



eaDirect User's Guide

eaDirect is a member of the eaSuite™ product line

V3.4
Document ID: DIGN-04-3.4-02
Data Published: 1.30.03

© 1997–2003 edocs® Inc. All rights reserved.

edocs, Inc., Two Apple Hill, 598 Worcester Road, Natick, MA 01760

The information contained in this document is the confidential and proprietary information of edocs, Inc. and is subject to change without notice.

This material is protected by U.S. and international copyright laws. edocs and eaPost are registered in the U.S. Patent and Trademark Office.

No part of this publication may be reproduced or transmitted in any form or by any means without the prior written permission of edocs, Inc.

eaSuite, eaDirect, eaPay, eaCare, eaAssist, eaMarket, and eaXchange are trademarks of edocs, Inc.

All other trademark, company, and product names used herein are trademarks of their respective companies.

Printed in the USA.

Table of Contents

- Preface.....9**
 - About Customer Self Service and eaSuite™.....9
 - About This Guide..... 12
 - Related Documentation 13
 - If You Need Help 13
- 1 Introduction to eaDirect Applications..... 15**
 - What is Electronic Statement Presentment?..... 15
 - The Statement Mastering Process..... 15
 - What is an Application? 16
 - What is a View? 16
 - How Do I Create an Application? 17
 - What is a DDF? 18
 - What is an ALF?..... 19
 - The HTML Template 20
 - What is DefTool? 20
 - What is Composer? 20
 - Using the National Wireless Sample Application..... 21
 - After Creating an eaDirect Application 23
- 2 Defining Data Extraction Rules with DefTool..... 25**
 - Overview of Creating a DDF with DefTool 25
 - Before Creating Your DDF 25
 - Steps for Creating a DDF..... 25
 - The DefTool Interface 26
 - The DDF Tree 28
 - The DefTool Work Area (Search Area)..... 28
 - Menus 29
 - Toolbars and Icons..... 36
 - Cutting, Copying, and Pasting 41

Creating a New Data Definition File	46
Creating a DDF for DJDE Files	47
Creating a DDF for AFP Files	56
AFP Font Settings	62
Creating a DDF for DB Extract Files	65
Creating a DDF for Line Printer Format Files	69
Creating a DDF for Metacode Files	73
Metacode Settings	76
Writing Data Extraction Rules	78
Identifying Unique Properties of Data to be Extracted	79
Search Area (Location)	79
Data Patterns (Content Search).....	81
Field Properties	82
Marker Properties.....	84
Table Properties.....	85
Creating an Extraction Rule (General Procedure)	87
Document Style (Start of a New Document).....	88
Document Recipient (Primary Key).....	95
Creating Additional Page Styles.....	100
Overview of Fields and Field Anchors.....	103
Creating a Field Definition.....	103
Custom Post-Conversions	106
Creating a new custom post-conversion.....	107
Editing a post-conversion.....	111
Removing a custom post-conversion from a field or column.....	111
Deleting a custom post-conversion from an application	112
Sharing custom post-conversions with other applications	112
Date and Time Format Constraints.....	114
Examples of Custom Post-Conversions	115
About Field Anchors	119
Creating a Field Anchor	122
Editing an Anchor.....	127
Deleting an Anchor.....	127
Tables.....	127
Extracting Tables for Text Presentation.....	128

Extracting Tables for Chart Presentation	128
Table Definitions	129
Creating a Table.....	130
Defining a Table in National Wireless	131
Defining a Table Start Marker	132
Defining a Table End Marker	137
Defining Table Columns.....	143
Defining the Table Anchor	146
Defining Groups.....	147
Identifying the Components of a Group Pattern	150
Table Columns	151
Creating a Group.....	152
Defining a Group.....	153
Defining a Group Start Marker	154
Defining a Group End Marker	160
Defining Group Properties	165
Presenting Sub-Account Data.....	169
Setting up Sub-Account Indexing.....	170
Creating a DDF for Indexing and Sub-Account Presentment.....	172
Modifying a Group Field or Subkey Properties	178
Deleting a Group Field or Subkey.....	178
The Extraction Simulator.....	179
Using the Simulator for the First Time	179
Navigating in the Simulator	180
Simulating Field Extractions.....	182
Simulating Table Extractions	184
Simulating Selective Extraction.....	185
Regular Expressions.....	188
Purpose of Regular Expressions	188
Using a Regular Expression with eaDirect	189
Writing Regular Expression Patterns	189
Illustration 1: Shorthand Date Formats MM/DD/YYYY	190
Illustration 2: Varied Shorthand Date Formats.....	191
Illustration 3: Currency Values	193
Illustration 4: Longhand Date Formats.....	195

Illustration 5: Data Values that Vary Widely in Pattern	197
Illustration 6: Static data strings	198
Illustration 7: Partially Static Data Strings	199
Regular Expressions Reference Chart	201
Dynamic Pattern Matching	205
Setting Dynamic Pattern Markers in DefTool.....	205
Simulating Dynamic Field Values	211
Completing the Dynamic Pattern Link in Composer	212
Element IDs	220
3 Defining Statement Presentment with Composer	223
Overview of Creating an ALF with Composer	223
Before Creating an Application Logic File (ALF).....	223
General Steps for Creating an ALF.....	223
Completed ALF and HTML Application Files.....	225
The Composer Interface	226
Application and Definition Trees	227
HTML Template Area.....	229
Menus	230
Toolbars	231
Creating a New ALF	234
For a Non-Subaccount View	234
For a Sub-Account View	236
Specifying a Default Template	238
Mapping Fields to the Template.....	239
Inserting Images, HTML Code, Hyperlinks, or HTML Tables	241
Creating Conditions	244
Creating a Logic Element.....	245
Creating an ALF Switch Condition	256
Adding Conditional Templates	261
Creating a Condition on a Statement Section.....	266
Deleting, Modifying, and Viewing Conditions.....	271
Creating Content	274
Creating New Content.....	274
Viewing Content	276
Changing Content	278
Deleting Unused Content	278

Mapping a Notification Link.....	279
Saving the ALF, HTML Template Files	280
4 Mapping Text Tables and Configuring Elements Using Composer	281
Mapping Tables and Groups.....	281
Mapping a Table as Text	281
Mapping a Group	282
Modifying the Group Template.....	284
Deleting a Group	284
Formatting Tables.....	285
Formatting an Entire Table	286
Formatting Individual Columns	286
Creating Conditions on a Table (Line Item).....	287
Creating Table Content	291
Deleting, Changing, or Viewing Table Conditions	296
Applying Formatting or Adding Content to a Pattern of Rows	298
Sorting Tables and Groups	301
Table Sorting.....	301
Group Sorting.....	303
Creating a Sort Element.....	304
Creating a Link to a Sort Element.....	306
Modifying a Sort Element.....	312
Deleting a Sort Element	312
Paging Statements	312
Creating a Page Element.....	313
Creating a Page Set.....	315
Inserting Navigational Paging Links.....	315
Modifying a Page Element	316
Removing a Page Element	317
Filtering Data	317
Table Filtering	318
Group Filtering	318
Creating a General Filter Element	319
Creating a Top/Bottom Filter	325
Linking a Filter Element to a Statement.....	328
Modifying a Filter Element	332

Deleting a Filter Element.....	333
Using Combine Elements to Page, Sort, & Filter Across Table or Group Occurrences	333
Creating a New Combine Element.....	334
Viewing and Deleting Combine Elements.....	335
Totaling with Arithmetic Elements	336
Creating a New Arithmetic Element	337
Modifying an Arithmetic Element	340
Deleting an Arithmetic Element.....	340
Using the Simulator	341
Simulating Dynamic Field Values	343
Simulating Sorting and Filtering Elements.....	345
5 Mapping Tables as Charts Using Composer.....	349
Chart Information in the ALF	349
Charting Attributes	352
Chart Information Contained Within the ALF.....	352
Chart Properties	353
Changing Chart Properties	364
Deleting Charts.....	366
Appendix A: Data Definitions.....	367
EBNF Description	367
Data Definitions	367
Appendix B: How Composer Assembles a Document	377
How Composer Assembles a Document.....	377
Index	379

Preface

About Customer Self Service and eaSuite™

eaSuite

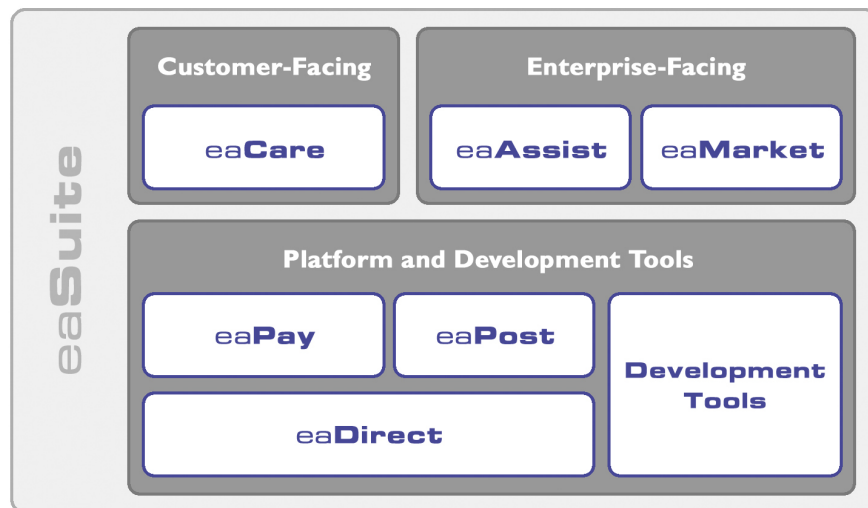
edocs has developed the industry's most comprehensive software and services for deploying Customer Self-Service solutions. eaSuite™ combines electronic presentment and payment (EPP), order management, knowledge management, personalization and application integration technologies to create an integrated, natural starting point for all customer service issues. eaSuite's unique architecture leverages and preserves existing infrastructure and data, and offers unparalleled scalability for the most demanding applications. With deployments across the healthcare, financial services, energy, retail, and communications industries, and the public sector, eaSuite powers some of the world's largest and most demanding customer self-service applications. eaSuite is a standards-based, feature rich, and highly scalable platform, that delivers the lowest total cost of ownership of any self-service solution available.

eaSuite is designed to support how organizations approach designing and deploying Customer Self-Service applications:

Customer-Facing Solutions present customers with the sophisticated functionality to meet customers' self-service needs. eaSuite offers a full set of capabilities to enable the range of business and consumer customer service activities, along with the flexibility to completely customize the solution to meet vertical industry and specific company requirements.

Enterprise-Facing Solutions empower employees within an organization and external partners to leverage the edocs platform to facilitate self-service and to support assisted service. Customer service representatives (CSRs), sales agents, account managers, marketing managers, broker-dealers and channel partners all play a role in delivering customer service, creating content, accessing information and performing activities for the benefit of customers.

Platform and Development Tools are designed to meet the rigorous infrastructure demands of the most technologically advanced organizations. These components of the eaSuite power edocs solutions with the functionality and development tools necessary to make account data available, and to create the customer- and enterprise-facing applications that enable customer self-service.



eaAssist

eaAssist™ reduces interaction costs and increases customer satisfaction by enabling enterprise agents – customer service representatives (CSRs), sales agents, broker-dealers and others – to efficiently access critical account data and service-related information to effectively service customers. Through its browser interface designed especially for the enterprise agent, eaAssist enables agents to take advantage of customer-facing online capabilities to provide better service by more efficiently resolving customer account inquiries at the point of customer contact.

eaMarket

eaMarket™ is the personalization, campaign and content management solution that enables organizations to increase revenue and improve customer satisfaction by weaving personalized marketing and customer service messages throughout the Customer Self-Service experience. The transactional account data that provides the foundation for a Customer Self-Service solution – such as transaction activity, service or usage charges, current task and prior service history – bring valuable insight into customers and can help optimize personalized marketing and customer service campaigns. eaMarket leverages that data to present relevant marketing and customer service messages to customers.

eaDirect

eaDirect™ is the core infrastructure of enterprise Customer Self-Service solutions for organizations large and small with special emphasis on meeting the needs of organizations with large numbers of customers, high data volumes and extensive integration with systems and business processes across the enterprise. Organizations use eaDirect with its data access layer, composition engine, and security, enrollment and logging framework to power complex Customer Self-Service applications.

eaPay

eaPay™ is the electronic payment solution that decreases payment processing costs, accelerates receivables and improves operational efficiency. eaPay is a complete payment scheduling and warehousing system with real-time and batch connections to payment gateways for Automated Clearing House (ACH) and credit card payments, and payments via various payment processing service providers.

eaPost

eaPost® is the account content distribution system that handles all the complexities of enrollment, authentication and secure distribution of summary account information to any endpoint, while also bringing customers back the organization's Website to manage and control their self-service experience.

Development Tools

eaSuite Development Tools™ are visual development applications that provide intuitive graphical user interface (GUI) environments for designing and developing Customer Self-Service solutions. The Development Tools encompass data management, workflow authoring, rules management and accounts receivable integration, as well as a full Software Developers Kit for custom application development.

About This Guide

This guide describes the tasks required to design and create an eaDirect application for presenting statements online. It provides instructions on how to use the eaDirect DefTool and Composer design and development tools.

This guide is intended for the application developer and those involved in the process of designing an eaDirect application.

This guide assumes you have:

- Completed a Statement Mastering Plan.

- Installed the eaDirect tools, DefTool and Composer, on a Windows NT or 2000 system.

Related Documentation

Online Help and a PDF version of this guide are also available.

Online	How to Access
Help	In DefTool or Composer, select Help>Help Topics .
A PDF of this guide	A PDF of this guide is available on the eaDirect product CD-ROM.

This guide is part of the eaDirect documentation set. For more information about implementing your eaDirect application, see one of the following guides:

Print Document	Description
eaDirect Installation and Configuration Guide	How to install eaDirect and configure it in a distributed environment.
eaDirect Production Guide	How to set up and run a live eaDirect application in a J2EE environment.

If You Need Help

Technical support is available to customers who have valid maintenance and support contracts with edocs. Technical support engineers can help you install, configure, and maintain your edocs application.

To reach the U.S. Service Center, located in Natick, MA (Monday through Friday 8:00am to 8:00pm EST):

- Telephone: 508.652.8400
- Toll Free: 877.336.3362

Preface

- E-support: support.edocs.com (This requires a one-time online registration)
- E-mail: support@edocs.com

When you report a problem, please be prepared to provide us the following information:

- What is your name and role in your organization?
- What is your company's name?
- What is your phone number and best times to call you?
- What is your e-mail address?
- In which edocs product did a problem occur?
- What is your Operating System version?
- What were you doing when the problem occurred?
- How did the system respond to the error?
- If the system generated a screen message, please send us that screen message.
- If the system wrote information to a log file, please send us that log file.
- If the system crashed or hung, please tell us.

Introduction to eaDirect Applications



eaDirect® is a high-performance enterprise Web statement and customer management solution that provides re-purposing and delivery of information over the Web. eaDirect is dependent on a number of distributable components, services, and web-based rapid application development (RAD) tools.

eaDirect provides a set graphical design tools for application development and a platform-independent web tool called the Command Center that provides centralized control of the activity in the production environment.

What is Electronic Statement Presentment?

Electronic statement presentment (ESP) allows customers to view their financial statements on the Web. The primary functions of electronic statement presentation applications are: triggering email notifications, time-sensitive sticky content, and a targeted use of web page design, advertising, incentives and messaging.

The Statement Mastering Process

Statement Mastering is the process of determining your organization's online presentment requirements and defining the eaDirect application needed to enable your users to view statements on the Web.

During the mastering process, the development team analyzes printed copies of each bill type, along with a representative production file, and identifies the online statements your custom application must present. This process identifies the data, text, and graphics your statements must bring together, along with any conditional business logic required.

The result of Statement Mastering is a Requirements Definition Document defining the custom views and design files your application needs.

Use the Requirements Definition Document along with the instructions in this guide to create an application for your organization. This guide describe how to use the eaDirect DefTool to specify how eaDirect can extract data from your data input file (the data extraction rules), and Composer to define how data and other elements map to HTML templates for Web presentment.

To make an ESP system, statements must be converted from a normal print format to an electronic presentation. A printed file originates from a data source. The resulting data stream from the source is formatted for printing functions. eaDirect also uses the data source, structuring the data by using set rules, then displaying the data on a pre-configured HTML template. There can be several different sets of rules and templates, and the templates can also contain business rules to modify the type of marketing and customization that is displayed. The end result is the same account information provided in a print statement is also provided in an electronic presentation over the Web.

What is an Application?

An eaDirect Application is a customized set of eaDirect files (DDFs, ALFs, statement HTML templates, and group HTML templates) created exclusively to extract and present the statements of a service provider online. You must design a custom application with unique DDF, ALF, and HTML files for each input data source. Each application has a unique name. A single service provider can have more than one application if they need to present data from input data sources with different formats or content.

What is a View?

A view is a set of design files that result in a particular presentation of statement data. The view files enable a user to dynamically display formatted statements live on the Web, receive email notifying them that an online statement is available, or to view account data in a static format online.

An eaDirect application can have multiple statement views for presenting different levels of information such as a summary page and statement detail pages. Dynamic Web views can present statement data in HTML, XML, CSV, or Chart format.

A typical HTML-based statement view (dynamic or static) consists of a pair of DDF and ALFs, and one or more associated HTML templates. These files identify which data to extract from the input file and how to display it to the user.

Refer to the Requirements Definition Document for your organization. This document resulted from the Statement Mastering process and defines each view you need and the design files you must create for each view.

How Do I Create an Application?

Defining an eaDirect application requires two phases:

- Defining data extraction
- Presentation design

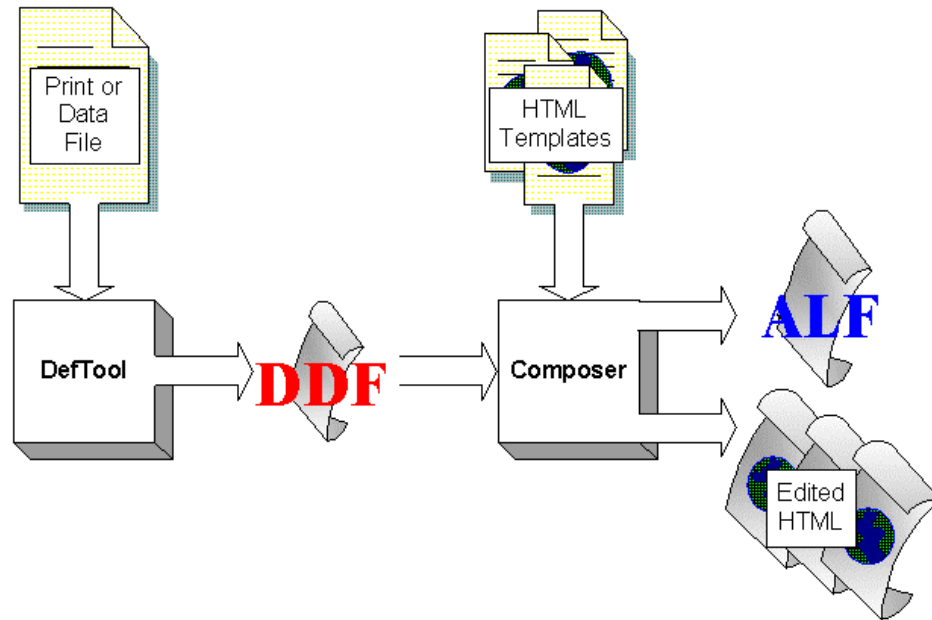
Defining data extraction consists of:

- Defining the file format properties of the input data file.
- Using the eaDirect DefTool to analyze and define the rules for extracting the data fields, tables, and groups from the input file.

Presentation design consists of:

- Creating the HTML templates used to compose the statements. (Use the tool of your choice to create HTML templates.)
- Using the eaDirect Composer to map the fields, tables, and groups (defined using DefTool) to the HTML templates.
- Using Composer to create business rules that let you customize statements for marketing purposes.

The following diagram shows a high-level view of the eaDirect application development process:



What is a DDF?

A Data Definition File, or DDF, contains the rules for finding and extracting data from an application's input data source. You create one or more DDF using eaDirect's DefTool exclusively for use with your organization's particular data source format and application.

The DDF is used by the Indexer job during production to prepare a data source and the entire eaDirect application for live statement retrieval. The DDF is then used on a regular basis during statement live retrieval to extract account data for dynamic display on the Web.

What is an ALF?

The Application Logic File, or ALF, contains the rules to present the data extracted from the original data source in a customized HTML format. You create one or more ALF's using the eaDirect Composer tool. The ALF (which is in XML format) is then used along with the DDF during live statement retrieval to display extracted account data on the Web, in email notifications, etc. The ALF also contains business logic (conditional statements that consider current statement data) for marketing and other customization purposes.

In addition to dynamic statement presentment, ALFs are also developed and used for the composition of static output, most commonly notification emails. An application can have multiple ALFs for use in presenting different statement Views.

The ALF defines:

- Formatting rules for presenting the data on an HTML template
- Paging, sorting, and filtering configuration information
- Business logic to customize and personalize a statement
- File names of the dynamic DDF and any HTML templates used for the statement view

eaDirect-specific tags are added to the HTML templates during the Composition process, indicating where the data should be inserted and where logic arguments are applied. These logic arguments, as well as other assembly instructions, are stored in the ALF.

You must create an ALF for each view, including a separate ALF for the notification email. You can also specify the conditions for the use of different ALFs.

The HTML Template

You use an HTML editor to create the HTML or HTM templates for presenting your statements online. However, they inherit some information about the fields, tables, groups and charts that are to be extracted and displayed when you map your data and other design elements in Composer.



Tip

Use placeholders in your template design. There are so many values in most bills and statements that it is easy to overlook one without a visual cue. This is particularly helpful if different people design the templates and map the values.

You can use Composer as an HTML editor while developing your application's ALFs for presentment.

What is DefTool?

DefTool is the eaDirect GUI application design tool you use to define the rules for extracting data from your application's data input file. DefTool presents the source file in a standard format, making it easy to see the statements and identify the data you want to extract. You can visually select the data elements required for presentment and business logic.

You must create a rule for extracting each data field, table, and group that you want to display in an online statement. As you work in DefTool, you can use the Extraction Simulator to test each data extraction definition.

The result of using DefTool is a Data Definition File (DDF) containing the data extraction rules for your application.

What is Composer?

Composer is the eaDirect GUI design tool you use to map the data extracted from your data input file to an HTML template for online presentation.

The result of using Composer is an Application Logic File (ALF) for your application and an edited version of one or more HTML templates.

When a customer makes a request to view a statement in a live eaDirect application, the eaDirect server uses the information in the ALF to locate the associated DDF, HTML template, digital images, and the data input file for extracting and present the statement.

Using the National Wireless Sample Application

A sample eaDirect application named “NatlWireless” is included with your system to let you demonstrate and practice using the eaDirect design tools (DefTool and Composer) with sample files and data. NatlWireless simulates telecommunication statements in an ASCII file format.

With each procedure in this guide, specific instructions for using NatlWireless are included so you can follow along (if you choose). The National Wireless input instructions appear in parenthesis after each step.

The NatlWireless application consists of the following files:

NatlWireless File	Purpose
NatlWireless.txt	Sample data input file (source).
NatlWireless.htm	HTML template for presenting the summary statement view (named HtmlDetail).
NW_LocSummary.htm	HTML template for presenting the detail statement view (named NW_LocSummary).
NWEmailDefault.htm	Default HTML template for notification emails.
NWEmail.htm	Alternate layout for notification emails (its use is determined by business logic).
NatlWireless.DDF	Contains the customized rules and business logic for extracting each field from each page of the original statement file for presenting different views of the data in online statements.
NatlWireless.ALF	Contains the customized rules and business logic for mapping and presenting the extracted data in an online summary statement.
NW_EmailDefault.ALF	Contains the customized rules and business logic for creating notification email.

NatlWireless File	Purpose
EDCSbd\samples\NW_Subdocument\NW_Subdocument.ddf	Contains the customized rules and business logic for extracting sub-document data from the original statement file for presenting a sub-account document view online.
EDCSbd\samples\NW_Subdocument\NW_Subdocument.alf	Contains the customized rules and business logic for creating a view of sub-account data.

After Creating an eaDirect Application

When you are finished using DefTool and Composer to create your application, refer to the eaDirect Production Guide for instructions on how to use the eaDirect Command Center to implement and run an application in a live production environment.

Defining Data Extraction Rules with DefTool

2

Overview of Creating a DDF with DefTool

Before Creating Your DDF

Before you can begin creating a DDF for your application, you must find a sample data source file that is representative of the format in which you deliver your statements. This file must contain a broad and varied sample of statement styles and data types.

You will use this sample data input file to:

- Identify the data you want to extract
- Define your DDF in DefTool

Much of the information required to identify data for extraction and create appropriate data extraction rules results from the edocs Bill Mastering process. You can use this book to assist in the Bill Mastering process.

Steps for Creating a DDF

To create a DDF with extraction rules for your application:

1. **Run DefTool.** From the Windows Start menu, select **Programs>eaDirect>DefTool**.

2. **Create a new (empty) DDF.** Follow the procedure for the particular type of data input file your application will use (DJDE, AFP, DB Extract, line printer, etc.). You must define your DDF using a sample data source file that is representative of the format in which you deliver your statements. The file you use to define the DDF must contain a broad and varied sample of statement styles and data types.
3. **Define the document style.** The document style defines how eaDirect can identify the start of a new statement in the data input file. It defines the first page of a new document based on a unique pattern that signals the start of a new document in the data source.
4. **Define the primary key.** The primary key contains the rules for locating the field defined as the unique identifier for each statement (and therefore recipient) in a billing cycle.
5. **Define additional page styles (if necessary).** If pages subsequent to the document style (first page) contain different content, you want to define additional page styles for each different type of page in your input file statements.
6. **Define the field, table, and group extraction rules.** You must define rules for extracting each element from your data input file that you want to display online. The rules tell eaDirect where to find the data (general location) in the file, and the type of information it contains (content). As you work in DefTool, use the Extraction Simulator to test each data extraction definition. DefTool also lets you apply formatting specifications to field and table column values after they are extracted (post-conversions).
7. **Save the DDF.**

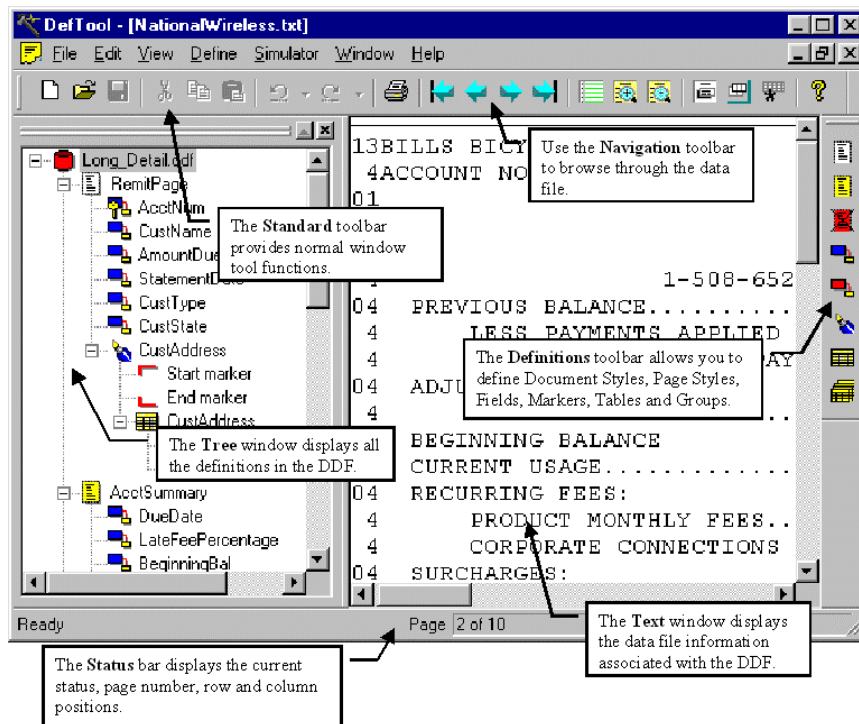
Your DDF is complete when it contains all the rules necessary to extract statement data from your input file for display online. When your DDF is complete, you can proceed to use Composer to map the fields to an HTML template for display and define any business logic you want to use.

The DefTool Interface

The DefTool interface consists of:

- The DDF tree on the left side
- The DefTool work area (search area) on the right side
- Menus
- Toolbars

The following diagram shows an open DDF in DefTool. The tree on the left shows the elements defined for extraction from the data input file. The work area on the right displays the content (statements) of the associated data input file.



The DDF Tree

The left side of the DefTool window shows all the rules created for the data input file to identify the various statement elements, including page styles, fields, tables, etc, in a tree format.

Click on the tree pages, tables, etc., to expand a list of elements.

When you first create a new DDF there are no elements in the tree.

The DefTool Work Area (Search Area)

The DefTool work area shows a digitized rendering of the sample data source in page-view format. This enables you to identify the search area location of elements for extraction graphically.

You can zoom in to get a close-up view of the file or zoom out to see more of the data file at a reduced size.

To enhance readability in DefTool, you can optionally view the search area, in colored bar-paper style shading.

To zoom in on a file:

- Select **View>Zoom In** or click .


To zoom out of a file:

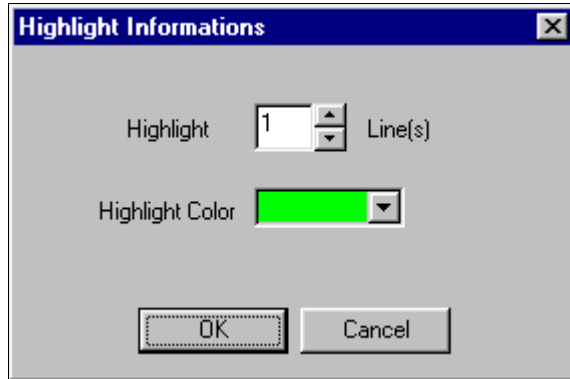
- Select **View>Zoom Out** or click .



To get a high-level view of a statement page, select **Zoom Out** several times until the entire page is visible in the work area. This is a good way to determine if the data is properly rendered on the page.

To view the data input file with color bar shading:

1. Select **View>Color Shading** or click . The Highlight Information dialog box appears.



2. Specify the number of lines to highlight.
3. Select a color from the Highlight Color drop-down menu.
4. To view more colors, select **More**. The Color dialog box appears.
5. Select a color.
6. To define a custom color, specify the color using the Color Matrix, the hue/saturation/luminosity, or the red/green/blue definitions. Click **Add To Custom Colors**.
7. Click **OK** at the next two dialogs.



Tip

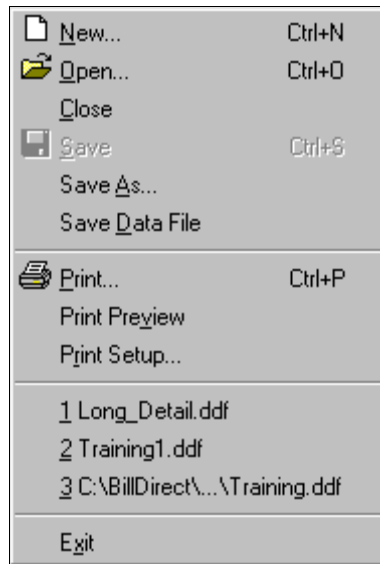
To remove the highlight bars, click the Highlight icon and select white as the Highlight Color.

Menus

The Standard Toolbar contains seven selections—each containing a number of user definable options.

File Menu

DefTool offers all the standard File Menu options, such as New, Open, Close, Save, Save As, Save Data File As, Print, Print Preview, Print Setup, and Exit. The Print Preview option enables a preview of the entire page displayed within the Work Area. This is very useful for quickly viewing the entire page to see where data falls within the defined boundaries.

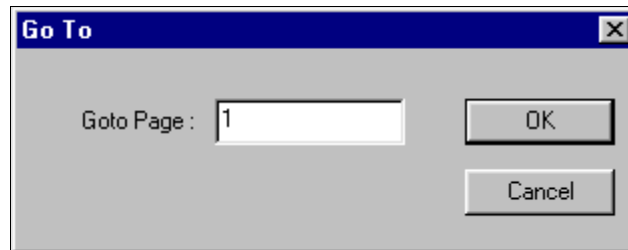


Edit Menu

The Edit menu allows moving quickly to any page with the **Go To** option. DDF Properties can change the data file properties of a DDF. The most recent operation can be undone (Undo), or an undone action can be redone (Redo).



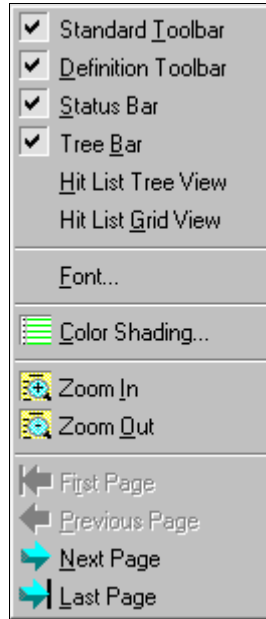
For example, you can go to a specific page within a file:



The **DDF Properties** option opens a window (shown below) that displays all the settings entered during the creation of the DDF, such as the Page Range, Normalization Characteristics, and Special AFP Settings. You can edit many of these properties by clicking on the **Edit** button, allowing any necessary changes without re-creating the settings in an entirely new DDF.

View Menu

The **view** menu option allows customization of the view for the DefTool main console and the work area.



You can elect to display any combination of the following items in the DefTool:

Standard Toolbar — located at the top of the DefTool main console and contains standard Windows tools, such as New, Open and Save.

Definition Toolbar — located on the right border of the DefTool main console and contains the tools needed to create rules for statement and page style detection and data extraction

Status Bar — located in the bottom frame of the DefTool main console and contains information about the current page in the work area and the most recent row/column coordinates in the work area where the mouse pointer resides.

Tree Bar — the left pane of the DefTool home window, containing the tree representation of the DDF rules.

The **Font...** option specifies a font for displaying the data source text in the Work Area.

Color Shading defines horizontal color bars that will overlay the text in the work area. A specific number of rows can be highlighted.

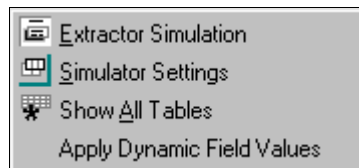
Define Menu

The Define options can be used to specify document styles, page styles, fields, markers, and tables. The Define options are also available in their own toolbar.



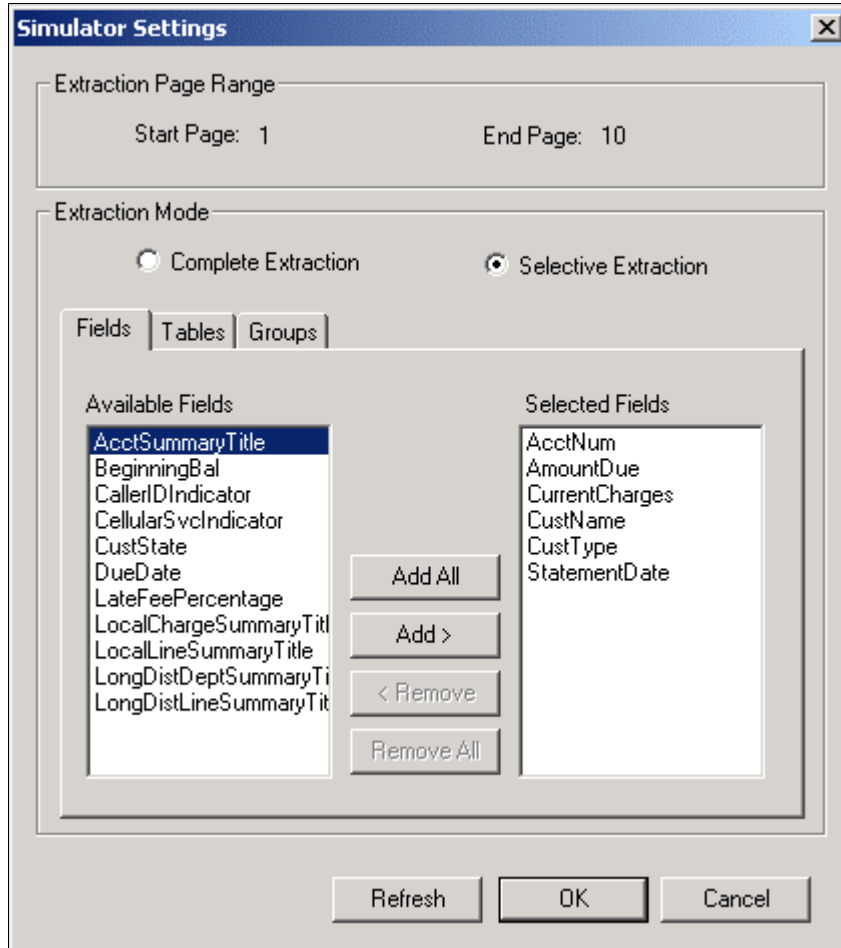
Simulator Menu

This brings up the Extractor Simulator that is used to test the definitions.



To test the extraction rules against the data source Page Range, select the **Extraction Simulator** option. By default, the Extraction Simulator will simulate extraction of all the rules.

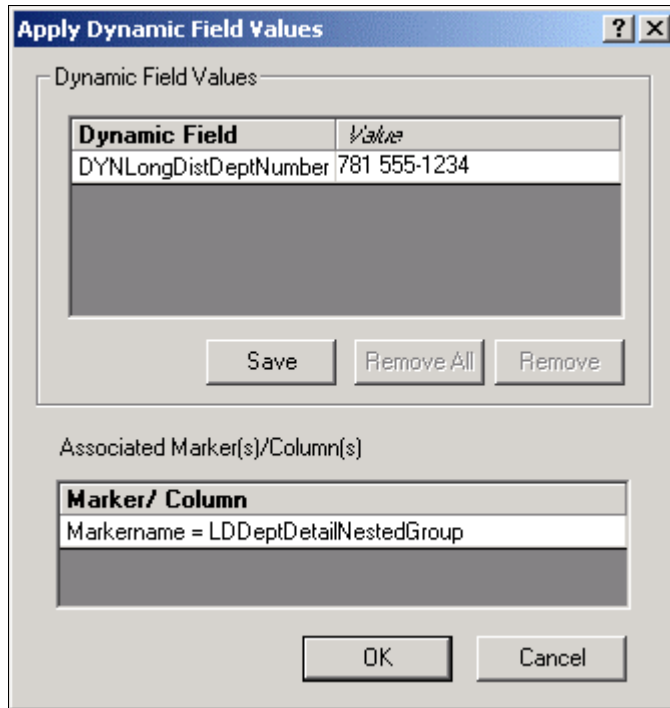
To perform selective extraction, i.e. to select particular rules for testing extractions, choose the **Simulator Settings** option (as shown in the following figure).



The selections made here will be saved to the Registry of the development server, so that each simulation will provide the results for the data items you want to view.

By default, the DefTool Simulator will show the first occurrence of a table within a group, and allow navigation to the subsequent occurrences as desired. To view all occurrences of your group's tables, select the **Show All Tables** option. This option works in a toggle mode; you do not need to select it repeatedly before each extraction.

If you are building an extraction rule to retrieve a specific group occurrence by dynamic pattern matching, you can test the extraction by selecting the **Apply Dynamic Field Values** menu option.



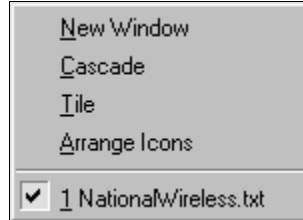
From the dialog that appears, specify a possible occurrence of the dynamic field by typing it into the Value field. You may elect to save this string to the Registry to streamline your testing process.



The use of the Apply Dynamic Field Values option requires that dynamic pattern matching be used in at least one of the group markers, table markers, or table columns.

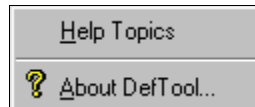
Window Menu

The Window options provide the standard tile, cascade, and arrange capabilities.



Help Menu

Provides access to the online Help for DefTool and Composer and to the About box.



Toolbars and Icons

There are two primary toolbars in DefTool:

- Standard Toolbar
- Definition Toolbar

Standard Toolbar

The Standard Toolbar, shown below, contains icons that allow quick access to the basic File, Edit, and View functions. By simply holding the mouse over the icon, a “Quick Help” description will appear.



Navigation Icons

The Navigation icons (First Page, Previous Page, Next Page and Last Page) allow movement from page to page of the sample data source within a previously selected page range.



Simulator Icons

The simulation icons are used to test the data extraction rules that you create in the DDF.



Definition Toolbar


The Definition Toolbar is used to activate the various data extraction options. These options are: Document Style, Page Style, Anchor, Field, Marker, Table, and group. With the exception of the Table and Group icons, a tool selection will remain until another tool is selected.



Tip

You can drag and drop the Definition Toolbar next to the Standard Toolbar.

Document Style

The Document Style icon  is used to create the Start of Statement setting. This mandatory setting is placed in all DDFs to determine where the statements begin. In addition to delimiting statements in a data source, this setting is also essential during the data source indexing process, when it indicates where the statements of the data source begin.


A document style represents the entire document for a given customer. The DocStyle identifier (data) must be a unique element that appears at the beginning of each document.

This data is stored in the index for all statements, and is used to locate statements in a data source during dynamic data retrieval.




The New Document Style icon is always the first tool that is used in your DDF. You must create the document style before creating any other extraction rule. The document style is only used once in each DDF.

Page Style


The Page Style icon  defines rules for detecting the “look and feel” of a unique style of page. The information needed to determine how many additional page styles your statements may have, if any, would be developed during your edocs Statement Mastering sessions.

In summary, a unique page style would not only have a different visual layout from the other pages in a statement, but it would also contain data elements that would not appear on other page styles. More than one statement page may use the same page style. An example of different page styles can be found on a statement from a telecom company. The first page of each statement is often a remittance page, presenting customer name, account number, and total amount due. The subsequent pages of the statement may have a different layout and contain different data elements. For example, one or more detail pages may present itemized tables for different calling plans. A new page style would be created and rules to extract the call itemization data would be associated with the detail page style, allowing the extraction components to only consider pages of this style when data elements are dynamically extracted.

Discard Style

You would define a discard page style  in the same fashion as an additional page style, however, the extraction components detect any pages that fall under this rule and completely disregard them during data extraction. The discard page style defines the page styles that are NOT considered for any type of data extraction.

New Field


The New Field tool  defines the rules for extracting data items whose value exists on one row and appears once per statement (multiple occurrences of the same value may appear on a statement, such as an Account Number, but there is only one real value for the Account Number in each statement).

Sometimes a field location may vary widely across a statement page. For example, the Total Amount Due may appear close to the top of a statement with few details, or near the bottom of a longer bill. In this case, you will need to create multiple locations for the field and an additional extraction rule to provide extra assurance that the correct data will be extracted. This additional extraction rule is called a field anchor and is also created with the New Field tool.



Use anchors with Markers and page styles also. Create them in the same way as fields — using the appropriate tool and selecting **Anchor For** in the dialog box.

New Dynamic Field


The New Dynamic Field tool  creates a dynamic field that is used in the pattern of a group marker, table marker, or a table column. You may want to quickly provide your end-users with a hierarchical and organized view of their statement data, especially if the statements are very large. For example, you may want to present the Summary information of a telecom bill and allow your recipients to “drill down” into a more detailed statement view by providing a link that will retrieve the Detail data when selected. If the statement is very large, it may be useful to design the links in the Summary view to perform retrieval of only a section of the Detail data.

For example, to retrieve Detail data for the Phone Line 508-652-8999 by clicking on a link in the Summary view, you can order that only detail that pertains to 508-652-8602 is extracted and presented. Part of the configuration to provide for this functionality starts at the DDF level. You can use a dynamic field in the pattern of a marker of the group that extracts phone line detail. During live retrieval, the pattern is dynamically filled with a specific phone line, such as 508-652-8602, guaranteeing that the extracted data pertains only to this Phone Line.

Local Line Summary		Local Line Detail			
(508) 652-8601	74.23	Phone Line: (508) 652-8602			
(508) 652-8602	28.98	FEATURES			
(508) 652-8603	9.40	02/25 - 03/24	FCC ACCESS CHARGE		8.25
(508) 652-8604	31.60	02/25 - 03/24	CALL FORWARDING		2.90
		02/25 - 03/24	TELECOMM. RELAY		10.07
			STATE TAX		1.31
			FEDERAL TAX		1.55
Local Line Grand Total	\$144.21		FEATURES SUBTOTAL		\$24.08
			ITEMIZED LOCAL USAGE		
		<u>DATE</u>	<u>CITY</u>	<u>MINS</u>	<u>USAGE</u>
		03/01	DIR ASST	1.0	0.95
		03/03	DIR ASST	1.0	0.95
		03/10	DIR ASST	1.0	0.95
		03/15	NEW BERN MN	2.5	0.50
		03/23	DULUTH MN	6.3	1.55
			ITEMIZED LOCAL USAGE SUBTOTAL		\$4.90
			SUBTOTAL FOR (508) 652-8602		\$28.98


New Marker



The New Marker tool  is used to define the start and end markers of tables and groups. Select a data string (static or variable) that always appears at or near the start of a table or group, and use the New Marker tool to create the rules to find that data string. The same tool is used to define rules to locate the data string that signifies the end of a table or group.

New Table




The New Table tool  creates rules for extracting one or more related data strings. These data strings may have multiple occurrences, and are extracted as table columns. A table can have only one column, such as a customer address section. The customer address is a field that can occur on more than one line. Since this data does not qualify as a field (more than one row of data), it must be extracted as a table.

A good example of multiple columns that are related to each other in some way are the components of an itemized phone call, containing a date, time, location, phone line called, number of minutes and charge. These data are not appropriate fields because they have iterations within a statement. A call itemization list can contain thousands of items, with thousands of occurrences in each of the columns suggested above. The table functionality is useful because it does not require knowledge of the number of rows a field may have, or how many occurrences of call items appear. The extraction components use the location of the start and end marker to determine how many rows should be searched for data.

Before creating a table, the start and end markers must be selected and defined using the New Marker tool. Then use the New Table tool to define the table name, select the markers used, and define the columns.

New Group

The New Group tool  is used to define which tables or groups should be considered for extraction together. A group can be a collection of tables, (or other groups), that appear in statements as a pattern. The pattern may have several iterations in a statement. The start and end location of each group occurrence must be determined and a start and end marker defined respectively for the data string. Next, identify and define the items that will be part of the group. These items can be tables or other groups. (Fields cannot be selected as part of a group.) Use the New Group tool to define the group name, select the markers to be used and select the items (tables and groups) that are in the group.

Names must not contain spaces or special characters, and must start with an alpha character. The names must not exceed 30 characters in length.

Cutting, Copying, and Pasting

You can use cut, copy, and paste functionality to move or copy the following DDF content within a DDF or to another DDF:

- Page styles
- Fields, including group fields

- Markers
- Tables and columns
- Groups

Using cut, copy, and paste functions saves time and reduces errors when building multiple application views.

(Note that you can only open a second DDF with a second data input file.)

If you copy and paste an entity into the same location as the original, you may be prompted to give it a new name. A “location” just refers to another place within the same DDF.

When you cut and paste an item, DefTool also moves or copies any associated child items (entities that belong to them) as child items in the tree.

The basic rules and behavior associated with copying each type of DDF entity are described here.

Page styles

When you copy and paste a page style (in the same DDF), DefTool prompts you to enter a new name. DefTool copies all the entities belonging to that page style onto the new one. Therefore, if a marker has a location in one page style, DefTool creates another marker in the target page style as well. (DefTool displays the marker under the root of the tree, not the node of the page style.)

Fields, group fields

You can cut, copy, and paste fields within the same page style or to a different one. You can also cut, copy, and paste group fields from one group to another.

If you paste a field within the same page style, you must provide a new name for the field.

If you paste a field to another page style, DefTool adds it as another location, so you do not need to rename the field.

Markers

You can move or copy a marker pair (start and end markers) to the same or another page style. DefTool moves or copies the associated tables and groups with the marker pair.

If you copy a marker pair to another page style, DefTool adds an additional location to the markers and displays the marker in the Definition Tree root (and not under the page styles).

If the source and target page styles are the same, DefTool adds a new location to each marker, and markers remain under their page node.

You can also move or copy *individual* start and end markers to the same or to another start or end marker.

Note that if you rename a marker location, it is equivalent to renaming the marker.

Tables, columns

To copy a table, you must paste it to the marker pair you want to associate it with.

When you copy and paste a table, DefTool prompts you for a new name and uses this name as a prefix to the columns of the new table. DefTool displays the renamed table under the marker node in the Definition Tree.

When you paste a table, DefTool copies or moves the associated anchor and columns.

You can also cut or copy a column from one table to another table. DefTool prompts you for a new name. If you cut a column that is an internal anchor, the anchor becomes external.

Groups

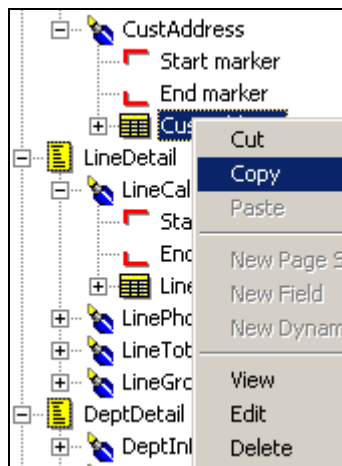
You can copy and paste a group to the same set of markers or to a new pair you want to associate the group with. When you copy and paste a group you must provide a new group name (only the top-most group is a copy; the table and column names remain the same).

When you paste a group, DefTool moves all child groups and tables belonging to the group.

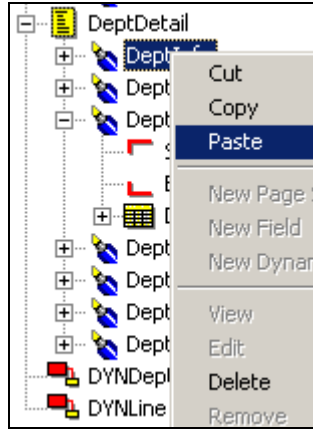
You can also copy a group and paste it as a child group under another parent group (in this case the group name stays the same).

To move or copy DDF entities:

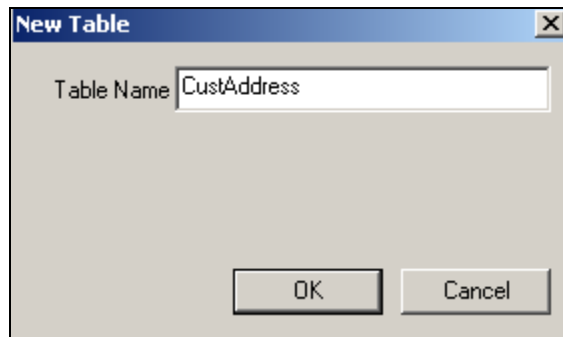
1. With the DDF open in DefTool, right-click the item in the tree you want to cut or copy and paste. To select multiple items, press **Ctrl**, then right-click on each item. To select a range, select the first item, then press **Shift** and click the last item.
2. Choose **Cut** or **Copy** from the right-click menu.



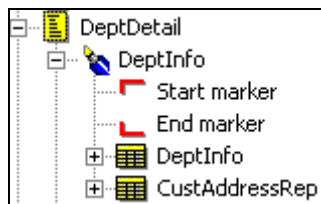
3. Right-click the target location or item in the open DDF (or switch to another open DDF and right-click the target item there) and select **Paste** from the right-click menu.



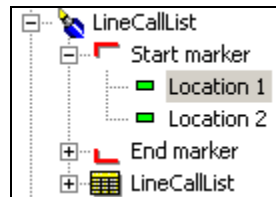
4. If DefTool prompts you for a new name, enter a unique name for the item and click OK.




5. DefTool redisplay the Definition Tree to reflect the result of your edits.



- Note that if you copy and paste a marker pair to a different page style, DefTool displays the markers under the root, with multiple location nodes beneath it. To see which page style a location node represents, right-click on the node, such as “Location 1” and select **E**d*i*t from the right-click menu. The Marker Properties dialog shows the page style name.



- Click  to save the DDF.

Creating a New Data Definition File

The DDF is used during live production to instruct the parser to extract key data elements from the defined input file type. eaDirect can accept input data from various file types, including:

- AFP (Mixed Mode and Fully Composed)
- DJDE (Dynamic Job Descriptor Entry)
- Metacode
- XML (Extensible Markup Language)
- Line Printer
- DB Extract

Essential to the creation of a DDF is the identification of the data source and its properties. When creating a new DDF based on a sample data source, you submit specific information about the data source in order for the DefTool to convert settings intended for a print- or storage-style format, into settings to render the data source in digital format. The information required by DefTool varies depending on the data source format and unique settings. The edocs Application Profile Document outlines the questions to ask about the data file. These answers, when used with the DefTool Configuration Wizard, will assist you in creating a DDF.

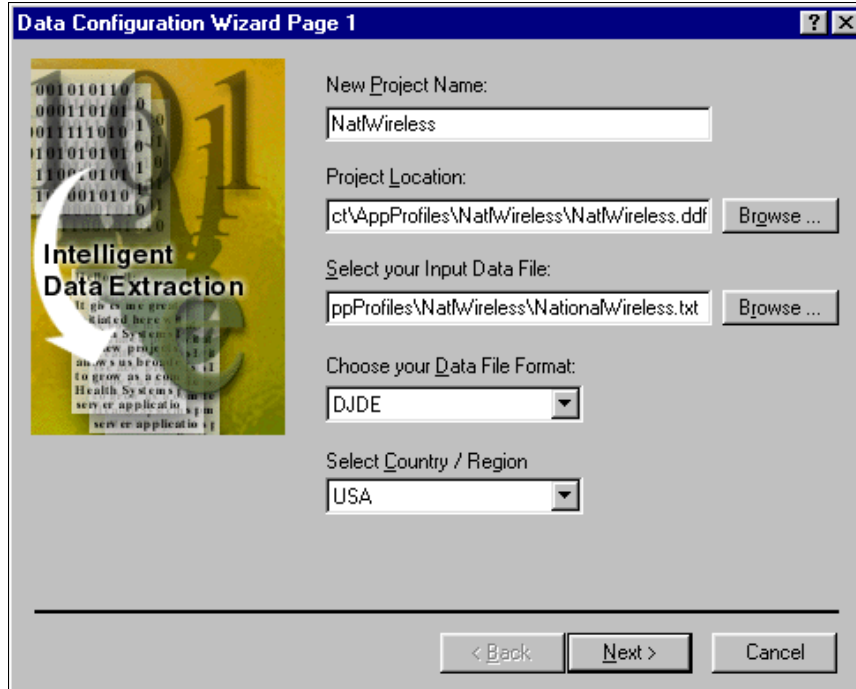
The Data Configuration Wizard pages for creating a DDF will vary, based on the file type selected and are presented by input data file formatting, starting with the DJDE format.

Creating a DDF for DJDE Files

The NatlWireless sample data source (NatlWireless.txt) is an ASCII text file from a Dynamic Job Descriptor Entry (DJDE) formatted input data file.

To create a new DDF for a DJDE data input file:

1. Select **Start>Programs>eaDirect>DefTool1**. The DefTool window appears.
2. Select **File>New** from the Main menu. Data Configuration Wizard Page 1 appears.



3. In the **New Project Name** field, enter the project name. This can be the application name (DDN) or the name of the statement view. (Enter NatWireless.)

Caution The project name is the descriptive application name assigned during project planning. Select a meaningful name. This name will be associated with the constructed DDF.

Names must not contain spaces or special characters, and must start with an alpha character. The names must not exceed 30 characters in length.

4. Next to the Project Location field, select **Browse...**, navigate to `C:\EDCSbd\Samples` and enter **ProjectName.ddf**. (Enter NatWireless\NatWireless.ddf.)

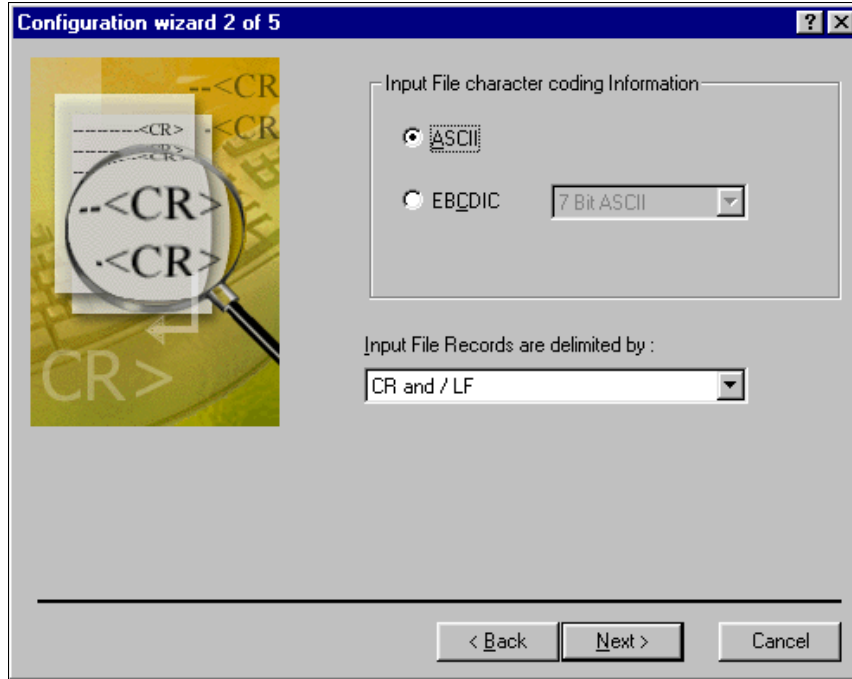
Caution The name of the DDF will be affiliated with the project name entered above, and should therefore match.

5. Click **open**. The Data Configuration Wizard Page 1 returns.
6. To select your input data file, click the **Browse** button to the right of the field and navigate to the location of the data source that will act as the broad sample of statements for building the rules for data extraction. (Select *C:\EDCSbd\Samples\NatlWireless\ NatlWireless.txt*.)
7. Click **open**. Data Configuration Wizard Page 1 appears.
8. To choose your data file format, click the corresponding dropdown list and select the data format. This will be the expected data format of all data sources for extraction. (Select *DJDE*.)
9. From the corresponding dropdown list, select the country/region from which the data source originates. The selection of country or region here will force DefTool to select the appropriate character set. (Select *USA*.)
10. Click **Next**. Configuration Wizard 2 of 5 appears.



Tip

From this point forward, the Data Configuration Wizard pages vary, based on the file type you select. For more information, see the section for each file type.

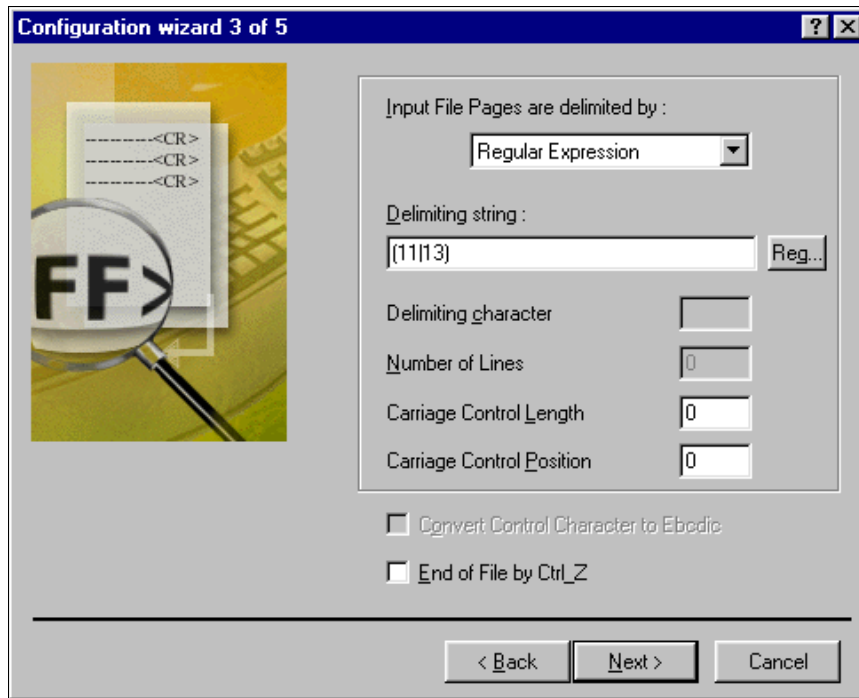


Depending on the data format, the DefTool can auto-detect configurations. For example, the DefTool knows that AFP files store data in the EBCDIC, rather than the ASCII, character set and will use a Carriage Return <CR> as the separator of file records.

Text, database extraction or line printer data sources can vary widely in format and design, and frequently do not contain the information needed to set the layout of the statements. Thus, the DefTool cannot automatically obtain settings, such as Character Set or table Separators. A general rule of thumb is that the less structured the data source, the more information the Data Configuration Wizard will require.

11. Select the appropriate type of input file character coding information. This informs the DefTool about the character set that is used in the input data source, and the mode of conversion to ASCII, if necessary. (Select ASCII.)
12. Select the method that input file records are delimited by, from the corresponding dropdown list. (Select CR / LF (Carriage Return and Line Feed).)

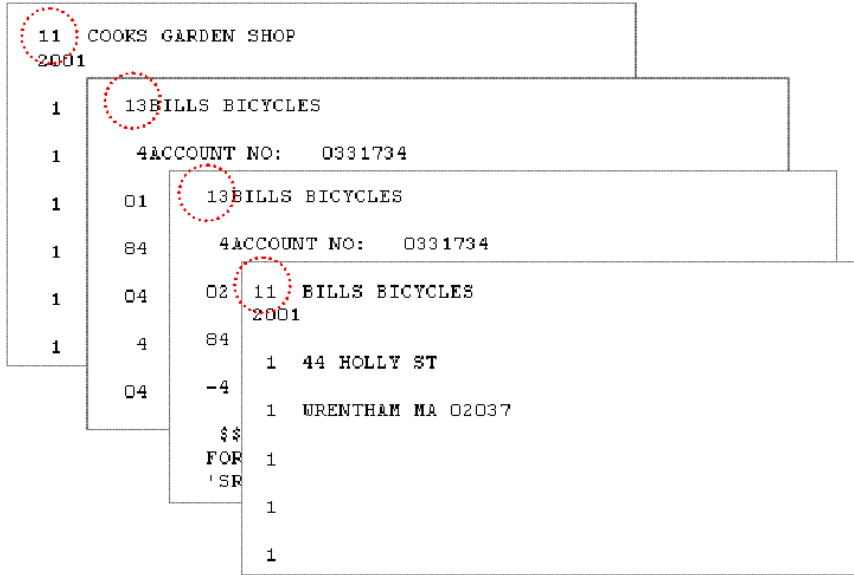
13. Click **Next**. Configuration Wizard 3 of 5 appears.



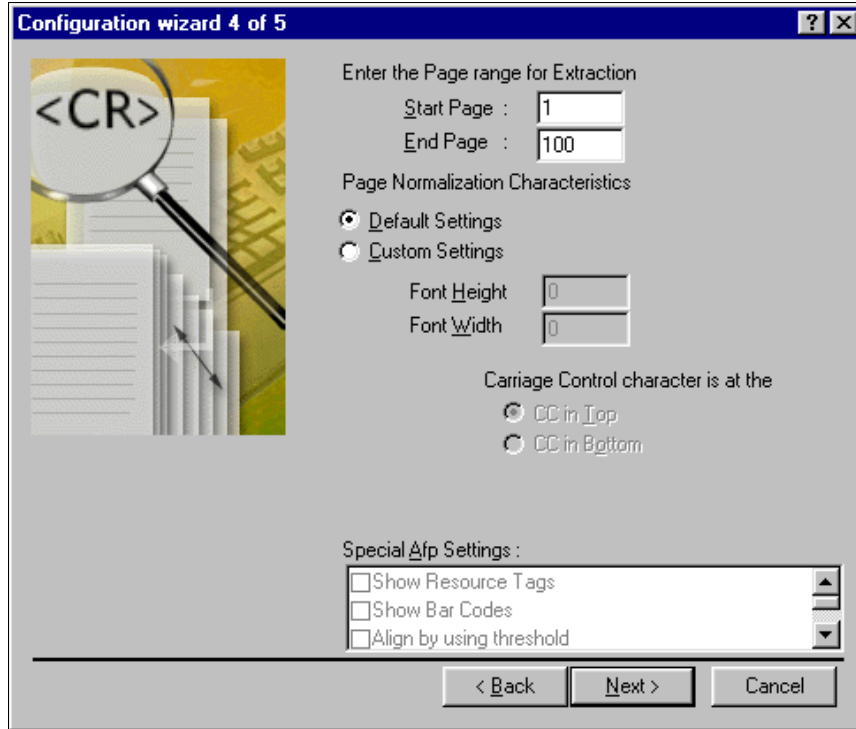
14. Specify the method that input file pages are delimited by from the corresponding dropdown list. (This information is usually obtained from your edocs Statements Mastering sessions.)

From the statement page segments of the National Wireless data shown below, it is determined that all of the Input File Pages contain the string 11 (on first pages) or a 13 (on subsequent pages), starting at the top left corner of the page. Therefore, the **Input File Pages** are delimited by **Regular Expression**, and the **Delimiting String** (using eaDirect regular expression syntax) is **(11|13)**. (Use this expression for the National Wireless input file page separator.)

Defining Data Extraction Rules with DefTool



15. If carriage controls have a known length or position, enter those values in the corresponding fields of Configuration Wizard 3. (For National Wireless, accept the defaults of Carriage Control Length and Position = 0, and click Next.) Configuration Wizard 4 of 5 appears.



16. Enter the page range for extraction; i.e. Start Page = 1 and End Page = 1000. The Page Range for Extraction setting allows a temporary selection of pages for use in developing the extraction rules (which will eventually apply to all the pages in the data source). Again, this setting is temporary to the current DefTool session, and has no bearing on the number of pages that will be extracted from a data source with this DDF during deployment of the application. Also, if it is discovered during a DefTool session that the selected Page Range does not contain an adequate sample of data, the Page Range is easily adjusted to open more pages of the data source while working in the DDF. (For NatlWireless.txt, set the Page Range for Extraction to a Start Page of 1 and an End Page of 100.)

Select a page range that includes a broad enough sample of statements, page styles and data elements, so the rules built into the DDF will capture all statement data. Sometimes, the sample data source you use can be very large, for example 100,000 or more pages. Testing your DDF rules on such a large data sample will create long run times for the extraction simulation. . A much smaller sample of pages will often suffice for obtaining a breadth of data styles. A recommended Page Range for DDF development is 10,000 or fewer pages

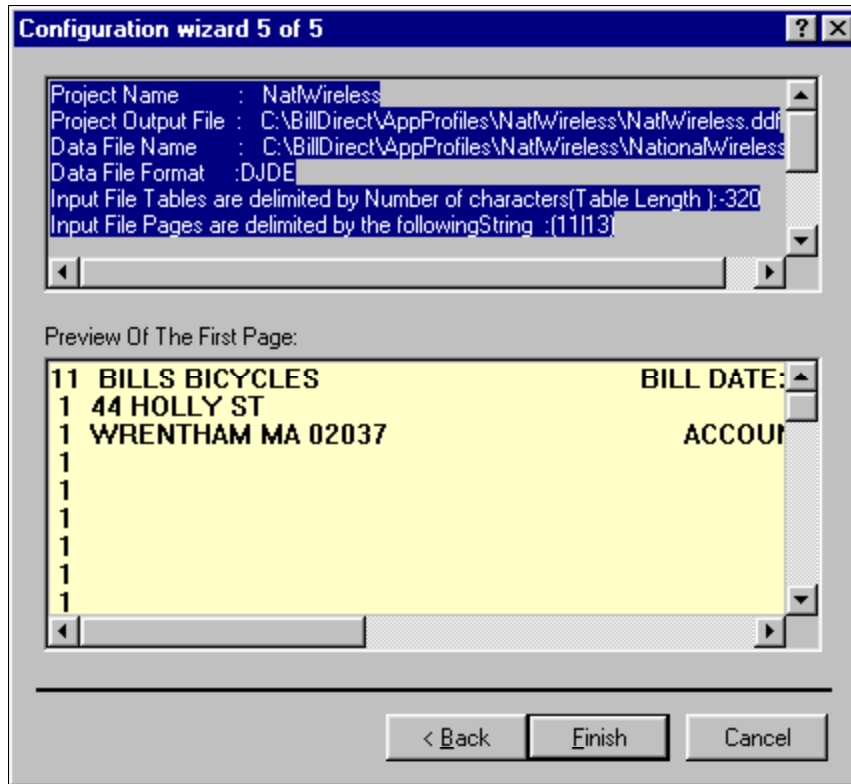
17. For page normalization characteristics, select **Default Settings**. This allows the DefTool to detect the high-resolution coordinate system of the data source, and translate this into the optimal lower-resolution coordinate system that will manage the digital rendering of the data source without sacrificing the data itself. The DefTool uses the resolution information (300 dpi, for example) and the page size (8 ½ × 11, for example), both of which are usually stored in the data source, and translates this information to build a system that is:
 - Of much lower resolution that is appropriate for a digitized view of the data, and
 - High enough resolution to accomodate all data in the pages of the data source.

The DefTool is often able to determine the appropriate lower-resolution coordinate set by using the Default Settings. In some cases, however, a higher-resolution coordinate set is needed to render all page data properly and in its own byte locations. A good example of this is a formatted data source (such as an AFP) that contains data formatted to very small font character widths. If left to the Default Settings, the new coordinate system created in the DDF to render the data source may cause the smaller-font data to be overlapped by other data on the page. The DefTool will often need you to make Custom Settings, such as increasing the Font Width, in order to allow all data to “occupy their own bytes.”



Font, height, and width relate to the grid size used when rendering print file data to the browser within DefTool, not the size of font displayed in the window. Adjusting the Font Height adjusts the number of vertical positions from the print file that map to a DefTool position. For example, Font Height 12 indicates that 12 print file positions map to one DefTool browser position. The same holds true for Font Width. Wider font spacing provides additional workspace on the page.

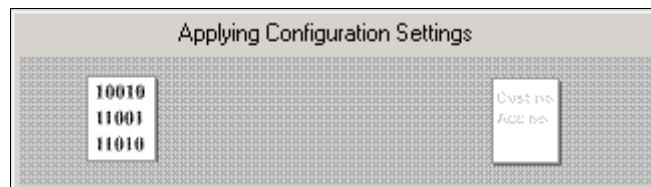
18. Enter the appropriate font height and width as required. (Since there are no very small font widths or other special formatting in *NatlWireless.txt*, so accept the Default Settings for the Page Normalization Characteristics.)
19. If your data source has any special AFP settings for extraction and presentation in the DefTool, check the appropriate items from the list in the corresponding scroll box. (For National Wireless, accept all other defaults, do not select any Special AFP Settings, and click Next.)
Configuration Wizard 5 of 5 opens.



The last page of every Configuration Wizard displays all of the settings created in the previous Configuration Wizard pages, regardless of the data format. It also provides a preview of the first page as it (and other pages) will be rendered in the DefTool work area. Scroll through this view to determine at first glance if the DDF properties are sufficient to accommodate all the data on the data source pages.

Confirm that the DDF properties match those presented in the configuration wizard and that all the data on the first page of Statement 1 appears in the preview of the first page.

20. Click **Finish**. The Applying Configuration Settings splash graphic appears, and the data source will appear in the DefTool work area. (A DDF icon appears in the Application Tree with NatlWireless.ddf as the parent tree node.)

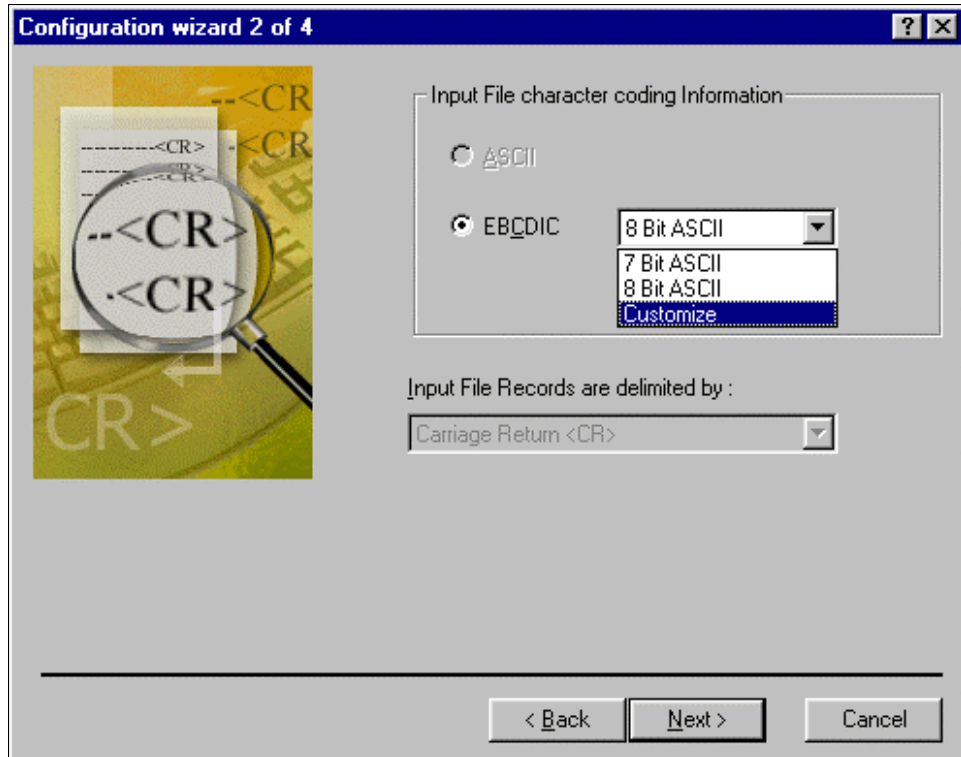


Creating a DDF for AFP Files

To create a new DDF for an Advanced Function Printing (AFP) file:

1. Select **File>New**. The Configuration Wizard 1 of 4 appears.
2. Specify the project name. The project name cannot be more than 16 characters, start with an alpha character, and should not contain spaces or special characters.
3. Specify the project location or click **Browse** and select a location. The default location is the *Samples* directory.
4. Specify the data file name and path or click **Browse** and select a path.
5. Select **AFP** as the file format.

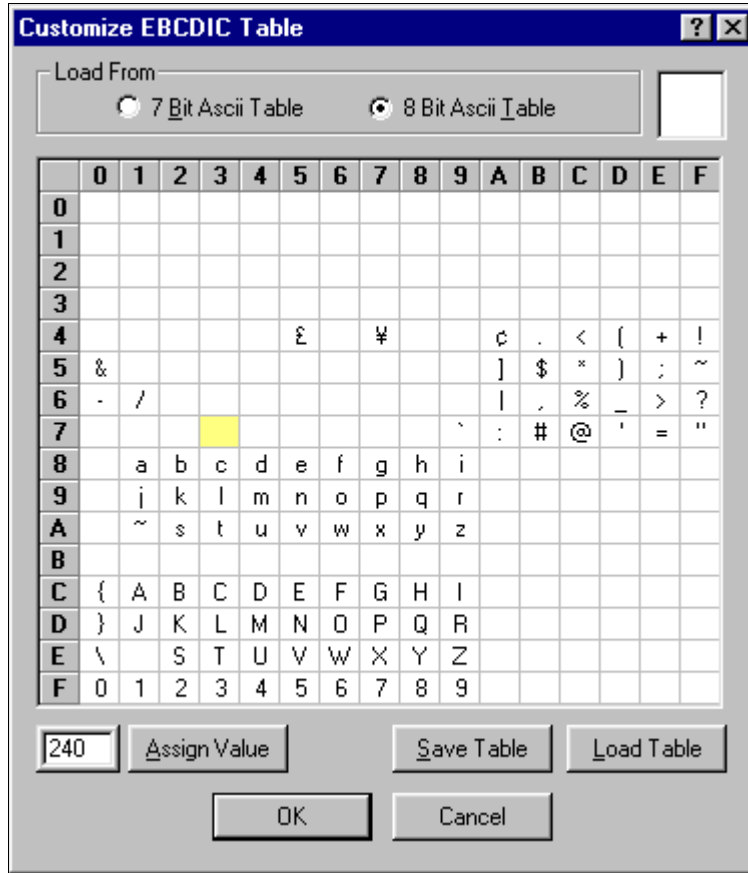
6. Select the country or region. The country/region is important for comparing currency fields, as the currency format differs for different locales.
7. Click **Next**. The Configuration Wizard 2 of 4 appears.



8. Specify the input file character coding. For AFP this must be EBCDIC.
9. Select **7-bit ASCII**, **8-bit ASCII**, or **Customize** to convert EBCDIC characters into ASCII.

If you want to support Extended Characters, you have to create a customized map of EBCDIC to ASCII, which is then saved in the DDF. DefTool also allows you to create a map and save it to an external file, which can be imported into DefTool. You can create a single map for every language, and then import that map for use with any DDF using that language.

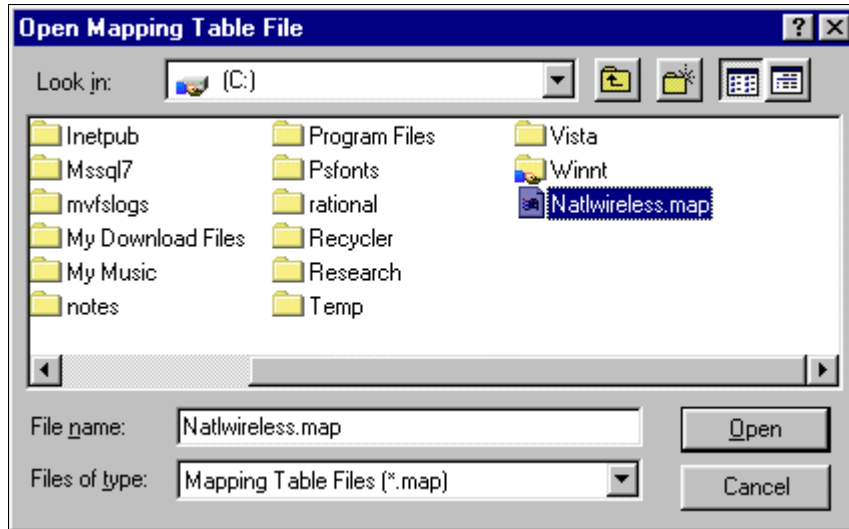
10. If you select **Customize**, the Customize EBCDIC Table dialog box appears.



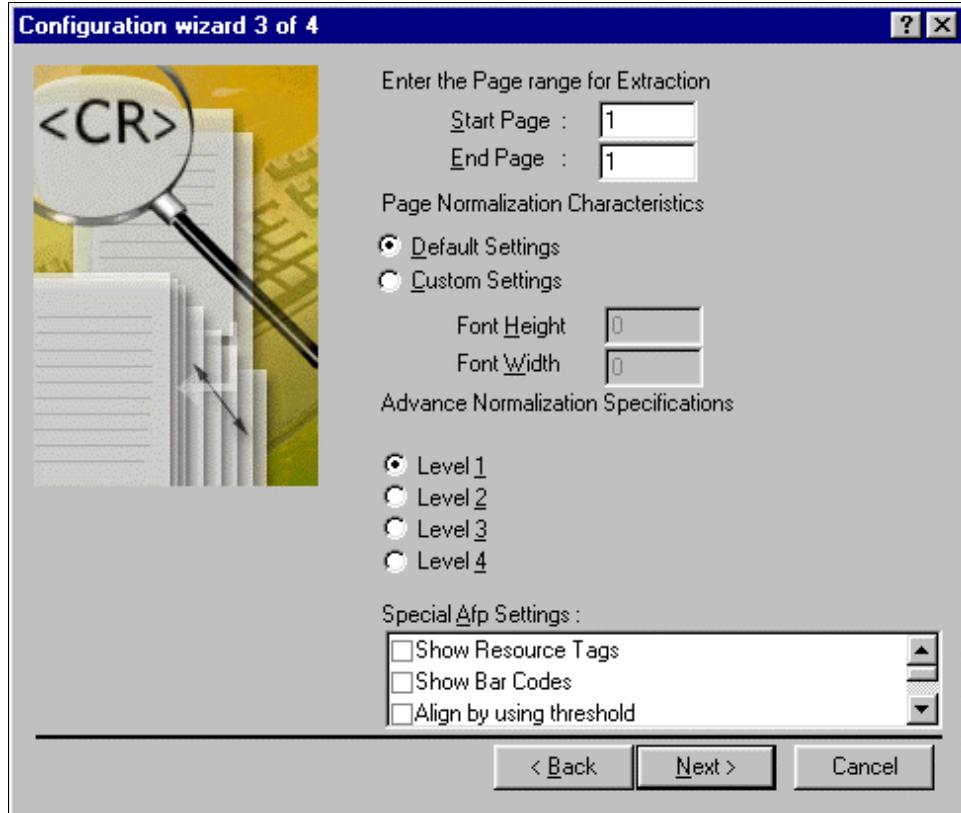
11. Select either the 7-bit EBCDIC table or the 8-bit EBCDIC table as the table from which to load the characters.



If you want to load an existing map, click Load Table and then click OK.



12. In the grid, select the location of the EBCDIC value that should be replaced by the Extended Character. The location will be highlighted. Enter the value for the character in hexadecimal in the data box next to Assign Value.
13. Click **Assign Value**.
14. To save the map to an external file, select **Save Table**.
15. Click **OK** to return to the Wizard 2 of 4.
16. Click **Next**. The Configuration Wizard 3 of 4 appears.



17. Specify the page range for extraction. It is not required that the start page be 1. For example, the page range can be 5-100. A large range, however, will impact simulation run times. Make sure that the selected range includes a representative sample of data.

18. Select either **Default** or **Custom Setting** as the page normalization characteristic.

The page normalization characteristics determine how the data file will be displayed in DefTool. DefTool assesses the data elements in the data file and makes a “guess” for presentment. It is better to select **Default** and preview the results, and then make any necessary changes as needed.

If **Custom** setting is selected, the font height and width must be specified. The font height and width refer to the spacing used when rendering data within DefTool, not the size of font displayed in the window. When you specify a value of one for width, you are telling DefTool to increase horizontal spacing by a factor of one. For each blank space encountered across, DefTool will add an additional space with this setting. When you specify a value for Font Height, you are telling DefTool to add additional vertical spaces at the factor indicated.

The maximum font height for an AFP printer is 72 points (1”) which translates out to 240pel, 300pel, 480pel or 600pel depending on the printer’s resolution. There is no limit to the font width - but it will usually be in direct proportion to the character represented. The only exception to this “rule of thumb” is when a font character is used for a company logo or similar image. In these situations, the font characters can be 1” tall by 2” or 3” wide.

For more information, see the subsequent section, “AFP Font Settings.”

19. The **Advanced Normalization Settings** determine the level of processing necessary to display the pages. Select one of the following as the advanced normalization settings:
- **Level 1** — Select this option for simple AFP data files where there is no overlapping data. You will not be able to specify the font height and width.
 - **Level 2** — Select this option for complex AFP data files where there is overlapping data. Level 2 is not as efficient as Level 1 and Level 3.
 - **Level 3** — A combination of Level 1 and Level 2. This is more efficient than Level 2.
 - **Level 4** — Level 4 provides the processing capabilities of Level 1, Level 2, and Level 3. It also provides Coded Font Support and BI tag extraction. Level 4 is more efficient than Level 3. Select Level 4 for all new DDFs based on AFP.

If you select Level 4 settings, the Coded Font Support settings will be displayed.

20. If you want to support coded fonts, enable the **Coded Font Support** check box and specify the Font File path. This can be any directory, but it must contain the following files: *icoded.fnt*, *CPDef.fnt* and *CSDef.fnt*. Also specify the font height controller and the font width controller.
21. Select any of the following special AFP settings:
 - **Show Resource Tags** — Displays image and overlay data in the header.
 - **Show Bar Codes** — Shows bar code tags when displaying the data file in DefTool.
 - **Align by using threshold** — Removes any unnecessary space lines and aligns the data when displaying the data file in DefTool.
 - **Extract No Operation (NOP)** from input file — Extracts No Operation (NOP) data from the data file.
 - **Extract Tag Logical Elements (TLE)** from input file — Any Tag Logical Elements (TLE) in the data file are extracted and presented in DefTool.
 - **Extract BI Tags** — BI overlay tags in the data file are presented in DefTool. This option is available only if you select Level 4 as the Advanced Normalization Settings.
22. Click **Next**. The Configuration Wizard 4 of 4 appears. All the configuration settings that were selected will be displayed along with a preview of the data file.
23. If the page formatting is correct, click **Finish** to open the data file in the Definition Tool; otherwise click the **Back** button to go back and change the configuration information.

AFP Font Settings

The DefTool parser takes the structured field that defines:

- The units of measure for the coordinate system (such as 240 dpi or 300 dpi)
- The page size (such as 8.5 X 11 inches)

It then computes the actual horizontal and vertical coordinate space size. For example, if it uses 240 dpi, it means a coordinate space of 2040 dots in the X direction and 2640 dots in the Y direction.

Source files, for example AFP and Metacode files, usually use a high-resolution coordinate system with many units per horizontal and vertical inch. The DefTool must translate the values to the much lower-resolution coordinate system of a computer screen, with many fewer units per horizontal and vertical inch.

This allows the DefTool to:

- Easily display the data inside one “screen” of information displayed in its window.
- Easily display the data inside a standard browser window.

The DefTool makes some assumptions about how to do the translation from a high-resolution system to a low-resolution system. The Width and Height settings allow you to increase the horizontal and vertical resolution of the grid system used in the DefTool. The default setting is 0,0, and changing these settings will increase the resolution of either the horizontal (Width) or vertical (Height) grid unit system in the DefTool.

The following describes a case where you would need to use character Width and Height information to successfully map AFP data to a text display inside the DefTool.

The X and Y coordinates where a text string starts on the page are extracted from the AFP composed text record, and these coordinates are translated into DefTool coordinates. For example, assume that the X, Y coordinate for the text string “eaDirect” is 120,200. Next, assume that the DefTool automatically scales down these coordinates to a coordinate system that has only 1/10th the number of units in both the X and Y directions. That means those coordinates are translated to 12,20.

This implies that for every 10 X direction coordinate units in the print document, there is only one coordinate position available in the X direction in the DefTool. For every 20 Y direction coordinate units in the print world, there is only one coordinate position available in the Y direction. It also means that it may encounter rounding issues: X direction print coordinates 1 to 10 are mapped to DefTool character position 1, X direction print coordinates 11 to 20 are mapped to DefTool character position 2, etc.

When the DefTool extracts data from the text string, it places each character in sequence into the DefTool display coordinate system, starting at the X coordinate calculated (12, in this case). For example, the DefTool places the 10 characters in the text string “eaDirect” into positions 12 through 21, accommodating the 10 characters in the string.

This system works well until the DefTool coordinate system induces too much “rounding error” into the resolution reduction process. For example, the DefTool often processes print files with extremely small font widths. If the characters in the word “eaDirect” were rendered using printer font characters that were only six dots wide, the total horizontal coordinate space occupied on the printed page would be 10 characters X 6 dots, or 60 total dots.

When you add this to the starting X position of 120, you have the string “eaDirect” occupying printer coordinate positions 120 through 180. The DefTool uses the resolution already defined for the X direction (1 unit or every 10 units of horizontal resolution in the print file), to place the characters out to position 21.

Now assume that the print file contains a text string at position 185,200. That places the text string 5 dots to the right of the end of the string “eaDirect” on the printed page (“eaDirect” ends at position 180). DefTool maps the X coordinate of 185 in the print document by dividing 185 by 10, or 18.5, which it rounds up to position 19. The DefTool will map anything from X position 181 to 190 in the print file to X position 19. However, it already placed a character - the “e” in “eaDirect” into this character position inside the DefTool. This is how character “overlap” can occur in the DefTool.

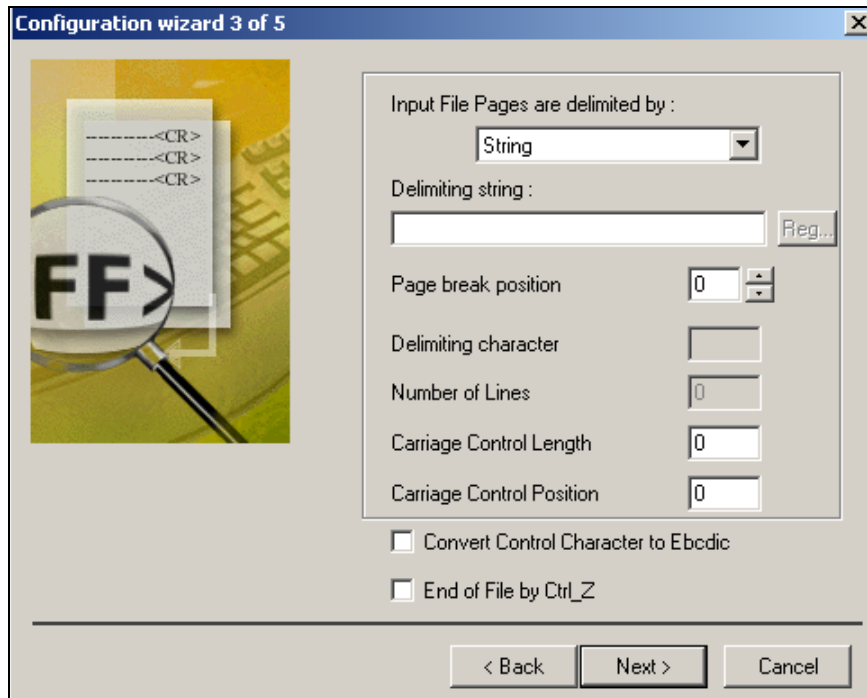
This problem is the result of too great a reduction in the X direction resolution when the DefTool converting from printer X direction coordinates to DefTool X direction coordinates using the current Font Width extraction value. You can adjust this value to increase or decrease the number of X direction printer units mapped to X direction DefTool units.

Creating a DDF for DB Extract Files

To create a new DDF for a DB extract data input file:


1. Select **File>New**. The Configuration Wizard 1 of 5 appears.
2. Specify the project name. The project end cannot be more than 16 characters and should not contain spaces or special characters.
3. Specify the project location or click **Browse** and select a location.
4. The default location is the Samples directory.
5. Specify the data file name and path or click **Browse** and select a path.
6. Select **DBExtract** as the file format.
7. Select the country or region. The country/region is important for comparing currency fields, as the currency format differs for different locales.
8. Click **Next**. The Configuration Wizard 2 of 5 appears.
9. The input file character coding for BD Extract can be either **ASCII** or **EBCDIC**.
10. Specify how the tables are delimited in the input file. Currently the valid options are:
 - **Carriage Return (CR)** — A control character that moves the printer head to the start of the next line, indicating the end of a table.
 - **Line Feed (LF)** — A control character that moves the printer head down one line indicates the end of a table.
 - **CR and / LF** — Table break is identified by both a <CR> and <LF> or either one.
 - **Number of Characters** — The table ends after a fixed number of characters.
 - **Record Split** — Use this option to split large Records. You must specify the number of fields in each line.

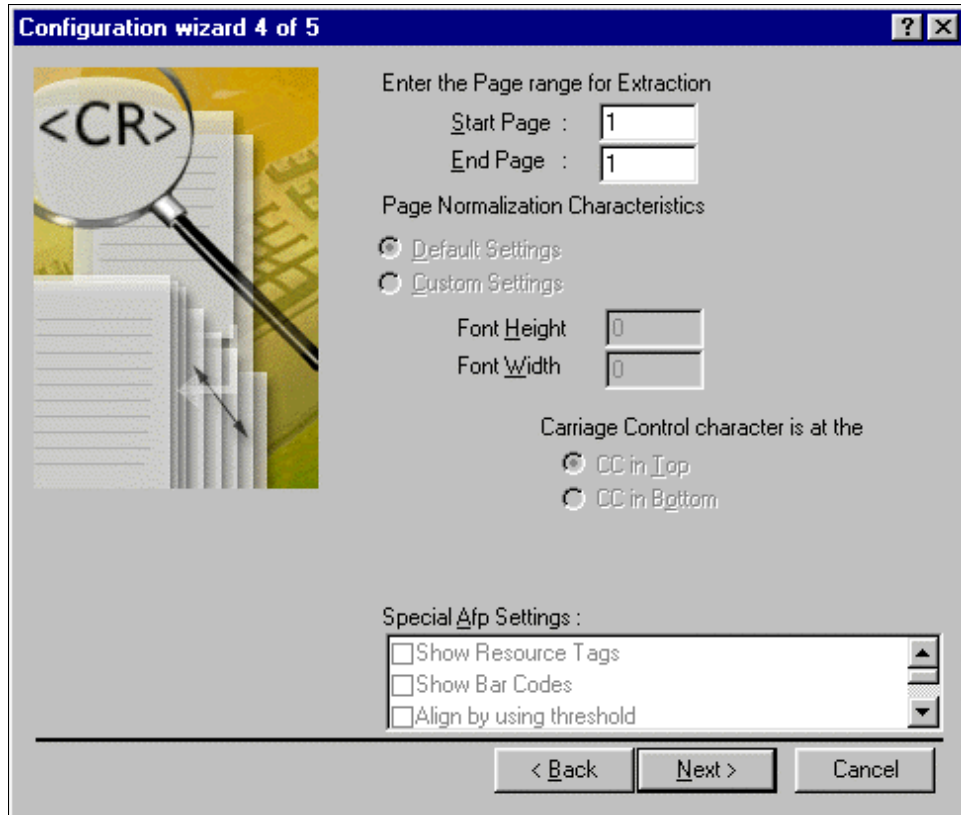
11. Click **Next**. The Configuration Wizard 3 of 5 appears.



12. Specify how the input file page breaks are delimited. The valid options are:

- Form Feed
- Carriage Return by Position
- Carriage Return by Length
- Number of Lines
- Character
- String
- Regular Expression (a changing string of characters that follow a logical pattern)

13. If you selected String as the page break delimiter, specify the exact string in the Delimiting String field. Enter the starting column position for the string in the Page Break Position field (if not 0).
14. If you selected Regular Expression as the page delimiter, click  and use the Regular Expression dialog to build the identifying expression.
15. If you selected Character as the page delimiter, identify the character in the Delimiting character field.
16. If you selected a Number of Lines per page denotes page breaks, specify the number of lines per page in the Number of Lines field.
17. If you selected Character or String as the page delimiter, select the **Convert Control Character to EBCDIC** option if the only way of entering the delimiting character is to specify its ASCII value and then internally convert it to EBCDIC.
18. If necessary, select the **End of File by Ctrl_z** option. The default end of file character is **0xFF**, but some data files can have **Ctrl z** as the end of file control character.
19. Click **Next**. The Configuration Wizard 4 of 5 appears.

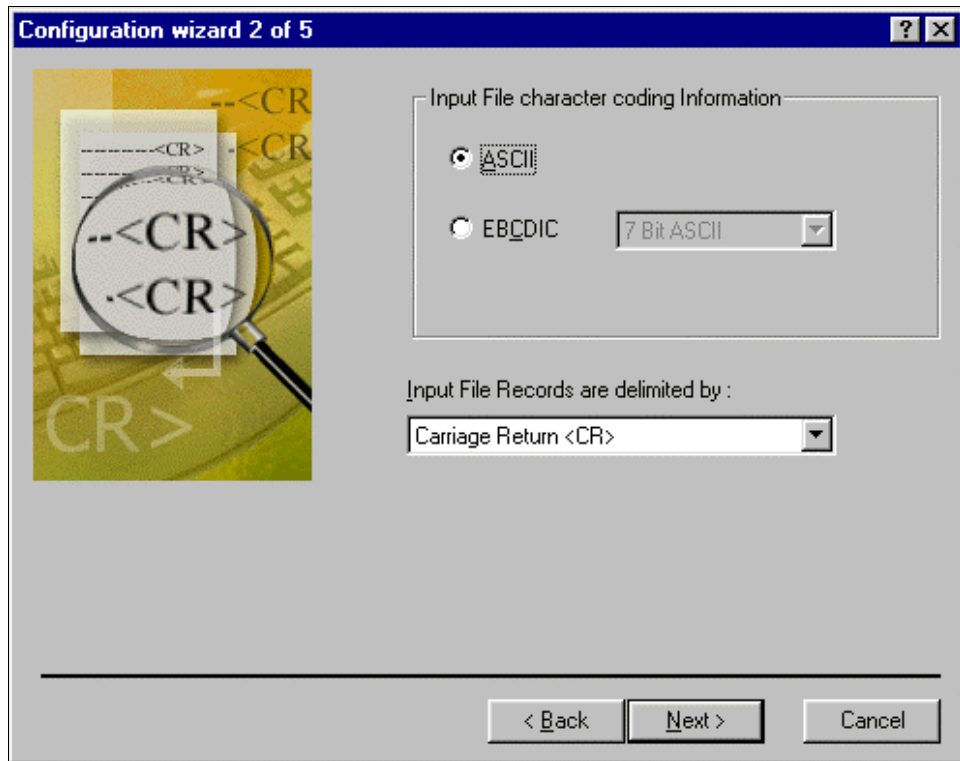


20. Specify the page range for extraction. It is not required that the start page be 1. For example the page range can be 5-100. A large range, however, will impact simulation run times. Make sure that the selected range includes a representative sample of data.
21. Click **Next**. The Configuration Wizard 5 of 5 appears. All the configuration settings that were selected will be displayed along with a preview of the data file.
22. If the page formatting is correct, click **Finish** to open the data file in the Definition Tool; otherwise click the **Back** button to go back and change the configuration information.

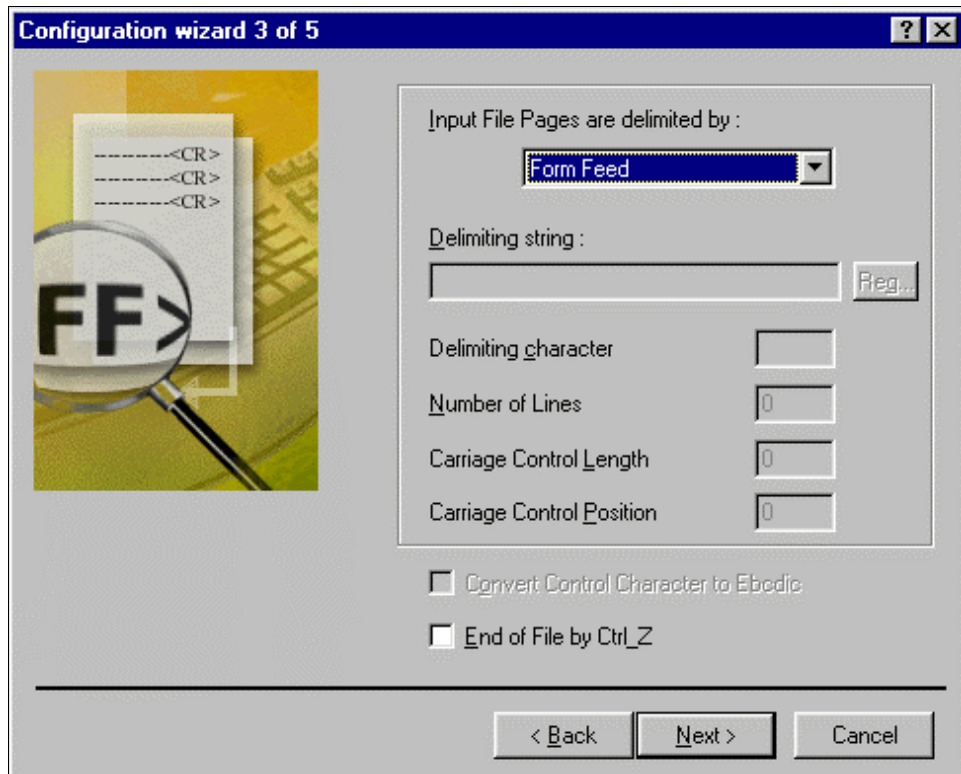
Creating a DDF for Line Printer Format Files

To create a new DDF for a line printer data input file:

1. Select **File>New**. The Configuration Wizard 1 of 5 appears.
2. Specify the project name. The project name cannot be more than 16 characters and should not contain spaces or special characters.
3. Specify the project location or click **Browse** and select a location. The default location is the Samples directory.
4. Specify the data file name and path or click **Browse** and select a path.
5. Select **Line Printer** as the file format.
6. Select the country or region. The country/region is important for comparing currency fields, as the currency format differs for different locales.
7. Click **Next**. The Configuration Wizard 2 of 5 appears.



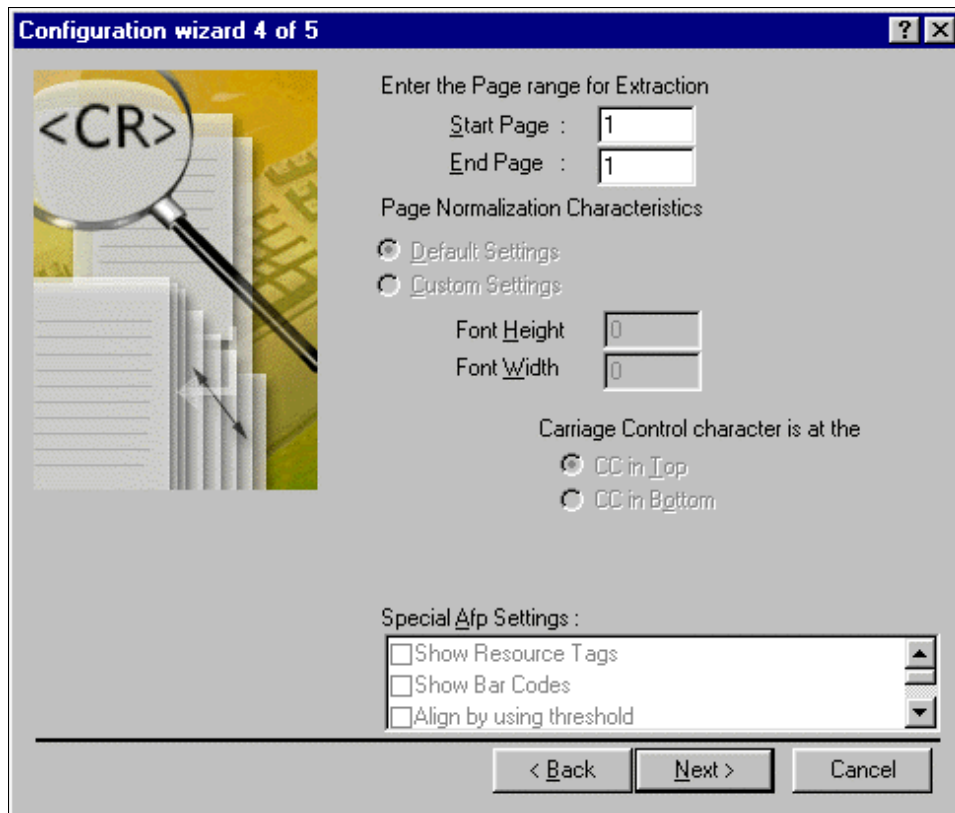
8. The default input file character coding for Line Printer is **ASCII**.
9. Specify how the records are delimited in the input file. Currently the valid options are:
 - **Carriage Return (CR)** — A control character that moves the printer head to the start of the next line, indicating the end of a record.
 - **Line Feed (LF)** — A control character that moves the printer head down one line indicates the end of a record.
 - **CR and / LF** — Record break is identified by both a <CR> and <LF> or either one.
10. Click **Next**. The Configuration Wizard 3 of 5 appears.



11. Specify how the input file pages are delimited. Currently the valid options are:
 - **Form Feed** — This control character directs the printer to advance the paper to the top of the next page. For example, a Form Feed can indicate the end of a page.
 - **Character** — A specific character denotes the Page breaks. You must specify the character.
 - **String** — A specific string identifies the Page breaks. You must specify the exact string.

12. Check the **Convert Control Character to EBCDIC** check box, if the only way of entering the delimiting character is to specify its ASCII value and then internally convert it to EBCDIC. This option is available only if you select **Character** or **String** as the page delimiter.

13. If necessary, check the **End of File by Ctrl_z** check box. The default end of file character is **0xFF**, but some data files can have **Ctrl z** as the end of file control character.
14. Click **Next**. The Configuration Wizard 4 of 5 appears.

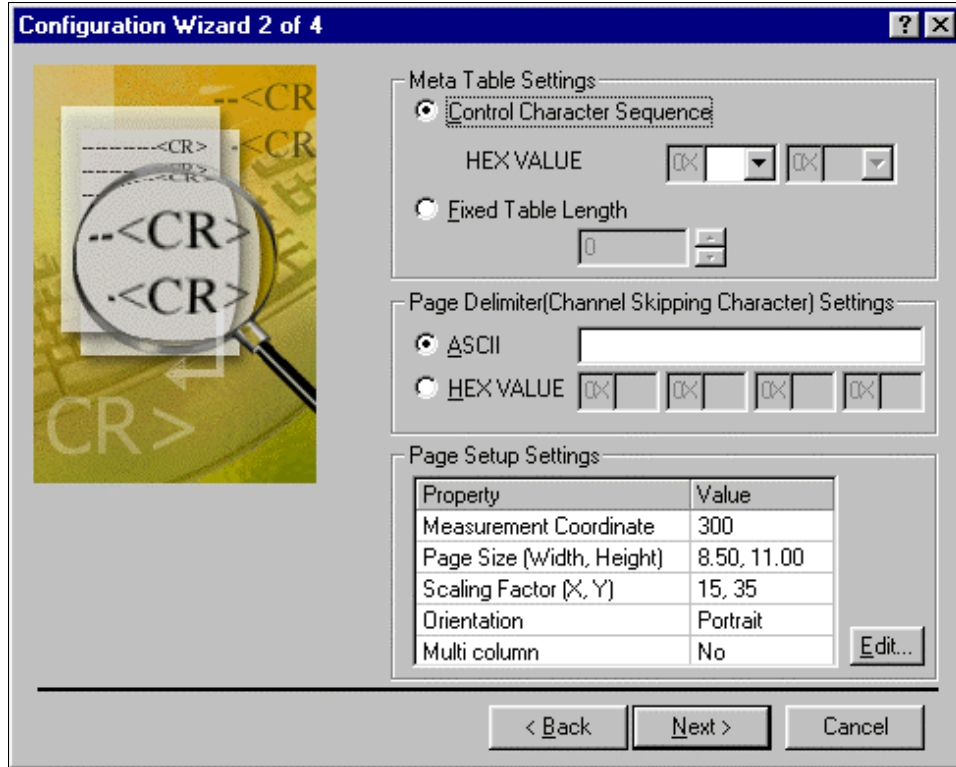


15. Specify the page range for extraction. It is not required that the start page be 1. For example the page range can be 5-100. A large range, however, will impact simulation run times. Make sure that the selected range includes a representative sample of data.
16. Click **Next**. The Configuration Wizard 5 of 5 appears.

Creating a DDF for Metacode Files

To create a new DDF for a Metacode file:

1. Select **File>New**. The Configuration Wizard 1 of 5 appears.
2. Specify the project name. The project end cannot be more than 16 characters and should not contain spaces or special characters.
3. Specify the project location or click **Browse** and select a location. The default location is the *Samples* directory.
4. Specify the data file name and path or click **Browse** and select a path.
5. Select **Metacode** as the file format.
6. Select the country or region. The country/region is important for comparing currency fields, as the currency format differs for different locales.
7. Click **Next**. The Configuration Wizard 2 of 4 appears.



8. Specify how the records are delimited in the input file. Currently the valid options are:
 - **Control Character Sequence** — A sequence of control characters signal the end of a record. This is normally given in hexadecimal. You can select commonly used values from the combo box.
 - **Fixed Record Length** — Records have a fixed number of bytes. You have to specify the number of bytes.
9. Specify how the input file pages are delimited (Channel Skipping Character). Currently the valid options are:
 - **ASCII** — Specific ASCII characters denote the Page breaks. You must specify the characters. The length is limited to four values.

- **Hexadecimal** — Page breaks are identified by specific Hexadecimal values. You have to specify the values. The length is limited to four values.
10. Click **Edit** to specify the page setup settings. The Page Setup dialog box appears.
 11. Specify the **measurement coordinates**. This can either be 300 dpi or 240 dpi. The default is 300 dpi.
 12. Specify the **page size**. Currently the valid options are:
 - **Letter** — 8.5 x 11.0 inches
 - **A4** — 8.27 x 11.69 inches
 - **Custom** — You have to specify the page width and height.
 13. Specify the **X and Y scaling factors**. This identifies the scaling factor used to display the file in DefTool. X specifies the number of pixels per column and Y specifies the number of pixels per row.
 14. Select **Portrait**, **Landscape** or **Portrait & Landscape** as the orientation. Some Metacode files have data in both Portrait and Landscape format, so you will have to select Portrait & Landscape for these files.
 15. If necessary, enable the **Multi column** check box, to allow multiple columns of data to be displayed in DefTool. The Right Column Properties section will be enabled in the Page Setup dialog box. These settings depend on the page orientation you select. In the screen shot shown below, Portrait and Landscape are selected as the page orientation. As a result the **Right Column Properties** for both are enabled. If only one orientation is selected, then only the relevant section will be enabled.
 16. Specify the column break position. This indicates the end of the first column to the left. Normally this would be the first blank space after the first column.
 17. Specify how to shift the column. i.e. Shift right or shift down. Also specify the number of columns by which to shift the next column. If you select Shift Down and do not specify the number of columns to shift by, zero will be taken as the starting column.



If you need to go back to the default page settings, click **Default**. The default settings are: Measurement coordinate –300 dpi, Page Size - Letter (8.5*11.0 inches), Scaling Factor - X = 15 and Y = 35, Orientation - Portrait.

18. Click **OK** to return to Configuration Wizard 2 of 4. The specified page setup settings will be displayed.
19. Click **Next**. The Configuration Wizard 3 of 4 appears.
20. Specify the **page range** for extraction. It is not required that the start page be 1. For example the page range can be 5-100. A large range, however, will impact simulation run times. Make sure that the selected range includes a representative sample of data.
21. Click **Next**. The Configuration Wizard 4 of 4 appears. All the configuration settings that were selected will be displayed along with a preview of the data file.
22. If the page formatting is correct, click **Finish** to open the data file in the Definition Tool; otherwise click the **Back** button to go back and change the configuration information.

Metacode Settings

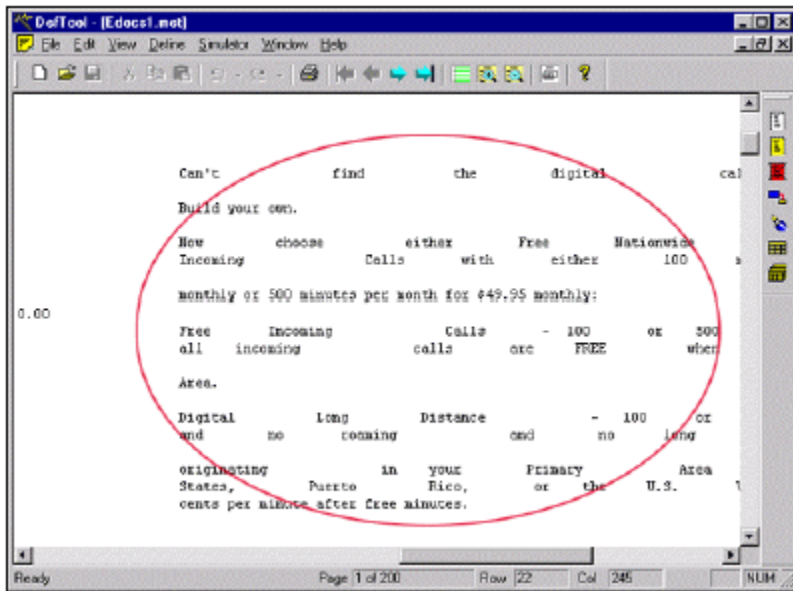
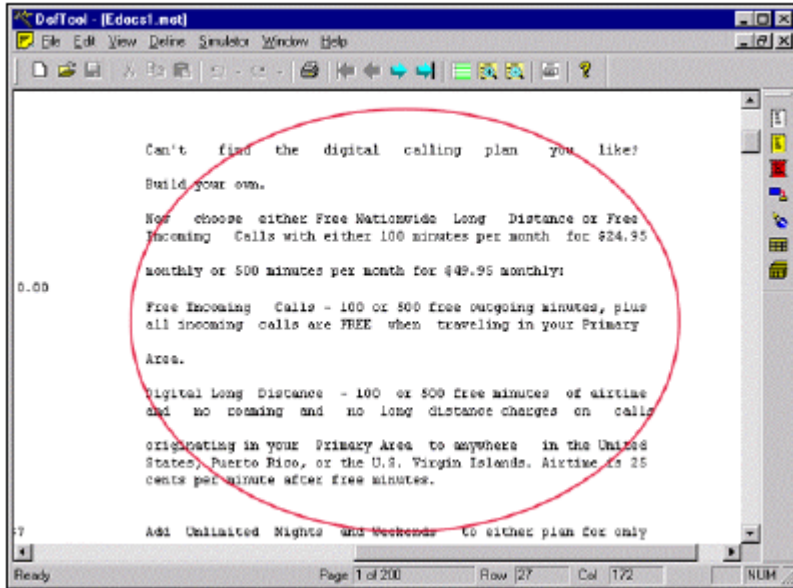
X and Y Scaling Factors

Print files specify information in pixels, but the DefTool requires row and column positions.

X Scaling factor — The number of pixels per column or the average width of each character. The default X scaling factor is 15 pixels. Decreasing the X scaling factor will introduce unnecessary spaces between the words.

Increasing the X scaling factor will reduce the space. The images below show how the X scaling factor changes how the DefTool displays data. In the first image, the X scaling factor is 17, while in the second image the X scaling factor is eight, which introduces more spaces between words.

Defining Data Extraction Rules with DefTool



Y Scaling Factor — The number of pixels per row or the average height of each character. The default Y scaling factor is 35 pixels. Decreasing the Y scaling factor will introduce unnecessary spaces between two rows.

Column Break Position

The column break position gives the start of the second column. (An approximate column value is sufficient.) Before setting this value, open the file in DefTool and make sure that the page appearance is similar to the printed-paper document. Once you get the correct output in DefTool, you must then define rules for extracting your data.

In the data file shown in the images below, there was data in both Landscape and Portrait orientations, the X scaling factor was 17 and the Y scaling factor was 35. With the selected scaling factors in Portrait orientation, some of the data that should be in column 1 is displayed in column 2.

In order to display the data correctly, the second column must shift to the right. Here it has been shifted 20 positions to the right.



You can also shift the second column down, in which case the second column will appear below the first column. If you shift the second column down, then you have to specify the starting position. The default start column position is zero.

Writing Data Extraction Rules

An extraction rule is what you write when you define a document style, primary key, additional page style, field, table, or group in DefTool.

Each of these rules identifies a unique, individual entity on a statement in the data input file.

Some extraction rules, such as field extraction rules, tell eaDirect where to search on a statement page and how to find data to extract to display it in an online statement or to use it with conditional business logic. We write other rules, such as the document style, for the purpose of telling eaDirect how to identify new statements within the data input file.

You must create well-defined rules that can successfully identify and extract these values. This requires an understanding of the data and all its possible forms, including its unique properties, so that you create a data pattern to use for extraction.

This section describes the general process of creating extraction rules.

Identifying Unique Properties of Data to be Extracted

For each statement element you want to extract, you must identify:

- The data's unique **location** in the DefTool search area.
- The unique **content** properties or characteristics of the data values that can be used in rules definition.

Search Area (Location)

Where does the data appear within the data source? As you graphically select your values in the work area to begin creating the rules to extract them, you need an understanding of where the data can appear on all the different types of statements that exist within each data source. You also need an understanding of the variability within each statement type. The window coordinates that you define (starting with the process of highlighting an area with the mouse, and refined within the Properties and Information dialogs) identify where the production components should look for the start of the data string.

An area, defined via the “grid” that has been applied to each statement page, of columns and rows that indicates where the **START** of the data string should be searched. This area is called a window coordinate set. The window coordinate set that you define should be as small as possible, defining the minimum search area necessary to locate the first position of the data string specified. This is because the eaDirect extraction components actually conduct the search for data from left to right, top to bottom.

It is often necessary to create window coordinate sets that span many rows on a page, for example. If you define a data string to be searched for somewhere between row 3 and row 100, then the extraction components will look on every row from row 3 until the correct data string is found. Assume, then, that the data string will always be found between column 40 and column 41. This means that your window coordinate set should not be any wider than that one expected column set. Expanding the set even by one column, i.e. to column 42, increases the searching area two-fold. Excessively large window coordinate sets will slow down the performance of live retrieval during deployment.

Example

A customer uses several of your services. Due to the detail in the statement, the Federal and State Taxes will appear near the bottom of the page (illustration below).

DR BARBARA WHITEHOUSE	MARCH 25, 2001
ACCOUNT NO: 4191463	
ACCOUNT SUMMARY	
	PAGE 1
CUSTOMER SERVICE:	
1-508-652-8600	1-877-336-3362
PREVIOUS BALANCE.....	392.75
LESS PAYMENTS APPLIED THROUGH 03/24/01.....	392.75CR
THANK YOU FOR YOUR PAYMENT	
BEGINNING BALANCE	\$.00
CURRENT USAGE.....	17.61
RECURRING FEES:	
800 NUMBER FEE.....	**WAIVED**
SURCHARGES:	
FEDERAL INTERSTATE/INT'L UNIVERSAL SVC FUND CHARGE.....	0.18
LIFELINE ASST/TELE RELAY - BUS.....	2.80
FEDERAL ACCESS CHARGE - BUS.....	30.17
OTHER SERVICES:	
LOCAL USAGE CHARGE.....	34.07
LOCAL SERVICE CHARGE.....	224.52
FEDERAL TAXES - LOCAL SERVICE.....	7.77
STATE TAXES - LOCAL SERVICE.....	7.77
TELECOM RELAY SURCHARGE.....	0.35
TAXES:	
FEDERAL TAX.....	0.45
STATE TAX.....	0.96
TOTAL CURRENT AMOUNT	\$ 344.13
	PLEASE PAY THIS AMOUNT \$ 344.13
	BY 04/19/01

Another customer subscribes to one of your services, or infrequently uses your services, requiring lesser detail in the statement. In this statement, the Federal and State Taxes appear in the upper third of the of the page, as shown below.

K J GREEN		MARCH 25, 2001	
ACCOUNT NO: 9001203			
	ACCOUNT SUMMARY		
		PAGE	1
	CUSTOMER SERVICE:		
	1-508-652-8600	1-877-336-3362	
BEGINNING BALANCE		\$	0.00
CURRENT USAGE.....			0.00
TAXES:			
FEDERAL TAX.....			0.00
STATE TAX.....			0.00
TOTAL CURRENT AMOUNT		\$	0.00
	NO AMT DUE	\$	0.00

The value for taxes starts in the same column on all the bills. To successfully extract the taxes on these two types of statements, the Taxes field window needs to span all rows from the upper third to the bottom of the page, yet only needs to span one column. This is indicated by the blue field indicator for the Taxes field.

Data Patterns (Content Search)

eaDirect only supports extracting data as a string. For each piece of data you want to extract, you must identify a data pattern that eaDirect can use to match any and all occurrences of the data string. Depending on the possible data occurrences, a data pattern can be

- Static data string (such as **ACCOUNT NO:**), or
- A variable regular expression (more common)

With a regular expression, you define, for each position of data, what characters are valid for extraction. You may want to extract only data strings that consist of 3 numeric values. In this case, you would write a regular expression that requires each of the 3 positions of data to be numbers, i.e. `[0-9][0-9][0-9]`.

What does the data look like? The knowledge of what characters are possible in each position of the data string, as well as how many characters make up any given string, will enable you to define precise rules that will ultimately extract only the data you want, as you want it to appear.

Example

On Biller A's statements, the Account Number is a 14-digit numeric string containing a dash after every four digits. Thus, it appears in the data source as `####-####-####`.

The Phone Number is a 10-digit numeric string with dashes that separate the area code, exchange and individual number, respectively. Thus, it appears in the data file as `###-###-####`.

The data on these statements floats slightly from statement to statement and are located near each other. Thus, there is a chance that the window coordinate sets that you define to extract these values will overlap.

To successfully extract these two values without confusion as to what value belongs to what field, you need to specify tightly defined extraction rules that detect the differences in pattern and length.

Field Properties

Fields are data strings that

- Occur on one row only, and
- Do not iterate within a statement; (they have only one value per statement).

When defining the rules to extract data fields, you must specify the following field properties:

Data Type — Data can be classified as a String, Currency or Numeric type. You select the type that is appropriate to the data, keeping in mind your plans for this field during the indexing process. Perhaps you may want to index a field called TotalAmountDue to the application database, for research or marketing reasons. It would make sense to classify TotalAmountDue as a Currency value, so as to ensure that the corresponding database field would also be defined as Currency.

Data type has an effect on field use in Business Logic. You may want to use the TotalAmountDue field in a Business Logic rule that offers your customers a different set of marketing promotions, based on the amount of the current statement. By classifying the TotalAmountDue Data Type as **CURRENCY**, you can be sure of accurate calculations of comparisons using Business Logic.

For example, a TotalAmountDue of \$169.48 can be compared to \$1000.00. Since the TotalAmountDue is less than \$1000, the customer would receive a statement with a less ambitious set of promotions.

Field Type — Specify the field type when creating your primary key only. You use this definition only once, on the first field you create.

Formatting Specifications — For applying post-extraction data conversions to the data. You can apply:

- Alphanumeric conversions, such as ‘Delete all trailing zeros or spaces’
- Date conversions, such as ‘Translate Aug 21, 2002 to 08/21/2002’

Force Field Length — You may want to extract a field by validating a select few characters of the string found, and then extracting all other characters to a specified number. This is useful in cases where the pattern of the field can vary widely, such as in a customer name or street address Line. In these cases, the regular expression that you would have to define as the valid pattern might be so broad that it could result in an impact to extraction performance. You could specify that the first character must be alphanumeric (i.e., *[A-Za-z0-9]*), but to extract the subsequent characters until a maximum number of 30 characters have been extracted. This is commensurate to enabling the Force Fixed Length option, and setting the length to 30.

**Tip**

All indexing fields will normally have the Force Fixed Length attribute engaged.

Marker Properties

For each table and group you want to extract from your data input file, you must create a pair of start and end markers that eaDirect uses to locate the start and end of the table or group.

Define the start marker on a data string (variable or static) that always appears on or above the table or group occurrences. Similarly, you must define an end marker on a data string that always appears on or below the table or group occurrences.

When defining the rules to create markers, you must specify the following properties:

Special Naming — The name you choose for your start and end marker must be the same. This identifies them as a marker pair. Also give the table or group associated with the marker the same name as the marker pair; this is a ‘best practice’ for organizational purposes.

Include Marker Line — Start and end markers can exist on the same row where there may be table or group data (including markers for the tables of a group). If this is the case, you check on the Include Marker Line option so that these rows are also searched for column data, or group information (including the group’s table markers and columns).

Use Blank Line as Marker — Sometimes, the best indicator of the start or end of a table or group is not a data string that appears with the table or group instances, but is rather a row in the statement containing no data. In this case, you can check on the Use Blank Line as Marker option.

Select the Line Only Containing This Pattern — If you are sure that the data string you are using as your marker is the only data on its line, you should enable this option. This requires that NO other characters appear between the rows that contain the marker string.

Look for the Start Marker on the Same Row — This option is useful when your start and end marker patterns are the same and expected to be found in the same location. Say, for example, that your start marker is string X in location Y, and your end marker is string X in the same location Y, yet is expected on a subsequent page. You would then enable this option in order to force the extraction components to search beyond the first occurrence of string X, thus avoiding the creation of a null search area for the data.

Use Dynamic Pattern — This option will allow you to insert a dynamic field in the marker's pattern. When this option is enabled, you will be given a dropdown list to select the dynamic field for the pattern, or part of the pattern. The use of dynamic fields as patterns is discussed in greater detail in section

Table Properties

Tables consist of one or more related data strings that you want to extract and present together. These data strings may have multiple occurrences. The data strings that you extract as part of a table are defined as columns.

Table Anchors

Every table you create must also contain an anchor definition. The table anchor is a data string that occurs with every occurrence of the table columns. If the table anchor pattern is not found where it is looked for on a row between the start and end marker, then this row becomes invalid for the search of the table columns. Essentially, the table anchor occurrences **MUST** appear on every line that there could be table data. The two options for your table anchor are:

- **Internal Anchor** — This is a table column that you expect to always be populated with data. For example, in a call itemization list, the charge column would be populated with data for every List Item. Therefore, you would select this column to be the anchor for itself and the remaining columns. By default, the table uses the first defined column as the Internal anchor.

- **External Anchor** — If you are not certain that one of your table columns will always be populated with data, then it may be necessary to define a data string that is external to the table itself as the anchor. A good example of external anchor data would be control codes that may exist with on the periphery of every table line. You can only define an external anchor if the appropriate data exists on every row of table column data.

Table Columns


You define the rules to extract table columns similarly to the same way you define a field. A major difference is in the window coordinate set that you define. When you are creating columns of a table, it is important to keep the following points in mind:

- Row coordinates of columns should only span one row. The rows that will ultimately be searched for column data will be defined dynamically by the detection of the table start and end markers during extraction. Therefore, you **do not** need to anticipate the row span of your columns when you are creating their rules. You only need to select one row (i.e. start row = 10 and end row = 11) when creating your window coordinate set.
- In most cases, choose the same row span for all columns in a table. More specifically, the row span that you choose for your columns must correspond to the visual pattern in which the column occurrences appear. For example, related column occurrences of a Call Itemization List would be the Date, Time, Location, Phone Line Called, Number of Minutes and Charge for a particular call. You would expect these data to appear on the same row, therefore you must define the columns to extract these data on the same row.
- By default, the order in which you create table columns becomes the order in which they appear to the user online. You can move your columns left or right in any order.

Creating an Extraction Rule (General Procedure)

To create an extraction rule:



1. Click on the appropriate tool icon  for the type of extraction rule you want to create.
2. In the DefTool work area, locate the data string you are building your rule upon, and while left-clicking your mouse, drag and drop your pointer over the start of the data string, and release the mouse-click. A ‘marquis-style’ black box appears over this selection in the work area. This is the basis for your window coordinate set, the properties of which you may edit later.
3. A Properties or Information dialog appears over DefTool for the particular type of extraction rule you are creating.
4. Name your extraction rule (if applicable). Keep names short but descriptive, as in TotAmtDue. Field, table, group, and marker names must start with a letter, use alphanumeric characters only (no special characters), and cannot contain spaces. Do not use the following reserved words in a name: Time, Date, INT, Double, Currency (“Date” is not permitted but “DueDate: is). Field, table and group names (but not marker names) must be unique. Maximum length is 30 characters. Names are not case-sensitive.
5. Specify the data pattern to search for.
6. Create additional properties for your rule, if applicable (formatting specifications, marker options, etc.)
7. Test your rule by clicking the **Test** button (this performs a quick check on the current statement page of your DefTool work area by highlighting in purple what could be found with your rule settings).
8. Save your rule by clicking **OK** to close the Properties or Information dialog. DefTool color-highlights the area indicated by your window coordinate set. The color indicates the type of rule you created:

Red - Document style and additional page styles

Blue - Fields

Bright Green - Field anchors, page style anchors and marker anchors

Yellow - Start and end markers

Aqua - Table columns

Document Style (Start of a New Document)

Every DDF requires a document style definition, indicating where each customer document starts in the data source. You must create the document style definition **before** creating a primary key or other defining other elements in the DDF.

You define the document style by specifying a name and unique pattern that the eaDirect Indexer production job can use to identify, upon locating this unique pattern, that the information that follows is the first page of a new statement.

Indexer considers all the pages from the start of document until the next occurrence of the pattern to be part of one statement. You can define document style directly or after defining a document style anchor.

In the following example of a data source, each section, denoted by a line, represents a different physical page of data:

<p>11BILLS BICYCLES FD2G6-6DF 2V0 0WG2XDF2GV3D+3 D20 OG 0D2S942F3DS,B3D63T63GD 6E3V3E3R4962G-6E*9E*5R6TG32ED</p>
<p>13BILLS BICYCLES WE WRKLEWKF WERMW FRT03S23RE2 E46F9512TG4Y64*58F12E63596*4* 342R*F*-64R559RT*WE65E5595D5E</p>
<p>13BILLS BICYCLES 2R8550G54T2F6E69R59F1C 6F*6R R92612R93E63RF2RT5493*35*R-33 -5,2G6E959T4E5498R58F1DR5R944</p>
<p>11BOBBYS BUILDING SERVICES 8JFKJRO9T59KDR45R9IFJDKRO4R02 E3D04EOTRKGKER00H;WS6R-R23F3E 6*6F6RT9WR-R-T692R36G52TR695R</p>
<p>13BOBBYS BUILDING SERVICES S2R6R3F3E21TR6253R6T6ER62E1RT R925+GED6TR*GFR256G*R3F+D+66R 4*TD3R-F650R2FR-RF2F3R3R-D+0</p>
<p>11COOKS GARDEN SHDP S2RR6R*F3TRF2T6G6T2T2Y-TYY-66 6R5*56Y20F0NTT52F2F1FR52T52T6 -R3F-R6R2F0F0FD3SER-R6R2EDE2R</p>
<p>13COOKS GARDEN SHDP S2E5R22D2R5RT1GV1D2R6TR6F2RD2 WS52E6F2RT0V2DE63R52R1E0D32FT D3R6T63G3T-*596365RF2F2F6T52T</p>

The following image shows how DefTool uses a document style defined as “11” to identify the beginning of a new statement:


Start of Statement 1 →	11BILLS BICYCLES FD2G6-6DF 2V0 0VG2XDF2GV3D+3 D20 0G 0D2S942F3DS,B3D63T63GD 6E3V3E3R4962G-6E*9E*5R6TG32ED
	13BILLS BICYCLES WE WRKLEWKF WERMW FRT03S23RE2 E46F9512TG4Y64*58F12E63596*4* 342R*F*-64R559RT*WE65E5595D5E
	13BILLS BICYCLES 2R8550GS4T2F6E69R59F1C .6F*6R R92612R93E63RF2RT5493*35*R-33 -5.2G6E959T4E5498R58F1DR5R944
Start of Statement 2 →	11BOBBYS BUILDING SERVICES 8JFKJRO9T59KDR45R9IFJDKRO4R02 E3D04EOTRKGKER00H;WS6R-R23F3E 6*6F6RT9WR-R-T692R36G52TR695R
	13BOBBYS BUILDING SERVICES S2R6R3F3E21TR6253R6T6ER62E1RT R925+GED6TR*GFR256G*R3F+D+66R 4*TR6R-F65OR2FR-RF2F2R3R-D+0
Start of Statement 3 →	S2RR6R*F3TRF2T6G6T2T2Y-TYY-66 6R5*56Y20F0NTT52F2F1FR52T52T6 -R3F-R6R2F0F0FD3SER-R6R2EDE2R
	13COOKS GARDEN SHOP S2E5R22D2R5RT1GV1D2R6TR6F2RD2 W552E6F2RT0V2DE63R52R1E0D32FT D3R6T63G3T-*596365RF2F2F6T52T



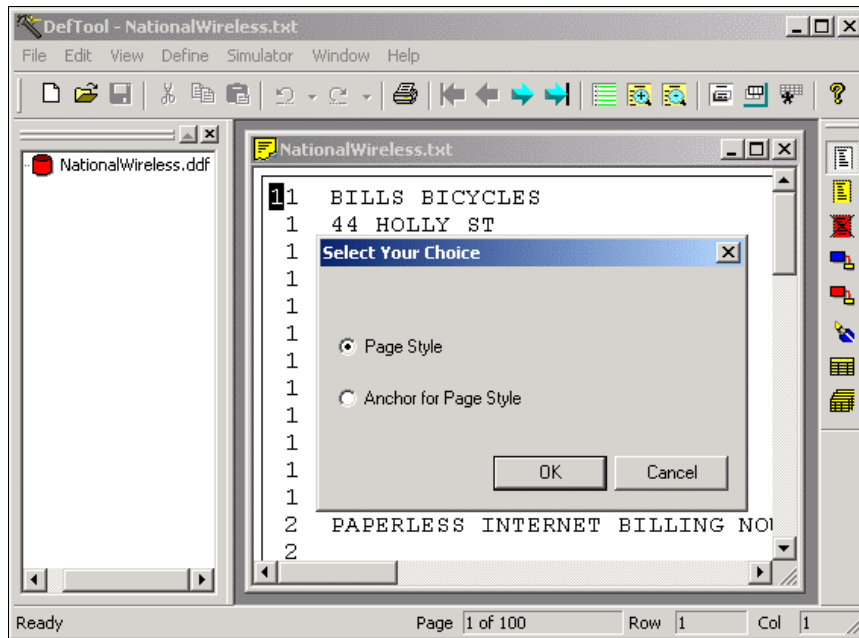
Anchors provide a second level of guarantee that the information you have specified is exact. If an anchor is defined, eaDirect looks on each page of the file for the anchor pattern. If eaDirect finds the anchor, it searches for the document style. If it does not find the anchor, the DefTool does not search for the document style on that page, and therefore does not consider that page to be part of a new document.

You can create a document style anchor with the same tool as for the actual document style, however anchors for document style should not be used unless absolutely necessary.

To define the document style:

1. Select the Document Style  icon .

2. In the DefTool's Work Area, locate the data string that will identify the start of statement, and then click and drag your mouse over the first position of the unique pattern that signifies the start of document, then release your mouse. (Select the first 1 in 11.) The Select Your Choice dialog opens, presenting two options: Page Style or Anchor for Page Style.
3. Accept the default of **Page Style**, since you can be sure that 11 will not appear in this location on subsequent statement pages. The Page Style Information dialog opens.



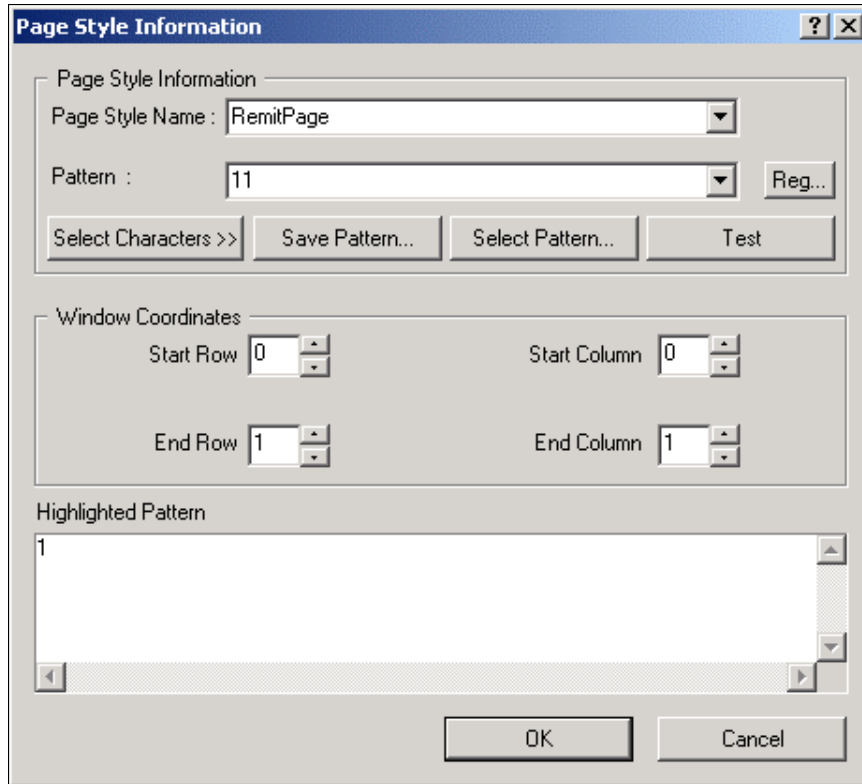
Tip

The window area you select will define the area that is searched for the **start of the data string** on all statement pages across the data source. You may need to adjust the resulting window coordinates if you find that this data string's start location can vary. You do not need to set your window coordinates to encapsulate the entire string.

Again, the window coordinates define for the extraction components where to search for the start (or position #1) of the data string you are trying to find. The number 11 is selected as the pattern for the document style because this pattern of characters appears on the first page of all statements in the data source. Thus, it can be used as a unique identifier for the first page of a document.

You would create an anchor for a page style when the data string you are using to identify your new page style may also appear on other page styles. You could, in this case, locate an additional data string that 1) appears with (and only with) the page style string and 2) appears in the same relative location as the page style string. There can be more than one document style pages in a statement. The location of a document style AND a primary key identifies the page as the first page of the statement.

Say, for example, that the string 11 could also appear on subsequent statement pages, but that on the 'first' pages, there also existed a string called REMIT TO: . You could create a page style anchor that first searched for REMIT TO: and, if found, searched for 11. It wouldn't matter if the string REMIT TO: appears on subsequent pages. It is the joint appearance of REMIT TO: and 11 that verifies the document (or first) page style.



4. Enter your page style end. Names must not contain spaces or special characters, and must start with an alpha character. The names must not exceed 30 characters in length. (Enter RemitPage.)
5. Enter the pattern for searching. (Enter 11.)

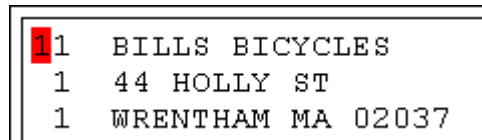


If you have selected the entire identifying data string in the work area, you can copy (Ctrl-C) and paste (Ctrl-V) the pattern in the Highlighted Pattern window.

6. If necessary, adjust the window coordinates (start and end rows; start and end columns) so that your coordinate set is sufficiently large to capture the start of all occurrences of your page style string, across all statements. Remember: the whole string DOES NOT need to be in the window. (The window coordinate set for the RemitPage properties should be as follows:)

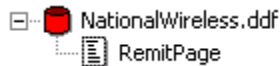
Start Row = 0 Start Column = 0
End Row = 1 End Column = 1

(If the first 1 in 11 is highlighted as shown below, your coordinates will match those above.)



Data sources often experience “float,” where data shifts position across the statements. Sometimes, this is as little as one character left or right. In other cases, data can span many rows up or down. You will need to familiarize yourself with your data source and take any “float” up/down or left/right into account when defining your window coordinates. To do this, you may have to navigate through several statements to determine what window coordinates will suffice to locate the start of your page style string. Since the DefTool main console is still active when the Page Style dialog is open, you can easily navigate to other ‘first’ pages with the blue arrow tools. When you need to adjust your window coordinates, you can use the up/ down arrows for each corresponding coordinate field, in the Properties’ Window Coordinates section. The live DefTool work area will be immediately updated with your new coordinates, by auto adjusting the original graphic selection that you made.

7. Click **OK**. The Page Style Information dialog closes, and the area you defined as the window coordinate set for the page style appears in red in the DefTool work area (shown below). A white document icon now appears in the Application Tree, representing your document style definition. (A white document icon named **RemitPage** appears in the Tree View, under the DDF parent tree node as shown below.)



**Caution**

The red highlighted area visually indicates the window coordinate set in which the DefTool will look for the start of the pattern that is identified. In the example above, the data string 11, the defined pattern, can only start in one position, between rows 0-1 and columns 0-1. If the string appeared between other rows or columns, the extraction components would not be able to locate it. Ensure that the window coordinate set is defined on a sufficient number of rows and columns to locate the desired string.

Document Recipient (Primary Key)

After you create the document style, you must define a primary key for your DDF. The primary key defines how to uniquely identify each statement contained in the data source, typically each recipient across a statement cycle.

The primary key field must be unique and reside on the document style page on every statement.

**Caution**

You must define the primary key AFTER defining the document style. You must also create the primary key before creating the rest of the rules or saving your new DDF.


The account number, if the statements describe only one account, may be a good choice for the primary key field. The customer name would be a poor choice, as it is possible for two people to share the same name.

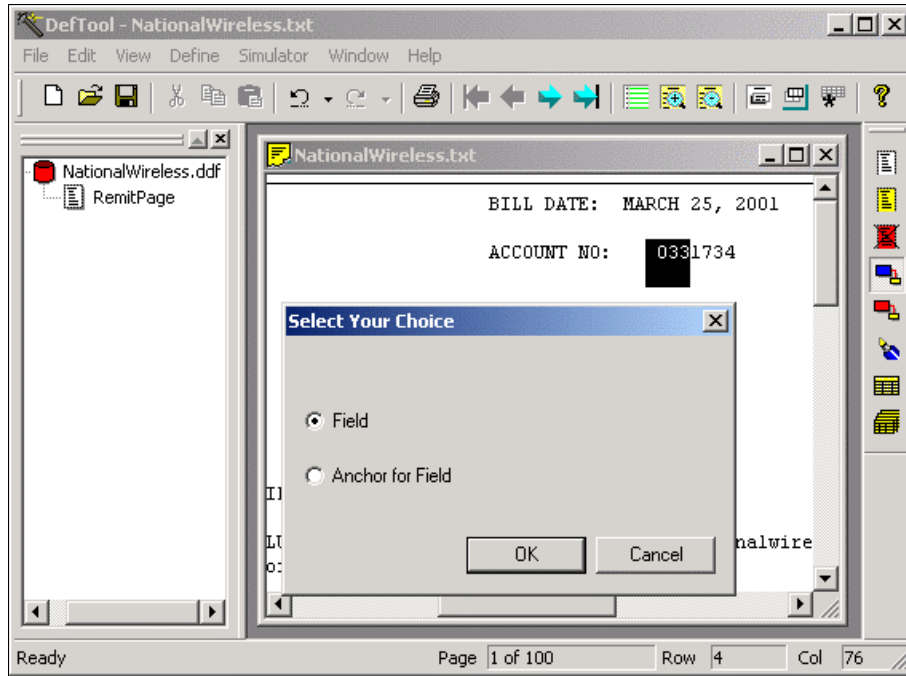
The primary key field is best defined directly, without an anchor. It is technically possible to use a field anchor with the primary key field, but is not recommended, given the importance of finding an occurrence of this field on every statement.

In the following example, the red selections indicate the data string used to define the document style. The blue selections show where the primary key field is defined. Here, the 7-digit account number is the primary key:

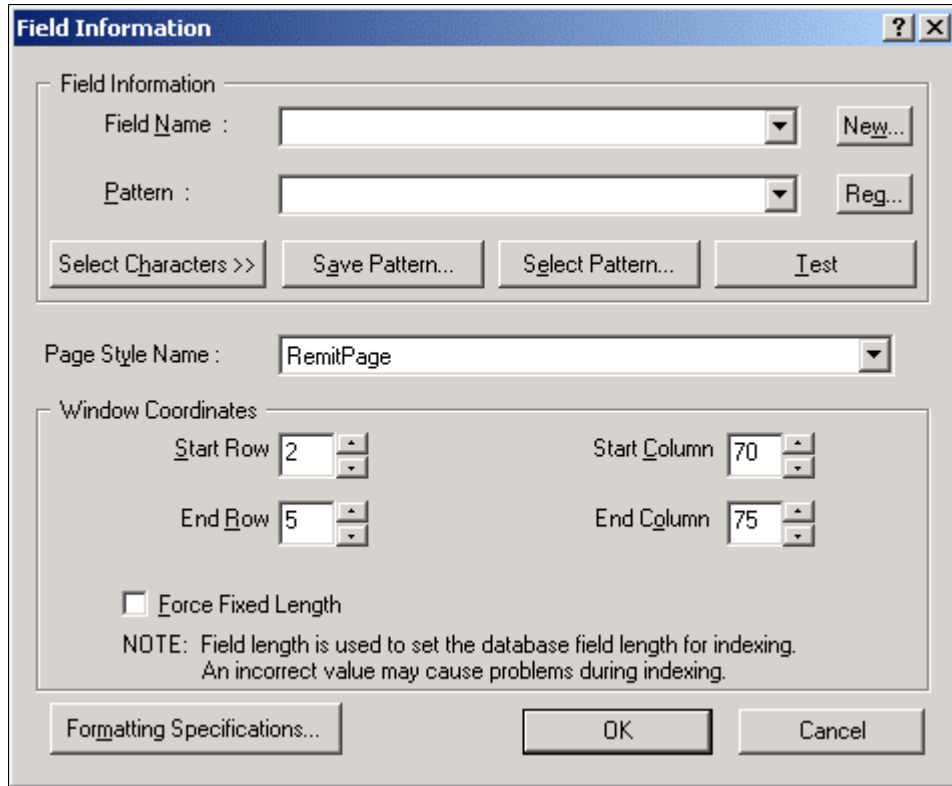
<p>New statement →</p> <p>New statement →</p> <p>New statement →</p>	<table border="1"> <tr> <td>11</td> <td>BILLS BICYCLES</td> <td>0293822</td> </tr> <tr> <td colspan="3">G2G2GH1T5HG32NOGH23Y5DF2GH-YH3TTY H02HY6HH22H2H6YH63HTGD SF5 TYH3Y66H 6TY2HY655T165TYH956Y21HERY3-Y63YH</td> </tr> <tr> <td>13</td> <td>BILLS BICYCLES</td> <td>0293822</td> </tr> <tr> <td colspan="3">WE66-H-6Y6-H2TY+B-TY*F32DH-6-YH2N E66E6G2E6R26G*76H3HB3Y6-FG3Y6H2G1 69G5T96Y9R5FGHHY6Y-T-G6YH5T2FF3R</td> </tr> <tr> <td>11</td> <td>B0BBYS BUILDING SERVICE</td> <td>0279518</td> </tr> <tr> <td colspan="3">G2G2GH1T5HG32NOGH23Y5DF2GH-YH3TTY 6TY2HY655T165TYH956Y21HERY3-Y63YH E66E6G2E6R26G*76H3HB3Y6-FG3Y6H2G1</td> </tr> <tr> <td colspan="3">13B0BBYS BUILDING SERVICE 0279518 H3Y4612H3Y6Y-HN3Y6Y6Y33H32Y6H63HY 5*T6G3H56H633G-Y62 OBG63G2H2G3G63 HG636H3B,G2G2T6YH6HG3G3B,,H-YGGF2</td> </tr> <tr> <td>11</td> <td>COOKS GARDEN SHOP</td> <td>4349043</td> </tr> <tr> <td colspan="3">DJCUR84UJRJ48RIR58RFUI49FKRI48UJ3 KFI58TUJJDH295JFU59060GMKVGJG75940 F050E4KFJT959RKIFKG096040RF0KVKTO</td> </tr> <tr> <td colspan="3">13COOKS GARDEN SHOP 4349043 H3Y4612H3Y6Y-HN3Y6Y6Y33H32Y6H63HY 5*T6G3H56H633G-Y62 OBG63G2H2G3G63 HG636H3B,G2G2T6YH6HG3G3B,,H-YGGF2</td> </tr> </table>	11	BILLS BICYCLES	0293822	G2G2GH1T5HG32NOGH23Y5DF2GH-YH3TTY H02HY6HH22H2H6YH63HTGD SF5 TYH3Y66H 6TY2HY655T165TYH956Y21HERY3-Y63YH			13	BILLS BICYCLES	0293822	WE66-H-6Y6-H2TY+B-TY*F32DH-6-YH2N E66E6G2E6R26G*76H3HB3Y6-FG3Y6H2G1 69G5T96Y9R5FGHHY6Y-T-G6YH5T2FF3R			11	B0BBYS BUILDING SERVICE	0279518	G2G2GH1T5HG32NOGH23Y5DF2GH-YH3TTY 6TY2HY655T165TYH956Y21HERY3-Y63YH E66E6G2E6R26G*76H3HB3Y6-FG3Y6H2G1			13B0BBYS BUILDING SERVICE 0279518 H3Y4612H3Y6Y-HN3Y6Y6Y33H32Y6H63HY 5*T6G3H56H633G-Y62 OBG63G2H2G3G63 HG636H3B,G2G2T6YH6HG3G3B,,H-YGGF2			11	COOKS GARDEN SHOP	4349043	DJCUR84UJRJ48RIR58RFUI49FKRI48UJ3 KFI58TUJJDH295JFU59060GMKVGJG75940 F050E4KFJT959RKIFKG096040RF0KVKTO			13COOKS GARDEN SHOP 4349043 H3Y4612H3Y6Y-HN3Y6Y6Y33H32Y6H63HY 5*T6G3H56H633G-Y62 OBG63G2H2G3G63 HG636H3B,G2G2T6YH6HG3G3B,,H-YGGF2			<p>← Statement for Bill's Bicycles</p> <p>← Statement for Bobby's Building Service</p> <p>← Statement for Cook's Garden Shop</p>
11	BILLS BICYCLES	0293822																														
G2G2GH1T5HG32NOGH23Y5DF2GH-YH3TTY H02HY6HH22H2H6YH63HTGD SF5 TYH3Y66H 6TY2HY655T165TYH956Y21HERY3-Y63YH																																
13	BILLS BICYCLES	0293822																														
WE66-H-6Y6-H2TY+B-TY*F32DH-6-YH2N E66E6G2E6R26G*76H3HB3Y6-FG3Y6H2G1 69G5T96Y9R5FGHHY6Y-T-G6YH5T2FF3R																																
11	B0BBYS BUILDING SERVICE	0279518																														
G2G2GH1T5HG32NOGH23Y5DF2GH-YH3TTY 6TY2HY655T165TYH956Y21HERY3-Y63YH E66E6G2E6R26G*76H3HB3Y6-FG3Y6H2G1																																
13B0BBYS BUILDING SERVICE 0279518 H3Y4612H3Y6Y-HN3Y6Y6Y33H32Y6H63HY 5*T6G3H56H633G-Y62 OBG63G2H2G3G63 HG636H3B,G2G2T6YH6HG3G3B,,H-YGGF2																																
11	COOKS GARDEN SHOP	4349043																														
DJCUR84UJRJ48RIR58RFUI49FKRI48UJ3 KFI58TUJJDH295JFU59060GMKVGJG75940 F050E4KFJT959RKIFKG096040RF0KVKTO																																
13COOKS GARDEN SHOP 4349043 H3Y4612H3Y6Y-HN3Y6Y6Y33H32Y6H63HY 5*T6G3H56H633G-Y62 OBG63G2H2G3G63 HG636H3B,G2G2T6YH6HG3G3B,,H-YGGF2																																

To define the primary key field:

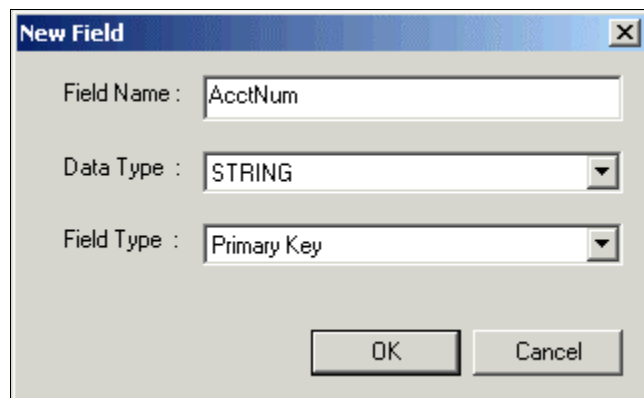
1. From the Definition Tool Bar, select the Field  icon.
2. In the work area, on a first page of any statement, scroll to the location of the data string that you have chosen as the primary key field, so that you can see the entire string in the work area. You can use any statement in the data source as the basis for the creation of your primary key field definition. In fact, this is true for any extraction rule you create. Just choose the correct page style (one that contains the data desired) so that you can be accurate in your window coordinates. (Scroll to the Account Number data string, in the upper right corner of the page.)
3. Use your mouse to left-click and drag over the area that contains at least the first position of the data string that you want. Select an area that includes the first numeric digit of the Account Number located to the right of the "Account Number" header. Customer names or addresses should not be used as the primary key. Social Security numbers, account numbers, or any other unique item is recommended. The area is highlighted in black, and the Select Your Choice dialog box appears, displaying two options: Field or Anchor for Field.



4. Select **Field** and click **OK**. The field Information dialog opens.



5. In the Field Information dialog, select the **New** button. The New Field dialog opens.



6. Enter Primary Key **field end**. Names must not contain spaces or special characters, and must start with an alpha character. The names must not exceed 30 characters in length. (Enter `AcctNum.`)
7. Select **STRING** from the Data Type dropdown list.

Caution Your **Primary Key field** must be defined as a **STRING**.

8. From the Field Type dropdown box, select **Primary Key**. The only time you will select a field type is during the Primary Key field definition.
9. Click **OK**. The New Field dialog closes and the name appears in Field Name.
10. Enter the pattern to extract the primary key. (Enter the regular expression `[0-9]+`)
11. Select the page style name from the drop down window. (Select `RemitPage.`)
12. Verify the window coordinates. Your window coordinates must include the first position of the Account Number data string. You should navigate through other 'first pages' to verify that these coordinates are sufficient to capture the data string on all statements. If you find that the window needs adjustment, you can make changes within the Field Information dialog, in the Window Coordinates section, by clicking the up/down arrows for each coordinate.

After navigating through the work area view of your statements' first pages, determine the appropriate window coordinate set. An appropriate window coordinate set would be

Start Row = 2	Start Column = 72
End Row = 4	End Column = 76

Since this definition is for the Primary Key field, you need to define a fixed length of characters for the extraction components. Use information from your edocs Bill Mastering sessions (and also from the hard-copy samples of your statements) to determine what this length should be.

Caution Use of an anchor for the Primary Key field is not recommended unless necessary.

13. Determine the maximum length of characters for the Account Number data string by either navigating through the hard copy or soft-copy of the National Wireless statement sample.
14. Enter the fixed length desired into the **Length** field, and check on the **Force Fixed Length** option. This setting will force the extraction components to verify the first part of the data string found, by the pattern, and then extract any subsequent characters to the number specified in the Length field. (Enter a length of 7 characters, and select the **Force Fixed Length** option.)
15. Click **OK** to close the Field Information dialog. The Field Information window closes. Your Primary Key field name now appears in Application Tree, as a child node of the Document Style tree node.

Creating Additional Page Styles

A page style is a rule you create to define the style for pages subsequent to the document style (first) page in a statement.

Your statements will likely have additional pages that contain different types of data and have an appearance different from the document style. For example, your statements may require additional page styles to include an Account Summary, Detail, Cover Letter, ID Card or other information.

Also, the same statement can have one or more pages of the same page style, such as a single account summary page followed by 25 detail pages on a credit card statement. You must create two page styles for a DDF to extract data from a statement like this. The first page style would be the account summary page. If it is also the first page of the statements, then this would be the document style. The second page style would be the detail page.

Creating an additional page style is nearly the same process as defining a document style. You must find a data string on the page (i.e. the title DETAIL) that uniquely identifies this page style, and define the area and pattern to allow the extraction components to locate this string during deployment.

You can define a page style in two ways:



- Directly
- After the definition of a page style anchor

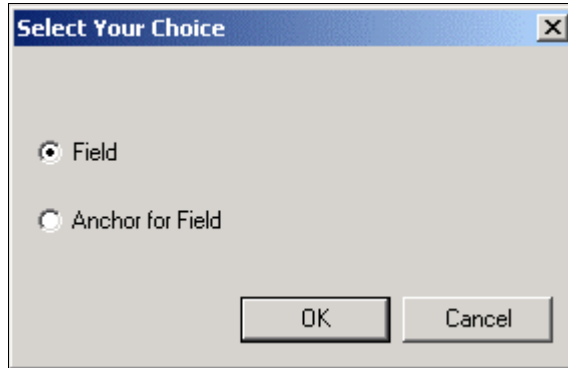
If you've already defined an anchor, the system looks for the anchor first and then the page style when extracting data. The directions below do not address defining a page style anchor. The process to create a page style anchor does not differ from the creation of a field anchor, with the exception that you must use the page style tool. Thus, refer to the Field Anchors Overview for more detail on anchors.

Caution You must define the rule to detect your page styles before you build rules to extract data from these pages. You cannot create a field, for example, to be extracted from a Detail page, until you have created a page style to properly locate the Detail pages of a statement.

To determine what page styles will need to be defined, refer to the information that you acquired during your edocs Bill Mastering sessions. (Refer to the hard-copy views of your National Wireless statements.)

To define a page style:

1. Navigate to the first statement's second page of the data source by clicking the Next Page  icon. (Go to Page 2 of the *NatlWireless.txt* data source in the DefTool work area.)
2. Click the Page Style  icon.
3. In the work area, left-click and drag your mouse over the first position of the data string that will identify the new page style. (In the work area, select the A in ACCOUNT SUMMARY.) The Select Your Choice dialog appears (shown below).



4. Select **Page Style** or **Anchor for Page Style**. (Select **Page Style**.)
5. Click **OK**. The Page Style Information dialog opens.
6. Enter the page style end. (Enter `AcctSummary`.)

Caution Names must not contain spaces or special characters, and must start with an alpha character. The names must not exceed 30 characters in length.

7. Enter a regular expression (static or variable) to validate the identifying string. (Enter `ACCOUNT SUMMARY`.)
8. Verify that your window coordinates are sufficient to capture the first position of your identifying string, on all pages of this type, across all statements in your data sources. (Verify that your `AcctSummary` page style window coordinates are:)

Start Row = 2 Start Column = 33

End Row = 3 End Column = 34

9. Click **OK** to close the Page Style Information dialog and save your page style. (Click **OK**. Your new page style, `AcctSummary`, is now represented in the Application Tree, and you can now build rules to extract data from the Account Summary statement pages.)

Overview of Fields and Field Anchors

Once you have defined the document style and primary key rules, you can write extraction rules for data fields.

Before you can begin writing field extraction rules, you must carefully analyze the input file to:

- Identify all the pieces of information you need to extract from the data input file to show customers in online statements, such as account number, address, services, total amount due, and statement date.
- Identify any data you want to extract for use with conditional business logic. For example, you may want to display different-looking statements for B2B or B2C customers. By extracting the customer type data field, you can write a conditional statement that looks at the field value to determine which HTML template to use for each customer. (You build conditional logic into your application using Composer.)

Note that you should hard-code static data into the HTML template instead extracting it from every statement. This includes labels or other descriptive text such as the “ACCOUNT NUMBER:” or “DUE BY:” labels that accompany the variable account number and due by fields every customer has. Occasionally, you may need to extract descriptive, static text along with the variable data. This is usually the case when the data does not always exist in every statement, such as Long Distance Charges. This way, you can drive the placement of descriptive text by the presence of relevant data.

You can define a field on its own or after the definition of a field anchor.

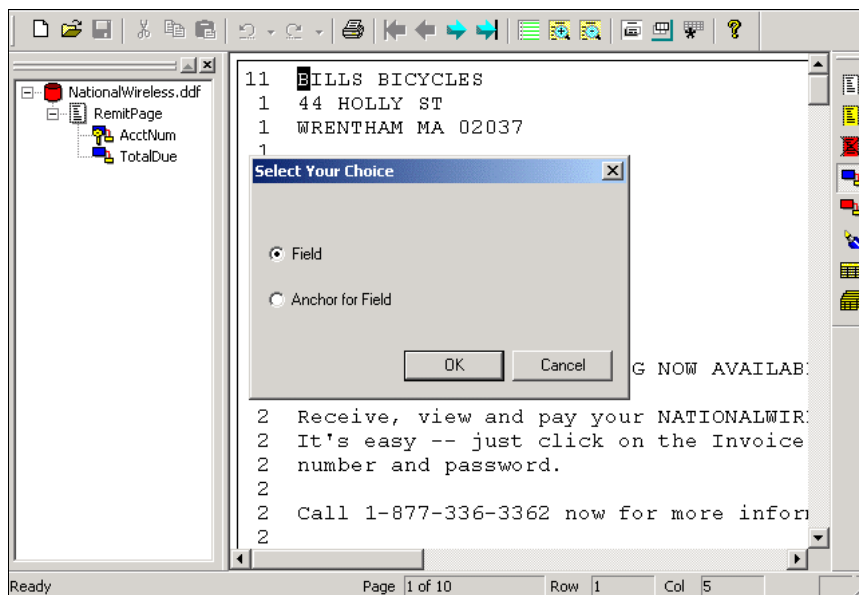
You can also define any post-extraction formatting rules you want.

Creating a Field Definition

This section describes how to create a field definition.

To create a field definition:

1. In the DefTool work area, navigate to any occurrence of the page style that contains your prospective field, and scroll to the location of the field.
(Navigate to the RemitPage of any statement, i.e. Page 1 of the data source. Scroll to the upper left corner of the page, where the customer name resides.)
2. If you are creating a field directly, ensure that the **New Field** icon is selected. If you have created an anchor for your new field, this icon will already be selected.
3. In the DefTool work area, left-click and drag your mouse over the area that covers the first position of all customer names in the statements of the data source. The Select Your Choice dialog appears. (Left-click and drag your mouse over the first character in the customer name, i.e. highlight the B in BILLS BICYCLES.)



4. Select **Field** and click **OK**. The Field Information dialog appears.
5. Select the **New** button. The New Field dialog opens.

6. Enter a name and data type for your field. Keep names short but descriptive, as in TotAmtDue. Field, table, group, and marker names must start with a letter, use alphanumeric characters only (no special characters), and cannot contain spaces. Do not use the following reserved words in a name: Time, Date, INT, Double, Currency (“Date” is not permitted but “DueDate: is). Field, table and group names (but not marker names) must be unique. Maximum length is 30 characters. Names are not case-sensitive. (Enter CustomerName as the field name, and accept the default of STRING as the data type.)

Caution Do not change Field Type (this box remains empty unless defining primary key field).

7. Click **OK**. The New Field dialog closes and the name appears in the Field Name box.
8. Click **OK** to close the New Field dialog.
9. In the Pattern box, enter an appropriate regular expression that will be used to validate and extract the field data. (Enter the regular expression `[A-Z]` to find only strings that start with a capital letter.)

At this point, you should decide if your field extraction rule will use a pattern to extract the entire data string, or if you will extract the field by Forced Fixed Length. If you want to extract the field by length, you may still enter a pattern that will validate at least the first portion of the data value. For example, to avoid extracting lines of spaces, you should use “`^`” (not a space) as your pattern. This will require that the first position contain a character other than a space.

10. Select the page style where the field resides from the corresponding dropdown list. (Select `RemitPage`.)
11. Verify that the window coordinates are sufficient to capture the first position of any occurrence of your field, across all statements. (Your window coordinates should be: Start Row = 0, Start Column = 4, End Row = 2, End Column = 5.)
12. To create custom post-conversions for the field, see “Custom Post-Conversions” on page 106.

Custom Post-Conversions

You can create custom post-conversions to use with the data extraction rules you define using DefTool. DefTool lets you apply post-conversion formatting specifications to a field or column value to further format the data for dynamic display online or for data storage purposes using the Command Center Indexer job. You can currently apply standard positional-based alphanumeric and date post-extraction conversions to extracted data, such as “Delete all trailing zeros or spaces™” and “Translate Aug 21, 2002 to 08/21/2002™.” You can create *customized* post-conversions to apply along with any sequence of existing (standard) post-conversions. You can apply multiple standard or custom post-conversions to a data field.

You can create custom post-conversions using:

- Regular expressions using the third-party PCRE 3.9 RegEx library for non-date and non-time data
- Date and time string formats

New regular expression library

When creating a regular expression for custom post-conversions in eaDirect, you must use the syntax of the third-party PCRE 3.9 RegEx library. For information about this library and the use of regular expressions, see “Mastering Regular Expressions” by Jeffrey E.F. Friedl, available at <http://www.oreilly.com/catalog/regex/toc.html>

It is important to note that is not the same library you use to create regular expressions for extracting data using the DDF.

Date and time formats

See “Appendix A – Data Definitions” in the *eaDirect User’s Guide* for a list of date and time formats you can use to build post-conversions for date and time data in eaDirect.

Reusing custom post-conversions

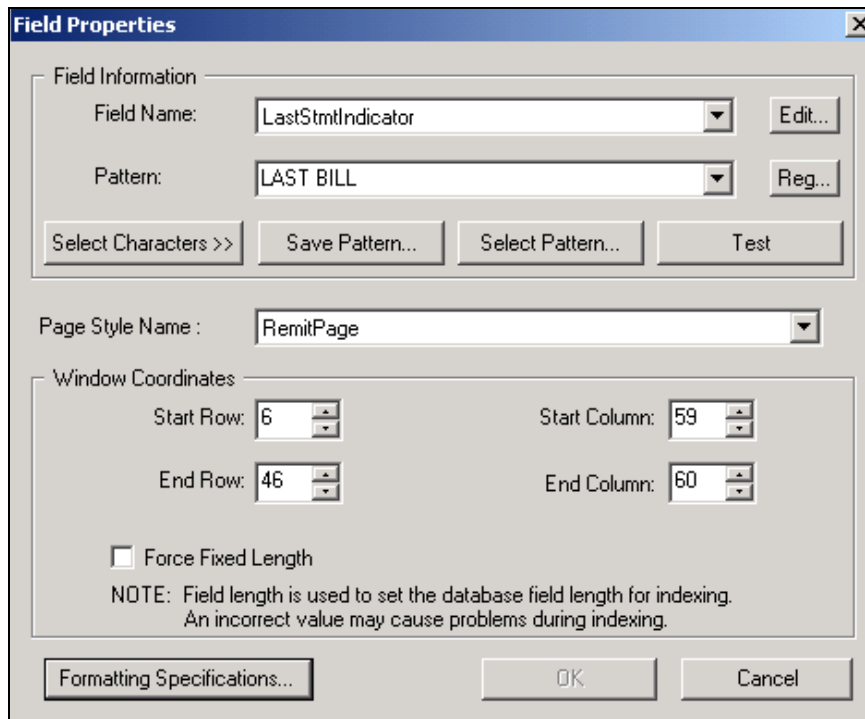
Each custom post-conversion you create becomes available for reuse with other data fields in the same eaDirect application (in the same DDF).

In addition, DefTool lets you save an application’s custom post-conversions to an xml file. This enables you to individually copy and paste the expressions into custom post-conversions you create in other another DDF.

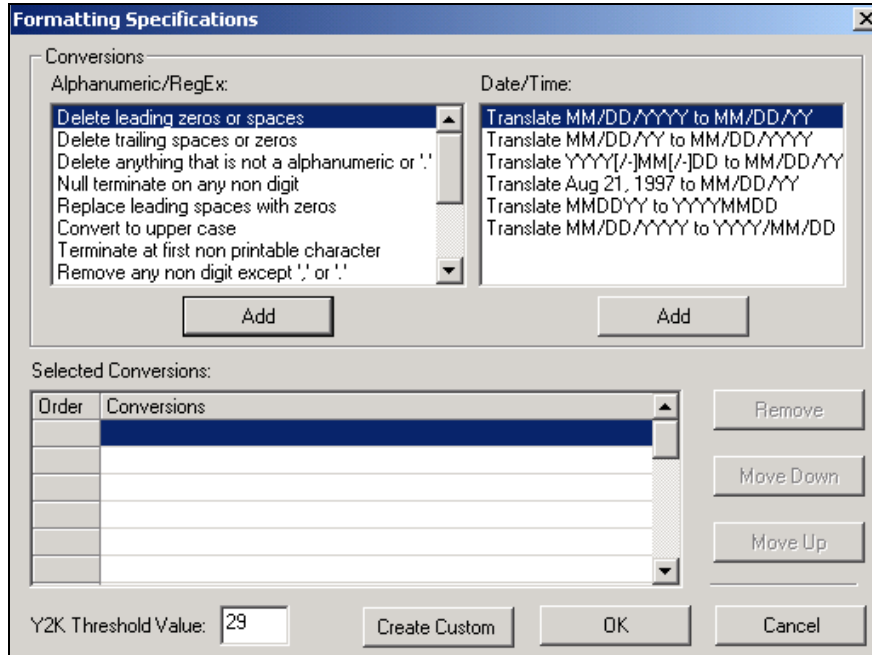
Creating a new custom post-conversion

To create a custom post-conversion for a field or column:

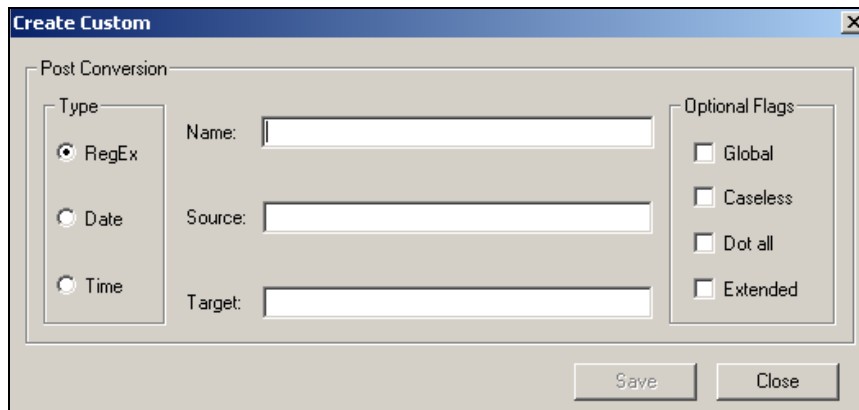
1. When defining the extraction rules for a field, DefTool displays the Field Properties dialog. (When defining a column, DefTool uses the similar Column Properties dialog.)



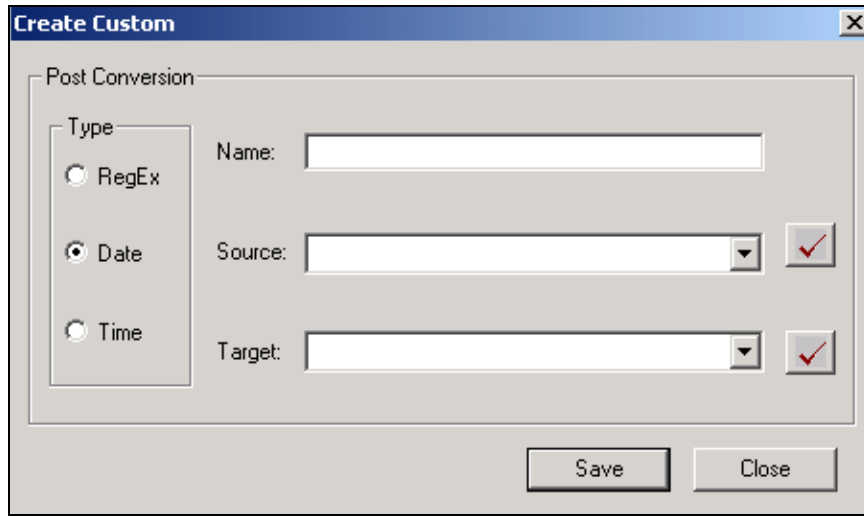
2. Click **Formatting Specifications**. DefTool displays the Formatting Specifications dialog. Use this dialog to choose any combination of standard or custom post-conversions and their application sequence to apply to the field or column.



3. Click **Create Custom**. DefTool displays the Create Custom dialog.

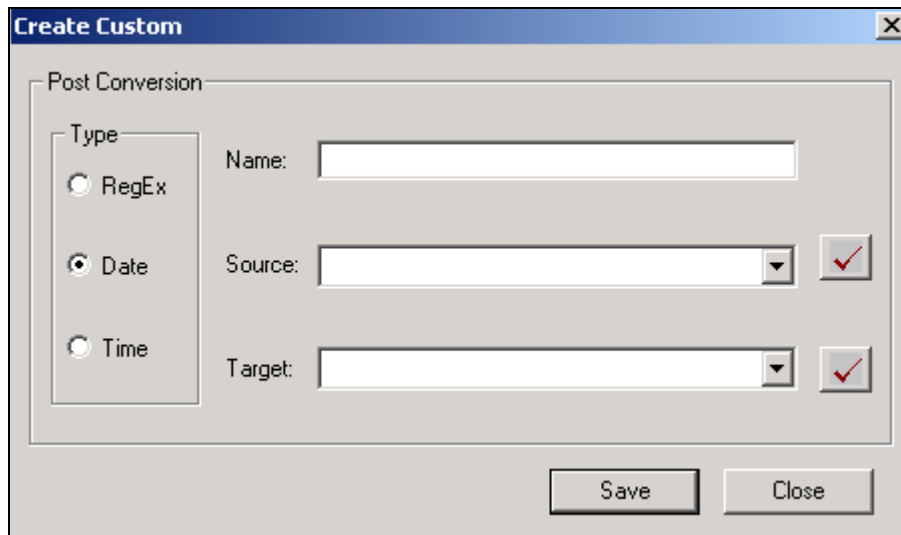


4. Specify the type of post-conversion you want to create: Date or Time to convert most date or time data, or RegEx to specify a regular expression for all other data types. If you choose a date or time, the Create Custom dialog redisplayes with different options.



5. Specify a meaningful name to identify the custom post-conversion. (This name appears in the “Alphanumeric/RegEx” or “Date/Time” conversion list on the Formatting Specifications dialog after you save, making it available to select with other fields or columns.)
6. In the Source field, enter the regular expression or date/time format functions you want to apply to the extracted data (this is on top of the regular expression defined on the Field Properties dialog that eaDirect first applies to the extracted data). For a regular expression, you can choose any of the following matching option letters; DefTool adds them to the expression between “(?) and “)””.
 - g - Global
 - i - Caseless
 - s - Dotall
 - x - Extended

7. In the Target field, enter the expression for converting the extracted source data for presentment.
8. When creating a date or time post-conversion, click after specifying the source and target functions to verify that the format is valid.



9. When you're through specifying the post-conversion functions, click **Save**. Continue defining any additional post-conversions you need, then click **Close**. DefTool displays the Formatting Specifications dialog with the new custom post-conversion in the appropriate conversion list.
10. To add the custom post-conversion to the field, highlight the conversion and click **Add**. Click **OK** when you are done defining formatting specifications for this field.
11. You can continue the process of defining extraction rules for this field. See the *eaDirect User's Guide* for complete instructions.

Editing a post-conversion

To edit a custom post-conversion:

1. With the DDF open in DefTool, select **View>Custom List**. The Custom Post Conversion List dialog appears.
2. Highlight the custom post-conversion you want and click **Edit**. The Edit Custom dialog appears.
3. Enter your edits. Click **Save**.
4. Click **Close**.

Removing a custom post-conversion from a field or column

To delete a post-conversion (custom or standard) from the formatting specification for a particular field:

1. With the DDF open in DefTool, right-click the field or column in the tree.
2. Select **Edit** from the right-click menu.
3. Click **Formatting Specifications**.
4. Highlight the custom post-conversion in the Selected Conversions list and click **Remove**.
5. Click **OK**.
6. Click **OK** on the Field (or Column) Properties dialog.

Deleting a custom post-conversion from an application

To delete a custom post-conversion from the application:

1. With the DDF open in DefTool, select **View>Custom List**. The Custom Post-Conversion List dialog appears.
2. Highlight the custom post-conversion you want and click **Delete**.
3. If the selected custom post-conversion is not in use, DefTool asks if you are sure you want to delete it. If it is in use, DefTool tells you which fields currently use the custom post-conversion and warns you that deleting it also removes it from the field formats. Click **Yes** to delete the custom post-conversion (or **No** to cancel). DefTool removes the custom post-conversion from the application and from any fields where it was in use.
4. Click **Close**.

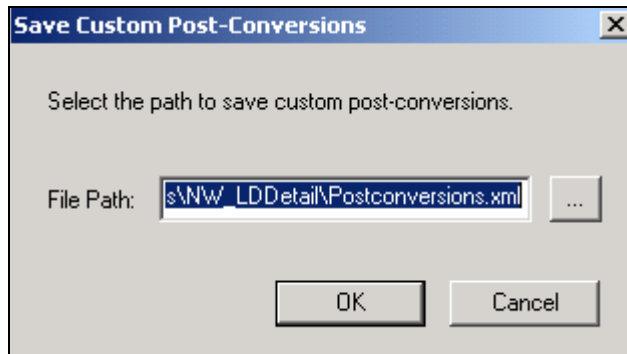
Sharing custom post-conversions with other applications


When you create and save a custom post-conversion, it becomes available for use in that application DDF only. To make those custom post-conversions available for use in another DDF, DefTool lets you save the custom post-conversions to a text file (in xml format). You must then recreate the custom post-conversions in the other DDF, copying and pasting the individual expressions from the file.

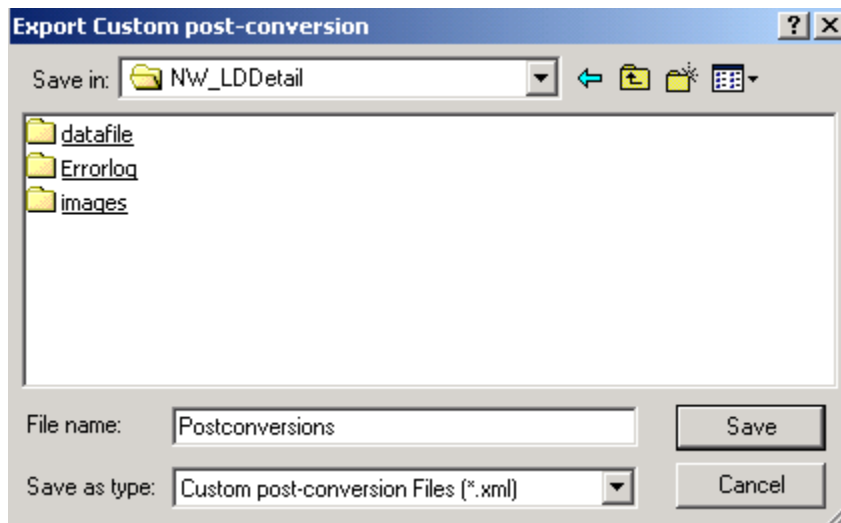
You can also add custom post-conversions to an existing file.

To save custom post-conversions to an external text file:

1. Select **File>Save custom post-conversions**. The Save Custom Post-Conversions dialog appears.



2. Click  to browse for a directory and/or edit the file name. DefTool displays the Export Custom Post-Conversion dialog. Specify the path and/or file name and click **save**.




3. Click **OK**.

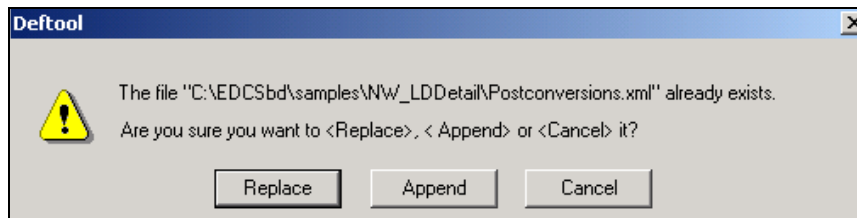
To copy saved custom post-conversions to another DDF:

1. In DefTool, open the DDF you want to add the saved custom post-conversions to.

2. In a text editor, open the xml file containing the saved custom post-conversions.
3. Follow the instructions in this chapter to create custom post-conversions, copying and pasting the individual expressions from the xml file to recreate the custom post-conversions in this DDF.

To append or replace an existing custom-post-conversions file:

1. Select **File>Save Custom Post-Conversions**. The Save Custom Post-Conversions dialog appears.
2. Click  to browse for the xml file you want to append or replace. DefTool displays the Export Custom Post-Conversion dialog:



3. Click **Append** or **Replace**. DefTool adds any new custom post-conversions to the file or replaces the file contents entirely.

Date and Time Format Constraints

See “Appendix A: Data Definitions” for details about the basic date and time formats available.

You must observe the following restrictions when using date and time formats to create custom post-conversions in DefTool:

- When converting a time duration format to another duration format, use only the following string components: %s, %Q, %K and %G

- You cannot convert a time format to a time duration or vice-versa. For example, you cannot convert the time duration “%K:%Q” to time format “%H:%M:%S” or the reverse. (It is OK to convert a time format to another time format, or to convert a time duration to a time duration, however.)
- Do not include both time duration formats and time formats in either the source or target. For example, you cannot have a source format of “%H:%Q:%S” and make the target “%T” because “%Q” represents a duration and “%H” is a time format.
- You cannot convert minutes to hours. Since the clock is in 1000 minutes, you can specify only %Q for the source. This only stores minutes in the tm structure and will contain values up to 999 and does not populate the hour field.

Examples of Custom Post-Conversions

The following examples show the regular expressions and formats you would use in the Source and Target fields on the Create Custom dialog to create various custom post-conversions.

Post-Conversion	Source (Regular Expression)	Target (Format)	Flag (Mode)
1,234,567.89 to 1.234.567,89 Convert dot(.) to comma(,) and comma(,) to dot(.)	<code>(\.) {1} (\,)</code>	<code>(?1,)(?2.)</code>	/g (Global)
1234567.89 to 1.234.567,89 Insert thousand separator as dot(.) and decimal separator as comma(,)	<code>(\.) {1} \G(\d{</code>	<code>(?1,)(?2\$&.)</code>	/g (Global)
00001234567.89 to 1.234.567,89 Remove leading zeros; insert thousand separator as dot(.); change decimal separator to comma(,)	<code>(\.) {1} ^0+ \G(\d{1,3}) (?=(?:\d\d\d)+(?!\d))</code>	<code>(?1,)(?2)(?3\$&.)</code>	/g (Global)
-1234567.89 to	<code>(\.) {1} \G\-</code>	<code>(?1,)(?2\$&.)</code>	/g

Post-Conversion	Source (Regular Expression)	Target (Format	Flag (Mode)
-1.234.567,89 Maintain negative sign; insert thousand separator as dot(.); change decimal separator to comma(,)	? (\d{1,3}) (=? (\d\d\d) (\d) + (?!\d))		(Global)
-1,234,567.89 to -1.234.567,89 Maintain negative sign; change thousand separator to dot(.) and decimal separator to comma(,)	(\.) {1} (,)	(?1,) (?2.)	/g (Global)
€1,234,567.89 to €1.234.567,89 Maintain the currency sign; change decimal separator to comma(,) and thousand separator to dot(.)	(\.) {1} (,)	(?1,) (?2.)	/g (Global)
€-78912354354.65767 to €-78.912.354.354,65767 Maintain currency sign; change decimal separator to comma(,); insert thousand separator as dot(.)	(\.) {1} \G(€? \ - ? €? \ - ?) (\d{1,3}) (=? (\d\d\d) (\d) + (?!\d))	(?1,) (?2\$&.)	/g (Global)
€-000058736545656.74647467 to €-58.736.545.656,74647467 Maintain currency sign; change decimal separator to comma(,); insert thousand separator as dot(.); trim off leading zeros	€? (0+) €\ - (0+) \ - ? (0+) (\.) {1} \G(€? \ - ? €? \ - ?) (\d{1,3}) (=? (\d\d\d) (\d) + (?!\d))	(?1€) (?2€-) (?3-) (?4,) (?5\$&.)	/g (Global)

Post-Conversion	Source (Regular Expression)	Target (Format)	Flag (Mode)
0000000.234 to 0,234 Trim off leading zeros; change decimal separator to comma(,)	(0){1,}(\.)	0,	/g (Global)
00000001.234 to 1,234 Trim off leading zeros; change decimal separator to comma(,)	(\.) 0+(\.){1} ^(0)+	(?1,)(?20,)(?3)	/g (Global)
999999999 to (999)999-9999 This requires you create and apply two separate, sequential post-conversions.	Post-conversion 1: \G^(\d{1}) \G(\d{5}) Post-conversion 2: \G(\.)(\d{3})	Post-conversion 1: (?1\(\$&)(?2\$&-) Post-conversion 2: (?1\$&\)	
10/14/2002 to October 14, 2002	%m/%d/%Y	%B %d, %Y	
18 julio 2002 or 20 agosto 2002 or 05 mayo 2001 to 18-07-2002 or 20-08-2002 Use these conversions for dates in languages other than English and French. This requires you create and apply two separate, sequential post-conversions.	Post-conversion 1 (Regular expression): (?:enero) (?:febrero) (?:marzo) (?:abril) (?:mayo) (?:junio) (?:julio) (?:agosto) (?:septiembre) (?:octubre) (?:noviembre) (?:diciembre) Post-conversion 2 (Date conversion): %d %m %Y	Post-conversion 1 (Regular expression): (?101) (?202) (?303) (?404) (?505) (?606) (?707) (?808) (?909) (?1010) (?1111) (?1212) Post-conversion 2 (Date conversion): %d-%m-%Y	/g (Global)

Explanation of first example

The first example in the table above shows that to convert 1,234,567.89 to 1.234.567,89, the post-conversion you specify must change the dot(.) to a comma(,) and the comma(,) to a dot(.).

Source – The regular expression looks for exactly one occurrence of dot(.) or a comma(,) in the source data:

```
(\\.) {1} | (\\,,)
```

Target – The target format specifies that if condition 1 is satisfied, replace the dot with a comma; else if condition 2 is satisfied, replace the comma with a dot:

```
(?1,,) (?2.,)
```

Flag (mode) – The /g or Global flag applies the formatting information globally (throughout the input string).

Explanation of second example

The second example in the table above shows that to convert 1234567.89 to 1.234.567,89, the post-conversion you specify must search for and insert the thousand separators as dot(.) and the decimal separator as comma(,).

Source – The regular expression looks for exactly one occurrence of a dot(.) or sets the anchor to the beginning, if a digit between exactly one to three occurrences, look ahead three digits followed by not a digit:

```
(\\.) {1} | \\G(\\d{1,3}) (?= (? : \\d \\d \\d) + (? ! \\d) )
```

Target – The target format specifies that if condition 1 is satisfied, then replace the dot with a comma, else if condition 2 is satisfied, then insert a dot(,):

```
(?1,,) (?2$&.,)
```

Flag (mode) – The /g or Global flag applies the formatting information globally (throughout the input string).

About Field Anchors

Field anchors provide a verification that the data that is found during extraction of a field is, in fact, the correct data. If a field anchor is defined, the extraction components look in the defined anchor window coordinates for the anchor pattern. If the anchor pattern is found, the extraction components then search for the field. If the anchor is not found, the DefTool does not search for the field on that page, and thus, does not extract any field data for that statement.

Field anchors are commonly used when the field data can “float” widely from statement to statement. If the window coordinate set of a field can span many rows, for example, then there is the strong possibility that other data strings of similar pattern could appear in the defined location. This means that the field rule you write could potentially extract the wrong data string. In this case, you would locate a data string that:

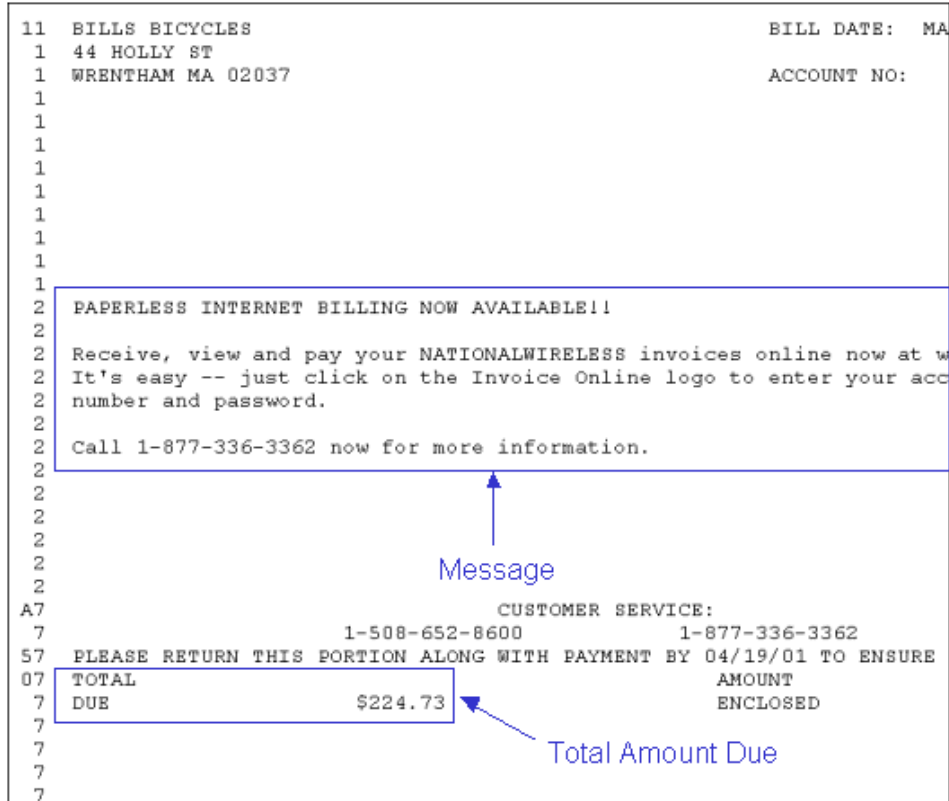
- Appears only if the field data exists on the statement
- Appears in a location relative to the field data.

Field anchors are commonly used with currency fields, especially when many currency values are on a page. Consider the following example:

On a biller’s statement, the Total Amount Due data resides on the Remittance Page. The Remittance Page (defined as the document style because it is the first page of the statement), may include a message to the recipient, the length of which can vary (illustration shown below).

Defining Data Extraction Rules with DefTool

```
11  BILLS BICYCLES                                BILL DATE:  MA
1   44 HOLLY ST
1   WRENTHAM MA 02037                            ACCOUNT NO:
1
1
1
1
1
1
1
1
1
1
2   PAPERLESS INTERNET BILLING NOW AVAILABLE!!
2
2   Receive, view and pay your NATIONALWIRELESS invoices online now at w
2   It's easy -- just click on the Invoice Online logo to enter your acc
2   number and password.
2
2   Call 1-877-336-3362 now for more information.
2
2
2
2
2
2
2
2
A7                                     CUSTOMER SERVICE:
7                                     1-508-652-8600          1-877-336-3362
57  PLEASE RETURN THIS PORTION ALONG WITH PAYMENT BY 04/19/01 TO ENSURE
07  TOTAL                                     AMOUNT
7   DUE                                     $224.73          ENCLOSED
7
7
7
7
7
```



The diagram shows a text-based bill with several sections. A blue box highlights a promotional message starting at line 2 and ending at line 8. An arrow labeled "Message" points to this box. Another blue box highlights the "TOTAL DUE" section starting at line 07 and ending at line 08, with a value of \$224.73. An arrow labeled "Total Amount Due" points to this box.

The Message here spans 7 rows, however, there is no guarantee that it will always span 7 rows on other statements. If the Message is shorter, then the Current Charges and Total Due values' will shift upwards. Additionally, the Message may not exist on some statements.

In-depth analysis of your data is needed to determine how widely the location of a data string can vary. So if the Message can be up to 10 lines or not exist at all (0 lines), then the row span that is necessary to extract the Total Due field is 10 rows.


```

11  BILLS BICYCLES                                BILL DATE:  MA
1   44 HOLLY ST
1   WRENTHAM MA 02037                            ACCOUNT NO:
1
1
1
1
1
1
1
1
1
1
1
2   PAPERLESS INTERNET BILLING NOW AVAILABLE!!
2
2   Receive, view and pay your NATIONALWIRELESS invoices online now at w
2   It's easy -- just click on the Invoice Online logo to enter your acc
2   number and password.
2
2   Call 1-877-336-3362 now for more information.
2
2
2
2
2
2
2
2
A7
7   1-508-652-8600                                1-877-336-3362
57  PLEASE RETURN THIS PORTION ALONG WITH PAYMENT BY 04/19/01 TO ENSURE
07  TOTAL                                          AMOUNT
7   DUE                                           $224.73      ENCLOSED
7
7
7
7

```



The data movement dictates The column span of a window coordinate set as well. Remember that your window coordinate set is used to locate the first position of the data string you are extracting. In the case of currency values, which tend to be right justified, you should compensate for large and small values, and adjust your column span to detect them (shown above).

The TotalDue field can contain the following definition:

```
Start Row/Column = 21, 4
End Row/Column = 30, 5
Pattern = ${0-9,}*\.[0-9][0-9][0-9][CR]*
```

Remember that the extraction components search the area defined in a left-to-right, top-to-bottom order. By defining a rule to locate another data string first, before searching for Total Due, we can safely extract the correct value.


When creating a field anchor, you must choose a data string that occurs in the same location relative to the data. The anchor string you choose can be on the same row as the data, or on the rows above or below, as long as the anchor string is always in that location relative to the data on every statement.

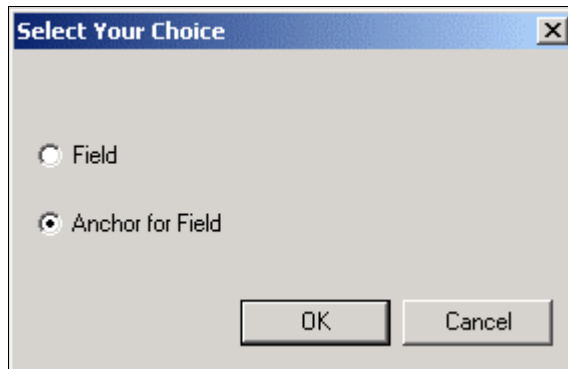
In the example above, after Bill Mastering, we can say that the string TOTAL always occurs on the same row as the TotalDue field value, therefore it would make a good anchor string.

Creating a Field Anchor

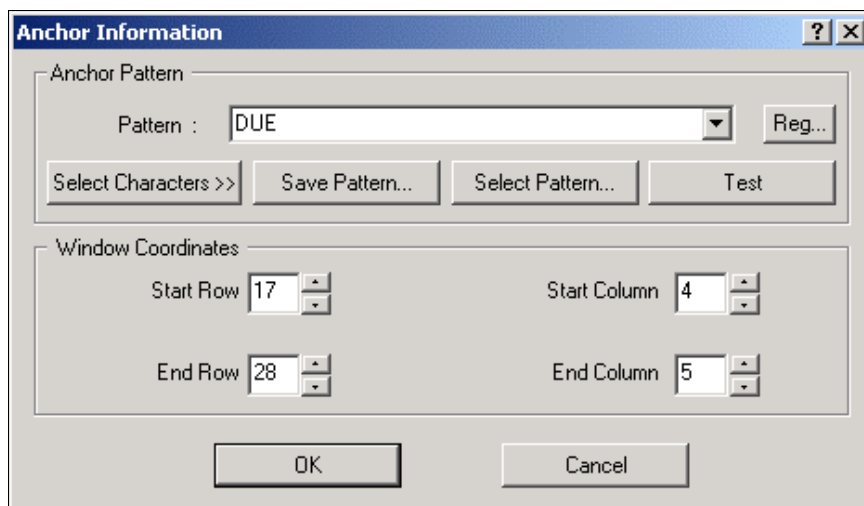
You create field anchors using the same tool as for field creation. When you need to create an anchor for a field, you must create it immediately before you create the field. This two-step process ensures that an internal bond is created between the anchor and the field.

To create a field anchor:

1. From the Definition Tool Bar, click the New Field  icon.
2. In the DefTool work area, select the area that you expect the anchor string to start on any statement. In this case, you would highlight rows 17 to 28, since you expect this string to be aligned with the Total Due value, and columns 4 to 5, since the string is left-justified and does not vary column-wise. The Select Your Choice dialog appears.
3. Click the **Anchor for Field** radio button (as shown) and click **OK**. The Anchor Information dialog appears (shown below).



4. In the Pattern field, enter the static string or variable regular expression that will detect your anchor string. In this case, you would enter **DUE**.

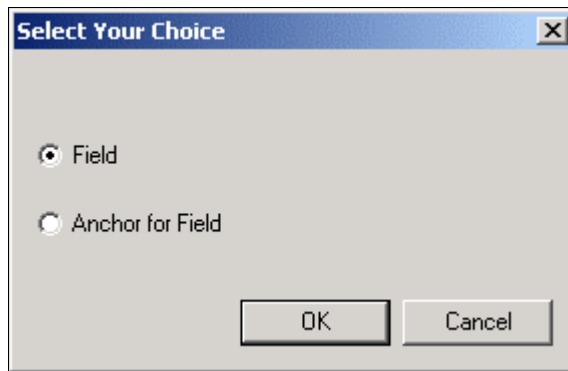


You can test this definition on the work area view by clicking the Test button. This will highlight, in purple, any string that this definition would be able to pick up.

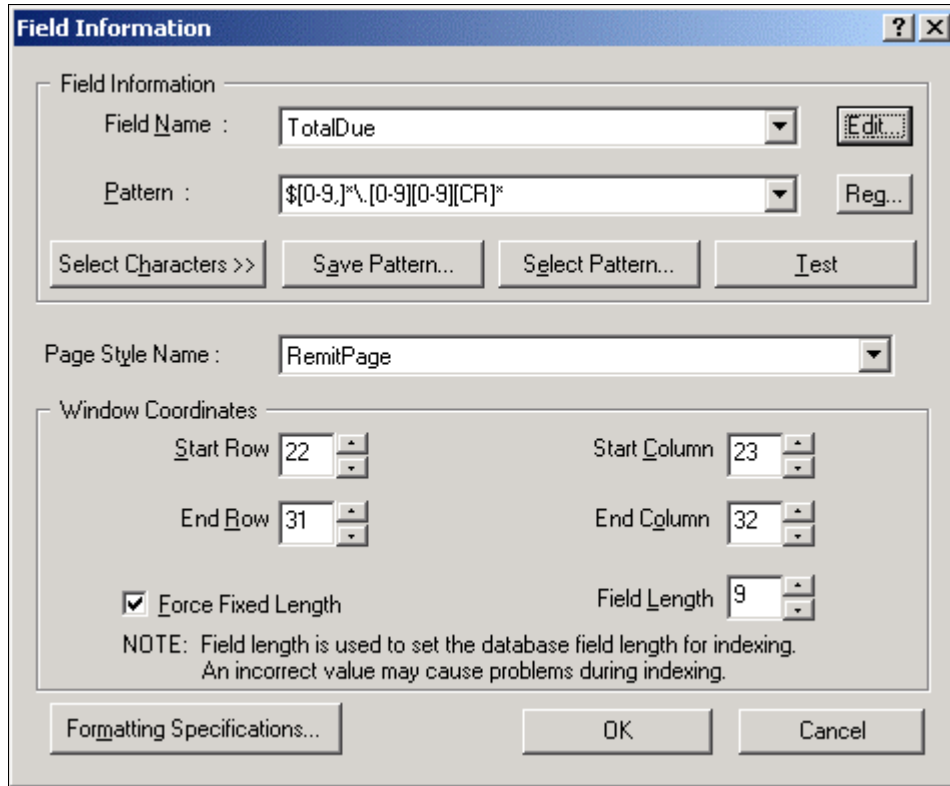
5. Click **OK** to save changes and close the Anchor Properties dialog.

Caution In the DefTool work area, you will notice that the area defined by the anchor's window coordinate set is now highlighted in bright green. There is no Application Tree item yet; the anchor definition will be contained within the field that you are about to create. This is why **you must create the field immediately after the anchor**.

6. With the New Field icon still selected in the Definition Tool Bar, highlight the work area to locate the field's first position. The Select Your Choice dialog appears again.



7. This time, specify Field (as shown) and click **OK**. The Field Information dialog appears (shown below).



8. Name the field in Field Name. Names must not contain spaces or special characters, and must start with an alpha character. The names must not exceed 30 characters in length.
9. Select the data type of the field.
10. Select the page style where the field resides in the statement.
11. Specify the pattern and/or set a length to extract by.
12. Make other formatting specifications as desired.
13. Adjust your window coordinates if necessary.
14. Click **OK** to save and close the Field Information dialog.



The anchor definition is bound internally to the field that is defined immediately after the anchor, and there is no Tree View representation of the anchor. To edit the anchor, you cannot delete an anchor without deleting the field to which it is attached.

The DefTool Tree View gains a new node for the field you created. When clicked, the work area is highlighted to show both the field and anchor windows (shown below).

```

11 ABC TRUCKING                                BILL DATE:  MARCH 25, 2001
 1  48 DELANCEY ROAD
 1  UPTON MA 01568                            ACCOUNT NO:   0123456
 1
 1
 1
 1
 1
 1
 1
 1
 1
 1
 1
 1
 1
 1
 2  ONLINE PAYMENT NOW AVAILABLE!!
 2
 2  Pay your NATIONALWIRELESS invoices online now at www.nationalwireless.net.
 2  It's easy -- just click on the Invoice Online logo to enter your account
 2  number and password.
 2
 2  Call 1-800-PAY-BILL now for more information.
 2
 2
 2
 2
 2
 2  CURRENT
 2  CHARGES                                $95.27
A7
07  TOTAL
 7  DUE                                $123.45
 7
 7
 7
 7
 7
  
```



Notice that the windows for the anchor and field shown above have the same number of rows. It is important that the row spans of the anchor and Field windows are the same. It is also important that the placement of the anchor window mimics the positional relationship of the anchor to the field. For example, if the anchor string is expected to be two rows above the field data, then the anchor window needs to start and end two rows above the Field window.

Editing an Anchor

The anchor definition is bound internally to the field that is defined immediately after the anchor, and there is no tree representation of the anchor.

To edit an anchor:

1. Locate and highlight the anchor's field in the DefTool Tree. As shown above, two windows will appear in the work area.
2. Locate the bright green window in the work area, and carefully double click this window. This will open the Anchor Information dialog where you originally defined the anchor.
3. Make the necessary changes to the anchor's pattern or window coordinate set, and click **OK** to close.

Deleting an Anchor

Since the anchor is not an entity of its own, it cannot be deleted independently. To delete an anchor, you must delete the field to which it is attached, and immediately repeat the steps of creating an anchor and its field .

Tables

You have already learned how to identify when data should be extracted as a field. To review, you build a field rule when your data exists on one row only, and does not have multiple iterations throughout the statement.

How do you extract data that exists on multiple rows, or is related to other data items, and/or has more than one unique occurrence in a statement? When your data fits any of these criteria, you will most likely extract it in the form of a table.

Simply put, a table consists of one or more fields that are combined into a unit for extraction and presentment purposes. In addition, the fields of a table, called columns, are related to each other in some way, and can have many instances in a statement. The table functionality allows you to build rules to extract these fields columns together, and look for multiple instances of the columns on all rows between a predetermined start and end location.

Extracting Tables for Text Presentation

The use of tables in customer applications will allow you to extract and present iterative or transactional information in a crisp, well-defined format that is easy to read in a browser. Some good examples of data that are best extracted as a table are:

A Customer Address — Some addresses consist of 2 lines of data, while other addresses require many more rows. The best way to extract an address entirely and simply is to create a table with one column (to extract all iterations of the Address Line)

A Credit Card Transaction List — The items of a Credit Card Transaction may include a date, retail outlet, location, and charge. Each of these items are related to each other in the transaction, therefore it makes sense to extract and present them together. Further, it is highly likely that a transaction list will contain multiple transactions, perhaps continuing the list onto many subsequent pages. If you extract this data as a table, with a predefined start and end location for the extraction components to search for data, the related items of every transaction will be presented with each other, and no special configuration is necessary.

Extracting Tables for Chart Presentation

eaDirect can extract data from your source, and present it dynamically in the form of a Chart. If you have the appropriate data, you can build a table to extract it and then use that table later on in the Composer, to build rules to compose the Chart during live retrieval of a statement. Charts are composed of two distinct pieces of information, namely the data to be charted and presentation directions. Using the eaDirect Composer tool, one can specify the data for charting. The charting functionality requires that you have KavaChart[®] installed.

KavaChart can load IIS or Oracle databases, and run via your own customized server-side tools. Charting in eaDirect involves the following steps:

1. In the Composer tool, insert a chart tag.
2. Define the look and feel properties for the chart.
3. Publish the properties file to the eaDirect application server.
4. Ensure display privileges are on the eaDirect web server.

You will need the appropriate data in your source in order to offer a Chart to your end-users. An example of Chart-appropriate data is:

- Current Service Charges — If, for example, your statements contain a Current Charge value for each service you provide, you can create a table that looks across an entire statement, with two columns to extract all instances of the following information:
- Current Charge Label — the descriptive text that occurs with the relative Current Charge (such as Local Service Charges)
- Current Charge Amount — the currency value of the Current Charge.

Essentially, you need to have data that will populate the following Chart components:

- The X-Axis, or Label
- The Y-Axis, or Value

See Chapter 5 for information about creating Charts in Composer.

Table Definitions

A table definition consists of:

- One or more columns, which are defined similarly to fields, and whose iterations occur in a pattern together.
 - A column cannot be searched for in more than one location

- A column cannot be defined with its own anchor. Rather, it uses the table anchor by default (table anchors are described below)
- A set of markers, which, during live retrieval, dynamically define the area in a statement that will be searched for instances of the defined columns. Your options for table markers are:
 - You can define only a start marker, or a start and end marker pair
 - Markers can have multiple locations
 - A start marker and end marker **do not** have to exist on the same page style
- An anchor, which guarantees that, on the row that it is found, there exists relevant data to be extracted in the table. The anchor of a table can either be:
 - Internal Anchor — A column that you have defined as part of the table, which will always extract data on all rows where there is table data. A good choice for an internal anchor might be the charge amount of a call itemization list, which depends on the logic that if there is a charge amount on a row between the start and end marker, then there is data on this row that should be extracted as part of the table.
 - External Anchor — A data string that is not actually part of the table data but occurs on rows, between the start and end marker, where there are relevant column data. A good example of an external anchor is the use of static control codes that are not part of the table, but indicate the existence of the table data on the row in which they are located.

Creating a Table

Once you have identified the data that you want to extract as a table, create the rules that define the table:

- Define the start markers of the table.
- Define the end marker (optional) of the table.
- Define the table format:

- Name the table.



The following rules must be observed when creating field, table, group, and marker names:

The names are not sensitive to case, but they must be fashioned from ONLY alphanumeric characters (no special characters are permitted).

The names must start with a letter, and cannot contain any spaces.

The names cannot contain any of the following reserved words alone: Time, Date, INT, Double, Currency. (For example: Date is not permitted, but DueDate is allowed.)

Field, table and group names must be unique (marker names are exempt from this requirement).

Keep the names short, e.g. TotAmtDue. Be descriptive, but use abbreviations instead of long names.

- Select the marker set that will detect the table and dynamically set the searching area for the column(s).
- Define the table column(s).
- Define the table anchor (internal or external).

Defining a Table in National Wireless

This section will lead you through each step of creating the rules that define a table using the National Wireless application. To illustrate each step, we will define the first table of the course on the customer address.

The goal will be to extract all of the customer information in one unit, including the customer end. This will allow us to define presentation of all this information with great ease in the Composer. We already extracted the customer end as a separate field, however we are not limited to extracting it once (see the Note below).

Sometimes, you will define rules to extract the same data in different ways. Since we want to present the customer name and address together, it makes sense to extract it as a unit, or in one table. Also, you may want to index the customer name as a field in the database. Since only fields can be indexed to the database, it makes sense to also extract the customer name as a field. This “double extraction” will not have an impact on application performance.

Defining a Table Start Marker

During live retrieval of a statement, a table start marker is used to inform the extraction components of the row on which to start searching for table data. You start the process of defining the start marker by first identifying a data string that appears on or above the row where table data begins. Start markers are mandatory, however, you may define more than one start marker for a table.

To define a table start marker:

1. Identify and locate the data string that indicates the start row of the table search area. This is where the extraction components will begin looking for column data. (Locate the string 11 in rows 0-1 and columns 0-1.)

2. Click to select the Marker icon  from the Definition Tool Bar.



You can easily find out the coordinates where your mouse is located. If you move your mouse into the work area, and then look at the Status Bar in the bottom pane of the screen, you can see the row and column coordinates of the cursor's location.

3. Move your mouse into the work area and left-click drag and drop your mouse over the first position of the string identified in Step 1. (Select the 1 in 11, so that one character position is selected.) The Select Your Choice dialog opens.
4. You will need to decide if you are creating a marker or an anchor for the marker (which you would create immediately after the anchor). (Accept the default of `marker`, and click `OK`.) The Marker Information dialog appears:

5. Enter the marker name. Names must not contain spaces or special characters, and must start with an alpha character. The names must not exceed 30 characters in length. (Enter CustAddress.)
6. Enter the pattern to be searched for as the start marker. Be sure that your pattern is sufficient to detect this start marker on all statements. (Enter 11 as the pattern.)

7. When creating a marker, the `Start Marker` radio button is selected by default. Accept this default for the start marker definition.
8. The start marker of a table can be located on or above the rows where table data exists. If your start marker string is on the same row where there may exist table data, then enable the **Include Marker Line** option. (The start marker string, `11`, appears on the same row as some customer address data. Click to enable `Include Marker Line`.)
9. The start marker string can be “re-used” as the end marker, meaning that you can define the end marker on the same exact string in the same location, with the intent that it be detected on a subsequent page. If you want to use the same string for the start and end marker and **not** create a null searching area for the table data, then enable **Look for the Start Marker on the Same Row** in the Start Marker Information dialog. (For `NatlWireless`, since your start marker and end marker will be different for the `CustAddress` table, **do not** enable `Look for the Start Marker on the Same Row`.)
10. If you are certain that your start marker’s row contains no other data than the start marker string itself, then enable the **Select the Line Only Containing This Pattern** option. (Since your start marker row does contain other data, **do not** enable `Select the Line Only Containing This Pattern`.)

Caution Select the `Line Only Containing This Pattern` will direct the extraction components to search the entire row for verification that no data besides the start marker string exists on that row. To use this option, you must be certain that no other data will ever appear on the start marker row.

11. You can use a row with no data on it as your start marker, by enabling the **Use Blank Line as Marker** option.
12. To **Use a Blank Line as a Marker**, you must be certain that a blank row will always indicate the existence of the table, and that there will always be a blank row preceding the table. (**Do not** enable `Use a Blank Line as marker`.)
13. Select the page style on which the start marker string resides. (Select `RemitPage` from the Page Style dropdown list.)

14. Verify that your window coordinates are sufficient to detect your start marker on all statements. If this table will be part of a group, verify that the window coordinates, especially rows, are sufficient to detect ALL occurrences of the start marker on a statement, and across all statements. (Verify that your window coordinates are:)

Start Row = 0

Start Column = 0

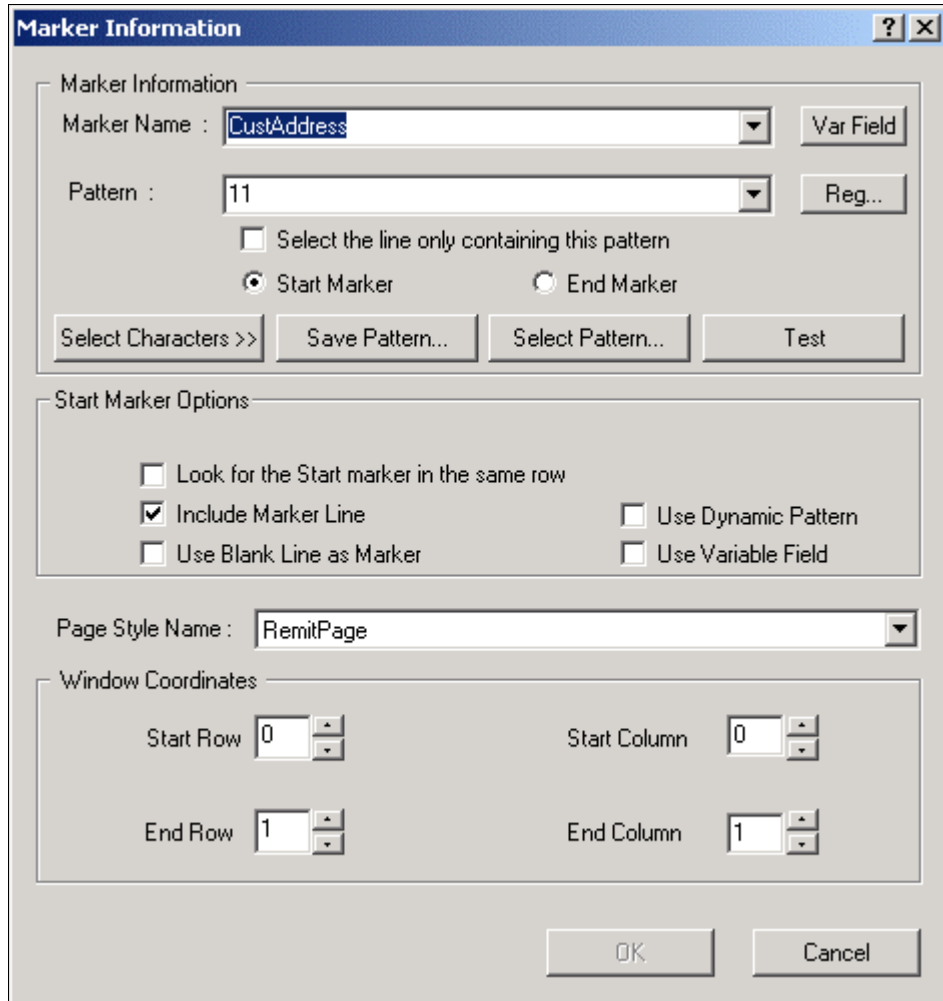
End Row = 1

End Column = 1

**Tip**

You can create your table rules so that only sections of a table are extracted and presented in a statement, which is driven by the choice of the recipient during statement live retrieval. To do this, you use a dynamic field, which dynamically sets a pattern to a table column, therefore requiring that only data lines where column X = Y will be presented. This feature appears in the marker information dialogs to define the same type of rules to extract specific group occurrences. Since markers for groups are created the same way as markers for tables, you see the feature here, however, selective table data extraction and presentment driven by dynamic pattern matching is defined at the table column level.

Your completed Marker Information dialog will appear as shown below.




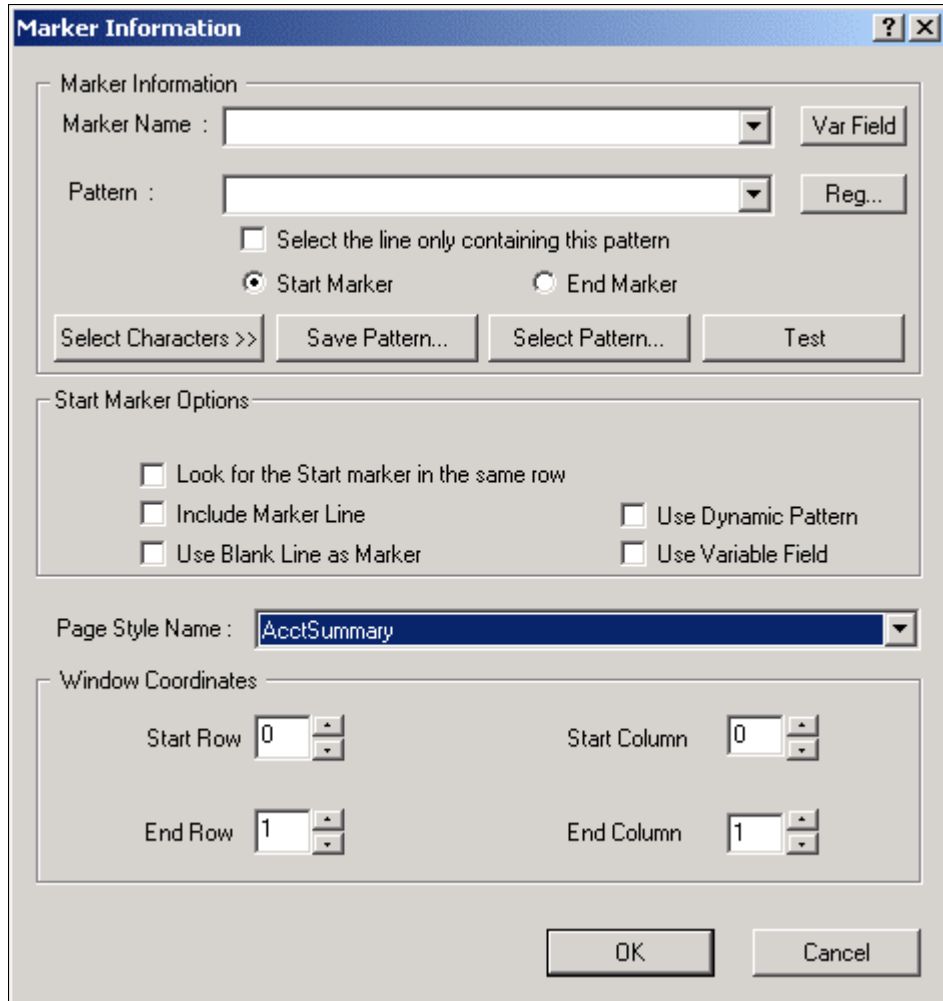
15. Click **OK** to close the Marker Information dialog and save your start marker definition. In the work area, notice that a yellow area defined by your marker's window coordinate set appears over your start marker string. Also, in the Tree a Start Marker node has been added to the tree.

Defining a Table End Marker

During live retrieval of a statement, a table end marker informs the extraction components of the row on which to stop searching for table data. You start the process of defining the end marker similarly to a start marker, by first identifying a data string that appears on or below the row where table data is expected to end. Whereas at least one start marker is mandatory, however, the definition of an end marker is optional, depending on the type of data you are extracting.

To define a table end marker:

1. Identify and locate the data string that indicates the end row of the table searching area. This is where the extraction components will stop looking for column data. (The string 2 in the left margin of the page is the best indicator of the end of the Customer Address section.)
2. Select the Marker icon  from the Definition Tool Bar.
3. Move your mouse into the work area and left-click drag and drop your mouse over the first position of the string identified in Step 1. (Select the first two control characters along the left margin of the page.)
4. The Select Your Choice dialog opens. From here, decide if you are creating a marker or an anchor for the marker (which you would create immediately after the anchor). (Accept the default of `Marker`, and click `OK`.) The Marker Information dialog opens (shown below).



5. Enter the marker name. Names must not contain spaces or special characters, and must start with an alpha character. Maximum length is 30 characters.

When you are creating a set of start and end markers, to use together with a table, you must give the end marker the same name as the start marker. This forces them to be attached as a marker pair. If you have already created and named the start marker to use with this new end marker, then you already have the end marker name. Select this name from the marker name dropdown list or enter it manually. (Select `CustAddress`.)

6. Enter the pattern to be searched for as the end marker. Be sure that your pattern is sufficient to detect this start marker on all statements. (Enter `2` as the pattern.)
7. When creating any marker, the Start Marker radio button is selected by default. Be sure to select the **End Marker** radio button before closing the Marker Information dialog.
8. The end marker of a table can be located on or below the rows where table data exists. If your start marker string is on the same row where there may exist table data, then enable the **Include Marker Line** option. (The end marker string, `2`, does not appear on the same row as any customer address data. **Do not enable** `Include Marker Line`.)
9. If you are certain that your end marker's row contains no other data than the end marker string itself, then enable the **Select the Line Only Containing This Pattern** option. (Since your end marker row does contain other data, **do not enable** `Select the Line Only Containing This Pattern`.)

**Caution**

Selecting the **Line Only Containing This Pattern** will direct the extraction components to search the entire row for verification that no data besides the end marker string exists on that row. To use this option, you must be certain that no other data will ever appear on the end marker row.

10. You can use a row with no data on it as your end marker, by enabling the **Use Blank Line as Marker** option. (**Do not enable** `Use a Blank Line as Marker`.)

**Caution**

To use a blank line as a marker, you must be certain that a blank row will always indicate the existence of the table, and that there will always be a blank row preceding the table.

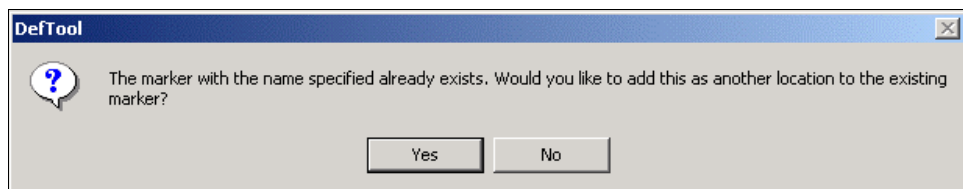
11. Select the page style on which the end marker string resides. Your end marker can reside on a different page style than the start marker it is matched with. (Select `RemitPage` from the Page Style dropdown list.)

12. Verify that your window coordinates are sufficient to detect your end marker on all statements. If this table will be part of a group, verify that the window coordinates, especially rows, are sufficient to detect ALL occurrences of the end marker on a statement, and across all statements. (Verify that your window coordinates are as follows:)

Start Row = 10	Start Column = 1
End Row = 14	End Column = 2

You can create your table rules so that only sections of a table are extracted and presented in a statement, which is driven by the choice of the recipient during statement live retrieval. To do this, you use a dynamic field, which dynamically sets a pattern to a table column, therefore requiring that only data lines where column X = Y will be presented. This feature appears in the Marker Information dialogs to define the same type of rules to extract specific group occurrences. Since markers for groups are created the same way as markers for tables, you see the feature here, however, selective table data extraction and presentation driven by dynamic pattern matching is defined at the table column level.

13. Click **OK** to close the marker Information dialog and save your end marker definition. The marker information dialog closes, and a dialog opens, alerting you to the creation of a marker pair (shown below).




14. Click **Yes** to create a marker pair with your end marker and start marker. In the work area, notice that a yellow area defined by your marker's window coordinate set appears over your end marker string. Also, in the Tree, an End Marker node has been added, creating the marker pair in the tree.

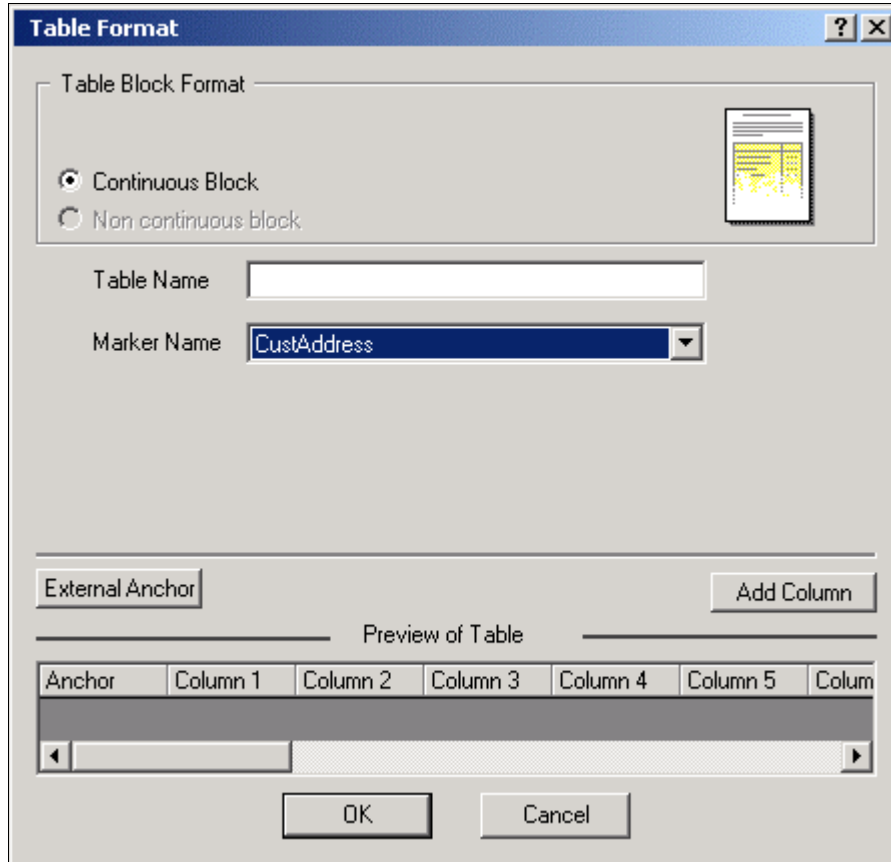
Tip

If you are defining an anchor for the marker used with a table group or a nested table group, make sure the anchor repeats for each occurrence of the group. If the anchor value is not present for an occurrence, that occurrence will not be extracted. The same is true if you are defining an anchor for the marker used with a nested group.

The next task is to define the format of the table. This includes naming the table, selecting the marker pair, defining rules to extract the data as columns and selecting an anchor for the table columns. You must create the marker pair before you can define the table format.

To define the table format:

1. In the Definition Tree, click the Table icon . The Table Format dialog opens (shown below).



2. Enter the table name. Names must not contain spaces or special characters, and must start with an alpha character. The names must not exceed 30 characters in length. (Enter `CustAddress`.)

It is a good idea to give your table the same name as the marker pair that will be used with it. This will help you organize your tables during presentment definition in the Composer, and also aid in DDF troubleshooting, if necessary.

3. Select the **Marker End** from dropdown list, to attach the marker pair to your new table. (Select `CustAddress` from the Marker Name dropdown list.)

Caution

Do not close the Table Format dialog at this point. Your next step is to define the table columns, which is done from within the Table Format dialog.

If you need to close the table Format dialog at any point before you have completed the settings for the table, **do not** click **Cancel**. This will cause you to lose any table format settings you may have made, including the end, markers and any columns created. Rather, click **OK**. You can re-enter this dialog from the Tree (right-click the table node and select **Edit**) at a later time to complete any work on the table that is left.

Defining Table Columns

After you have created the table markers, named the table, and selected its markers, your next step is to define the rules to extract the table data as a column or columns. First, you need to identify the columns that will be built into your table, by examining the data samples. (For the CustAddress table, you only need to build one column to extract each line of the data.)

Tip

Generally, you can think of table columns as data elements that iterate **vertically**, not horizontally.

To define a table column:

1. Open the Table Format dialog (if not already open) by right-clicking on the Table node in the Tree, and selecting Edit. (You should be working in the Table Format dialog for the CustAddress table.)
2. Near the lower-right corner of the Table Format dialog, click the **Add Column** button.
3. Your work area behind the Table Format dialog is live, and will be used as you create the column definition(s). Move the Table Format dialog away from the work area, and navigate in the work area to a view of your table data. Notice that your mouse cursor is a large plus sign + when you enable the **Add Column** feature.

4. To define your first column, left-click and drag your mouse over the first position of a data string that will be extracted as a column iteration. (On the first statement, select the first B in `BILLS BICYCLES`.) The Column Information dialog opens (shown below).

Column Information

Column Information

Column Name :

Data Type :

Pattern :

Use Dynamic pattern Extract Spaces

Fixed Length

Window Coordinates

Start Row

Start Column

End Row

End Column



Caution

The definition of a column closely resembles field definition, with one major exception. You **do not** define a span of rows where the column data is to be searched. (Remember, the number of rows that will ultimately be searched for your table columns' data will be dynamically set by the detection of the table start and end markers. Select a start and end row that create an area of only one line.)

5. Enter the column name. The name must start with an alpha character and cannot contain spaces or special characters. Maximum length is 30 characters. (Enter `CustAddressLine`.)

6. Enter the pattern that will extract all iterations of the column. (Enter [A-Z0-9#%] to require that the first position of the address line consists of an alpha-numeric character, a pound sign or a percent sign, since these characters are common to the beginning of an address line.)



As with field definition, your column pattern can be a regular expression that is used to validate and extract the entire string, or it can be used to validate the string by a few characters, while extraction is managed by the Forced Fixed Length.

7. If your column will be extracted by length, click **Fixed Length** and set the column length to the desired number of characters to be extracted. (Click **Fixed Length**, and in the resulting box to the right, enter 32 as the number of characters to extract.)
8. Verify that the window coordinate set is sufficient to capture the first position of any iteration of the column, including iterations on other statements. Do this by navigating through the statement pages in the work area. (Verify that your window coordinate set is as follows:)

Start Row = 0	Start Column = 4
End Row = 1	End Column = 5

9. Make additional Formatting Specifications, if desired, by clicking on the **Formatting Specifications** button and selecting post-extraction conversions from the resulting dialog.
10. Click **OK** to close the Column Information dialog. The Column Information dialog closes, and the area defined by the column's window coordinate set is highlighted in aqua. The first column name appears in the Table Format dialog Preview area, under Field 1.



Be sure that you choose a start and end row for your columns that mimic the visual pattern in the data itself. In most cases, the relative column instances occur on the same row (such as the columns of Date, Store and Charge in a Transaction List item), therefore, the definitions for each of these columns must have the same start and end row coordinates.

11. To define each additional column of your table, repeat Steps [2-10](#).



Each additional column appears in the Preview area of the Table Format dialog in the order in which you created them. This is the order that they will be presented during live retrieval. If you need to re-order your column, simply right-click the column you want to move, and select Move Left or Move Right until the column is in the desired position.

Defining the Table Anchor

Once you have defined your markers and a table format, including name, marker selection and columns, you are ready to define or select the table anchor. You need to decide at this point whether your table will have:

- **External Anchor** — Based on a data string that iterates with column data if and only if the column data exists, or
- **Internal Anchor** — One of the columns of your table, which is guaranteed to contain data on every row where there is table data.

By default, the first column that is defined becomes the internal anchor of the table. You override this default by either:

- Marking another column as the internal anchor, or
- Defining an external anchor.

(For the *CustAddress* table, there is no external string that iterates with every iteration of column data, therefore we must use a defined column as the Internal anchor. Further, since there is only one column in the *CustAddress* table, this column is the internal anchor by default.)

To define an external anchor:

1. Open the Table Format dialog (if not already open) by right-clicking on the Table node in the Tree, and selecting **Edit**.
2. Near the lower-left corner of the Table Format dialog, click the **External Anchor** button. The Anchor Properties dialog appears.
3. Enter the pattern to validate the anchor string found.

4. Verify that the window coordinates are sufficient (column-wise) to detect the start of the anchor string.
5. **Verify that the row span of the window coordinate set matches the visual pattern of the columns' row spans.** In most cases, choose the same start and end row for the anchor as for the table columns.
6. Click **OK** to close the anchor information dialog. The area defined by the window coordinate set for the anchor is highlighted in pink in the work area.

To select an internal anchor (if the first column is inappropriate):

1. Open the Table Format dialog (if not already open) by right-clicking on the Table node in the Tree, and selecting **Edit**.
2. Locate the column which you want to be the internal anchor, in the Preview area of the Table Format dialog.
3. Right-click that column in the Preview area, and select **Mark as Anchor**.
4. Click **OK** to close the Table Format dialog and save the table definition.

Defining Groups

Just as tables can be considered groups of fields (columns) that combine together in a pattern, you can extend this thought to other, more complex visual patterns in your statements. A group is a collection of tables (or data extracted in the form of tables) that iterate together as a pattern.

For example, a telecom company may include a statement section for long distance line charges. This section may contain a title, heading, detail and subtotal (illustrated below):

LONG DISTANCE CHARGES FOR (508) 652-2876							
DATE	TIME	NUMBER	CITY	TYPE	MIN	USAGE	
01/25	11:05 AM	336 373-5990	GREENLAND	MN D	.5	.05	
01/25	11:06 AM	336 373-5990	GREENLAND	MN D	1.1	.11	
01/25	12:55 PM	336 373-5990	GREENLAND	MN D	.7	.07	
01/25	01:14 PM	306 875-3700	BOSTON	MN D	1.6	.16	
01/25	01:24 PM	336 373-5990	GREENLAND	MN D	2.3	.23	
01/25	01:31 PM	704 292-2444	MURPHY	MN D	.5	.04	
01/30	12:36 PM	704 289-4343	MURPHY	MN D	5.7	.43	
02/07	02:08 PM	828 271-4800	ASHEVILLE	MN D	.9	.09	
02/15	02:46 PM	704 226-0711	MURPHY	MN D	7.2	.54	
02/20	01:12 PM	845 732-7942	PEARLHARBR	NY D	15.7	1.08	
02/21	11:14 AM	704 226-0711	MURPHY	MN D	3.2	.24	
SUBTOTAL		11 CALLS	39.40	MINUTES	USAGE	\$3.04	

Assume that customers may have one or more phone lines. In these cases, this type of data, or the long distance charges for each phone line, will be displayed in the statement in the same pattern (an illustration of this repeating pattern follows).

1st Occurrence

LONG DISTANCE CHARGES FOR (508) 652-2876						
DATE	TIME	NUMBER	CITY	TYPE	MIN	USAGE
01/25	11:05 AM	336 373-5990	GREENLAND MN	D	.5	.05
01/25	11:06 AM	336 373-5990	GREENLAND MN	D	1.1	.11
01/25	12:55 PM	336 373-5990	GREENLAND MN	D	.7	.07
01/25	01:14 PM	306 875-3700	BOSTON MN	D	1.6	.16
01/25	01:24 PM	336 373-5990	GREENLAND MN	D	2.3	.23
01/25	01:31 PM	704 292-2444	MURPHY MN	D	.5	.04
01/30	12:36 PM	704 289-4343	MURPHY MN	D	5.7	.43
02/07	02:08 PM	828 271-4800	ASHEVILLE MN	D	.9	.09
02/15	02:46 PM	704 226-0711	MURPHY MN	D	7.2	.54
02/20	01:12 PM	845 732-7942	PEARLHARBR NY	D	15.7	1.08
02/21	11:14 AM	704 226-0711	MURPHY MN	D	3.2	.24
SUBTOTAL		11 CALLS	39.40 MINUTES	USAGE	\$3.04	

2nd Occurrence

LONG DISTANCE CHARGES FOR (508) 652-1349						
DATE	TIME	NUMBER	CITY	TYPE	MIN	USAGE
01/25	11:29 AM	704 289-9599	AVALON MN	D	1.3	.10
01/25	11:31 AM	704 289-9599	AVALON MN	D	.3	.02
01/25	11:33 AM	704 555-1212	DIR ASST	D	1.0	.99
01/25	11:34 AM	704 555-1212	DIR ASST	D	1.0	.99
01/25	11:58 AM	803 738-3769	COLUMBIA NV	D	26.9	2.39
01/31	04:03 PM	262 781-6949	BEDFORD WI	D	.3	.03
SUBTOTAL		11 CALLS	30.80 MINUTES	USAGE	\$4.52	

3rd Occurrence

LONG DISTANCE CHARGES FOR (508) 652-9756						
DATE	TIME	NUMBER	CITY	TYPE	MIN	USAGE
01/27	01:37 PM	605 388-7710	RAPID CITY SD	D	.8	.07
02/02	09:40 AM	704 226-9396	MURRAY MN	D	10.3	.77
02/20	09:42 AM	704 932-0625	KANANTA MN	D	.3	.02
02/20	12:45 PM	262 781-6949	BEDFORD WI	D	9.2	.82
SUBTOTAL		4 CALLS	20.60 MINUTES	USAGE	\$1.68	

The goal of a group definition is to define the rules to extract a pattern of data once, and allow the extraction components to reuse these rules to detect and extract all other occurrences of the pattern.

You can use the information about the pattern to:

- Build extraction rules to extract each component of the pattern, based on one occurrence of the pattern. **Each “component” of the pattern must be identified properly and extracted as a table.**
- Combine each table into a group definition, so that other occurrences of the tables will be extracted together.

Identifying the Components of a Group Pattern

Building on our illustration above, it is easy to identify the components of this group pattern. To review, this pattern contains a title, heading, some lines of detail and a line of subtotal information. Thus, you should build a table to extract each component. The tables are identified in the illustration below.

Title	LONG DISTANCE CHARGES FOR (508) 652-2876							
Heading	DATE	TIME	NUMBER	CITY	TYPE	MIN	USAGE	
Detail	01/25	11:05 AM	336 373-5990	GREENLAND	MN D	.5	.05	
	01/25	11:06 AM	336 373-5990	GREENLAND	MN D	1.1	.11	
	01/25	12:55 PM	336 373-5990	GREENLAND	MN D	.7	.07	
	01/25	01:14 PM	306 875-3700	BOSTON	MN D	1.6	.16	
	01/25	01:24 PM	336 373-5990	GREENLAND	MN D	2.3	.23	
	01/25	01:31 PM	704 292-2444	MURPHY	MN D	.5	.04	
	01/30	12:36 PM	704 289-4343	MURPHY	MN D	5.7	.43	
	02/07	02:08 PM	828 271-4800	ASHEVILLE	MN D	.9	.09	
	02/15	02:46 PM	704 226-0711	MURPHY	MN D	7.2	.54	
	02/20	01:12 PM	845 732-7942	PEARLHARBR	NY D	15.7	1.08	
	02/21	11:14 AM	704 226-0711	MURPHY	MN D	3.2	.24	
Subtotal	SUBTOTAL		11 CALLS	39.40 MINUTES	USAGE	\$3.04		

Each table is defined as an ordinary table, each requiring markers, columns and an anchor. When creating tables that will be contained in a group, keep the following important points in mind concerning the table markers:

- **Each table marker set must be contained somewhere within the group pattern**
- **Each table marker set must iterate with the group occurrences, if there is data for that table in the occurrence.**
- Think ahead in terms of the window coordinate set you define. It must be large enough to capture ALL occurrences of the marker pattern, not just on the data example you are building it on.
- Think ahead in terms of the patterns you define for your markers. They must be sufficient to detect ALL iterations of the across group occurrences.
- As with standalone tables, the markers can be defined on or above the table data

- Be sure not to use the same exact string as the start and end marker, as this will create a null searching area for the extraction components.

Table Columns

As with standalone tables, your columns may iterate, however, iteration is not necessary. Good examples of tables whose columns won't iterate in the group occurrence are headings, titles and subtotals. However, a detail table would have columns that iterate in the group. Additional important information about table columns:

- As with standalone tables, you **do not** define your columns' window coordinates to search across all possible rows. During live retrieval, the rows that are searched for column data will be dynamically defined upon the detection of the tables' start and end markers. In short, define your columns on one row only.
- As with standalone tables, the rows that you choose to define your columns on reflect the visual pattern of the column data in the statement. In fact, most table columns (like those of a transaction list or call itemization list) need to be defined on the same row, since the list items iterate together on the same rows.
- Be sure that the patterns for each column are sufficient to capture all iterations of the column, across all occurrences of the group in the statement.
- Building further on the illustrations above, this is a visual depiction of the items identified as table columns (highlighted in aqua) for the long distance line charges, and where you might define each column:

LONG DISTANCE CHARGES FOR (808) 652-2876						
DATE	TIME	NUMBER	CITY	TYPE	MIN	USAGE
01/25	11:05 AM	336 373-5990	GREENLAND MN	D	.5	.05
01/25	11:06 AM	336 373-5990	GREENLAND MN	D	1.1	.11
01/25	12:55 PM	336 373-5990	GREENLAND MN	D	.7	.07
01/25	01:14 PM	306 875-3700	BOSTON MN	D	1.6	.16
01/25	01:24 PM	336 373-5990	GREENLAND MN	D	2.3	.23
01/25	01:31 PM	704 292-2444	MURPHY MN	D	.5	.04
01/30	12:36 PM	704 289-4343	MURPHY MN	D	5.7	.43
02/07	02:08 PM	828 271-4800	ASHEVILLE MN	D	.9	.09
02/15	02:46 PM	704 226-0711	MURPHY MN	D	7.2	.54
02/20	01:12 PM	845 732-7942	PEARLHARBR NY	D	15.7	1.08
02/21	11:14 AM	704 226-0711	MURPHY MN	D	3.2	.24
SUBTOTAL		11 CALLS	39.40	MINUTES	USAGE	\$3.04



Tip

By extracting all data from this pattern and avoiding hard-coded titles, headings and labels in your HTML statement templates, you effortlessly control the display of these strings, since they will not be presented unless the relevant data exists in the statement.

To review, groups are combinations of tables that are extracted as a unit for presentment purposes. You will often use groups to extract data in a moderately complex application that contains multiple or combined statement information (such as a consolidated invoice or phone statement) and you want to keep related data elements together when they are presented.



Caution

If a table is not part of a group, then the extraction components will carry out ONLY ONE search for the table start and end markers, and thus, only extract ONE occurrence of the table. By creating a group for tables that have multiple occurrences, you order the extraction components to search for as many occurrences of the tables within the group's start and end marker.

Creating a Group

The first step in creating a group is to identify the components of the group. Then, for each component, or table, you follow the steps to create a table.

Select the components of the group. These components can be:

- Tables that you have defined and are part of the group pattern, or

- Inner groups that you have defined, that iterate within the group pattern.

Defining a Group

To illustrate each step, we will define a group to extract all occurrences of the *Local Line Detail* sections of a National Wireless statement.

The goal is to extract all of the local line detail information for a phone line in one unit. This will allow us to define presentment for all of the information in the Composer with ease.

You must have identified and built the tables (13 in all) to extract the following *Local Line Detail* information:

Local Line Phone Number *Features* section, containing:

- Heading
- Detail Breakdown

Subtotal *Local Usage* section, containing:

- Heading
- Subheading
- Detail Breakdown
- Subtotal

Itemized Calls section, containing:

- Heading
- Subheading
- Detail Breakdown
- Subtotal

Local Line Subtotal

Each of the above types of information may or may not occur with the group, which we will call `LocalLineDetailgroup`. When you recognize a pattern in your statements as described above, you must:

- Build a separate table to extract each “type” of information. This will allow us complete freedom in formatting each of these items uniquely in the Composer.
- Consider any possible item that may appear in the group pattern, and build a table to extract it.

Sometimes, one or more of the items listed above (such as the Itemized Calls section), will not occur with the group pattern on a particular statement. Regardless, you still need to include these items in the group, since they may occur with the group pattern.



The following rules must be observed when creating field, table, group, and marker names:

The names are not sensitive to case, but they must be fashioned from ONLY alphanumeric characters (no special characters are permitted).

The names must start with a letter, and cannot contain any spaces.

The names cannot contain any of the following reserved words alone: Time, Date, INT, Double, Currency. (For example: `Date` is not permitted, but `DueDate` is allowed.)



Field, table and group names must be unique (marker names are exempt from this requirement).

Keep the names short, e.g. `TotAmtDue`. Be descriptive, but use abbreviations instead of long names.

Defining a Group Start Marker

During live retrieval of a statement, a group’s start marker is used to inform the extraction components of the row on which to start searching for group data. You start the process of defining the start marker by first identifying a data string that appears on or above the row where group data begins. Start markers are mandatory, however, you may define more than one start marker for a group.

To define a group start marker:

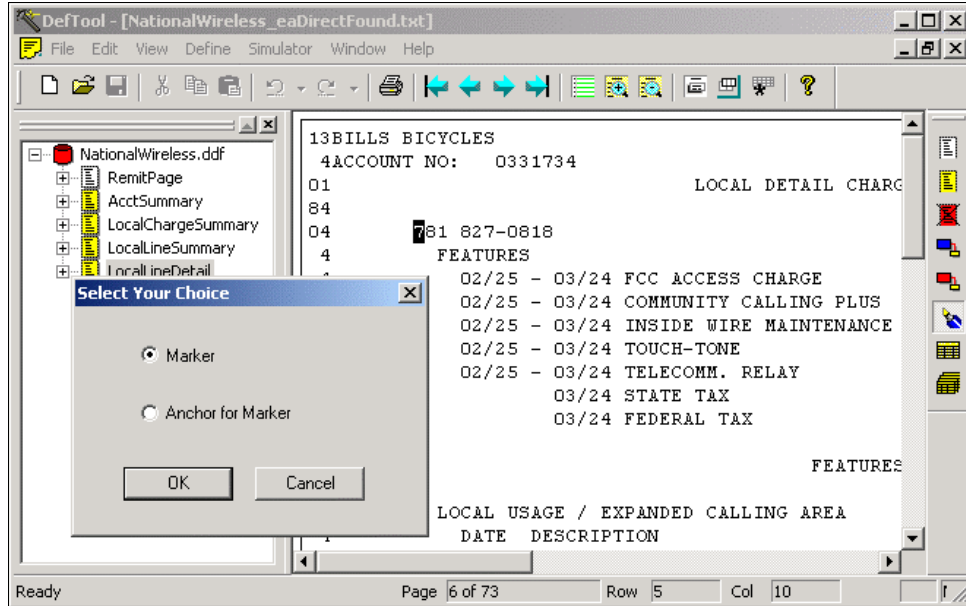
1. Navigate to a LocalLineDetail page of any National Wireless statement in the DefTool work area.
2. From the Definition Tool Bar, select the Marker  icon.
3. Identify and locate the data string that indicates the start row of the group's searching area. This is where the extraction components will begin using the group's table and inner group definitions (markers, columns, etc.) to extract your new group's data. (Since there is no non-data string that occurs on or above every LocalLineDetail group occurrence, use the phone number that always occurs at the start of the group. Locate any phone number between columns 9-10.)
4. Select the Marker  icon from the Definition Tool Bar.

Tip

You can easily find out what coordinates your mouse is on. If you move your mouse into the work area, and then look at the Status Bar in the bottom pane of the screen, you can see the row and column coordinates of the cursor's location.

5. Move your mouse into the work area and left-click drag and drop your mouse over the first position of the string identified in Step 2. (Select the first numeric character in the phone number as shown below.) The Select Your Choice dialog opens.

Defining Data Extraction Rules with DefTool



Tip

It is sufficient at this point to just select the first character (shown above). You must edit the window coordinates later in the Marker Information dialog to be large enough to capture other Phone Numbers.

- From here, decide if you are creating a marker or an anchor for the marker (which you would create immediately after the anchor). (Accept the default of marker, and click OK.) The Marker Information dialog opens.
- Enter the marker end. Keep names short but descriptive, as in TotAmtDue. Field, table, group, and marker names must start with a letter, use alphanumeric characters only (no special characters), and cannot contain spaces. Do not use the following reserved words in a name: Time, Date, INT, Double, Currency (“Date” is not permitted but “DueDate: is). Field, table and group names (but not marker names) must be unique. Maximum length is 30 characters. Names are not case-sensitive. (Enter LocalLineDetailgroup.)

8. Enter the pattern to be searched for as the start marker. Be sure that your pattern is sufficient to detect this start marker on all statements. (Enter a regular expression to extract the phone number as the pattern. You should have already created this regular expression for use in other tables and columns. The pattern to extract the phone number is:)


```
[0-9] [0-9] [0-9] [0-9] [0-9] [0-9] \- [0-9] [0-9] [0-9] [0-9]
or \n\n\n \n\n\n\-\n\n\n\n
```
9. When creating a marker, the Start Marker radio button is selected by default. Accept this default for the start marker definition. (Accept the default radio selection of Start Marker.)
10. The start marker of a group can be located on or above the rows where group data exists (including markers of tables and inner groups). If your group start marker string is on the same row where there may exist table data, and/or used as a marker for an inner table or group, then enable the **Include Marker Line** option. (The start marker string, or the phone number, is part of the group data, and is also used as the start marker for tables within the group. Click to enable **Include Marker Line**.)
11. The start marker string can be “re-used” as the end marker, meaning that you can define the end marker on the same exact string in the same location, with the intent that it be detected on a subsequent page. If you want to use the same string for the start and end marker and NOT create a null searching area for the group data, then enable **Look for the Start Marker on the Same Row** in the Start Marker Information dialog. (Since your start marker and end marker will be different for the LocalLineDetailGroup, **do not** enable “Look for the Start Marker on the Same Row.”)
12. If you are certain your start marker’s row contains no other data than the start marker string itself, enable the **Select the Line Only Containing This Pattern** option. (Since your start marker row does contain other data, **do not** enable “Select the Line Only Containing This Pattern.”)

Caution "Select the Line Only Containing This Pattern" directs the extraction components to search the entire row for verification that no data besides the start marker string exists on that row. To use this option, you must be certain that no other data will ever appear on the start marker row.

13. You can use a row with no data on it as your start marker, by enabling the **Use Blank Line as Marker** option. (For the National Wireless application, **do not** enable "Use a Blank Line as Marker.")

Caution To use a blank line as a marker, you must be certain that a blank row will always indicate the existence of the table, and that there will always be a blank row preceding the table.

14. Select the **page style** on which the start marker string resides. (Select `LocalLineDetail` from the Page Style dropdown list.)
15. Verify that your window coordinates are sufficient to detect your group start marker on all statements. (Since the phone number can appear near the top or bottom of the page, your row coordinates will span much of the page. Verify that your window coordinates are:)

Start Row = 2

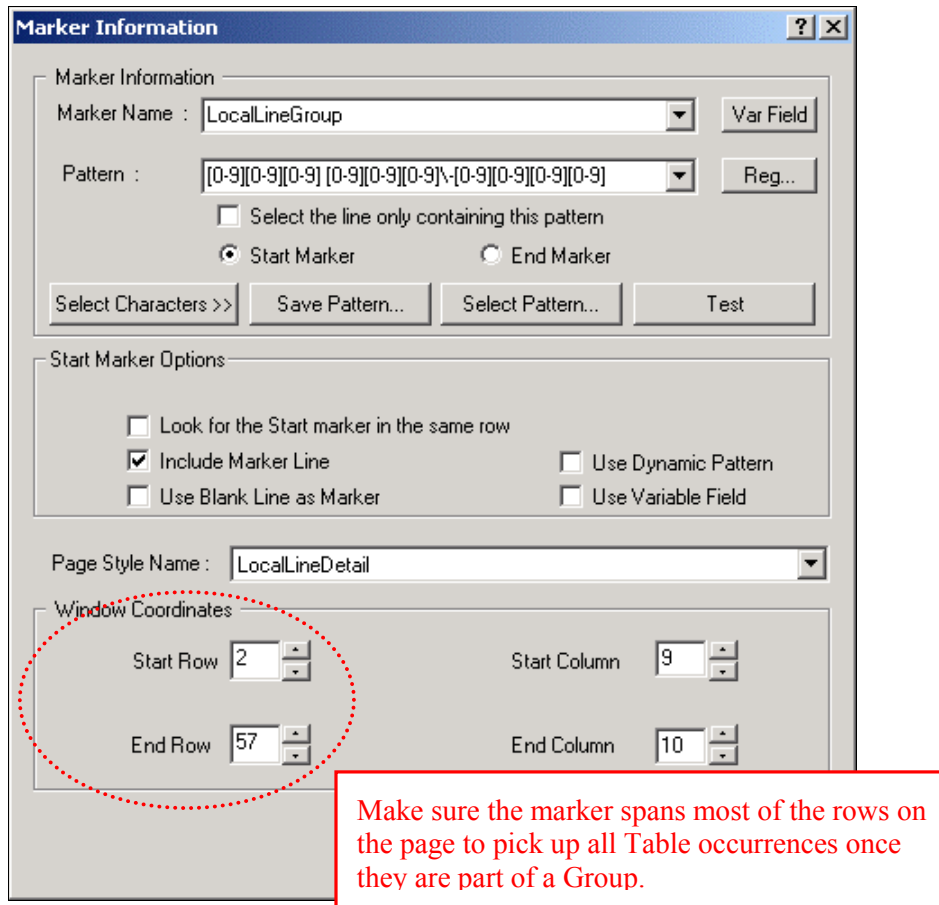
Start Column = 9

End Row = 57

End Column = 10

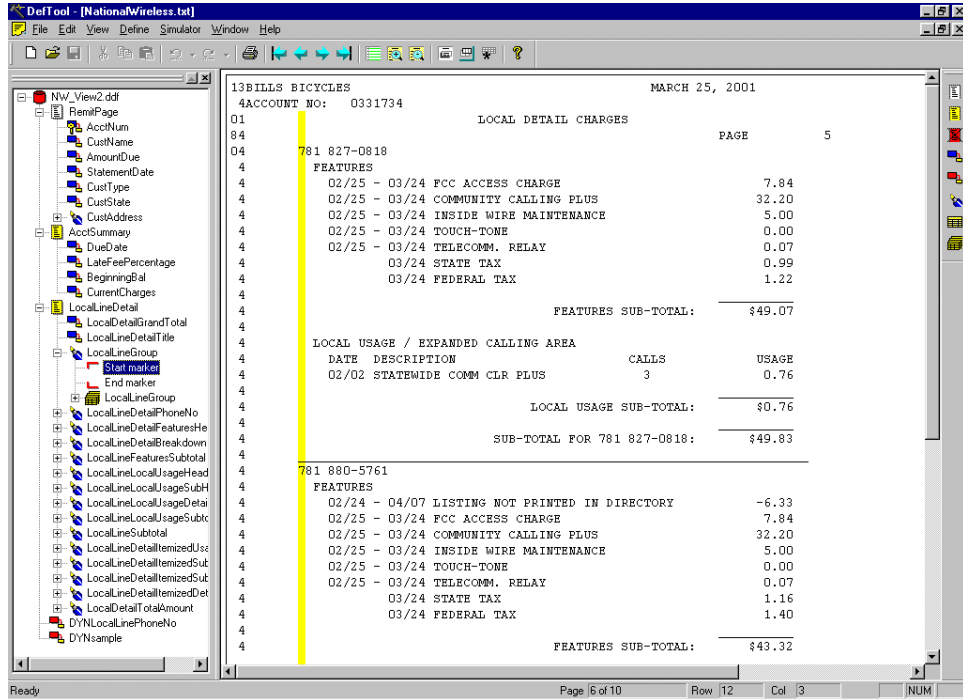
Tip You can create your group rules so that only certain occurrences of a group are extracted and presented in a statement, which is driven by the choice of the recipient during statement live retrieval. To do this, you insert a dynamic field in the pattern, which dynamically sets a pattern to a group marker, therefore requiring that only data where group marker = X will be presented. Dynamic pattern matching for selective group extraction is discussed in "[Setting Dynamic Pattern Markers.](#)"

The completed Marker Information dialog will appear as follows:



- Click **OK** to close the Marker Information dialog and save your group start marker definition. In the work area, notice that a yellow area defined by your marker's window coordinate set appears over your start marker string. Also, in the Tree, a Start Marker node has been added to the tree. This is shown below.

Defining Data Extraction Rules with DefTool




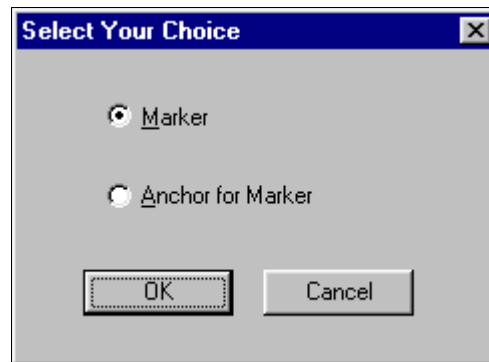
Defining a Group End Marker

During live retrieval of a statement, a group's end marker is used to inform the extraction components of the row on which to stop searching for group data. You start the process of defining the end marker similarly to a start marker, by first identifying a data string that appears on or below the row where table data is expected to end.

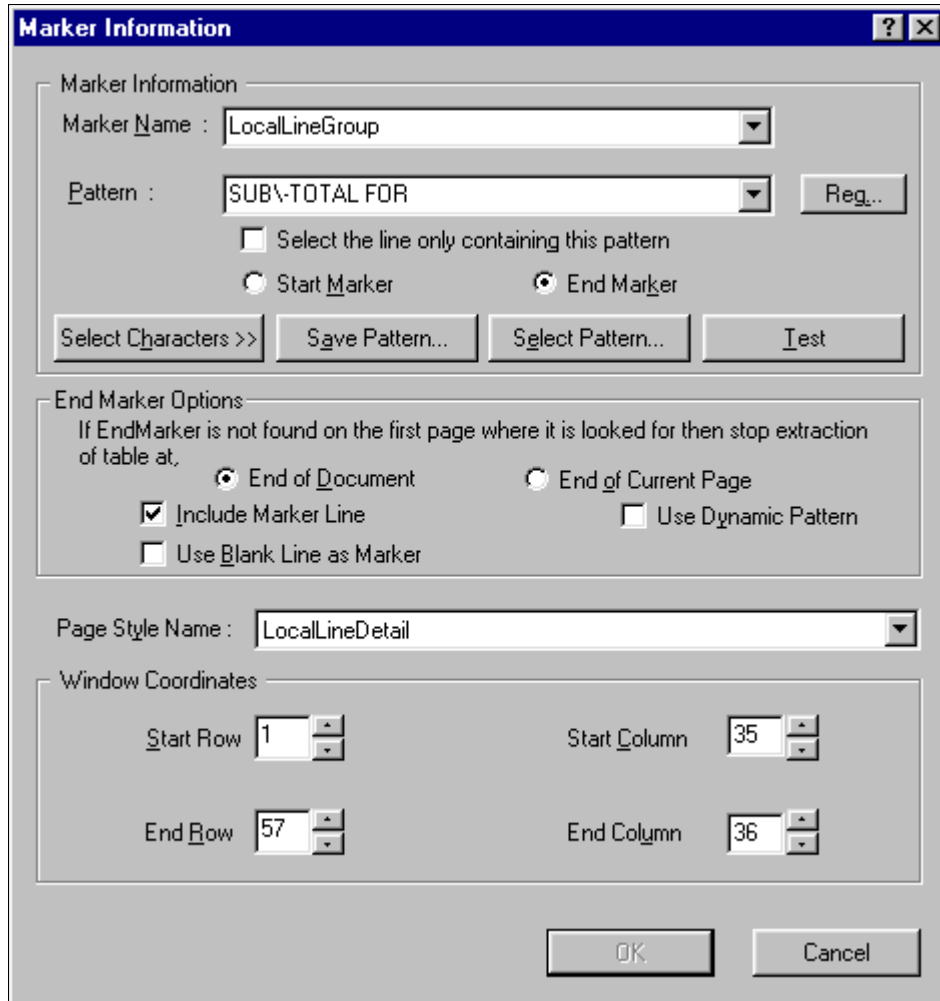
To define a group end marker:

1. Identify and locate the data string that indicates the end row of the group's searching area. This is where the extraction components will stop looking for column data. (The string *SUB-TOTAL FOR*, starting between columns 35-36, is the best indicator of the end of the LocalLineDetail for a Phone Number (or the group occurrence)).

2. Select the Marker  icon from the Definition Tool Bar .
3. Move your mouse into the work area and left-click drag and drop your mouse over the first position of the string identified in Step 1. (Select the first character S of SUB-TOTAL FOR between columns 35-36.) The Select Your Choice dialog opens.



4. From here, you decide if you are creating a marker or an anchor for the marker (which you would create immediately after the anchor). (Accept the default of `Marker`, and click `OK`.) The Marker Information dialog opens (shown below).



5. Enter the marker end. If you have already created and named the start marker that will be used with this new end marker, then you already have the end marker name. Select this name from the Marker Name dropdown list or enter it manually. Names must not contain spaces or special characters, must start with an alpha character, and cannot exceed 30 characters in length. (Select LocalLineDetailgroup.)

6. Enter the pattern to be searched for as the end marker. Be sure that your pattern is sufficient to detect this start marker on all statements. (Enter `SUB\ -TOTAL FOR` as the pattern.)
7. When creating any marker, the **Start Marker** radio button is selected by default. Be sure to select the **End Marker** radio button before closing the Marker Information dialog. (Click to select the End Marker radio button.)
8. The end marker of a group can be located on or below the rows where group data exists. If your group end marker string is on the same row where there may exist table data, then enable the **Include Marker Line** option. (The end marker string, *SUB-TOTAL FOR*, does appear on the same row as group data, and is used as an end marker for some inner tables. Click to enable the Include Marker Line option.)
9. If you are certain that your end marker's row contains no other data than the end marker string itself, then enable the **Select the Line Only Containing This Pattern** option. (Since your end marker row does contain other data, **do not** enable Select the Line Only Containing This Pattern.)

**Caution**

Select the Line Only Containing This Pattern will direct the extraction components to search the entire row for verification that no data besides the end marker string exists on that row. To use this option, you must be certain that no other data will ever appear on the end marker row.

10. You can use a row with no data on it as your end marker, by enabling the **Use Blank Line as Marker** option. (**Do not** enable Use a Blank Line as Marker.)

**Caution**

To **Use a Blank Line as a Marker**, you must be certain that a blank row will always indicate the existence of the group, and that there will always be a blank row preceding the group.

11. Select the page style on which the end marker string resides. (Select `LocalLineDetail` from the Page Style dropdown list. Verify that your window coordinates are:)

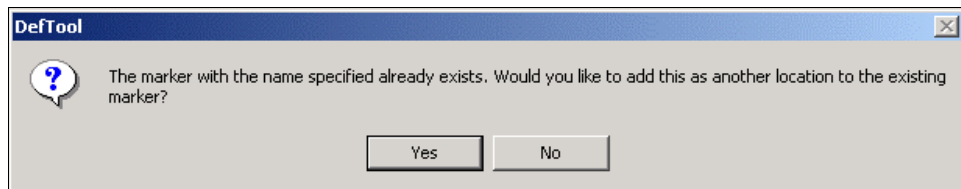
Start Row = 1	Start Column = 35
End Row = 57	End Column = 36



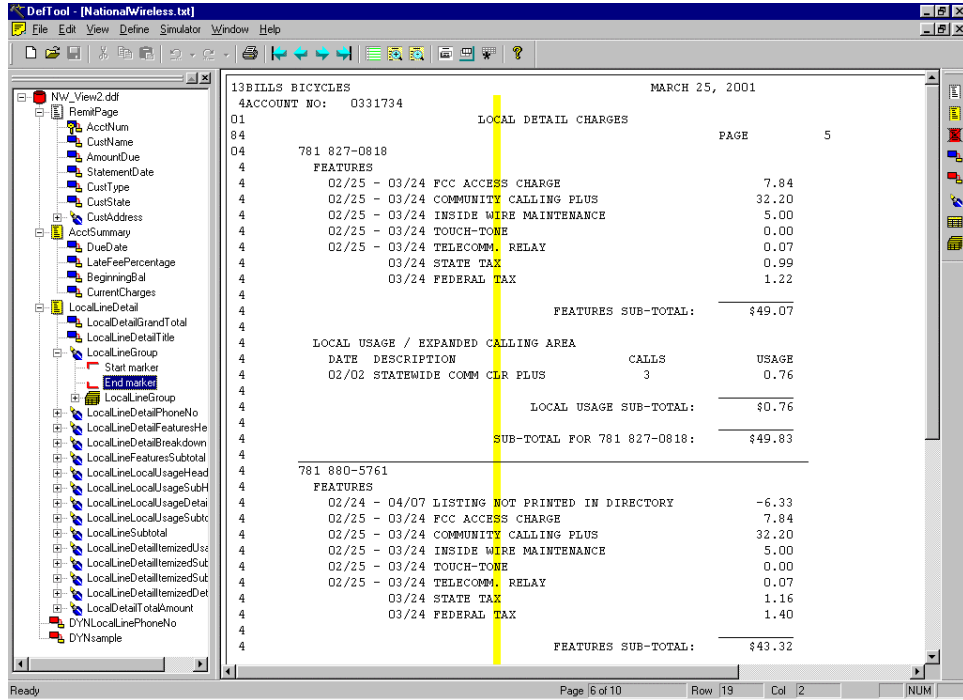
Tip

Your end marker can reside on a different page style than the start marker it is matched with.

12. Click **OK** to close the marker Information dialog and save your end marker definition. The marker Information dialog closes, and a dialog opens, alerting you to the creation of a marker pair (shown below).




13. Click **Yes** to create a marker pair with your group end marker and group start marker. In the work area, notice that a yellow area defined by your group end marker's window coordinate set appears over your group end marker string, and also spans most rows on the page. Also, in the Tree, an End Marker node has been added, creating the Marker pair in the tree (shown in the figure that follows).

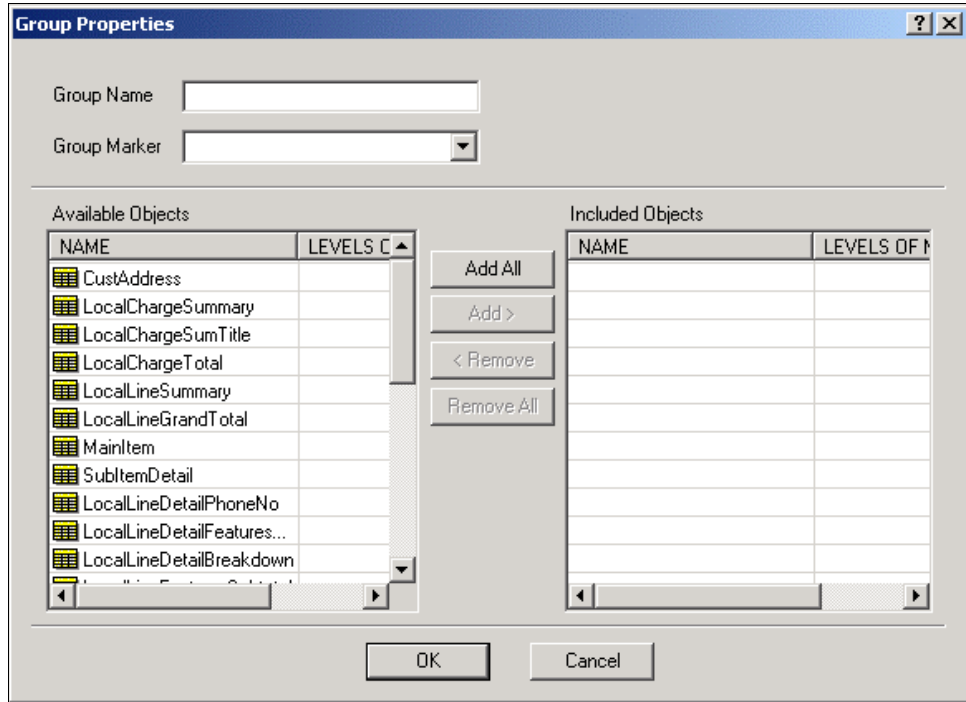


Defining Group Properties

The next task is to define the properties of the group. This includes naming the group, selecting the marker pair, and selecting the data items that will be searched for and extracted as part of the group. To review, the data items that you can include as part of a group are tables and inner groups. You cannot include fields as part of a group. You must create the marker pair before you can define the group properties.

To define group properties:

1. From the Definition Tool Bar, click the Group  icon. The Group Properties dialog opens.



2. In the Group Properties dialog, enter the group name in the corresponding field. (Enter `LocalLineDetailGroup`.)



Just as for tables, it is a good idea to give your group the same name as the marker pair that will be used with it. This will help you organize your groups during presentment definition in the Composer, and also aid in DDF troubleshooting, if necessary.

3. Select the group marker from the corresponding dropdown list, to attach the marker pair to your new group. (Select `LocalLineDetailGroup` from the group marker dropdown list.)

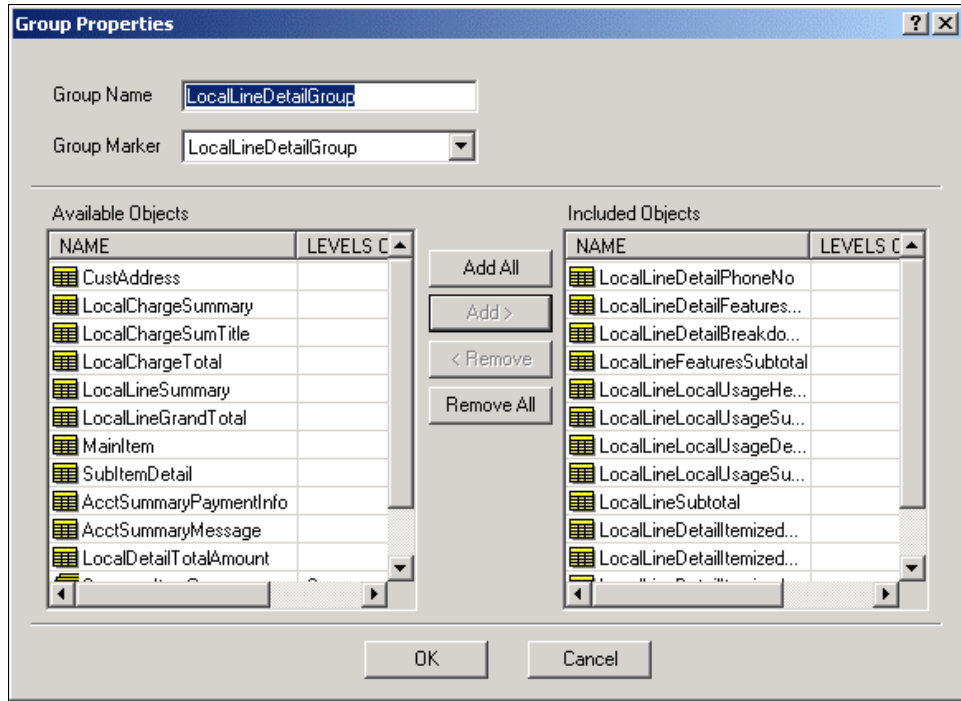
4. Your next step is to define the group's objects. All of the tables and groups that you have created in your DDF will be displayed in the Available Objects section of the Group Properties dialog. Select the object that you want to include in the group, from the Available Objects section, and click **Add >**. The object will appear under the Included Objects section. (Select the following tables that you have built to extract items from the Local Line Detail statement sections:)

LocalLineDetailPhoneNo
LocalLineDetailFeaturesHead
LocalLineDetailFeaturesBreakdown
LocalLineDetailFeaturesSubtotal
LocalLineDetailLocalUsgHead
LocalLineDetailLocalUsgSubhead
LocalLineDetailLocalUsgBreakdn
LocalLineDetailLocalUsgSubtotal
LocalLineDetailItemizedHead
LocalLineDetailItemizedSubhead
LocalLineDetailItemizedBreakdn
LocalLineDetailItemizedSubtotal
LocalLineDetailSubtotal

 **Tip**

You can also double-click an object in the **Group Properties Available Objects** list to add it to the group. To add ALL available tables and groups to your new group, click **Add All >**.

Your group properties will be defined as shown.



5. Click **OK** to close the Group Properties dialog and save the new group settings.

About available objects and nested groups

All tables and groups will appear in the Available Objects list. This can be confusing if you are creating nested groups.



If creating nested groups, **do not** select tables that are part of a group.

For example, if you want to build a nested group to include three tables and one inner group (containing five tables), the three tables, the inner group, **AND** the five tables of the inner group appear in Available Objects. The correct selection will be the three tables and the innergroup. **Do not** select the five tables of the inner group, because they are already grouped.

Presenting Sub-Account Data

The eaDirect sub-account indexing feature lets you create a view specifically for presenting sub-account data. Use sub-account indexing if you need to:

- Quickly access sub-account data in a data input file for presentment in a view.
- Include sub-account information in a history list, or “hit list.”

For example, you could define the cell phone detail sections of a large B2B telecommunications statement as sub-accounts, speeding Web access to line detail on request.

To enable the sub-account indexing feature, you must define the sub-account data area of an input file as a group, then define a group field on a piece of data that uniquely identifies the sub-account (or sub-document), called a subkey. Just as a primary key identifies a document, the subkey identifies each sub-document group.

You can also define additional group fields for indexing and presentment. eaDirect indexes the group fields that belong to the sub-account group as part of the regular Indexer production job, storing exact offsets and other information about the start and end of each sub-account in the database. This index data enables you to include sub-accounts in a history list and to click a link to quickly access sub-account data in a view (without having to scan the entire statement to find the group information).

You must create a separate view (DDF/ALF pair) for presenting sub-account information to users. You can also define more than one sub-account for an application; create one indexing DDF to use for all the sub-accounts and a separate ALF for each sub-account view.

The larger document that a sub-document belongs to is called the root, or parent, document. One parent document can consist of many instances of one or more sub-documents (sometimes referred to as the “children” of the parent document).

The following section describes the general process you must follow to set up sub-document indexing in your application.

Sample sub-document indexing application

eaDirect includes the following sample application view files to demonstrate sub-document indexing:

- *EDCSbd\samples\NW_Subdocument\NW_Subdocument.ddf*
- *EDCSbd\samples\NW_Subdocument\NW_Subdocument.alf*

Use the *NationalWireless.txt* data file with this application.

Setting up Sub-Account Indexing

This section describes the general process required to implement sub-document indexing in a new or existing eaDirect application. You must tailor these procedures to the particular data in your input file and the views you need to present. Include these steps in the overall plan, design, and implementation of your eaDirect application.

Note that the procedure to implement sub-document indexing in an *existing* eaDirect application is different from creating the feature in a new application. For an existing eaDirect application, you can use your existing DDF, however you must create a separate ALF for presenting the sub-document view.

To implement sub-account indexing in a *new* eaDirect application:

1. Using DefTool, create a DDF specifically to index and extract data for the sub-account view. See “Creating a DDF for Indexing and Sub-Account Presentment” below along with the instructions in this guide for details about creating a DDF.

2. Define the doc style and the **primary key** in the indexing DDF, along with any other **pure fields you need to index** for your application. (You must index fields to make them available for inclusion in the history list.)
3. In this DDF, define the sub-document data as a **group**. (This includes creating group markers, creating the group, and assigning the group markers to a group.)
4. Identify a piece of data that uniquely identifies each sub-account in the data input file, such as sub-account number, and define this as the **subkey** group field for the sub-account. The subkey must be located within the sub-account group.

eaDirect uses the subkey to locate the requested sub-document during dynamic retrieval. When the Indexer job indexes the subkey, it stores the exact offsets to each sub-document. By storing offsets for each sub-document, eaDirect has the information it needs to directly access any sub-document in the input file, eliminating the need to search the entire data file to identify the sub-document boundary.

5. Define any **group fields** you want to index, such as the name or other information associated with the sub-account number. Define any group fields for indexing within the sub-account parent group.
6. In Composer, create a new Application Logic File (ALF), selecting the **Use sub-document** option to identify it as a sub-document view. Map the tables and fields to the ALF's default or parent template, which is essentially the group template. (See "For a Sub-Account View" on page 236 for details.)
7. When you configure the Indexer job using the eaDirect Command Center, publish the indexing DDF with the Indexer job. eaDirect automatically indexes all group fields defined in the parent group of a sub-account in the DDF. See the *eaDirect Production Guide* for details about fields you need to index in your application and other Indexer job configuration details.

To implement sub-account indexing in an existing eaDirect application:

1. Make sure that you have defined the sub-account data as a **group** in your existing DDF.


2. Identify a piece of data that uniquely identifies the sub-account, such as a sub-account number. Right-click on the group name in the Definition Tree and select Edit from the right-click menu. On the Group Properties dialog, click Create. Define the field you want to use as a **subkey**.
3. On the Group Properties dialog, click Create again to define **any additional group fields** you want to index and/or include in the view, such as the name associated with the sub-account number. Define any group fields that need to be indexed within the sub-account parent group.
4. In Composer, you **must** define a new ALF for presenting the view of sub-account data, selecting the “Use sub-document” option to identify it as a sub-document view. (See “For a Sub-Account View” on page 236 for details.)
5. In the Command Center, publish the updated DDF.
6. Run the Indexer job. eaDirect automatically indexes all group fields defined in the parent group of the sub-account in the DDF.


Creating a DDF for Indexing and Sub-Account Presentment

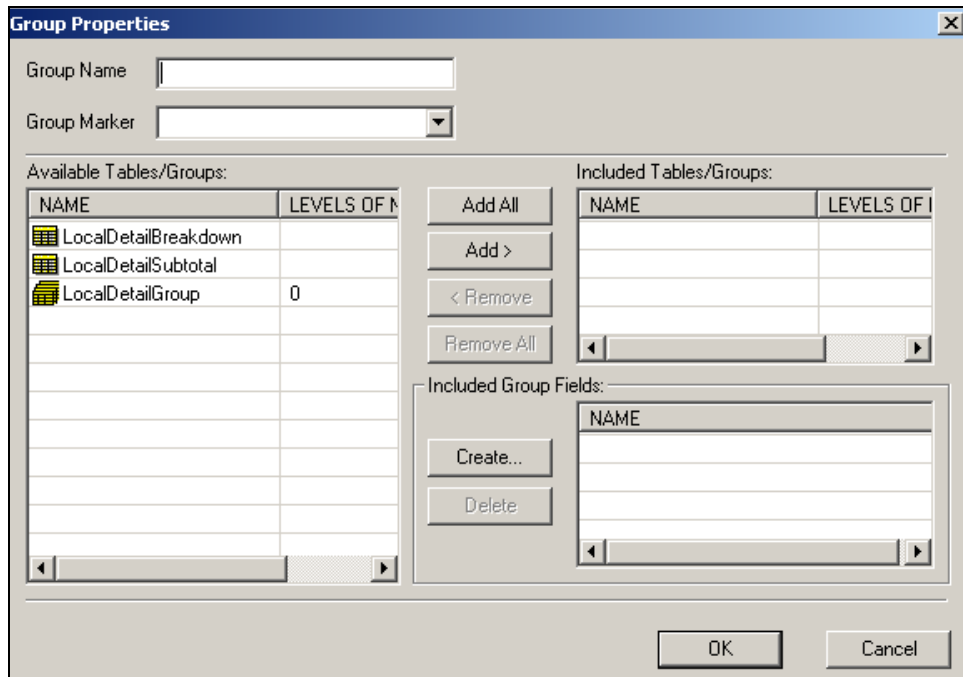
Follow the procedures in this section if you are creating a *new* eaDirect application and need to implement sub-account indexing. In the DDF you create for sub-account indexing, you must define all the fields that you need to index for the application.

To create a new DDF for indexing and sub-account presentment in DefTool:

1. In DefTool, select **File>New**. DefTool begins the Data Configuration Wizard. Complete the wizard and follow the procedures described in this guide for creating a document style, primary key, and any additional page styles.

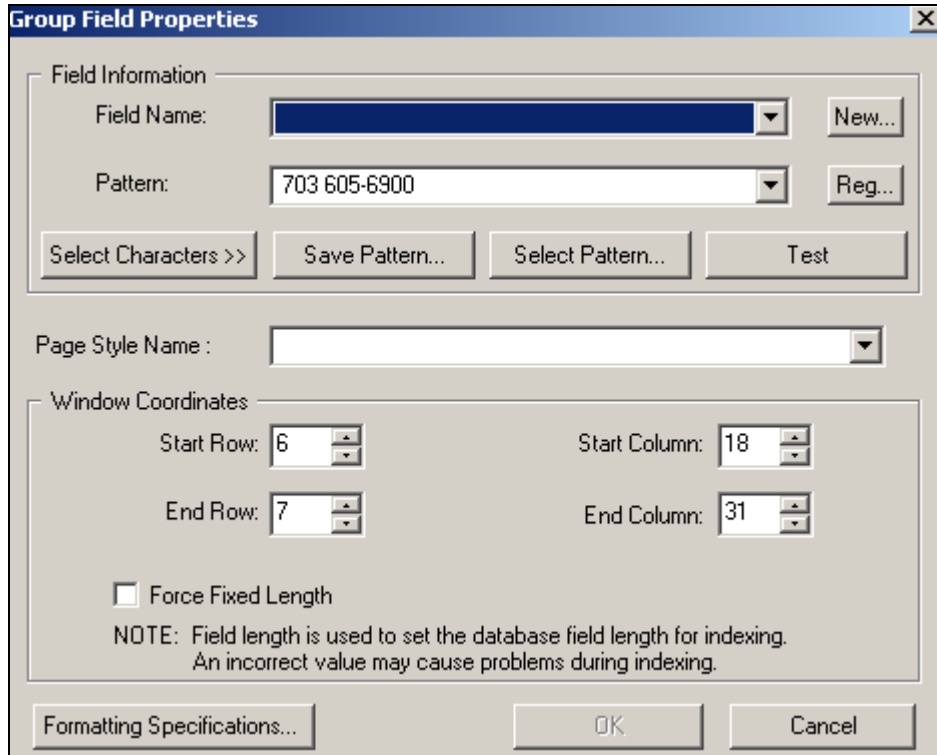
2. Click the New Marker icon  and define the start and end markers for the sub-account group. You define the start and end markers for the group indicating the group offsets – the exact location in the data input file where eaDirect should begin searching for the group data. Once you define a group, it appears in the Definition Tree on the left of the DefTool window for the DDF.

3. Click the Group icon  to display the Group Properties dialog. Define the properties of the sub-account group. Specify the group name, associated markers, and select the tables in the group.

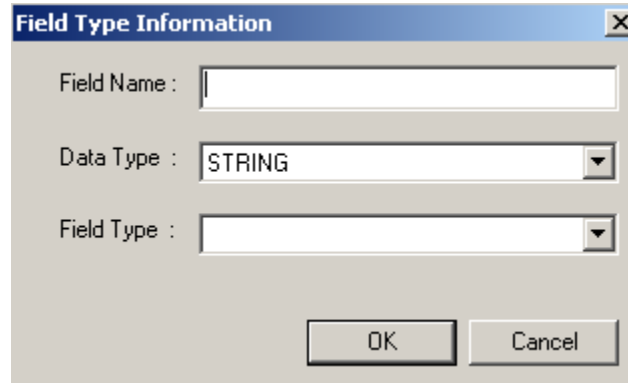


4. Create a subkey for the sub-documents: identify a piece of data in the data input file that uniquely identifies each sub-account. The subkey and any anchor you specify must be between the group's start and end markers.

5. On the Group Properties dialog, click **Create**. Use the left mouse button to drag the crosshair selector box around the data you want to use for the subkey group field. At the Select Your Choice dialog, select **Field** and click **OK**. DefTool displays the Group Field Properties dialog:




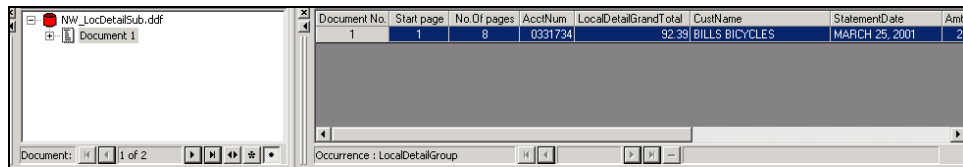
6. Click **New**. DefTool displays the Field Type Information dialog. Specify a name for the subkey field, select the data type, and select **Sub key** from the Field Type drop-down list.



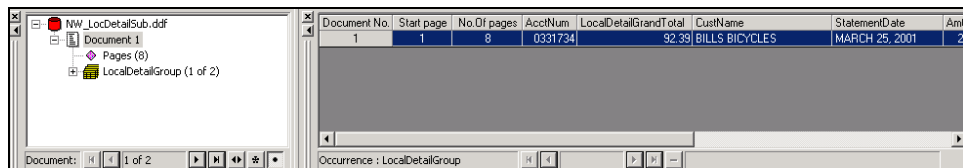
7. Click **OK**. Continue to define the subkey group field extraction rules (properties) including the page style to associate with the group field, and click **OK** when you are done. DefTool displays the new subkey group field under the group and its associated markers in the Definition Tree; note that the subkey group field icon looks different from the regular group field icon.
8. On the Group Properties dialog, define any additional group fields you want to index within the sub-account parent group. You can also define group fields in nested groups. You can define more than one group field for a group, but you must define only one as the subkey. You can use a group field in multiple locations.
9. Use the DefTool Simulator, described below, to test extraction of subkeys and group table data from your data input file. To edit data extraction rules, right-click a group or on a marker in the Definition Tree and select **E*dit*** from the right-click menu. To edit a subkey or other group field extraction rules, see “Modifying a Group Field or Subkey Properties.”
10. Save the DDF.

To simulate extraction of sub-account group data and subkeys in DefTool:

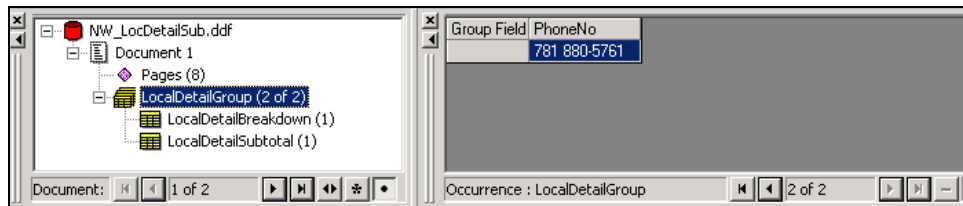
1. Click the Extractor Simulation icon  in the Simulation toolbar or select **Simulator>Extractor Simulation**. The Simulation pane appears at the bottom of the DefTool main window. Nodes shown in the tree on the left represent each document detected in the input file. On the right, Simulator displays the document boundaries and extracted fields.



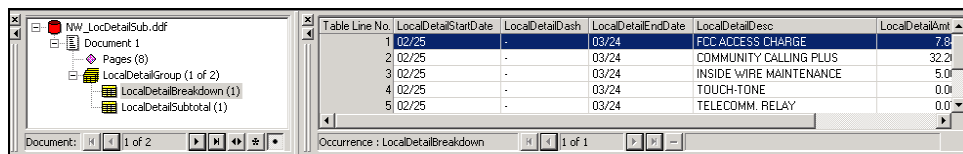
2. Click the document to expand the list of pages, groups, and fields.

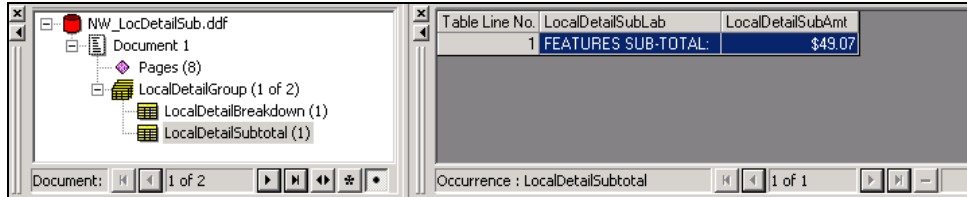



3. Click the sub-document group to display the subkey or group fields for the first occurrence.

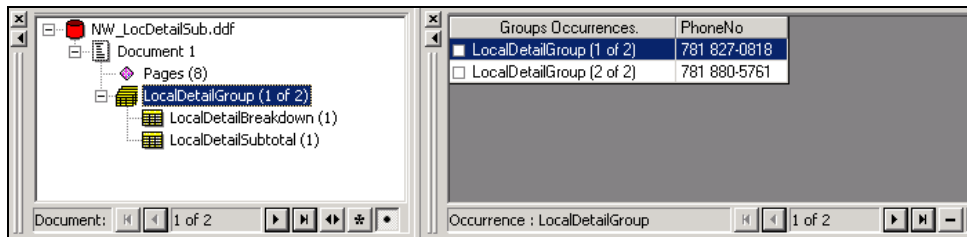


4. To display group data extraction, click the table.

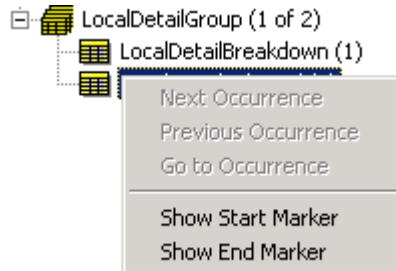





- To test the extraction of all subkeys, click the Show All Tables icon , click the Document to expand the list of fields, tables, and groups, then click the sub-document data group. DefTool displays the extracted subkey and group fields for each group occurrence in the document.



- You can use all the existing features of the DefTool Simulator to test the groups, tables, and markers. For example, you can right-click on a group or table to display a menu of options:



- As with any groups, tables, or fields, you can choose to selectively simulate a sub-document group (select **Simulator>Simulator Settings** and the Group tab or click the Simulator Settings icon ). By default, DefTool Simulates all tables, group fields, and subkeys associated with the group.

Modifying a Group Field or Subkey Properties

You can edit the properties of a group field, including a subkey, after you have created it. For example, the simulation process may reveal a problem that requires you to edit the group field extraction rules, such as the field pattern or window coordinates.

To edit the properties of a group field:

1. With the DDF open in DefTool, expand the Definition Tree to expose the group field.
2. Right-click on the group field (can be a subkey), and select **E*dit*** from the right-click menu. The Group Field Properties Dialog appears.
3. Make your edits using the Group Field Properties Dialog and click **O*K***.

Deleting a Group Field or Subkey

You can delete a group field, including subkeys, from a DDF. If a group field or subkey has multiple locations, you can select and delete by location (similar to how fields work).

To delete a group field or subkey:

1. With the DDF open in DefTool, expand the Definition Tree to expose the group field.
2. Right-click on the group field (can be a subkey), and select **D*el*ete** from the right-click menu. DefTool asks if you're sure you want to delete the field.
3. Click **Y*es***. DefTool deletes the group field.

To delete multiple group fields:

1. With the DDF open in DefTool, expand the Definition Tree to expose the group.


2. Right-click the group and select **E****d****i****t** from the right-click menu. DefTool displays the Group Properties dialog.
3. Under the Included Group Fields, select a field to delete and click **D****e****l****e****t****e**. DefTool asks if you're sure you want to delete the field.
4. Click **y****e****s** to delete the field. DefTool deletes the group field.
5. Continue deleting group fields as necessary.
6. Click **o****k**.

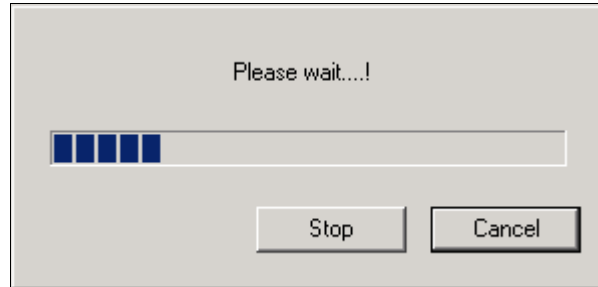
The Extraction Simulator

The Extraction Simulator utility is built into the DefTool to allow you to test your DDF extraction rules as you are defining them, without having to go through the process of indexing the data source by the production components. This utility prevents the “blind design” problems often associated with application development. After you have defined the Document Style and Primary Key Field, you can use the Extraction Simulator at any time while you are working in the DefTool.

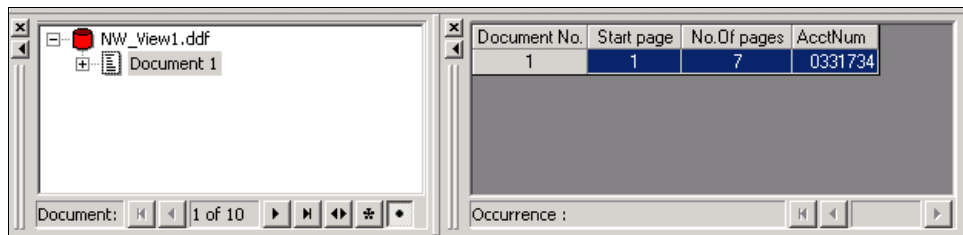
Using the Simulator for the First Time

By default, the Extraction Simulator will test all of the definitions you have made simultaneously, every time you run it. You do not need to make any configurations to use the Extraction Simulator for complete simulation.

- Locate the **S****i****m****u****l****a****t****o****r**  icon in the Simulation Tool Bar or select **S****i****m****u****l****a****t****o****r**>**E****x****t****r****a****c****t****i****o****n** **S****i****m****u****l****a****t****o****r**. (Click the **E****x****t****r****a****c****t****o****r** **S****i****m****u****l****a****t****o****r** Icon to test your Document Style and Primary Key field definitions.) A Please Wait...! prompt appears (shown below):

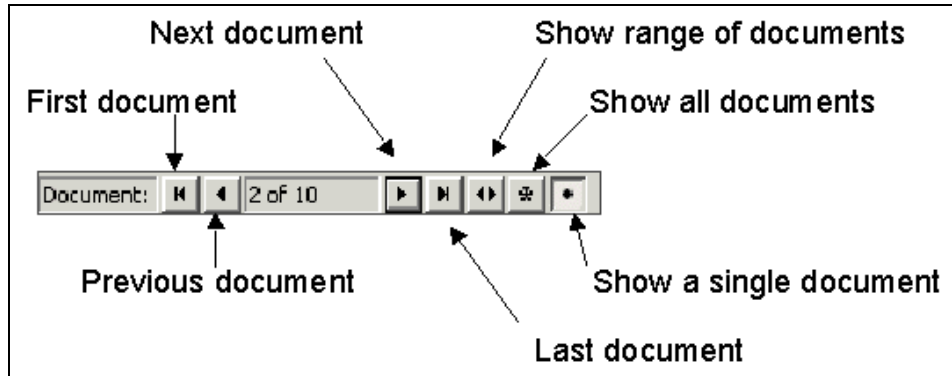



A second horizontal pane will appear in the DefTool main console. On the left side, the Tree contains nodes representing each statement (referred to as documents here) detected in the data source, and on the right side, a table displays the document number, start page, number of pages and fields extracted for each statement detected in the data source (shown below).

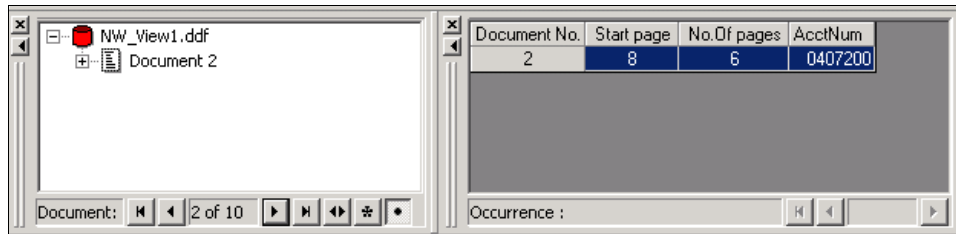



Navigating in the Simulator

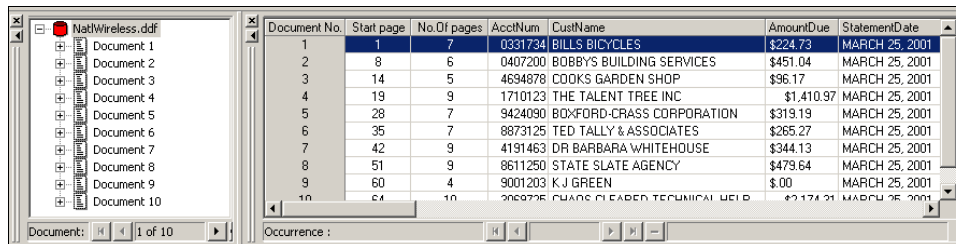
By default, the Simulator view only displays the first statement detected. The Simulation View contains a Navigational Bar, located below the Simulation Tree (shown below). The navigational tools can modify the Simulation view to show a subsequent statement, include a range of statements, or display all statements in the data source.




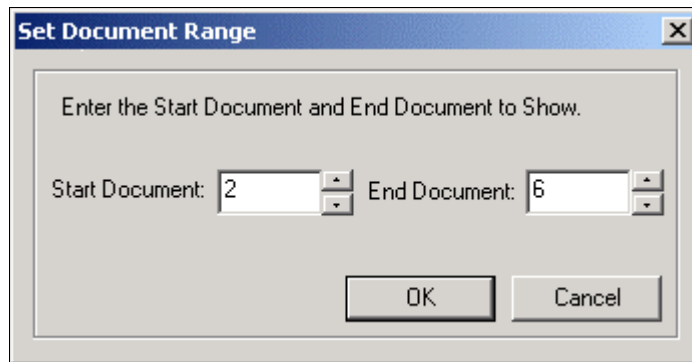
- Click the **Next Document**  icon to verify the detection of Statement 2 and its Primary Key field extraction (0407200). The Tree View displays a node for Document 2, and the Simulation View displays the document number, start page, number of pages and (AcctNum) Primary Key field extraction.



- Click the **Show All Documents**  icon. If you have properly defined your Document Style and Primary Key field, all statements in the data source will be represented in both the Tree View and Simulation View (shown below).



- Click the **Show Range of Documents**  icon. The Set Document Range dialog appears. Adjust the start and end documents as desired (shown below). Since this is a live dialog, you need to set the end document first. The Simulation View will display the extraction for the document range you specified.



At this point in the DDF creation process, the Simulator is useful for determining if the statements will be detected. The Document Style and Primary Key field are used together to detect a statement in your data sources. If both of these settings are not properly defined, and therefore not detecting particular statements, then you will notice this immediately by the absence of the statements in the Simulation View. The Simulation View would consider these statements as part of the previous statements. This is similar to what would happen in a production environment. If your Simulation View doesn't appear as expected, you will need to edit your definitions and rerun the Simulator.




Tip

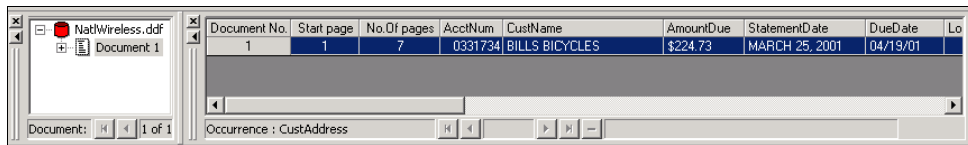
Information in the Simulation View is only as current as the last time you ran Extractor Simulation.

Simulating Field Extractions

The Extraction Simulator will allow you to test your rules for fields, tables and groups before you deploy the application. You can perform complete or selective extraction simulation on any of the rules you have created.

By default, the Extraction Simulator will test all of the rules (fields, tables and groups) that you have built in your DDF. All the fields that you have defined will appear in the main Simulator View after you have run the Simulator.

Click the **Simulator**  icon to test your new field definitions. After the Please Wait...! window appears and closes, your DefTool main console will include the Simulator Data View (right-pane) and Tree (left-pane), both shown below.




The default Simulator View appears displaying the fields of the first statement (or document, as called here) in your sample data source. Follow the instructions below to modify the View of your simulation.

To See This...	Do This...
View the fields of document 1	Scroll to the right in the Simulator Data View
View the fields of document 2	Click the Next Document icon in the left-pane
View the fields of all documents	Click the Show All Documents tool in the left-pane. Scroll to the right in the Simulator Data View
Verify that your fields are extracting the right data	Use the Simulator Data View and the chart of fields below.
Sort each field in descending order.	Click once on each field name, one at a time, in the Simulator Data View.
Sort each field in ascending order.	Click twice on each field name, one at a time, in the Simulator Data View.
Troubleshoot your field definitions	Look for “Hit Not Found” where there should be field occurrences. Double click “Hit Not Found” instances in the Simulator View to navigate to the corresponding area in the data source.

Simulating Table Extractions

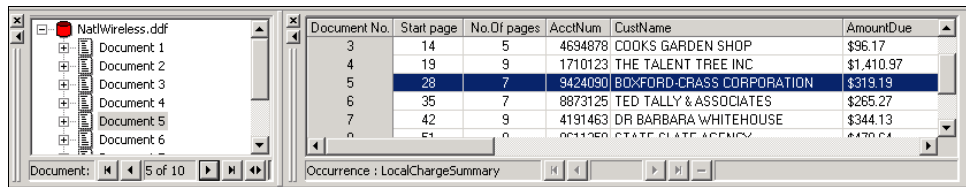
The Extraction Simulator will allow you to test your rules for fields, tables and groups before you deploy the application. You can perform complete or selective extraction simulation on any of the rules you have created.

By default, the Extraction Simulator will test all of the rules (fields, tables and groups) that you have built in your DDF.

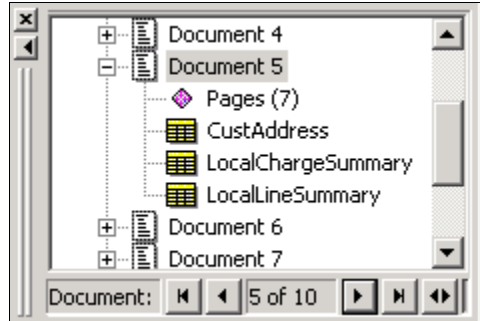
Click the **Simulator**  icon to test your new table definitions. Your Simulation Data View appears as it has after previous extractions, with all fields being presented in the main view area.

The extraction results for your table definitions are accessed from the Simulation Tree, one document at a time. To view the table extraction results for any document (or statement), follow the steps below:

Locate the document in the Simulation Tree — enable Show All Documents, Show Range or Show Single Document, if necessary, to view the desired document in the Tree. (Enable Show All Documents to locate Document 5 in the Simulation Tree as shown below.)



Expand the tree node for the document by clicking on the corresponding plus box in the Simulation Tree. (Expand Document 5 in the Simulation Tree as shown below.) Each table that was detected and extracted from the document will be listed as sub-nodes.



To view the extraction results of a table for this specific document, click the desired table in the now-expanded Simulation Tree. (In the Simulation Tree, click **LocalChargeSummary** to view the extraction of the Local Charge Summary data in the *BOXFORD-CRASS* statement, or Document 5 as shown below.) The data extracted as your table will be shown in the Simulation Data View.

The screenshot shows the Simulation Data View with a table of extracted data. The table has three columns: 'Table Line No.', 'LocalChargeDesc', and 'LocalChargeAmount'. The data is as follows:


Table Line No.	LocalChargeDesc	LocalChargeAmount
1	1 YEAR TERM DISCOUNT	-6.27
2	FCC ACCESS CHARGE	49.50
3	BUSINESS LINE	174.00
4	RANDOM HUNTING PACKAGE	35.25
5	TELECOMM. RELAY	0.42
6	TOUCH TONE	0.00
7	STATE TAX	6.09
8	FEDERAL TAX	7.56

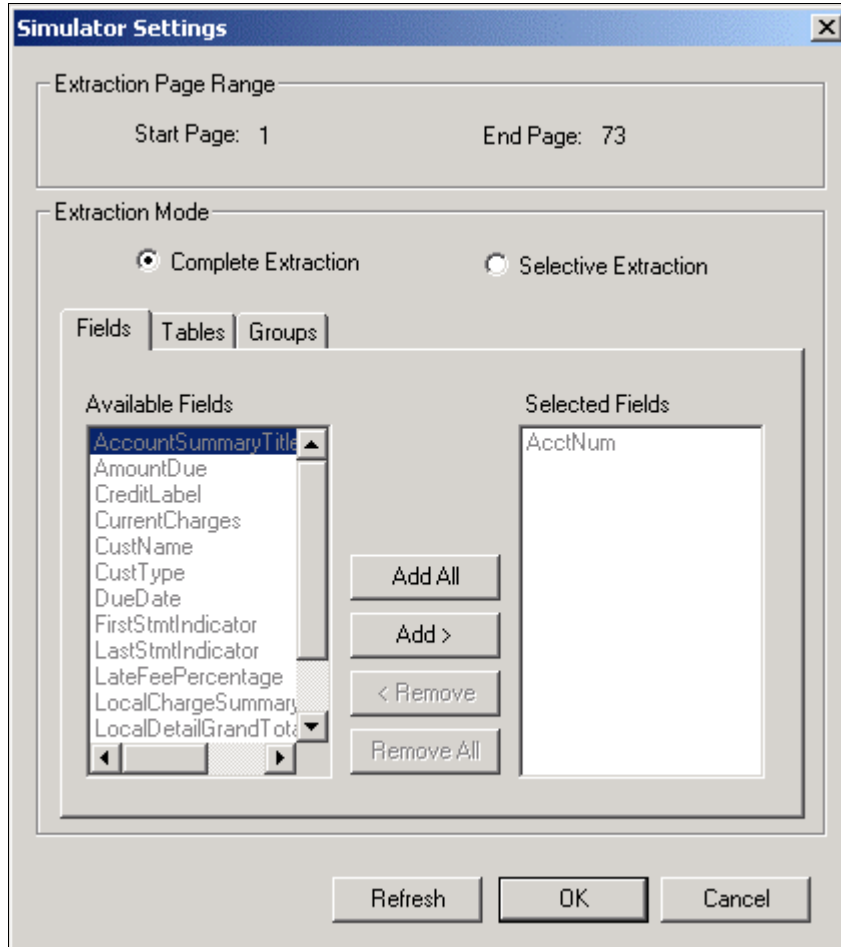
At the bottom of the window, the status bar reads 'Occurrence : LocalChargeSummary 1 of 1'.

(Compare the results in your Simulation Data View with the print-copy and DefTool work area view of the *BOXFORD-CRASS* statement, and with the image above.)

Simulating Selective Extraction

You can configure the Extraction Simulator to test the rules for a selection of fields, tables and groups within the Simulator Settings dialog. To create Simulator Settings:

Click the Simulator Settings  icon in the Simulation Tool Bar or select **Simulator>Simulator Settings**. The Simulator Settings dialog opens (shown below).



By default, the Simulator is set to perform extraction with all of the rules you have defined. To override the Complete Extraction and select only the items you want, click the **Selective Extraction** radio button. Items in the Fields, Tables and Groups tabs will become active.

To select the fields you want extracted, go to the Fields tab, Available Fields window. For each desired field:

- Click the field name, and click **Add...**, or
- Double-click the field name.

(Select the following fields for extraction so that they appear in the Selected Fields window: **CustName**, **AmountDue**, and **CustType**.)

To select the tables you want extracted, go to the Tables tab, Available Tables window, and for each desired table:


- Click the table name, and click **Add** or
- Double-click the table name.

(Select the following tables for extraction so that they appear in the Selected Tables window: **LocalChargeSummary** and **LocalLineSummary**.)

To select the groups you want extracted, go to the Groups tab, Available Groups window, and for each desired group:

- Click the group name, and click **Add**, or
- Double-click the group name.

Click **OK** to save your settings and close the Simulator Settings dialog. (Click **OK**.)

Click the **Simulator**  icon to re-simulate your newly selected extraction rules.



Tip

You can later re-configure your Simulator Settings within the same dialog, by selecting fields, tables and groups, respectively, and clicking **Add >**, **Add All**, **Remove >** or **Remove All**.

You can delete your **Selective Extraction** settings quickly within the same dialog by clicking on **Refresh**.

Regular Expressions

A regular expression is a pattern that you use in a DDF rule, to validate a data string that you want to extract. The data string that your rule is looking for is usually of a specified format. You use regular expressions in all of your DDF rules to establish the data pattern that is to be looked for, whether the data that your rule is looking for is static or variable. In order to write regular expressions that are effective in extracting all the data values you want in your statements, and allow for optimal extraction performance, a good understanding of regular expression syntax is necessary.

Purpose of Regular Expressions

To review, a pattern in any DDF rule can:

- Validate the data string that identifies your document style or additional page styles
- Verify the start and end marker positions of tables and groups in your statements
- Extract the appropriate data values for fields and table columns.

A regular expression in a DDF rule tells the eaDirect extraction components, position by position, exactly what characters are valid as part of the data value.

Every DDF contains rules to extract data values that conform to a well-known format. Some common data formats that will likely appear in your data (and that you will want to extract) are:

- Dates
- Currency
- Phone Numbers

City, State and Zip Code Numeric values Statement data will usually fit a proprietary pattern, such as:

- Account Numbers

- Customer ID Numbers

Finally, you will likely extract some static text, or at least use static data strings as markers and page style identifiers. A regular expression is needed to find the data strings during live statement retrieval. These strings can be:

- Descriptive labels, such as TOTAL AMOUNT DUE:
- Print or database file control codes
- Headings, such as LOCAL DETAIL CHARGES

To extract the data that fit common formats like the ones listed above, you can write a regular expression that will uniquely identify a data value and exclude other data values that do not follow the format.

Using a Regular Expression with eaDirect

When your customers click to view their dynamically composed statement, the eaDirect extraction components use the regular expression pattern that you define to validate the data value found in a left-to-right, one-ahead manner. This means that if the regular expression does not match a position of the data value found where it is expected, it will proceed to the next position in the regular expression in its attempt to validate the position. If the data value found does not then match the next position in the regular expression, the data will not be extracted.

Writing Regular Expression Patterns

The illustrations below will introduce you to the concept of writing a regular expression to detect and extract data values of specific formats with your DDF rules. Each illustration contains

- A sample of data that follow a common format
- An examination of the data sample to find the general pattern that exists
- A regular expression that will find and extract all data that fit the pattern that was found

- An explanation of each new regular expression operator used.

You will become acquainted with all of the most commonly used regular expression operators through the succeeding illustrations.

Illustration 1: Shorthand Date Formats MM/DD/YYYY

Consider a field, *DueDate*, whose representative sample of values across statements are:

- 03/16/1997
- 12/31/2000
- 10/05/1999

To write a regular expression to extract these values as *DueDate*, you start by finding out what the different examples have in common. Given that this sample includes all possibilities of *DueDate* values, you can see that the general pattern is (per position):

Pattern position	Used for	Valid characters
Position 1	The first digit of the MONTH	0 or 1
Position 2	The second digit of the MONTH	Any number
Position 3	Delimiter of MONTH and DAY	A forward slash
Position 4	The first digit of the DAY	0, 1, 2 or 3
Position 5	The second digit of the DAY	Any number
Position 6	Delimiter of DAY and YEAR	A forward slash
Position 7	The first digit of the YEAR	1 or 2
Position 8	The second digit of the YEAR	Any number
Position 9	The third digit of the YEAR	Any number
Position 10	The fourth digit of the YEAR	Any number

Therefore, using regular expression syntax, your regular expression pattern to validate and extract values of the DueDate would be:

[01] [0-9] / [0-3] [0-9] / [12] [0-9] [0-9] [0-9]

To specify that multiple characters are valid for a position in the data value, enclose the characters within hard brackets (examples shown below):

Regular Expression	Signifies
[01]	0 or 1
[AF]	A or F
[Aa]	A or a
[az09]	a, z, 0 or 9

To specify a range of characters that are valid for a position in the data value, separate the first and last character in the range with a dash (examples follow):

Regular Expression	Signifies
[0-3]	0, 1, 2 or 3
[0-9]	Any number
[A-F]	A, B, C, D, E or F
[A-Z]	Any uppercase letter
[a-z]	Any lowercase letter



To specify that only ONE character is valid for a position in the data value, simply enter that character into the pattern. In the example above, Positions 3 and 6 must be a forward slash, therefore the regular expression contains a / in each of these positions.

Illustration 2: Varied Shorthand Date Formats

Consider what would happen if DueDate values could also be:

- 3/25/1998

- 12/1/2000
- 05-07-1996

This changes the general pattern of the DueDate values.

Pattern position	Used for	Valid characters	Appearance frequency
Position 1	The first digit of the MONTH	0 or 1	Optional
Position 2	The second digit of the MONTH	Any number	Exactly once
Position 3	Delimiter of MONTH and DAY	A forward slash or a dash	Exactly once
Position 4	The first digit of the DAY	0, 1, 2 or 3	Optional
Position 5	The second digit of the DAY	Any number	Exactly once
Position 6	Delimiter of DAY and YEAR	A forward slash or a dash	Exactly once
Position 7	The first digit of the YEAR	1 or 2	Exactly once
Position 8	The second digit of the YEAR	Any number	Exactly once
Position 9	The third digit of the YEAR	Any number	Exactly once
Position 10	The fourth digit of the YEAR	Any number	Exactly once

Therefore, your regular expression would change to allow these varying formats in the DueDate:

```
[01]*[0-9][/\-][01]*[0-9][/\-][12][0-9][0-9][0-9]
```

To specify that a character could appear multiple times, or not appear at all (i.e. 0 or more times), enter an asterisk after the character. To specify that a selection of characters could appear zero or more times, enter the asterisk after the closing hard bracket that contains the characters (examples follow).

Regular Expression:	What it means:
[0-9]*	A numeric character is optional, and could occur more than once (no limit)
[0-9,]*	If a character appears, it must be a numeric or a comma, and there could be multiple occurrences of any of these characters

To extract a regular expression syntax character as normal data, you must enter a backslash before that character, whether it appears as part of a selection of characters for a position (i.e. within a set of hard brackets) or in its own position (examples follow).

Regular Expression	Signifies
\-	The dash is the only character that can and must appear in the position
[^/]	The valid characters for this position are the forward slash and dash
[\-+*]	The valid characters for this position are the dash, plus and asterisk
No\.	Look for the data string <i>No.</i> (i.e. Position 1 = N, Position 2 = o, Position 3 = .)

Illustration 3: Currency Values

Consider a field, AmountDue, whose representative sample of values across statements are:

- \$964.23
- \$1.99
- \$65.00
- \$0.03

Again, to write a regular expression to extract these values as AmountDue, you start by finding out what the different examples have in common. It is not necessary, however, to try and decipher the pattern from left to right at this point. Simply look for the common traits of the sample data, and build your regular expression around those traits. Given that this sample includes all possibilities of AmountDue values, you can see that the general pattern is (per position):

Pattern position	Used for	Valid characters
First Position	The currency symbol	\$
Next Position(s)	The dollar amount	Any number, one or more times
Next Position	The decimal point	.
Next Position	The 10th of a cent figure	Any number
Last Position	The 100th of a cent figure	Any number

Therefore, using regular expression syntax, the regular expression pattern to extract values that match the above pattern would be:

`$ [0-9] + \. [0-9] [0-9]`



Tip

Currency can be sorted to three decimal places. For further information, see "Sorting Tables and Groups" on Page 301.

To specify that a character could appear 1 or more times (i.e. always occurs at least once, but may occur in subsequent positions), enter a plus symbol after the character in the regular expression. To specify that a selection of characters could appear 1 or more times, enter the plus after the closing hard bracket that contains the characters (examples shown below).

Regular Expression	Signifies
[0-9]+	At least one numeric character must appear, but there could be more than one (no limit)
A+	At least one A must appear, but there could be more
[A-Za-z0-9]+	Any alpha-numeric character is valid for one or more positions in a data value
+ (space, then +)	One or more spaces are required in the data string

Illustration 4: Longhand Date Formats

Consider a field, *StatementDate*, whose representative sample of values across statements are:

- May 25, 2001
- August 3, 1998
- January 20, 1999

Again, to write a regular expression to extract these values as *StatementDate*, determine what the different examples have in common and build your regular expression as specifically as possible, given those common traits. Given that this sample includes all possibilities of *StatementDate* values, you can see that the general pattern is:

Pattern position	Used for	Valid characters
First Positions	The longhand month of the year	January, February, ..., and December
Next Position	A space between the month and day	A space character
Next Position(s)	The 1- or 2-digit numeric day of the month	One or two numbers
Next Position	A comma after the day	A comma
Next Position	A space between the comma and year	A space character
Last 4 Positions	The numeric 4-digit year	[12][0-9][0-9][0-9]

Therefore, using regular expression syntax, the regular expression pattern to extract values that match the above pattern would be:

```
(January|February|March|April|May|June|July|August|September|October|November|December) [0-3] * [0-9] , [12] [0-9] [0-9] [0-9]
```

To specify that a selection of data strings are valid for detection or extraction by your DDF rule, delimit the strings with the pipe symbol and enclose the entire string set with soft brackets. The data strings you specify can be static (as in the case of longhand months, above) or variable (i.e. using an inner regular expression pattern) (examples shown below).

Regular Expression	Signifies
(cat dog)	The string cat, or the string dog
([A-Z][a-z]+ [0-9]+)	An alpha string starting with an uppercase letter, e.g. Apple, or a string of numbers, e.g. 01760
([a-z0-9.\-]+@[a-z.\-]+\ ([0-9][0-9][0-9]\)[0-9][0-9][0-9]\-[0-9][0-9][0-9][0-9])	An email address, e.g. biller-2@bbb.com, or a phone number, e.g. (508) 652-8600

Illustration 5: Data Values that Vary Widely in Pattern

Consider a table column called AddressLine that is defined to extract each line of a customer address. Examine the given representative sample of address lines below:

- 2 Apple Hill
- # 4-B
- Three Canal Park
- c/o Libby Mae Brown
- % John Talley

These potential values of the AddressLine table column vary widely, both in character composition and in length.

Therefore, it can be difficult, to find a common pattern among a data sample like above.

A regular expression to extract each position of each AddressLine value might look like:

```
[A-Za-z0-9#%] [A-Za-z0-9#%\-\ (\) \. &@, /: ' " ] +
```

This regular expression requires that:

- Position 1 must be an alpha-numeric character, a pound sign or a percent sign
- Positions 2 and beyond can each be an alphanumeric, a pound, a percent, a dash, etc.

This regular expression is very broad, so that it must check many characters per position. This practice is:

- Inefficient for extraction performance, and possibly incomplete, given that there may be characters that appear in AddressLine that you have not anticipated.

You can write a regular expression that specifies which characters are NOT valid for a position or positions. This will:

- Exclude unwanted data strings that exist within the search area, and
- Extract the correct data string(s) as desired.

A regular expression that would extract all Address Lines, yet exclude space lines that may appear between rows of meaningful data (i.e. no space in the first position), is shown below:

`^ .+`

Pattern position	Used for
Position 1	All characters are valid except for a space
Position 2 and on.	All characters are valid

Illustration 6: Static data strings

Suppose that you want to build a DDF rule that needs to locate (or extract) a static data string. For example, you will likely use static strings as your anchors, page styles, and table/group markers. Sometimes, you will even want to extract static strings that are related to some variable data, so that you can dynamically present that information when its related data appears in the statement, rather than hard-code it into your HTML templates.

Points to keep in mind when you use a static string as the basis for an anchor, marker, page style, or any other rule:

- The pattern you write will specify exactly ONE character per position, rather than a range or selection of characters
- If only one character is valid for a position, there is no need to surround it with brackets or place repeat operators (such as + or *) after it (unless you expect it to repeat, of course!)
- The pattern is, however, technically still a regular expression, even though it may not use regular expression operators.

Suppose you want to use the string `DETAIL` as a marker for a table or group? Given that you expect the string to be exactly the same across all statements, the pattern to find this string is:

Pattern position	Used for
Position 1	An uppercase D only
Position 2	An uppercase E only
Position 3	An uppercase T only
Position 4	An uppercase A only
Position 5	An uppercase I only
Position 6	An uppercase L only

It is easy to see that the regular expression to find the data string `DETAIL` is:

`DETAIL`

Illustration 7: Partially Static Data Strings

Sometimes, only a portion of the data string you are looking for is static. Then, your pattern will be partially static as well.

Points to keep in mind when you use a partially static string as the basis for an anchor, marker, page style, or any other rule:

- Part of the pattern you write will specify exactly ONE character per position, whereas the other portion of the pattern will be variable.
- If only one character is valid for a position, there is no need to surround it with brackets or place repeat operators (such as `+` or `*`) after it (unless you expect it to repeat, of course!)

Suppose you want to extract the `DeptNumber` as a field, including the descriptive text:

- `DEPT 1`
- `DEPT 2`

- DEPT 10

Given that the sample above is broad enough to represent all values of DeptNumber in the statements, you can see that the pattern is:

Pattern position	Used for
Position 1	An uppercase D only
Position 2	An uppercase E only
Position 3	An uppercase P only
Position 4	An uppercase T only
Position 5	A space only
Position 6	A number
Position 7 (optional)	Another number, if anything

Therefore, using regular expression syntax, the regular expression pattern to extract values that match the above pattern would be:

```
DEPT [0-9]+
```


Regular Expressions Reference Chart

The chart below is intended as a job aid for you to use during the development of your DDFs.

Regular expression	Used for	How it is used in a regular expression	How it is NOT used in a regular expression
[]	Contains a set of characters that are valid for a position Can be used with -	Surrounds a set of 2 or more characters that are valid [01] Used with - , surrounds a range of characters that are valid (used with the dash - ; see below): [a-f] Used with - , surrounds both single characters and a range that are valid (used with the dash - ; see below): [0-9,]	NOT used to contain data strings: Bad = [January February] Good = (January February) NOT used in a nested fashion: Bad = [[xyz]0-9] Good = [xyz0-9] NOT used for one character that is valid: Bad = [\$] Good = \$
-	Delimits a range of characters Always used within []	Always used within [], separates the first and last character of a range that is valid [5-9] Can be used multiple times per position: [A-Za-z0-9]	NOT used to state two characters that are valid: Bad = [0-1] Good = [01] NOT used without a first or last character: Bad = [A-] Good = [A-Z]

Regular expression	Used for	How it is used in a regular expression	How it is NOT used in a regular expression
+	One or more of the preceding character(s) are valid	After a single character that is valid: 0+ Used outside [], after a set of characters that are valid: [0-9]+	NOT used within hard brackets: Bad = [0-9+] (will mean numbers and a + are valid) Good = [0-9]+ NOT used before a character or selection of characters: Bad = + (preceding a space) Good = + (i.e. one or more spaces are valid) Bad = +[a-z] Good = [a-z]+
*	“Optional” Zero or more of the preceding characters are valid	After a single character that is valid: 0* Used outside [], after a set of characters that are valid: [0-9,]* Will be treated as normal character, not operators, if it appears inside square brackets: [0-9+]	NOT used within hard brackets: Bad = [0-9*] Good = [0-9]* NOT used before a character: Bad = *\$ Good = \$* NOT used before a selection of characters: Bad = +[a-z] Good = [a-z]+ NOT used consecutively (will yield unpredictable results): Bad = [0-9]**,*[0-9]*\.*[0-9]*[0-9]* Good = [0-9,]*\.[0-9]*

Regular expression	Used for	How it is used in a regular expression	How it is NOT used in a regular expression
.	“Wildcard” Any character found is valid	Used by itself in one position: . Used to extract widely varying strings of unknown length: .+	NOT used within hard brackets as part of a selection of valid characters: Bad = [a-f.] (ambiguous)
()	Surrounds a selection of data strings that are valid	Used with , surround the data strings that are valid: (Total Amt Due Final Amt Due)	NOT used within [] Bad = [(cat dog)A-Z] Good = (cat dog [A-Z])
	Delimit a selection of data strings that are valid Always used within ()	Used with (), in between each valid data string: (MA CT RI VT NH ME)	NOT used without (): Bad = Local Long Distance Good = (Local Long Distance)
\	Extract the following regular expression operator as a normal character Used with W, w, N or n, extract appropriate set of characters (see below)	Used within [] or (), precedes the regular expression operator character you want to extract: (Jan\. Feb\. Mar\ [0-9,\. Used in a static position, precedes the regular expression operator you want to extract: No\. of Shares Used to extract itself: \\	NOT used after a regular expression operator you want to extract: Bad = +\ Good = \+

Regular expression	Used for	How it is used in a regular expression	How it is NOT used in a regular expression
^	The following character is NOT valid	Used before [], to signify that the characters contained within are not valid: ^[a-z] Used before (), to signify that the data strings contained within are not valid: ^(Dept Section group) Used before a single character: ^ (I.e. a space is not valid)	NOT used after a character: Bad = \$^ Good = ^\$
\w	Extract any alpha character (upper- or lowercase)	Used within [], if part of a bigger set of characters that are valid: [01\w] Used alone in a position: \w	NOT used within ()
\W	Extract any NON-alpha character	(see above)	(see above)
\n	Extract any numeric character	(see above)	(see above)
\N	Extract any NON-numeric character	(see above)	(see above)

Dynamic Pattern Matching

A dynamic field is a parameter that you create in your DDF, for use in the pattern of a group marker, table marker, or table column, to allow for the selective extraction of the group or table data. When a statement is retrieved dynamically in an application using dynamic pattern matching, the dynamic field parameter in the group marker, table marker, or table column is replaced with a static string of the recipient's choice during live retrieval, and only the data occurrences whose marker or column match the string get extracted.

Dynamic pattern matching allows you to build an application that will extract and present data hierarchically, and specific to the recipient's requirements. For example, you may want your recipient to be able to view their statement in "pieces," rather than all at once. The recipient's first "piece," or view, of their statement, may show top-level summary data only, from which he or she can select the portion of detail to be viewed dynamically, based on a selection in the Summary view. This functionality allows for faster extraction and presentment performance, and also facilitates easier viewing for recipients, in that they would no longer be forced to scroll through potentially thousands of lines of data in their browser.

With dynamic pattern matching implemented in your application, your recipients determine 1) the level of statement detail to view and 2) only the sections of detail that they want to view. Statement data that is not chosen for view by the recipient is simply not extracted, thus presenting statement data faster to the recipient and freeing up system resource in your application environment.

Setting Dynamic Pattern Markers in DefTool

Example 1:

A telecom company wishes to present basic account information and summary data to recipients as the first view of the statement.

Recipients can have multiple Local and/or Long Distance Phone Lines. On the first view of the statement, the Local Line Summary and Long Distance Line Summary sections are shown, with each Phone Line and its charge in the appropriate list. Since dynamic pattern matching was implemented here, a link has been placed around each phone line that will make a second request for live retrieval, this time for the charge detail related to the phone line.

When the recipient clicks on a phone line for which they want to see the charge detail, (such as local line detail charges, as shown below), the local detail charges data for this phone line ONLY is extracted and presented in the recipient's browser.

Example 2:

The same telecom company wishes to allow recipients to view portions of their Long Distance Itemized Calls list, based on date, for a selected phone line.

To create the settings for the above scenario in the DefTool, you will create the extraction rules for the detail data (or target) to use a dynamically passed value (the Local Line Phone Number) to extract a specific occurrence of the detail data (detail information pertaining only to the phone line that is selected).

To create settings in DefTool:

1. You must first find the hierarchy in your statement data, and how the different levels of hierarchy relate to one another. For example, the Local Line Summary section of a statement contains basic charge information about a number of local phone \Lines; perhaps it contains the phone line number and current charge for each phone line. The detail data (such as the features, local usage and itemized calls) for each phone line follows later in the statement. Therefore, each item in the summary data is a "parent" to a section of detail data that follows in the statement.
2. Find a data string that is common between:
 - An item in the high-level data that you plan to link FROM (i.e. Summary data), and
 - The item at the marker level (for a group) or table column level (for a table) of the lower-level data section that you plan to link TO (for example, Detail data).

For example, each item in the Local Line Summary section (shown below) contains a reference to a Phone Line.

LOCAL LINE SUMMARY	
252 566-4910	39.48
252 566-5874	45.75
252 566-9616	68.25
252 566-9617	68.42
252 566-9618	61.98
LOCAL LINE GRAND TOTAL:	\$291.88


The Local Charge Detail section for each Phone Line starts with the Phone Line instance (shown below).

		LOCAL DETAIL CHARGES	PAGE	5
01				
84				
04	781 827-0818			
4	FEATURES			
4	02/25 - 03/24	FCC ACCESS CHARGE	7.84	
4	02/25 - 03/24	COMMUNITY CALLING PLUS	32.20	
4	02/25 - 03/24	INSIDE WIRE MAINTENANCE	5.00	
4	02/25 - 03/24	TOUCH-TONE	0.00	
4	02/25 - 03/24	TELECOMM. RELAY	0.07	
4	03/24	STATE TAX	0.99	
4	03/24	FEDERAL TAX	1.22	
4				
4				
4		FEATURES SUB-TOTAL:	\$49.07	

Caution If your Target data is an occurrence of a group, the common data string you choose must exist at one of the markers of the section of Detail data you are trying to link to. If your Target data is a portion of a table (based on a column constraint, such as all Calls made in the evening), then your common data string must exist in a column of the table.

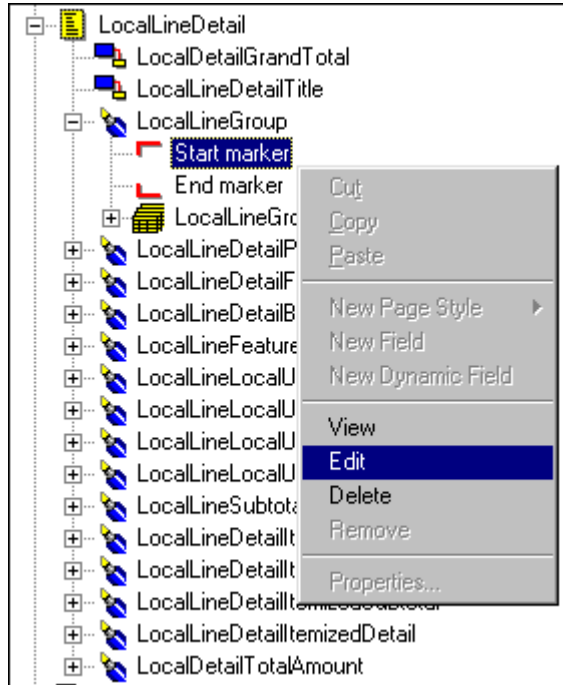
3. Create your extraction rules for the Detail section (also known as the Target) in the DDF, using the DefTool. Most likely, this type of data will be best extracted as a group with several tables. In the example shown above, a group would be created to extract all Detail data for a Phone Line, starting at the row containing the Phone Line instance itself and ending at the row containing the Phone Line Subtotal data.

Therefore, the Phone Line example (variable regular expression suitable for validating a phone number string) in the detail group would be the best start marker, and the Subtotal string (i.e. SUB-TOTAL FOR) would be the best end marker for the detail group.

4. Create a new dynamic field in your DDF, by clicking the New Dynamic Field icon  in the DefTool toolbar and locating the resulting Dynamic Field node in the tree. It appears in the tree in rename-mode; give it a name that you will recognize later (Such as *DYNLocalLinePhoneNo*.)



5. Set the pattern for the marker of your detail group to use a dynamic pattern rather than just a specific regular expression.



6. In the Marker Properties dialog, Enable the **Use Dynamic Pattern** checkbox. (Note that the Test button is now changed to Dynamic Fields.)
7. Click **Dynamic Fields** and select your new dynamic field by locating it in the resulting drop down list and double-clicking on it. Your Pattern field should now be populated with the name of your dynamic field enclosed in angle brackets (such as <DYNLocalLinePhoneNo>).



8. Erase any other non-static characters from the Pattern field, and click **OK** to save changes and close the Start Marker Properties dialog.

Marker Information

Marker Name : LocalLineGroup Var Field

Pattern : <DYNLocalLinePhoneNo> Reg...

Select the line only containing this pattern

Start Marker End Marker

Select Characters >> Save Pattern... Select Pattern... Dynamic fields >>

Start Marker Options

Look for the Start marker in the same row

Include Marker Line Use Dynamic Pattern

Use Blank Line as Marker Use Variable Field

Page Style Name : LocalLineDetail

Window Coordinates

Start Row 2 Start Column 9

End Row 57 End Column 10

OK Cancel



Tip

It may be prudent to leave a partial regular expression in the Pattern field along with the Dynamic Field reference. For example, if the phone line in the detail group was preceded by the string LOCAL LINE: , then you would set the start marker pattern to locate:

LOCAL LINE: <DYNLocalLinePhoneNo>

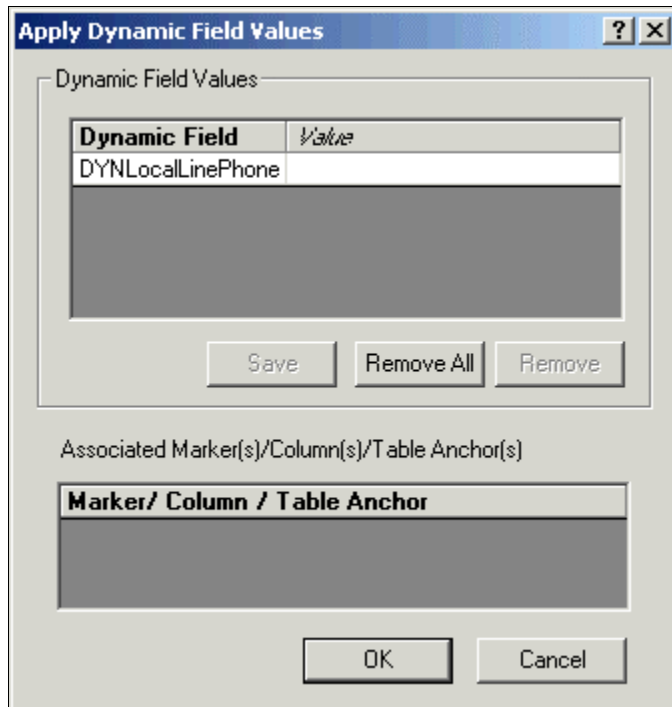
You need to create a hyperlink around the common data item in the high-level view (i.e. the summary view). (In this instance, you can create a link around each Local Phone Line in the LocalLineSummary.) This link, when clicked, will make a specific request for data retrieval and presentment. (In our example, it will request extraction of the LocalChargeDetail group, but only the occurrence of the group where the Phone Line matches the Summary Phone Line that is selected).


Simulating Dynamic Field Values

Continuing with the procedures from above, you can check your work by simulating using the newly created dynamic pattern.

To simulate using dynamic field values:

1. Click . The Apply Dynamic Values dialog appears.



2. Click to select the dynamic field for simulation.
3. Double-click in the Value field to enter the dynamic value. (Enter 202.)
4. Click OK. Only extractions for the number “202” will appear.
5. Save the DDF by clicking the Save icon .

Completing the Dynamic Pattern Link in Composer

Now you will need to complete the linking in Composer.

To link the dynamic pattern in Composer:

1. Select **Start>Programs>eaDirect>Composer**.

Tip

All DefTool configurations for dynamic pattern matching are performed at the Target view level in the DDF.

All Composer configurations for dynamic pattern matching are performed within the source view's ALF (i.e., the summary view).

2. Select **File>Open**.
3. In the resulting Open dialog, browse to *C:\EDCSbd\Samples\NW_LocSummary.alf*.
4. Go to the Default Template in the WYSIWYG area of the Composer GUI, and locate the table that contains the column whose data will be wrapped with a hyperlink to selectively extract detail information. (Locate the LocalLineSummary table tag: `[E]LocalLineSummary,R[/E]`.)
5. Right-click on the table tag and select **Configure Table**. (For National Wireless, right-click `[E]LocalLineSummary,R[/E]` and select **Configure Table**. The WYSIWYG area displays the ALF's placeholder for the HTML table, `[E]LocalLineSummary,R[/E]`.)
6. You can edit the LocalLineSummary table to include static text or image items for linking. For example, the original table configuration in our National Wireless example appears as follows:

[E]LocalLinePhoneNo[/E]	[E]LocalLineAmount[/E]
-------------------------	------------------------

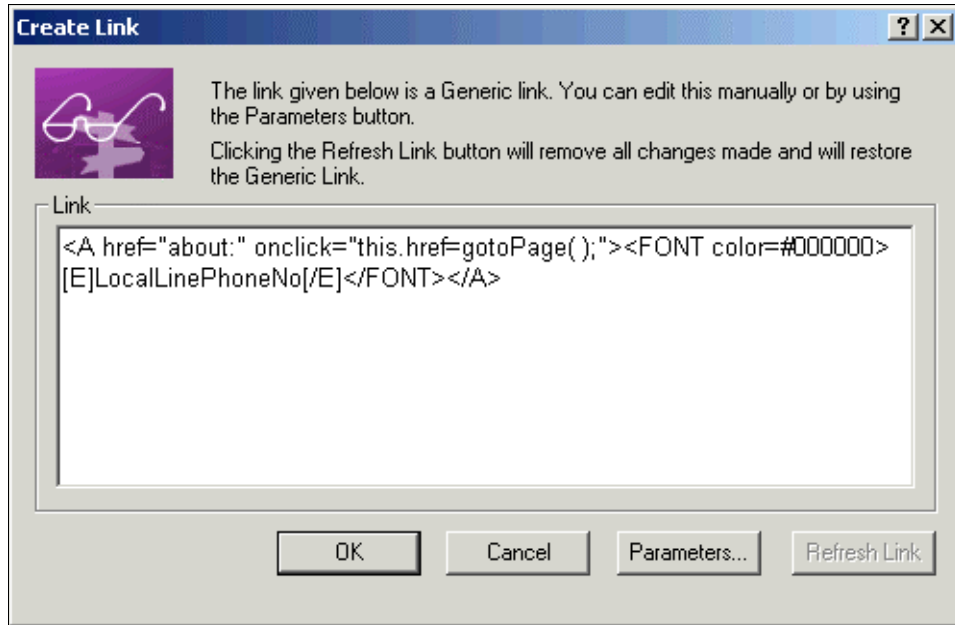
You can add your own text link, "Click here for details," or add a graphic.


7. Locate the column tag for linking and completely highlight the tag. (Highlight [E]LocalLinePhoneNo[/E].)

Caution It is essential to highlight the entire tag before proceeding, so that your new link is wrapped around the tag properly.

8. Right-click the highlighted tag and select **Create Link** from the right-click menu. The Create Link dialog appears, containing generic link syntax.
9. Edit the link to provide the following information:
 - The target view (DDF/ALF pair) that contains extraction and presentment rules for the Detail data
 - The name of your dynamic field

The name of the data element that will be replaced by the value inserted into the dynamic field.



If the Link to View dialog already has a link in it, click  to remove the link and restore the generic link. You can edit this link directly, or from the Parameters dialog.

10. Click the  button. The Add Parameters dialog appears.

Add Parameters [?] [X]

Specify the Target View, Dynamic Fields, and Elements to be used in the link.

Target View:

Dynamic Fields

Name	Value	Type

Add ...
 Modify ...
 Remove
 Clear All

Elements

Sort Elements : Add To Link Configure and Add

Filter Elements : Add To Link Configure and Add

Top/Bottom Elements: Add To Link Configure and Add

Added Elements

Element Type	Element Name

Remove
 Clear All
 Reset URL

Sample Link

```
<A href="about:" onclick="this.href=gotoPage( );"><FONT face=Arial size=2><STRONG>DESCRIPTION</STRONG></FONT></A>
```

OK Cancel

On the Add Parameters dialog, you must specify:

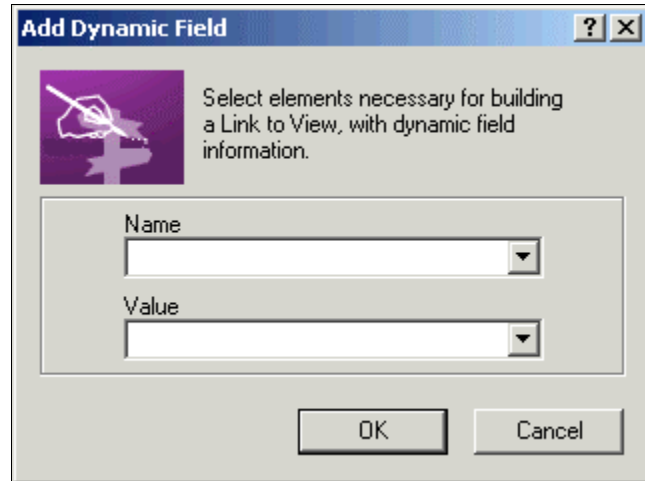
- Your **Target View** name
 - The **Dynamic Field** that is used in the target group marker; when the link is clicked, this “parameter” will be populated with the actual link string and sent back during the request for detail data, to extract only the group occurrence that matches this string.
 - Sort, filter, and top/bottom Elements
11. In the **Target View** field, enter the name of the view that contains extraction and presentment rules for the target, or detail, data (e.g. NatlWireless).

Tip

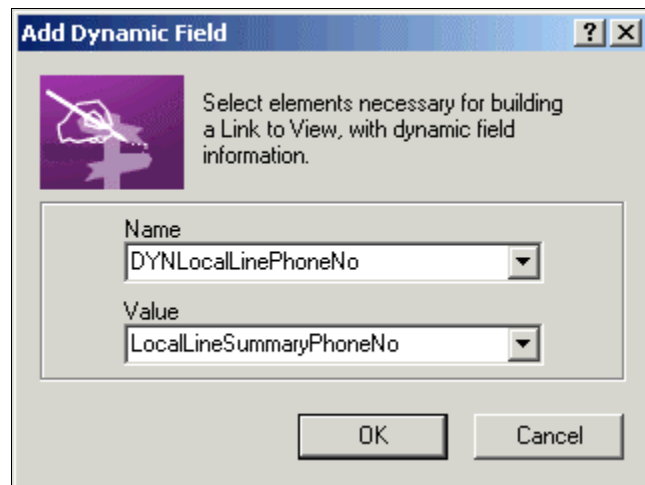
In the Dynamic Fields section, you will add the parameters that will be used to verify the correct occurrence of detail for presentment. You can specify one parameter, or multiple parameters to certify the correct occurrence of detail data to be presented.

12. Click . The Add Dynamic Fields dialog appears.

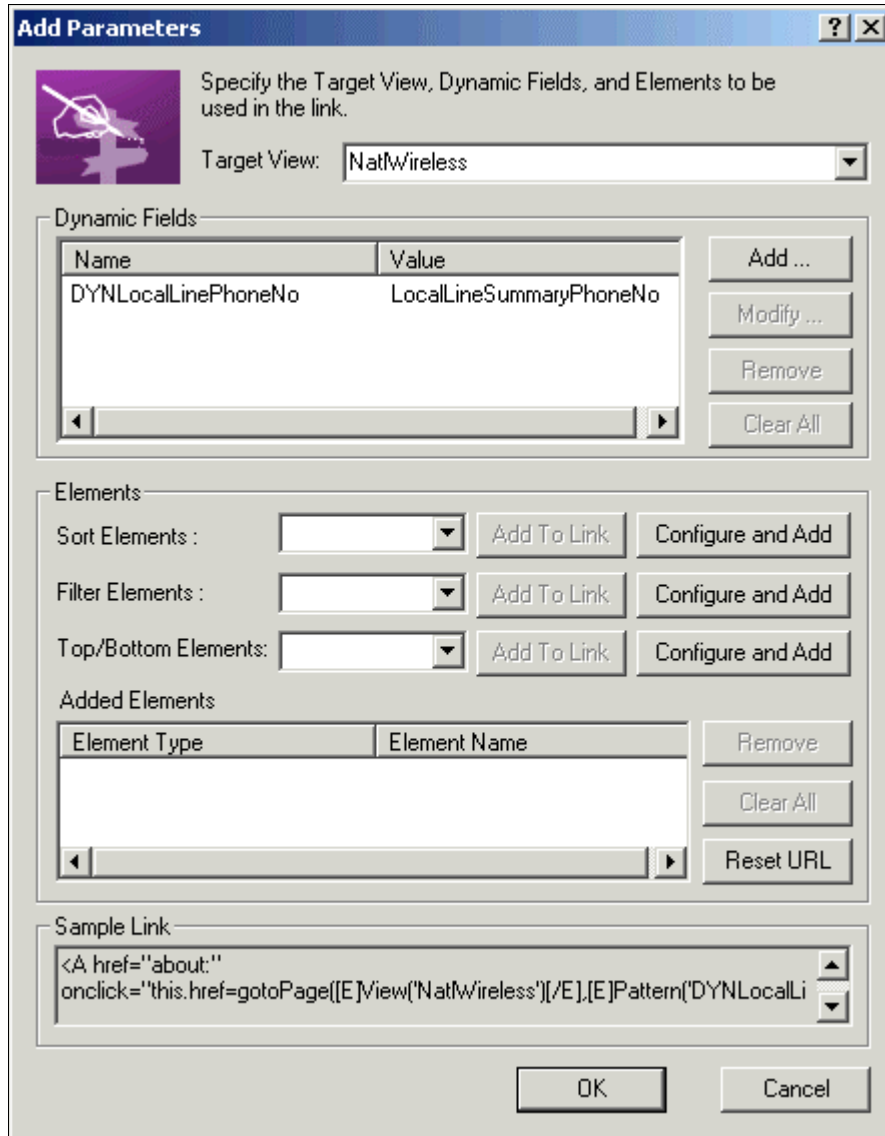
Dynamic fields define the requirements for extracting Detail data. The end is your dynamic field end, and the Value is the data that will populate the dynamic field when the link is clicked. The Value can be static or can be a field or column occurrence. Most likely, the Value will be the name of the column you are wrapping the link around, so that the link string is used to match to and extract the correct portion of Detail data.



13. In the **Name** area, enter the dynamic field name in HTML syntax. (Enter `DYNLocalLinePhoneNo`.)
14. In the **Value** area, enter the column name that represents data to match the selective extraction in HTML syntax. (Enter `[E] LocalLineSummaryPhoneNo [/E]` .)

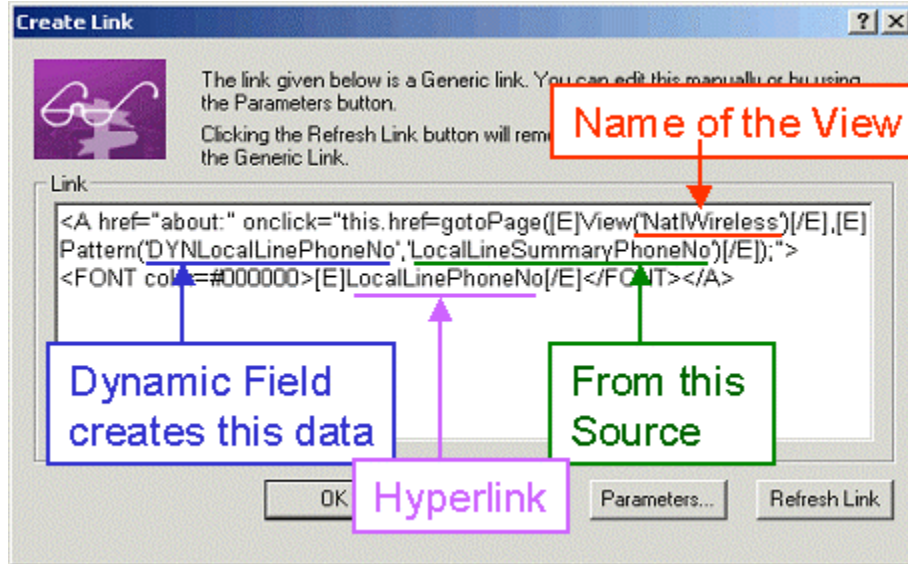


15. Click **OK**. The new name/value pair appears in the Dynamic Fields section of the Add Parameters dialog.



16. Click **OK** to return to the Create Link window.

View the changes to the hyperlink code that appears in the Link to View dialog. (Below is an example using National Wireless.)



This link will not only send a request to see the Detail View of the statement, but only the Detail View that pertains to the Local Line Summary Phone Number that is requested by the link.

Tip For more information about the JavaScript function that is used in this link, contact the edocs Training department.

- Click **OK** to close the Link to View dialog. In the WYSIWYG area of the Composer, you will notice that your table column name is now displayed as a link. This link code, along with other table formatting information, is stored in the ALF and referenced during live retrieval of the Summary data. (An example using National Wireless is shown.)

[E]LocalLinePhoneNo[/E]	[E]LocalLineAmount[/E]
-------------------------	------------------------

Element IDs

Element IDs uniquely identify each item of data in a dynamically composed statement, including each field (standalone, non-repeating data), group, table, and row within each table. eaDirect does not assign element IDs to table columns or cells.

Element IDs are transparent to the user, and require customization for any type of implementation with eaDirect.

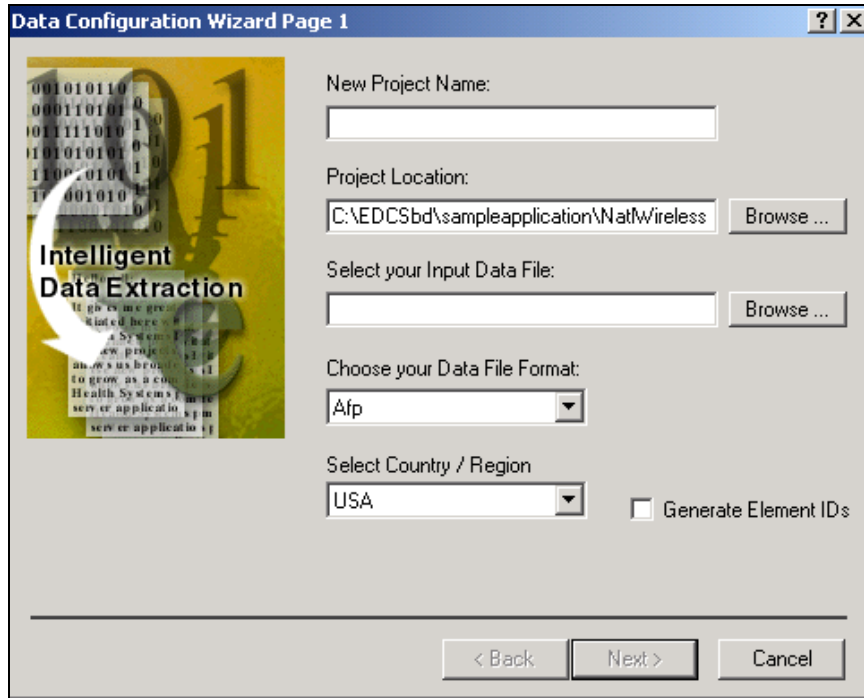
Enable the element ID generation feature in a new or existing DDF if your eaDirect application requires unique line item identification in a view, such as for use with a customized annotation and dispute feature or other custom functionality.

If you are not sure whether you need to turn on this option for a new DDF, consult your application designer or leave the feature off. You can enable or disable the feature at any time if necessary.

Anytime you turn the element ID feature on or off in an existing DDF, you must republish the view. See the *eaDirect Production Guide* for details about publishing version sets.

To enable element IDs when creating a new DDF:

- On Page 1 of the Data Configuration Wizard, select the **Generate element IDs** option.



To enable/disable element IDs in an existing DDF:

1. Open the statement DDF in DefTool.
2. Select **Edit>DDF Properties**. DefTool displays the Edit Properties of DDF dialog.

Defining Data Extraction Rules with DefTool

The screenshot shows the 'Edit Properties of DDF' dialog box. The 'Name of the DDF File' is 'C:\EDCSbd\sampleapplication\NatWireless\NW_LocSummary.ddf' and the 'Name of the Data File' is 'C:\EDCSbd\sampleapplication\NatWireless\NatWireless.txt'. The 'Country/Region Settings' is set to 'USA'. The 'Page Range' is from 'Start Page : 1' to 'End Page : 10'. The 'Generate Element IDs' checkbox is unchecked. The 'Page Info' section has 'Pages are Delimited by' set to 'Regular Expression', 'Delimiting String' set to '1[13]', and 'Delimiting Character' is empty. 'Carriage Control Length' and 'Carriage Control Position' are both set to '0'. The 'Table info' section has 'Delimited by' set to 'CR and / LF' and 'Table Length' set to '0'. The 'Special File Settings' section has the 'End of File by Ctrl_Z' checkbox unchecked. The 'Page Normalization Settings' section has 'CC in Top' selected. The 'Input File Character Coding is' section has 'ASCII' selected.

3. Click the **Edit** button.
4. Enable the **Generate Element IDs** option.
5. Click **OK**.
6. Republish the application view. (See the *eaDirect Production Guide* for details about publishing version sets.)

Defining Statement Presentment with Composer

3

Overview of Creating an ALF with Composer

Before Creating an Application Logic File (ALF)

You must create the following files for your application before you can begin using Composer to create the rules for statement presentment:

- The Data Definition File (DDF) using DefTool
- The HTML templates you plan to use for presentment

You must create a separate ALF for each application view. Become familiar with the different views your application requires before starting to create ALFs.

General Steps for Creating an ALF

To create an ALF:

1. **Run Composer.** From the Windows Start menu, select **Programs>eaDirect>Composer**.
2. **Create a new ALF.** Associate the new ALF with the DDF.
3. **Define the default template** for the view (ALF).

4. **Map field, table, and group definitions to your template.** Composer displays all the data definitions from your DDF in a tree and lets you click and drag the definitions onto your HTML template. Be sure to work in the WYSIWYG template view.
5. **Format mapped fields and tables.** Composer lets you specify font and cell properties.
6. **Customize the ALF** to present the email notification or statement data as needed:
 - **Add content.** Content can include images, text, hyperlinks, HTML code, HTML statement templates, and certain table formatting.
 - **Create any simple or complex data-driven conditions (business logic, called logic elements) on sections or tables in the statement.** A condition contains the data-based business logic that determines which image, text, HTML code, HTML template, or *conditional* table formatting to display.
 - **Specify the HTML template to use for each occurrence in a statement.**
 - **Create any sort, paging, filter, combine, or arithmetic elements and add corresponding element hyperlinks.**
 - **Add any conditional templates or switch ALFs.**
7. **Use the Simulator to view the composed sample statements.** Simulator lets you see how the composed view will appear in a browser. This lets you make necessary changes without running the production jobs to test the application.
8. **Save your ALF and the HTML templates.**
9. **Create any additional ALF your application requires (if any)**

A completed ALF contains all the information about how to assemble an email or statement view. Composer adds eaDirect-specific tags into the HTML templates during the ALF creation process, indicating where data should be modified and where logic arguments need to be applied. The actual logic arguments, as well as other assembly instructions, are stored in the ALF.

Once you have finished creating your application ALFs, you can proceed to test and implement your application in a live production environment. See the *eaDirect Production Guide*.

Completed ALF and HTML Application Files

Be sure to save your ALF whenever you make additions or changes to composition rules. Whenever you save an ALF, Composer automatically saves the open template as well. (You can save the template independently.)

A complete ALF contains the following information:

- File path/name of the associated DDF
- Version of eaDirect
- File path/name of the associated default HTML template

In addition, a complete ALF includes the following information for any features you add, including:

- Content name for:
 - Switch ALFs
 - Alternate HTML statement templates (conditional templates)
 - HTML group templates
 - Business logic-driven content containing images, text, or HTML files
- Conditional statements containing business logic
- Logic elements (pieces of comparative logic you use in conditions)
- Group assembly statements

- Table formatting information
- Sorting configuration details (sort elements)
- Paging configuration details (page elements)
- Filter configuration details, including top/bottom filters (filter elements)
- Arithmetic elements to sum a column or count rows in a table
- Configuration details to combine tables for paging or sorting (combine elements)

The completed HTML template can contain the following eaDirect application features:

- Tags for mapped fields, tables, groups, and charts
- Formatting (except table formatting, which is saved in the ALF)
- Open and close tags for sections of HTML on which you create one or more conditions
- Links to sort, page, and filter elements
- Links to subsequent views, such as a detail view.

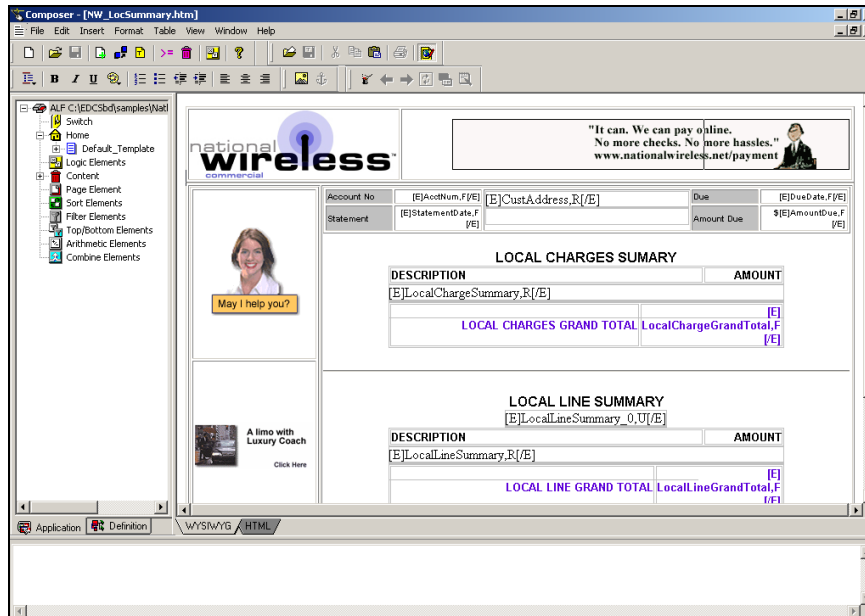
See also “Saving the ALF, HTML Template Files” on Page 280.

The Composer Interface

The Composer interface consists of:

- Menus
- Toolbars
- Application and Definition Trees on the left
- The open HTML template on the right

The Trees, menus, and toolbars often provide more than one way to select a Composer feature or perform a task.



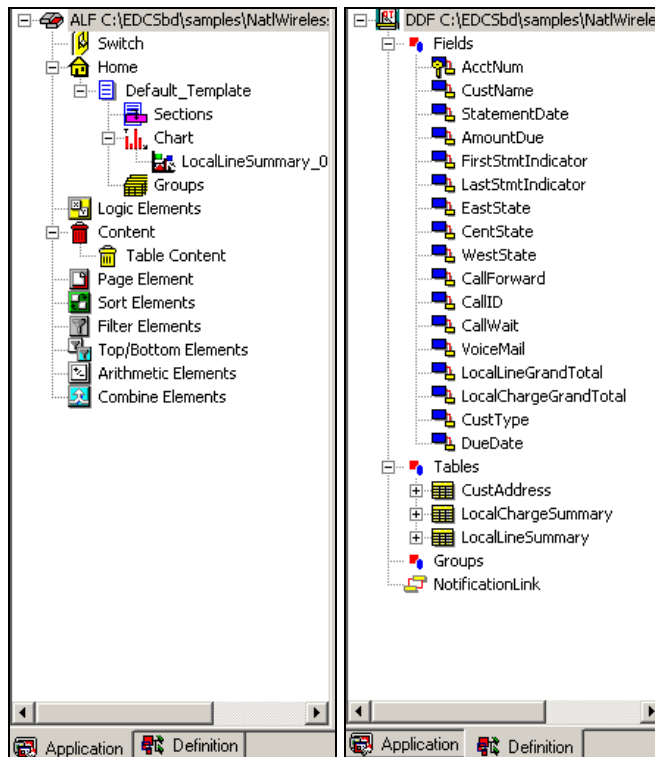
Application and Definition Trees

The left side of the Composer window lets you alternately display either of two trees that let you add data to your statement or create additional features for presentment:

- Application Tree** – The Application tab displays a list of composition rules and features that have been created in the Application Logic File (ALF), such as any switch ALFs or alternate HTML templates added, conditions on sections, content, logic elements, and sorting, paging, filtering, arithmetic, or combine elements. You can right-click on categories and items in the Application tree for a shortcut to creating new features (such as new content) or other design tasks. The Application tree is empty for a new ALF.

- **Definition Tree** – The Definition tab displays an expandable list of all data fields, tables and groups defined in the associated Data Definition File (DDF). You can click and drag items from the Definitions tab to the HTML Template area to map these items to the statement for presentation.

The Application Tree tab appears on the left, the Definition Tree tab on the right:



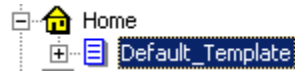
To switch between trees:

- Click the  Definition or  Application tab.

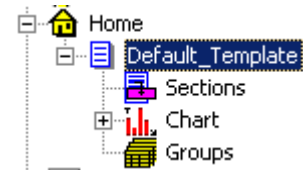
To expand or collapse a tree node:

- Double-click the node name or click the plus (+) or minus (-) sign in front of the node to expand or collapse the list of items.

Collapsed:

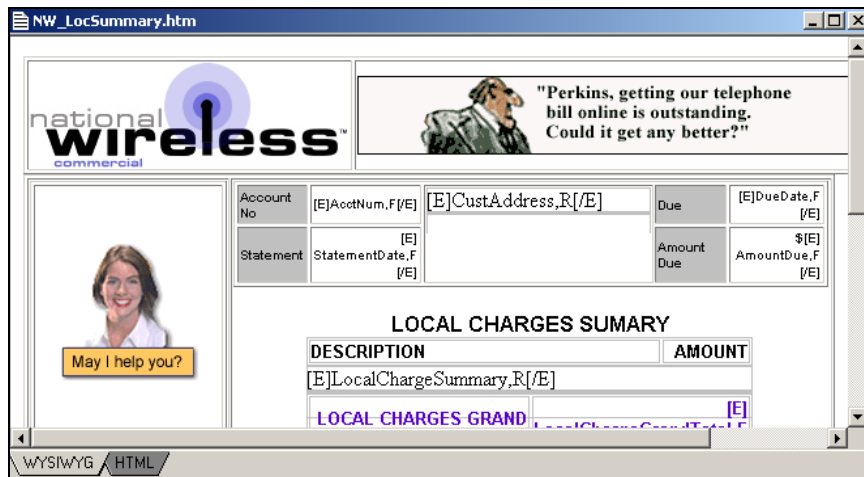


Expanded:



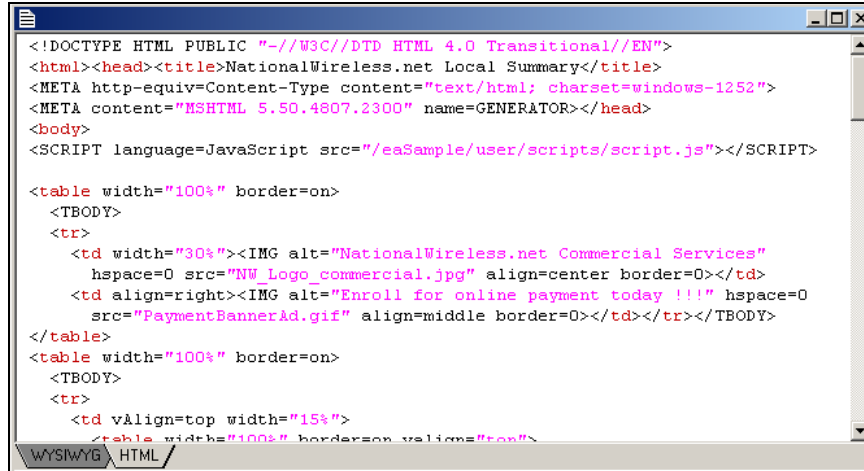
HTML Template Area

The open HTML template appears in the right pane of the Composer window. Use this work area to view and edit your statement graphically in WYSIWYG (what-you-see-is-what-you-get) mode:



You can switch to HTML mode to view and edit the source HTML:

Defining Statement Presentment with Composer



```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html><head><title>NationalWireless.net Local Summary</title>
<META http-equiv=Content-Type content="text/html; charset=windows-1252">
<META content="MSHTML 5.50.4807.2300" name=GENERATOR></head>
<body>
<SCRIPT language=JavaScript src="/eaSample/user/scripts/script.js"></SCRIPT>

<table width="100%" border=on>
  <TBODY>
  <tr>
    <td width="30%"><IMG alt="NationalWireless.net Commercial Services"
      hspace=0 src="NW_Logo_commercial.jpg" align=center border=0></td>
    <td align=right><IMG alt="Enroll for online payment today !!!" hspace=0
      src="PaymentBannerAd.gif" align=middle border=0></td></tr></TBODY>
</table>
<table width="100%" border=on>
  <TBODY>
  <tr>
    <td vAlign=top width="15%">
      <table width="100%" border=on vAlign=top>
```

To switch HTML template edit mode:

- Click the  or  tab.

Menus

File Menu

The Composer File menu let you open, close, or save your current ALF and associated HTML template files, or access recently opened ALFs directly.

Edit Menu

Use the Edit menu for standard edit and text search functionality in HTML or WYSIWYG mode, including Undo and Redo actions on the template.

Insert Menu

The Insert menu lets you insert images, anchors (bookmarks), or HTML text in the template, or create content or logic elements.

Format Menu

Use the Format menu for standard formatting options such as style, font, alignment, indent and bulleted and numbered lists. You can use it to directly edit the HTML template.

Table Menu

Provides standard table options to insert and edit HTML tables in your template (not the data tables you map from your DDF).

View Menu

Use the View menu to define which Composer panes and toolbars you want to view (toggle on/off), view a list of conditions, logic elements, or content, go into edit mode, or select the Simulation options.

The ALF Objects Bar option displays/hides the Composer Application and Definition Trees. (You can also right-click on the white space in the trees and click the Hide button that appears.)

Window Menu

The Window options provide standard tile and cascade capabilities.

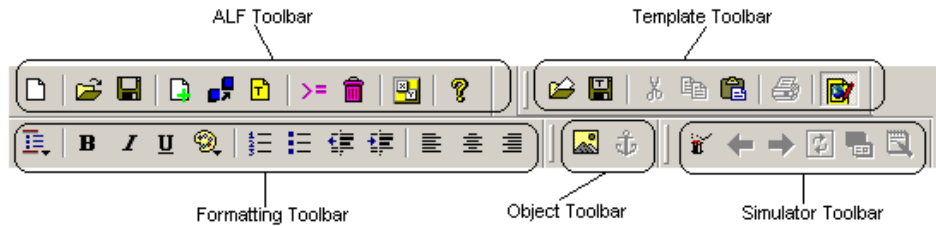
Help Menu

The Help menu provides access to online help topics for Composer and the product About box.

Toolbars

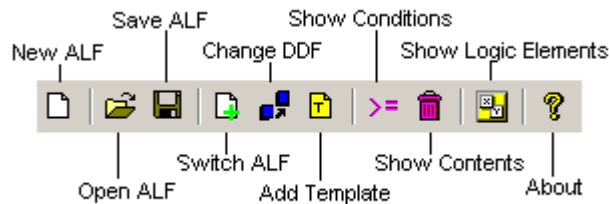
The Composer includes several toolbars, shown below, containing icons that let you quickly access many key functions. To find out what an icon represents, hold the mouse over the icon and the Quick Help description appears.

Defining Statement Presentment with Composer



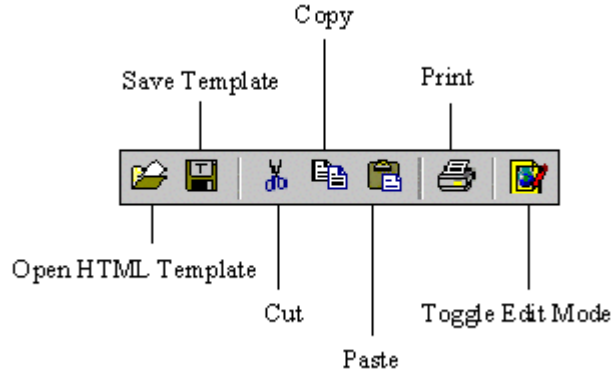
ALF Toolbar

The ALF toolbar icons let you to perform functions associated with the ALF. You can create a new ALF, opening an existing ALF, save the file, create a switch ALF condition, change the associated DDF, view a list of conditions, contents, or logic elements, add a conditional template, or view the Composer About box.



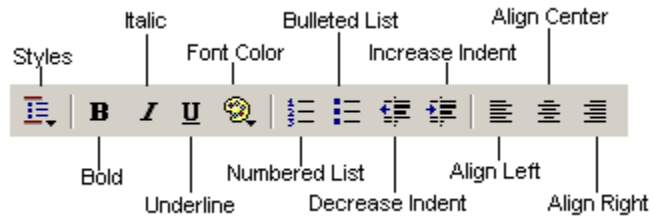
Template Toolbar

The Template toolbar icons affect the open template file. You can open a template, save changes, cut/copy/paste items to it, print, or toggle in and out of edit mode.



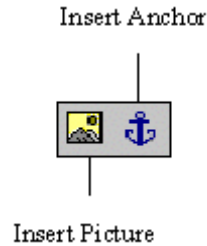
Formatting Toolbar

Use the Formatting toolbar icons to format the template. You can apply various character and paragraph formatting, including bold, italics, underline, color, numbering, bullets, justification, and indents. Use the Style Format icon to apply a variety of formatting.



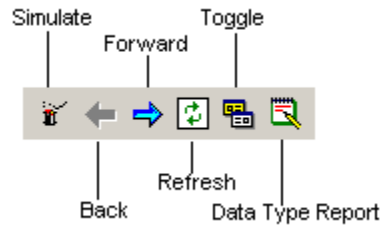
Object Toolbar

The Object toolbar icons let you insert graphics or hyperlinks into the template (first select an object on the template to associate a hyperlink with it).



Simulator Toolbar


The Simulator toolbar icons let you use a test data file to view the statements as they will appear composed in a browser. You can page backward and forward through multiple file documents, verifying that the conditional logic is defined correctly and that all values map correctly.

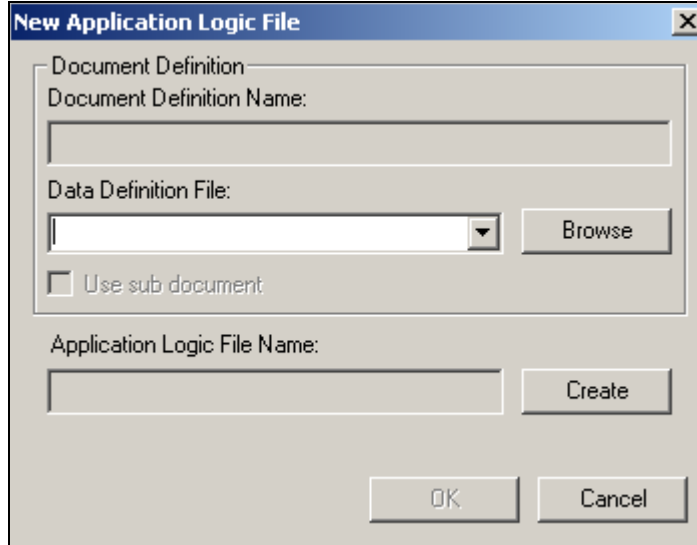


Creating a New ALF

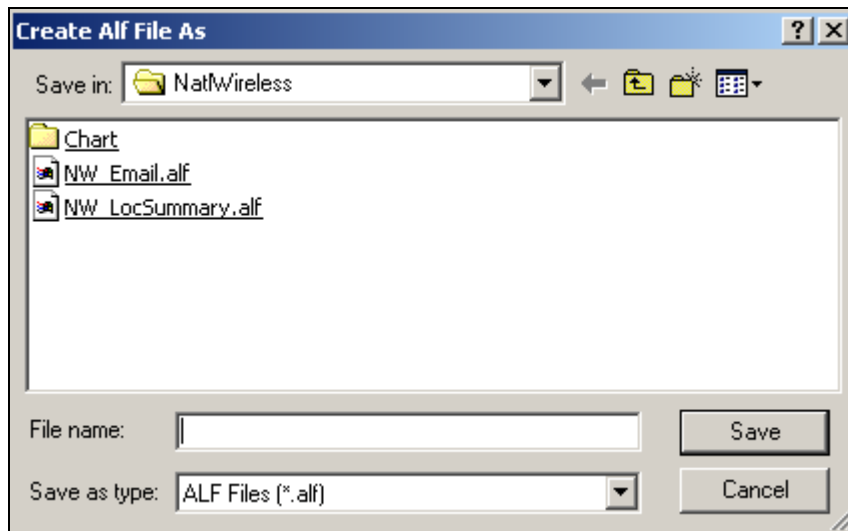
For a View that is Not a Sub-Account

To create a new ALF (statement or email notification):

1. Maximize the Composer window.
2. Select **File>New ALF** or click the **New Document**  icon. The New Application Logic File window opens.



3. Click **Browse** and select the appropriate path to the DDF associated with the ALF. (If the selected DDF has a subkey group field, the Use Sub-document option becomes active; you must select this option for a view designed to present a sub-document. See “For a Sub-Account View” on page 236.)
4. Click **Create**. The Create Alf File As dialog opens.



5. Verify that the Save In field points to the correct folder and enter a meaningful name for the ALF.
6. Click **save**. The New Application Logic File dialog reappears.
7. Click **OK**. The new ALF appears in the Application Tree.

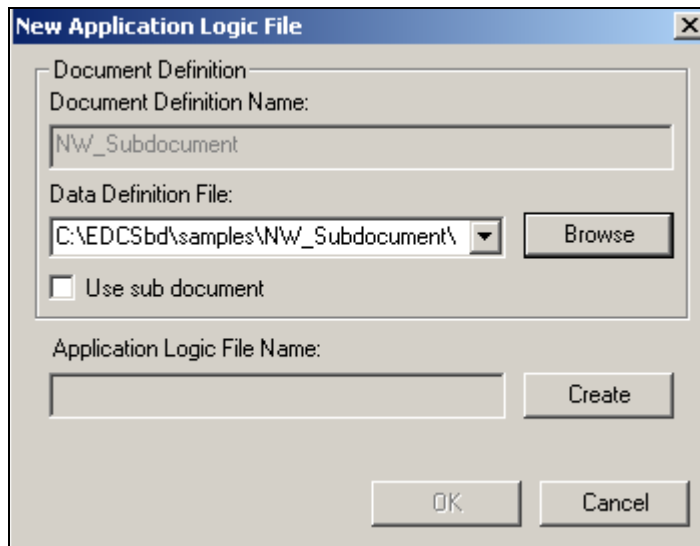
For a Sub-Account View

To use the sub-document indexing feature you **must** create and use a separate Application Logic File (ALF) for presenting a sub-document view.

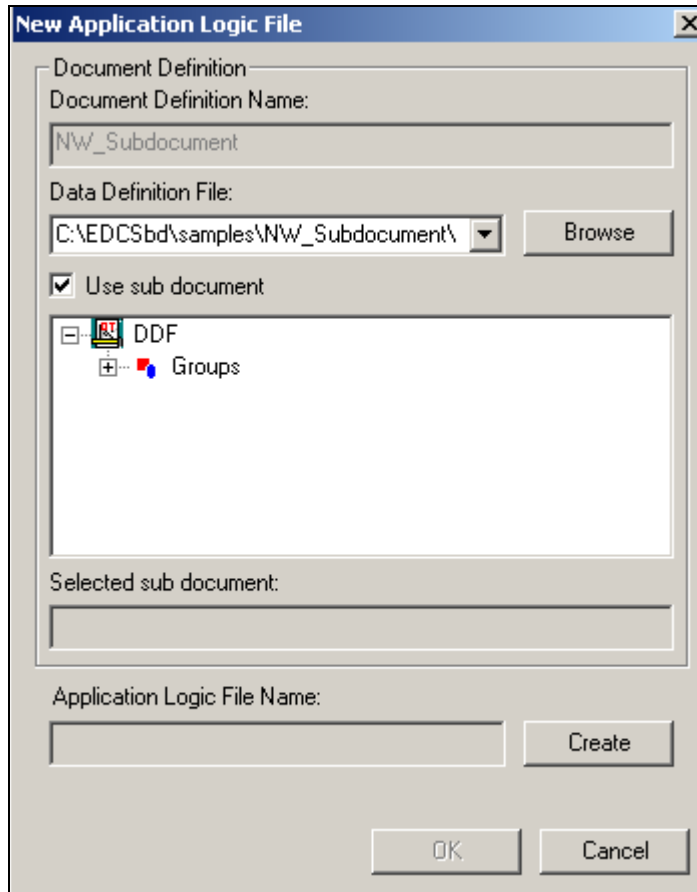
You must also map the group and subkey to the default template. (You do not need to specify a group template for the sub-documents; the group becomes the main document in this view.)

To create a new sub-account ALF and map the data for presentment:

1. In Composer, select **File>New ALF**. Composer displays the New Application Logic File dialog. Select the application DDF you created for this view.



2. Select the **Use sub-document** option. Composer displays a tree showing the sub-document groups in the DDF.




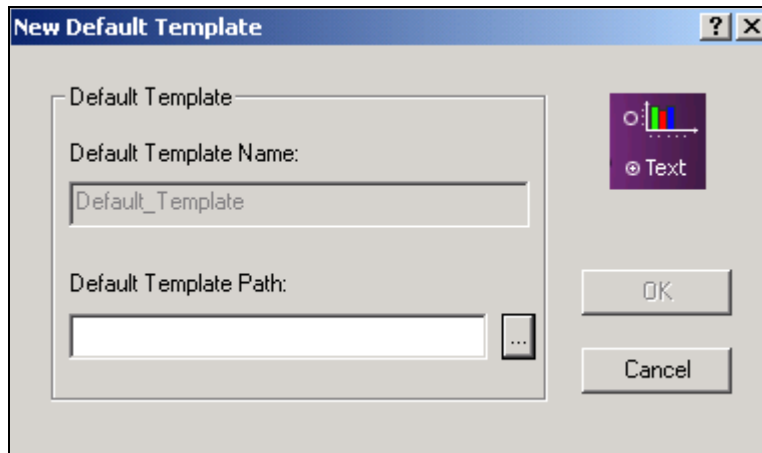
3. Click on the Groups node to expand the list. Select the group containing the subkey for the sub-account you want to use in this ALF view.
4. Click **create**. The Create Alf File As dialog opens.
5. Verify that the Save In field points to the correct folder and enter a meaningful name for the ALF.
6. Click **save**.


7. Click **OK**. Note that in the Definition Tree, Composer converts the sub-document group into pure tables, fields into pure fields, and the subkey to the primary key. Composer treats the sub-document as a normal document, not a group (and there is no group template for the group).

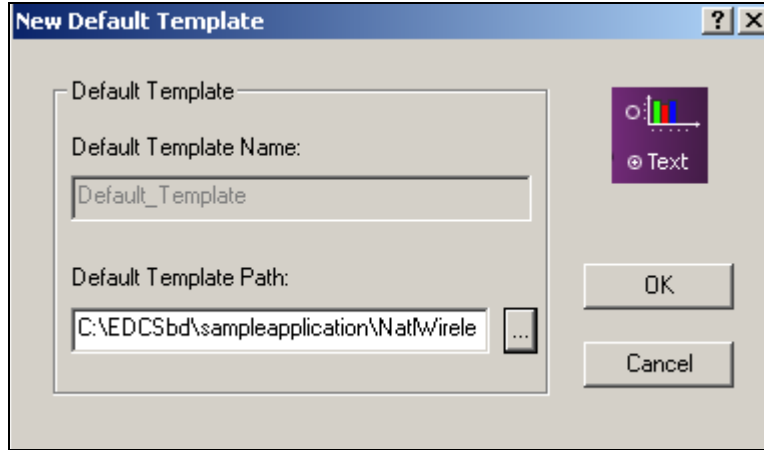
Specifying a Default Template

To specify a default template for a statement or email notification ALF:

1. Right-click the  **Home** icon in the Application Tree.
2. Select **Add Default Template** from the right-click menu. The New Default Template dialog opens.



3. Click the  button next to the Default Template Path. The Open dialog appears.
4. Select the HTML template you want to use as the default.



5. Click **OK**. The template appears in the work area of Composer, and `Default_Template` appears under the Home node in the Application Tree.



To open ALFs created in version 1.8 of Composer or earlier, convert the ALF to the new XML format using `ALFConverter.exe`, found in the `EDCSbd/bin` directory. To open XML files from version 1.9, rename the files with an ALF extension.



Mapping Fields to the Template

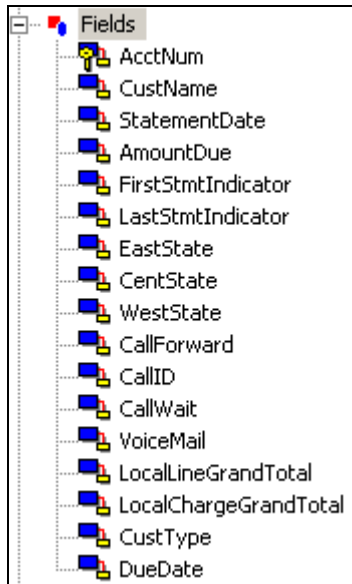
After creating an ALF and specifying the default template, you can map the data fields defined in the DDF to the template for presentment. (You can also map fields to any conditional HTML templates added).

(You can also map tables and groups to the template. See Chapter 4 for details about mapping and formatting tables in Composer.)


To map data fields to a statement or email notification template:

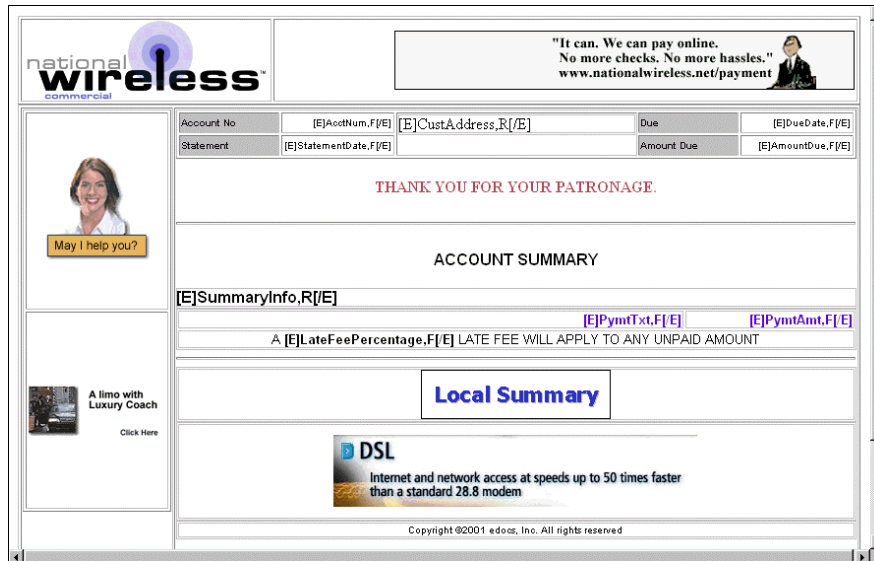
1. Click the **WYSIWYG** tab in the Work Area (if not there already) to display the Default HTML template graphically.


2. Click the  **Definition** tab in the Tree area and double-click the  **Fields** node to expand the list of items in your DDF, for example:



3. Click the field you want to map, then drag and drop it where you want it to appear in the template. (Fields in the template appear as [E] Fieldname, F [/E]. The field assumes the properties of the area in which you place it. “F” indicates the element is a field.)
4. Add any field formatting you want. To format a field, highlight the entire field tag, including the [E] and [/E] tags, and use the Format menu or the Formatting toolbar to apply HTML formatting.
5. You can optionally add a notification link to the template. A notification link creates a direct link to the production web server, based on the configuration settings in the eaDirect Command Center. See “Mapping a Notification Link” on Page 279 for details.
6. If your HTML template used “XX” placeholders for fields or tables, delete them. Select **Edit>Find Text** to search the HTML for any placeholders; verify that you’ve mapped all the required values to the template and deleted all the placeholders.

- Click the  icon to run the Simulator and check your work. Make any changes necessary, running Simulator as often as needed. (The NatlWireless statement template should look like this:)




- Click the Save Template  icon to save the changes to your template.

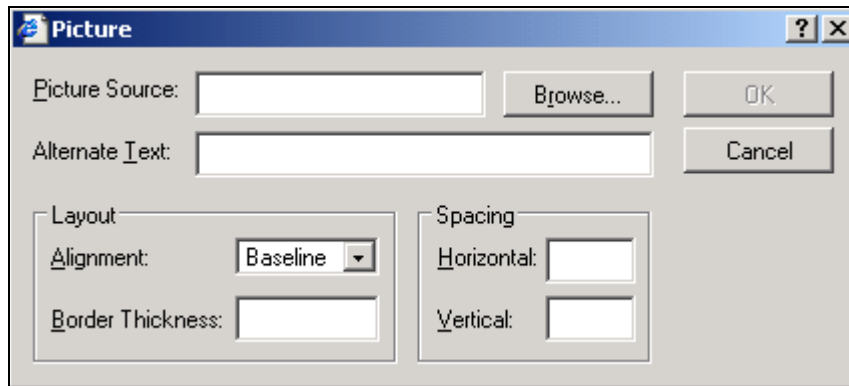
Note that if you delete a field, table, or group from the DDF (using DefTool) after mapping it to an HTML template in Composer, you must manually delete the associated tag in the template.

Inserting Images, HTML Code, Hyperlinks, or HTML Tables

You can insert images, HTML code, hyperlinks or HTML tables into the statement template.

To insert an image:

1. With the ALF and statement template open in Composer, place the cursor on the template where you want to insert the image, and click  or select **Insert>Picture**. The Picture dialog appears.



2. Click **Browse** and select the image.




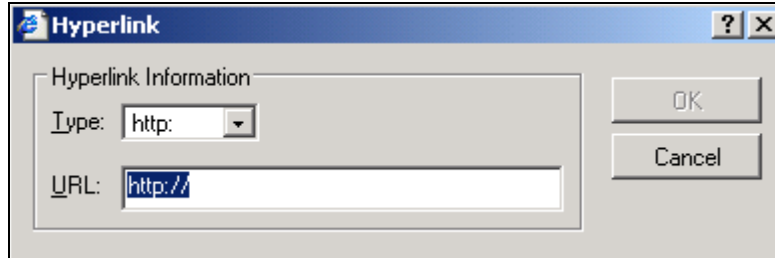
When referencing images or other files in the template, don't use literal paths, such as "img src=c:\images\logo.gif". Instead, use URL addresses, such as:

<http://www.edocs.com/images/logo.gif>

3. Enter any text to use as a possible alternative to the image.
4. Specify the layout and spacing information and click **OK**.

To insert hyperlinks:

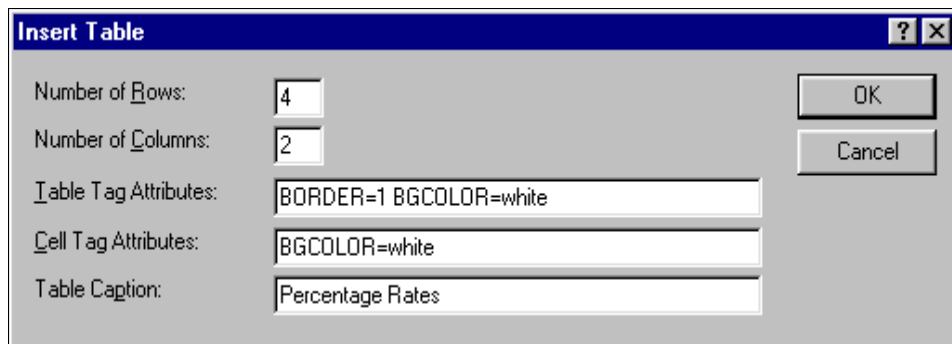
1. With the ALF and statement template open in Composer, select the text or object to use for the link.
2. Click  or select **Insert>Anchor**. The Hyperlink dialog appears.



3. Choose the hyperlink type and enter its URL information.
4. Click **OK**. Text links appear underlined and blue in the template.

To insert HTML tables:

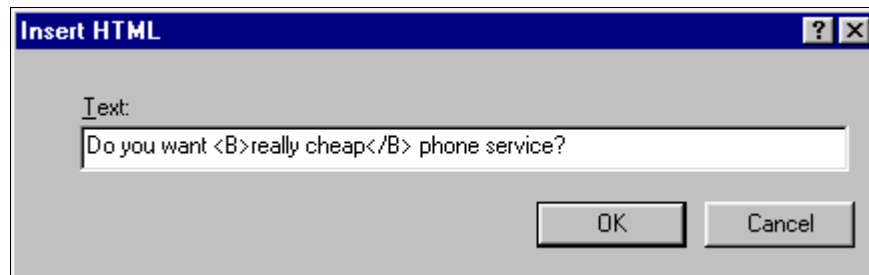
1. With the ALF and statement template open in Composer, place the cursor in the desired location and select **Table>Insert Table**. The Insert Table dialog appears.



2. Specify the number of rows and columns.
3. Specify the table attributes.
4. Specify the cell attributes.
5. Enter a table caption, if desired.
6. Click **OK**.

To insert HTML text:

1. With the ALF and statement template open in Composer, place the cursor in the desired location and select **Insert>HTML**. The Insert HTML dialog appears.



2. Enter the text, with any HTML commands.
3. Click **OK**. The text appears.

Creating Conditions

Composer lets you specify data-driven conditions that can instruct eaDirect to do one of the following when composing an individual statement or email:

- Use an alternate ALF (switch ALF condition)
- Use an alternate HTML template for presenting the statement
- Insert or use alternate images, text, or HTML file in a section of a statement or in an individual table row (conditions on a table can also include font and cell formatting)

For example, you could define a condition that causes a different HTML template to be used depending on the value in the total amount field of a statement. You can target marketing campaigns and promotions to specific users based on the data in their statement.

In addition, a non-data driven condition on a table can apply formatting and content to a select pattern of rows.

The Composer interface lets you build the conditional statements by selecting the individual operators, logic elements, content, etc. Similarly, you can pick and choose components to create individual logic elements to use in your conditions. You can create and name logic elements anytime, making them available to choose from a drop-down list when creating conditional statements.

Creating a condition on a table lets you add images, links, formatting, etc. directly into the line of a statement, letting you focus your marketing campaign at the table data level. Creating a “condition on a line item” lets you format the cells or rows in an alternating pattern you define. See Chapter 4 for information about using conditions on tables.

A simple conditional statement takes the form:

```
IF (logic element) THEN content
```

For example:

```
IF (B2Ccustomer) THEN B2CMessage
```

Creating a condition on a section adds `<EDOCSECTION = SECTIONNAME>` open and close tags to the HTML template.

Creating a Logic Element

A logic element is an expression that compares the values in two data fields or between a data field and a specific value. You use one or more logic elements to build conditional statements in Composer.

You can create logic elements while defining a condition or create logic elements first, making them available to select from a drop-down list when creating the condition. You define and save logic elements with a name, making them available for reuse in other conditions.

A logic element can compare:

- The values in two fields, group fields, or table columns
- The value in a field, group field, or table column to a specific value such as a date, time, amount, NULL or EMPTY, or a regular expression

- A table or group with NULL or EMPTY (to test for existence)

Content is what eaDirect inserts or uses in place of something else in a statement when the condition is true. Content can be an image, text string, HTML section or HTML template within a section of a template. In tables, content can also include font and cell formatting; see “Creating Conditions on a Table (Line Item)” on Page 287 for a description of content you can apply to tables (table content). (Any alternate HTML templates or ALFs are “content” for those types of conditions only.)

You can create conditions with multiple logic elements using the AND operator or the OR operator, for example:

```
IF (B2Ccustomer AND AmountDue1000) THEN B2Cmessage
```

Additional conditions in a statement must start with ELSE IF.

Logic element syntax

Each logic element requires the following information and syntax:

- **The data type;** DATE, TIME, CURRENCY, INTEGER, or DECIMAL (not necessary for a string).
- **An open parenthesis** (not needed if the data type is a string).
- **The field, group field, table, column, or group** in format /BillDate [] .
- **The data format** if the data type is DATE, TIME, CURRENCY, or DECIMAL. For DATE or TIME you must manually enter any separators, such as forward slashes in the date: %d/%m/%Y.
- **A close parenthesis** (not needed if the data type is a string).
- **An operator for the condition;** ==, != (not equals), <, >, <=, >=, ~, or TRUE, AND, OR or NOT. (Use the tilde “~” before a regular expression.)
- **A value to compare with the first part of the logic element.** This can be another field, group field, table, column, specific date, time, integer, or string, or a regular expression, along with the data type and format (plus separators), where appropriate.

The following examples of logic elements show a variety of logic comparisons:

```
/FirstStmntIndicator [] !=NULL
```

```
/CustType [] == "B2B"
```

```
/SummaryInfo [] /SummaryInfoCR [] == "CR"
```

```
DATE (/StatementDate [], "%d/%m/%Y") >= DATE  
("11/18/2002", "%d/%m/%Y")
```

```
CURRENCY (/CurrentCharges [], '.', "") > CURRENCY ("200.00",  
'.', "")
```

You can use a relative path if the section (condition) is on extracted data in the table/group HTML. You can always use an absolute path.


Creating logic elements on group fields

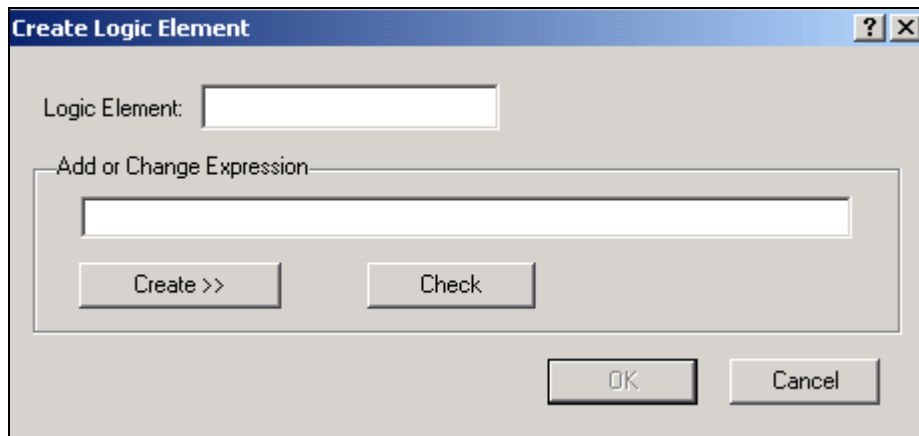
You can create a logic element on sub-account group fields, including a subkey, or on a group field and a table column. These elements can be at different levels, but must both be within the same sub-account group hierarchy.


In the case where you have nested sub-account groups, any conditions you create on the nested sub-account group apply only to that group and not to the higher-level, or parent sub-account group. Any condition you create on the parent sub-account group likewise applies to the parent sub-account only, and not to any nested sub-account groups. (Nested sub-account groups act like regular groups since they are just a subset of the parent group view.)

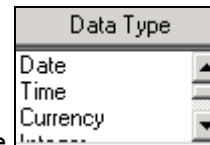
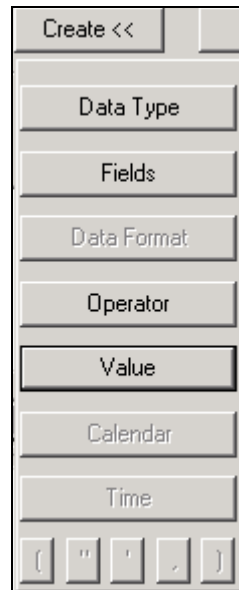
If a condition on a group field adds a section to a group template, the group field must belong to that group or to a child group of the one where the section appears. The condition cannot add a section to a group that is a child of the group the field belongs to.

To create a logic element:

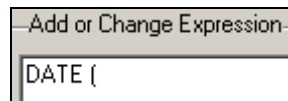
1. With the ALF open in Composer, right-click the  Logic Elements node in the Application Tree and select **Create Logic Element** from the right-click menu (or click the appropriate Create button while creating a condition). Composer displays the Create Logic Element dialog.



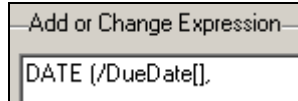
2. Type a meaningful name for the logic element in the Logic Element box.
3. Click  to display the buttons for selecting fields, operators, dates, etc. to add to the logic element. Select or enter items in the sequence you want them to appear in the logic element. (You can manually enter the logic element in the Add or Change Expression box, but choosing from the menus reduces typing and syntax errors.)



4. If comparing a field or column, click **Data Type** and select the type of data. It is not necessary to select a data type for strings. Composer adds the data type and an open parenthesis to the expression:

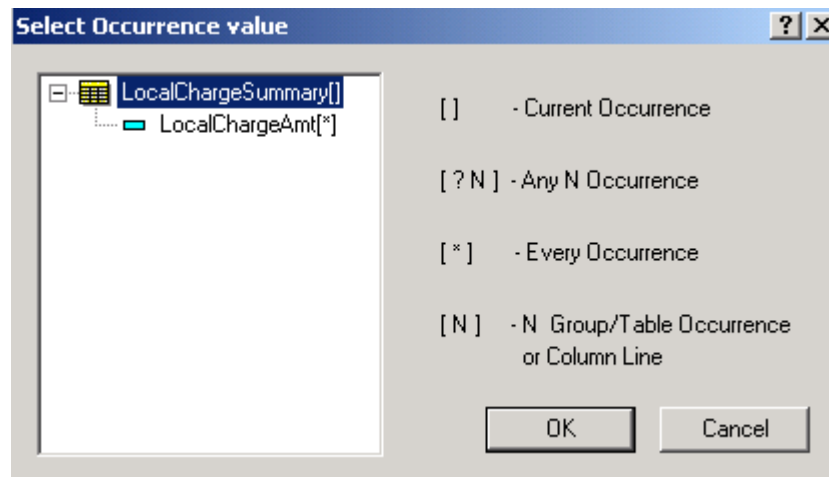


5. Click **Fields**, expand the list, and select the field, table, column, group, or group field you want to compare. Composer adds the column name (and a comma, if data type is not a string) to the expression:

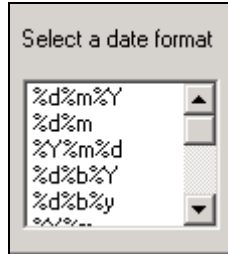


6. If you choose a table column, the Set Occurrence Value dialog appears. Specify the row occurrences you want eaDirect to check. To specify a basis other than the default (empty brackets default to Current Occurrence), click on the table name and edit the value between brackets [] as described here:
- **[] Current Occurrence** – Evaluate each row in the table and apply the content to each qualifying row
 - **[?N] Any N Occurrence** – If N rows of qualifying data exist in the table, then apply the content
 - **[*] Every Occurrence** – Every row must meet the condition, then apply the content
 - **[N] N Group/Table Occurrence or Column Line** – If the Nth table row qualifies, then apply the content

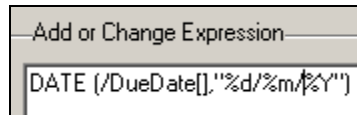
Be sure and specify the same row occurrences on both sides of a logic element if you are comparing two column values.



- If the data type of the comparison field or column is a date, time, currency, or



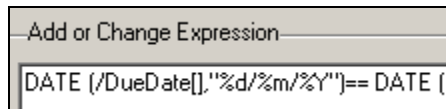
decimal, click **Data Format** and select the format of the data. (For additional information about valid format strings you can enter, see Appendix A.) Composer adds the data format and a close parenthesis to the expression. Manually add any required separators.



- Click **Operator** and select an operator from the dropdown list. (Choose the tilde “~” operator with a regular expression.) Composer adds the operator to the expression.



- Click **Data Type** and select the same data type you selected in the first part of the logic element (if any). Composer adds the data type and an open parenthesis to the second part of the expression.



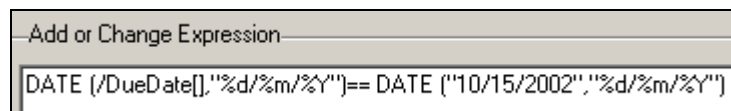
10. Specify the field, group field, column, date, time, or value for comparison on the right side of the logic element expression. You can only compare groups and tables with NULL or EMPTY.

- **To compare the data to a field or column**, click **F**ields and choose the field or column, adding any necessary separators. (Note that if you are comparing group fields, this field must be higher in the hierarchy than the one on the left side of the expression.) Composer adds the column name in quotation marks, followed by a comma, to the expression.
- **To specify a date for comparison**, click **C**alendar and choose a specific month, year, and day. Composer adds the selected date in quotation marks, followed by a comma, to the expression.

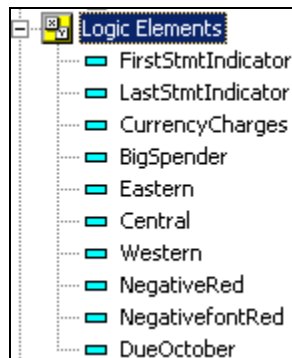
March		2002				
Sun	Mon	Tue	Wed	Thu	Fri	Sat
24	25	26	27	28	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

- **To specify a time for comparison**, click **T**ime. Click the hour, minutes, seconds, or AM/PM to highlight, and use the up and down arrows to specify the value: . When the time you want appears, double-click to enter the time in the expression. Composer adds the selected time in quotation marks to the expression.
- **To specify an integer, decimal, or string value for comparison**, manually type the value between quotation marks followed by a comma. Add a close parenthesis after an integer value. (Click buttons or type directly into the expression.)

- **To compare a string, table, or group to NULL or EMPTY**, click **Value** and select NULL or EMPTY. Composer adds the word NULL or EMPTY to the expression.
11. If the data type is a date, time, currency, or decimal, click **Data Format** and select the same format you selected in the first part of the expression, manually entering any required separators. Composer adds the data format, followed by a close parenthesis, to the expression.

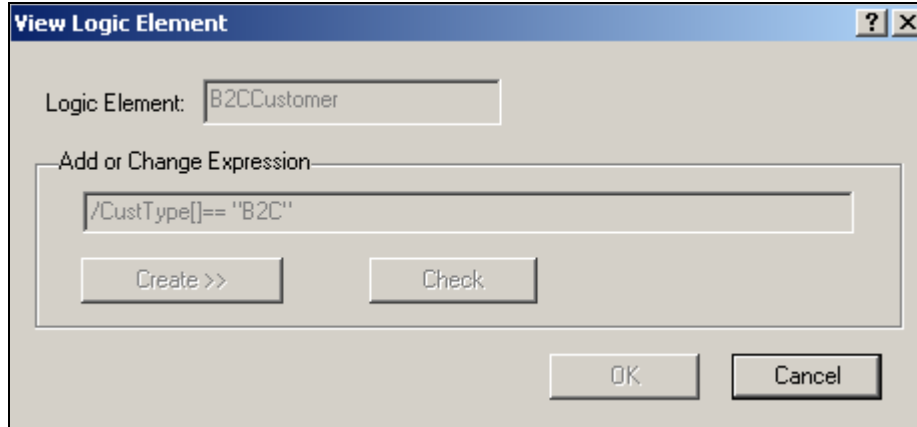



12. When the logic element expression is complete, click **Check** to verify that the logic element is valid. Correct or complete the logic element as needed, checking again to verify any changes.
13. Click **OK**. Composer adds the logic element to the list in the Application Tree.

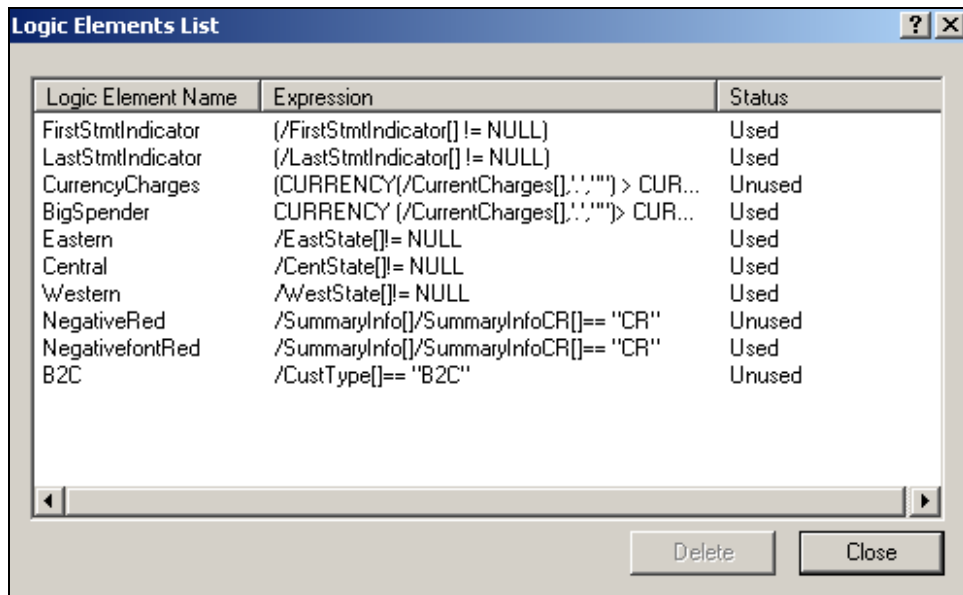


To view logic elements:


- **To view a particular logic element expression**, with the ALF open in Composer, expand the Logic Elements node and right-click the logic element name in the Application tree. Select **View Logic Element** from the right-click menu.

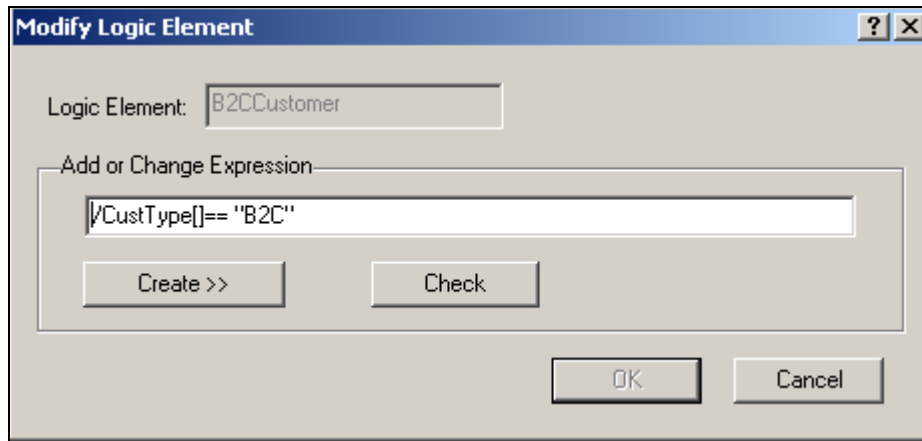


- **To view a list of all logic element expressions**, with the ALF open in Composer, right-click the  Logic Elements node in the Application Tree and select **Logic Elements List** from the right-click menu. Composer displays the Logic Elements List.




To edit a logic element expression:

1. With the ALF open in Composer, expand the  Logic Elements node and right-click the logic element name in the Application tree.
2. Select **Edit Logic Element** from the right-click menu. Composer displays the Modify Logic Element dialog.



3. Type your edits or use the Create drop-down menu. See Creating a Logic Element on Page 245 for details.
4. Click **Check** to verify that the edited expression is valid.
5. Click **OK**.

To delete an unused logic element:

1. With the ALF open in Composer, right-click the  Logic Elements node and select **Logic Elements List** from the right-click menu. Composer displays the Logic Elements List.
2. Select the unused logic element and click **Delete**.
3. Click **Close**.

Creating an ALF Switch Condition



You can create a data-driven condition that instructs eaDirect to compose and present a customer's statement using an entirely different ALF.

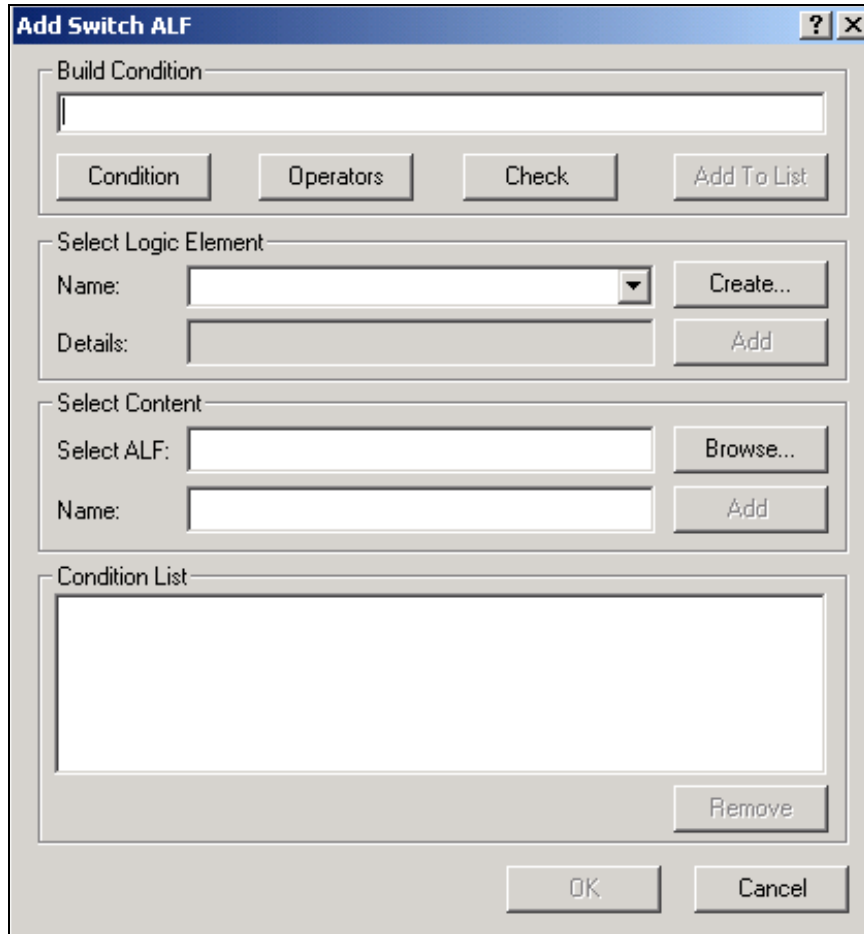
An ALF switch condition gives you the greatest level of conditional control over what you present to a customer. Using the same DDF, you can create a completely different-looking statement and present an alternate set of data for targeted customers. For example, you could create a condition to specify switching ALFs for larger customers whose data requires different table or charting formats.

Before creating a switch ALF condition, you must determine whether it makes more sense to use entirely separate views or to create a condition on a view.

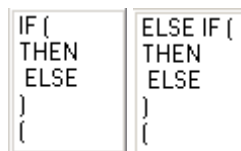
Each ALF switch condition requires one or more logic elements. A logic element is an expression that compares the values in two data fields or between a data field and a specific value. You can create logic elements while defining a condition or create logic elements first, making them available to select from a drop-down list when creating the condition. See [Creating a Logic Element](#) on Page 245 for details.

To create an ALF switch condition:

1. Right-click the  **Switch** node in the Application Tree and select **Add ALF** from the right-click menu, or click the **Switch ALF**  icon in the ALF toolbar. The Add Switch ALF dialog appears.



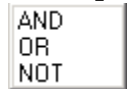
2. Use the options on this dialog to “build” a conditional statement in the Build Condition field. (You can manually enter the text in the Build Condition field, but choosing from the menus reduces typing and syntax errors.)



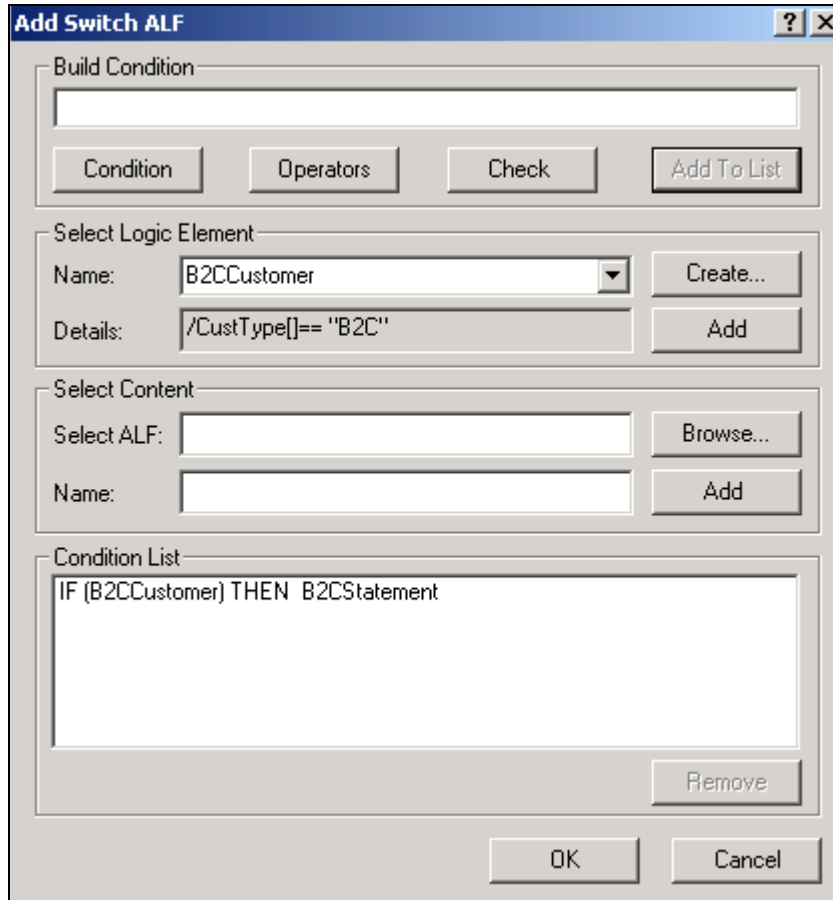
3. Click **Condition**. If this is the first ALF switch you’re creating, select **IF** from the drop-down list. If you’ve already added one or more ALF switch conditions, select **ELSE IF**.

4. Under Select Logic Element, click **Create** to create a logic element for the ALF switch; see “Creating a Logic Element” on Page 245 for details. If you have already created the logic element, choose it from the Name drop-down list and click **Add** to add it to the condition.

5. Click **Operators** to select any operators your conditional statement requires



6. Add any additional logic elements the condition requires.
7. Click **Condition** again and select the right parenthesis from the drop-down list. One set of parenthesis must enclose all logic elements in a condition.
8. Click **Condition** and select **THEN**.
9. Click **Browse** to select the ALF (content) to switch to.
10. Type a name for the switch ALF content in the Name field and click **Add** to add it to the condition.
11. When the switch ALF condition is complete, click **Check** to verify that it is valid. Correct or complete the statement as necessary, checking the condition again.
12. When the condition is complete and valid, click **Add to List**. Composer displays the condition in the Condition List at the bottom of the dialog.



13. Click **OK**. The added switch condition appears in the Application Tree:




To view properties of all ALF switch conditions:


1. With the ALF open in Composer, right-click **IF= Condition** below the **Switch** node in the Application Tree.

2. Choose **View Properties** from the right-click menu. Composer displays the Switch Condition dialog showing all ALF switch conditions defined.
3. Click **OK**.


To edit an ALF switch condition:

1. With the ALF open in Composer, right click the  **Switch** node in the Application Tree.
2. Select **Modify Condition** from the right-click menu. Composer displays the Modify Switch ALF dialog.
3. Double-click the condition from the Condition List and edit it as necessary.

To change the ALF used in a switch condition only:

1. With the ALF open in Composer, right click the content name of the ALF listed under **IF= Condition** below the  **Switch** node in the Application Tree.
2. Select **Change** from the right-click menu.
3. Composer asks if you are sure you want to change the ALF. Click **Yes**.
4. Choose a new ALF from the Change ALF dialog and click **OK**.

To delete an ALF switch condition:



1. With the original ALF open in Composer, right click the content name of the ALF listed under **IF= Condition** below the  **Switch** node in the Application Tree.
2. Select **Delete** from the right-click menu.

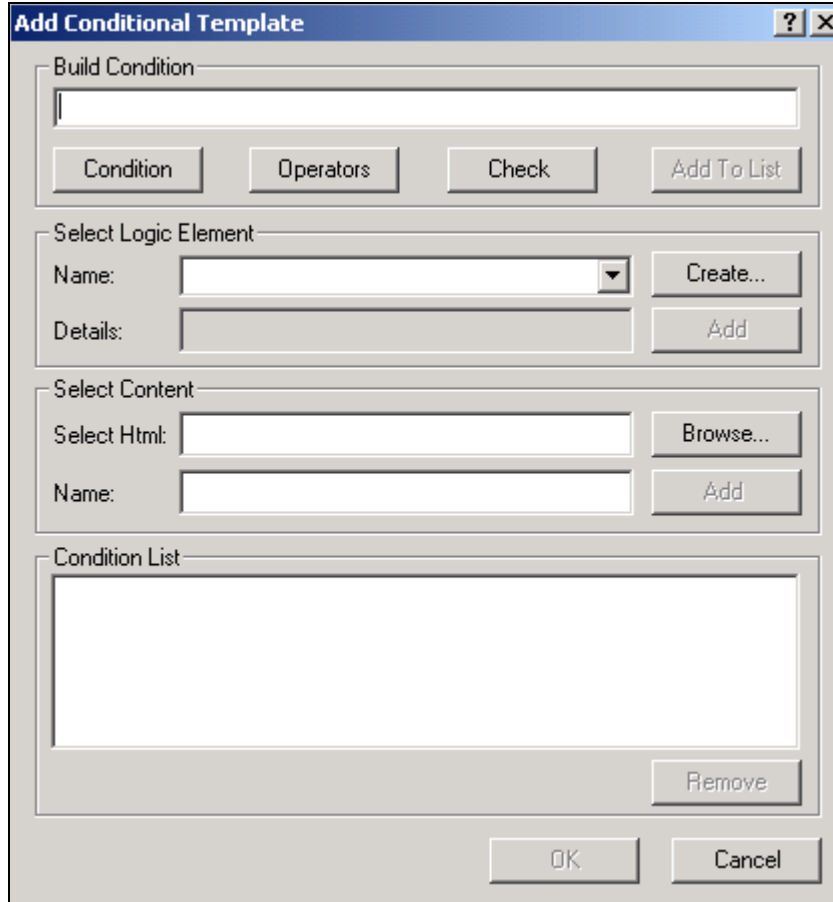
Adding Conditional Templates

You can create a condition that instructs eaDirect to compose a statement or document using an alternate template (not the default template) under specific data-driven circumstances. This lets you customize the presentation of statements based on the data they contain.

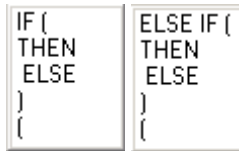
Each condition requires one or more logic elements. A logic element is an expression that compares the values in two data fields or between a data field and a specific value. You can create logic elements while defining a condition or create logic elements first, making them available to select from a drop-down list when creating the condition. See [Creating a Logic Element on Page 245](#) for details.

To add a conditional HTML template:

1. With the ALF open in Composer, right-click the  Home node in the Application Tree and select **Add Template** from the right-click menu, or click . Composer displays the Add Conditional Template dialog.



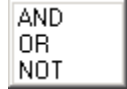
2. Use the options on this dialog to “build” a conditional statement in the Build Condition field. (You can manually enter the text in the Build Condition field, but choosing from the menus reduces typing and syntax errors.)



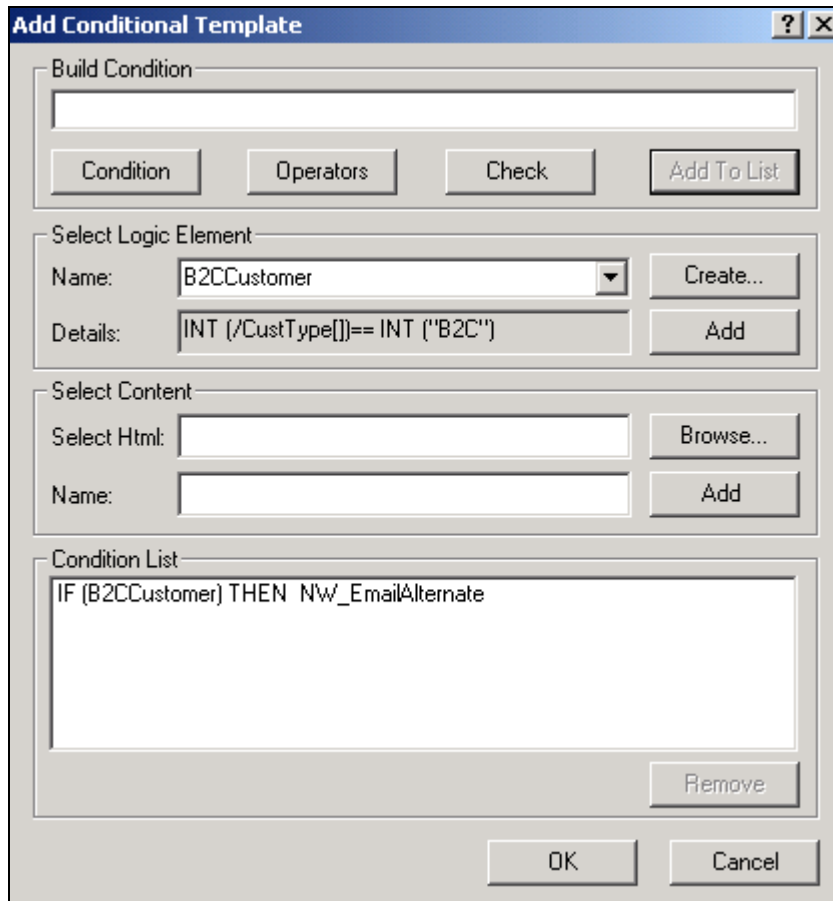
3. Click **Condition:** If this is the first alternate template you’re adding to the ALF, select **IF** from the drop-down list. If you’ve already added any additional templates, select **ELSE IF**.

4. Under Select Logic Element, click **Create** to create a logic element for the template condition; see Creating a Logic Element on Page 245 for details. Click **Add** to add it to the condition.

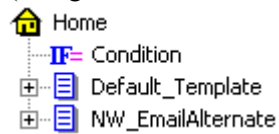
5. Click **Operators** to select any operators your conditional statement requires





6. Add any additional logic elements the statement requires.
7. Click **Condition** again and select the right parenthesis from the drop-down list. One set of parenthesis must enclose all logic elements in a condition.
8. Click **Condition** and select **THEN** from the drop-down list.
9. Click **Browse** to select the alternate HTML template file (content).
10. Type a name for the alternate HTML template in the Name field and click **Add** to add it to the condition. Names must not contain spaces or special characters, and must start with an alpha character. The names must not exceed 30 characters in length.
11. When the condition is complete, click **Check** to verify that it is valid. Correct or complete the statement as necessary, checking the condition again.
12. When the condition is complete and valid, click **Add to List**. Composer displays the condition in the Condition List at the bottom of the dialog.




13. Add any additional conditional templates, if necessary.
14. Click **OK**. Composer adds an **IF=** Condition node listing the template conditions (along with the default template) for the ALF:



To view properties of all template conditions:

1. With the ALF open in Composer, expand the  Home node in the Application Tree and right-click  Condition .
2. Choose **View Properties** from the right-click menu. Composer displays the Template Condition dialog showing all template conditions defined.
3. Click **OK**.

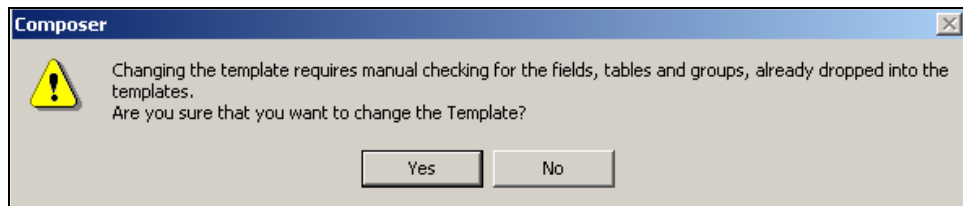
To edit a template condition:

1. With the ALF open in Composer, right-click the  Home node.
2. Select **Modify Condition** from the right-click menu. Composer displays the Modify Conditional Template dialog.
3. In the Condition List, double-click the condition you want to change and edit as necessary.

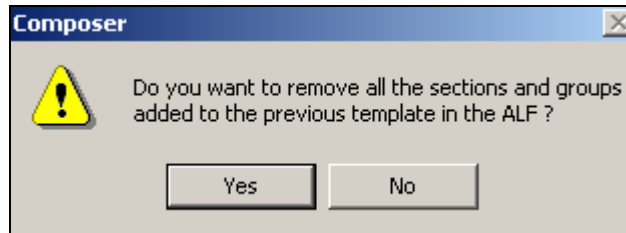
To change the default template file or the template specified as a conditional alternate:

You can change the HTML template file you want to use as the default for a view. You can also change the template file specified in a condition as an alternate template.

1. With the ALF open in Composer, right-click the content name of the template in the Application Tree.
2. Select **Change** from the right-click menu. Composer displays the following message:



3. Click **Yes**. Composer displays the Change Template File dialog.
4. Choose the new template you want to use. If you added sections and groups to the old template in the ALF, Composer displays the following warning:



5. Click **Yes**. Composer removes all the sections and groups defined on that template from the ALF.

To delete a conditional template from an ALF:

1. With the ALF open in Composer, right-click the template you want to delete.
2. Select **Delete** from the right-click menu. Composer asks if you are sure you want to delete the template.
3. Click **Yes** to delete the template. Composer removes the template from the Application Tree.

Creating a Condition on a Statement Section

You can create a data-driven condition on a selected area of a statement that instructs eaDirect to replace the section with a different image, HTML file, text content, or NULL when composing the statement. This lets you tailor advertisements or marketing messages based on the data in a customer's statement.

You can define a condition on any section of a template, such as a block of text, or an inserted field, or image.

Each condition requires one or more logic elements. A logic element is an expression that compares the values in two data fields or between a data field and a specific value. You can create logic elements while defining a condition or create logic elements first, making them available to select from a drop-down list when creating the condition. See *Creating a Logic Element* on Page 245 for details.

You must identify each image, block of text, and HTML file you plan to use in a conditional statement as content in Composer. You must name each item you want to conditionally insert in the template. The content name acts as a pointer to the image, text, or HTML file. You can create content while building a condition on a section or create content independently; see “*Creating Content*” on Page 274.

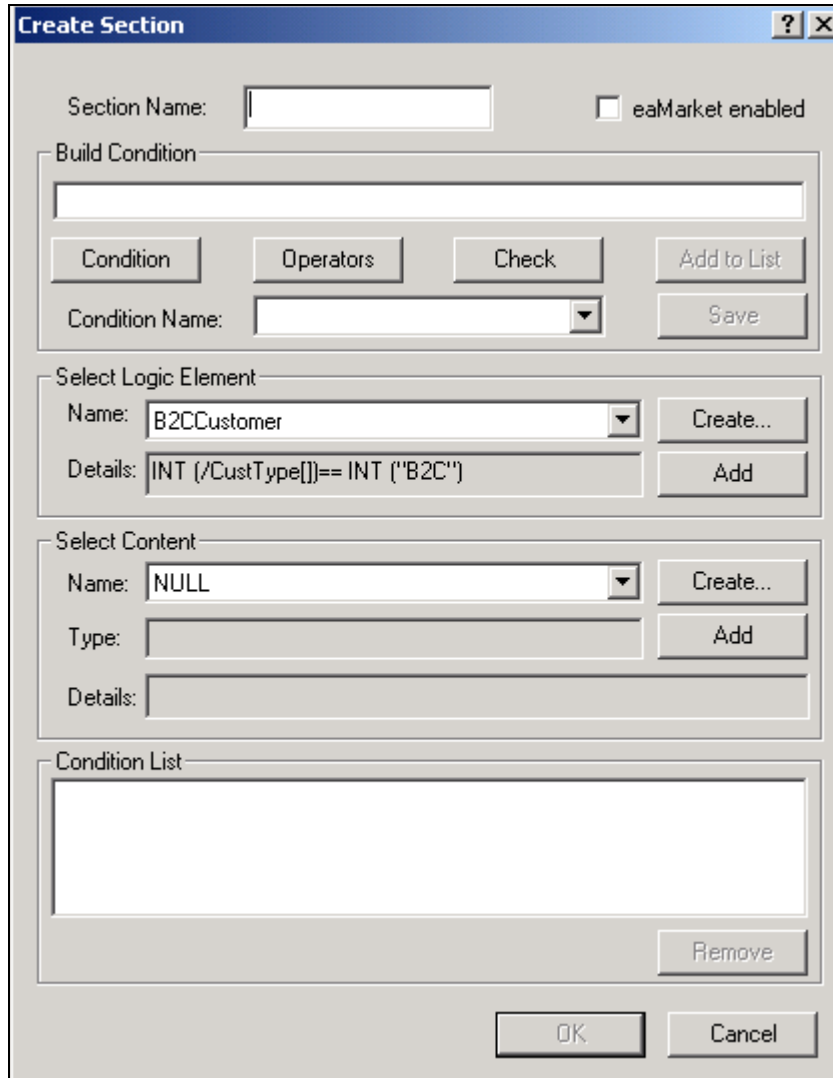
You can create compound conditions on a section (IF THEN...ELSE IF) and add additional conditions at any time. To create a compound condition on a section, you must build, check, and add the first part of the expression and the ELSE IF portion separately. For example, to create the compound statement:

```
IF (StateFL) THEN OrangeJuiceAd ELSE IF (StateGA) THEN PecanPieAd
```

you must build and check “IF (StateFL) THEN OrangeJuiceAd” and add it to the conditions list. Then you can build, check, and add “ELSE IF (StateGA) THEN PecanPieAd” to the list of conditions for the section.

To create a condition on a statement section:

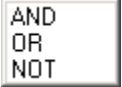
1. With the ALF open in Composer, select the section (text, image, etc.) on which you want to create a condition. (Select **THANK YOU FOR YOUR PATRONAGE** on the default template in NatlWireless.alf).
2. With the section highlighted, right-click and select **Add Section** from the right-click menu. The Create Section dialog appears:



3. Enter a name for the section. Names cannot contain spaces or special characters, and must start with an alpha character. Maximum length is 30 characters. Use meaningful names; “BigSpenderInsertAirlineOffer” is more descriptive and easier to identify than “ALFsection1”.

4. Use the options on this dialog to build a conditional statement in the **Build Condition** field. You can manually enter the text in the Build Condition field, but choosing from the menus reduces typing and syntax errors. (You can optionally choose a previously saved condition from the drop-down list if you've already created a condition you want to reuse in this section. If so, skip to Step 13.)



5. Click **Condition** and select **IF** from the drop-down list.
6. Under Select Logic Element, click **Create** to create a logic element for the condition; see “Creating a Logic Element” on Page 245 for details. If you have already created the logic element, choose it from the Name drop-down list and click **Add** to add it to the condition.
7. Click **Operators** to select any operators your conditional statement requires
 
8. Add any additional logic elements the conditional statement requires.
9. Click **Condition** again and select the right parenthesis from the drop-down list. One set of parenthesis must enclose all logic elements in a condition.
10. Click **Condition** and select **THEN**.
11. Choose the content to insert from the **Name** drop-down list under Select Content, or click **Create** to create new content. See “Creating Content” on Page 274 for details.
12. Click **Add** to insert the content name in the condition.

Defining Statement Presentment with Composer

Create Section [?] [X]

Section Name: eaMarket enabled

Build Condition

Condition Name:

Select Logic Element

Name:

Details:

Select Content



Name:

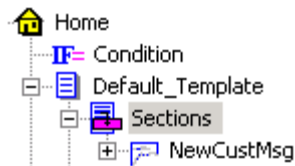
Type:

Details:

Condition List


13. When the section condition is complete, click **Check** to verify it is valid.
Correct or complete the statement as necessary, checking the condition again.
14. If you want the conditions on this section to be available in eaMarket, click the **eaMarket enabled** checkbox.

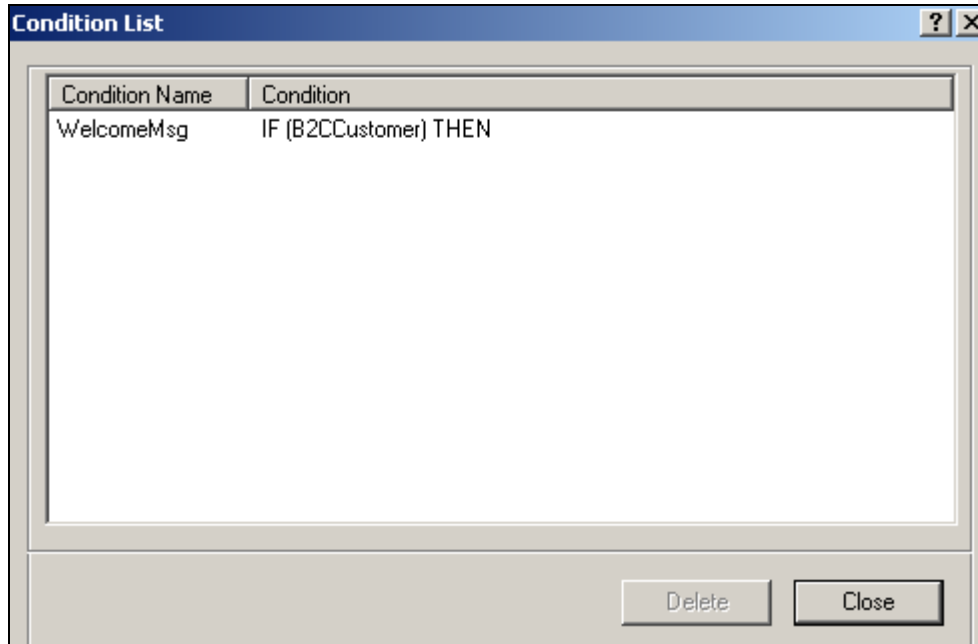
15. When the condition is complete and valid, click **Add to List**. Composer displays the condition in the Condition List at the bottom of the dialog.
16. If you want to save the condition for possible reuse in another section, click the condition in the Condition List and type a name for the condition in the Condition Name drop-down box (next to the Add to List button). Click **Save**.
17. If necessary, build any additional conditions to create a compound condition on the section. Begin subsequent conditions with ELSE IF.
18. Click **OK**. The condition appears in the  **Sections** node for the template. (If you eaMarket enabled a section, it appears with the eaMarket enabled section  icon instead.)






Deleting, Modifying, and Viewing Conditions

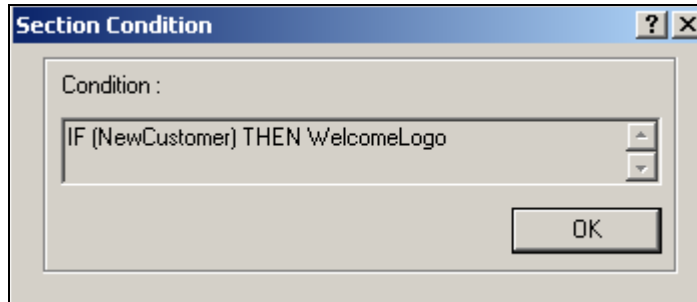
To display a list of saved conditions:

- Click the Show Conditions icon  on the ALF toolbar. (Note table conditions are not included.)





To view the properties of a condition:

1. With the ALF open in Composer, double-click the  Sections node (below the  Home node). Composer displays the conditions (sections) defined in this template.
2. Double-click the name of the condition (section) you want to view.
3. Right-click on the  Condition node and select **View Properties** from the right-click menu. Composer displays the Section Condition dialog with the conditional statement.





To modify a condition on a section:

You can modify a condition on a section or add one or more ELSE IF conditions to a section (creating compound conditions).

1. With the ALF open in Composer, double-click the  Sections node (below the  Home node).
2. Right-click the name of the condition you want to modify.
3. Select **Modify Section** from the right-click menu. The Modify Section dialog appears.
4. To modify an existing condition, double-click on the condition and edit as needed. To add a new condition to the section, follow the procedures above to add an ELSE IF condition.

To delete a condition on a section:

1. With the ALF open in Composer, double-click the  Sections node (below the  Home node).
2. Right-click the name of the condition you want to delete.
3. Select **Delete Section** from the right-click menu.
4. Composer asks if you're sure you want to delete the section. Click **Yes**.

Creating Content

Content is any image, “block” of text, or HTML that you conditionally insert in a section of a statement based on the user’s data. You must identify each item you plan to use in a data-driven condition as content, and give each a name. The content name acts as a pointer to the image, text, etc.

You must define each of the following items as content to use them in a condition:

- Images
- Blocks of text
- HTML files
- Alternate (switch) ALFs
- Alternate HTML template files


By naming content, it becomes available to select from a drop-down list when building conditions on a section. (You can also create new content when building a condition.)

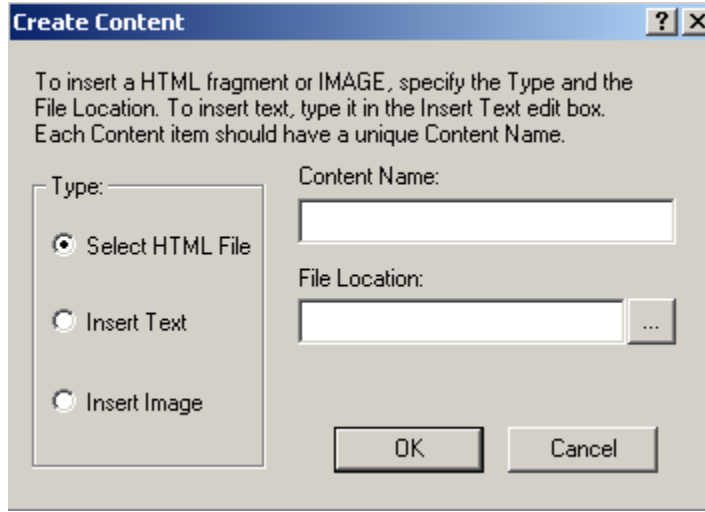
You must create content to use in table conditions separately; table content is not available for use in general (non-table) conditions and vice-versa. Composer ties table content directly to the particular table condition, and it can include font and cell formatting, unlike non-table content. (You can create some reusable table content for use with other table conditions only.) See “Creating Conditions on a Table (Line Item)” on Page 287 for information.

Creating New Content

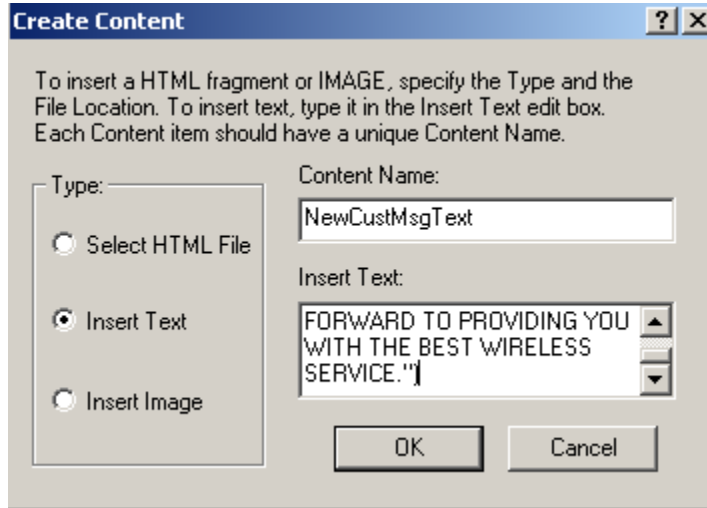
Follow these instructions to create content for use in conditions on non-table sections of a statement. (See “Creating Table Content” on Page 291 for information about creating table content.)

To create new (non-table) content:

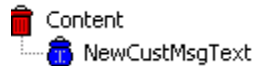
1. In the Application Tree, right-click the  Content node and select **Create** from the right-click menu. The Create Content dialog appears.



2. Click the content type: Select HTML File, Insert Text, or Insert Image. (Click Insert Text for NatlWireless.)
3. Enter a unique and meaningful name for the content. Names cannot contain spaces or special characters, must start with an alpha character, and cannot exceed 30 characters in length. (Enter NewCustMsgTxt.)
4. Browse the **File Location** field to select an HTML or image file, or type the text content in the **Insert Text** box. (Enter “WELCOME TO NATIONAL WIRELESS!
WE LOOK FORWARD TO PROVIDING YOU WITH THE BEST WIRELESS SERVICE.”)



5. Click **OK**. The content name appears in the Application Tree:





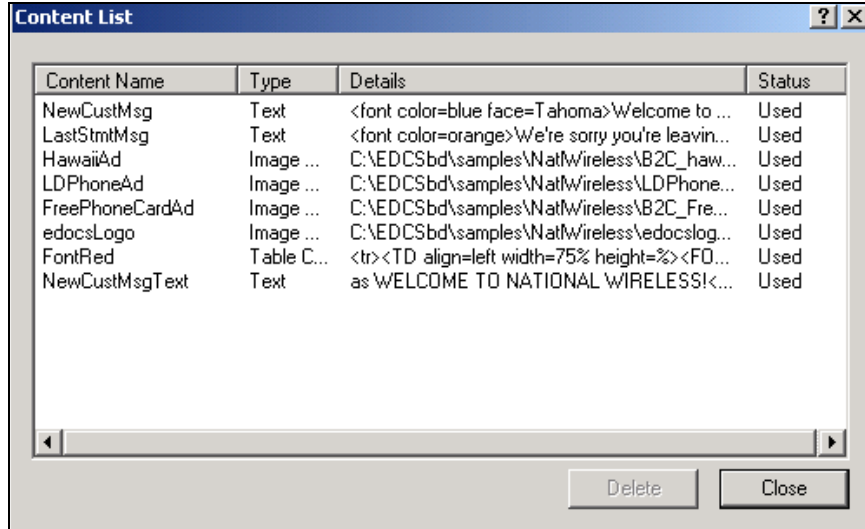
Viewing Content

You can display a list of all contents defined for an ALF showing the content type, details, and status (used or unused). You can optionally display a list of table content only.


You can also view the properties of individual contents.

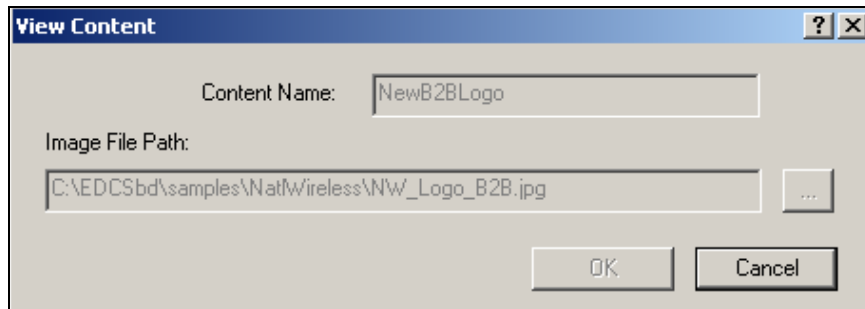
To view a list of all content and properties:

1. With the ALF open in Composer, right-click the  **Content** node (or click the  **Table Content** node for a list of table content only).
2. Select **Content List** from the right-click menu. Composer displays the Content List dialog.



To view properties of individual content:



1. With the ALF open in Composer, double-click the  Content node to display a list of content names in the Application Tree.
2. Right-click the content name and click **View Properties** from the right-click menu. Composer displays the View Content dialog (or the View Reusable Content dialog for reusable table content).



Changing Content

You can modify change the HTML template file, text, or image file associated with a content name by editing the content properties. (You can also modify table content and reusable table content.)



To edit content properties:

1. With the ALF open in Composer, double-click the  Content node to display a list of content names in the Application Tree. (Double-click the  Table Content node to display a list of table content names, if necessary.)
2. Right-click the content name you want to edit.
3. Select **Edit Properties** from the right-click menu. Composer displays the Modify Content dialog (or the Modify Reusable Content dialog for reusable table content).
4. Edit the content as necessary and click **OK**.

Deleting Unused Content

You can delete any content that is not in use in a condition, including reusable table contents.



To delete unused content:

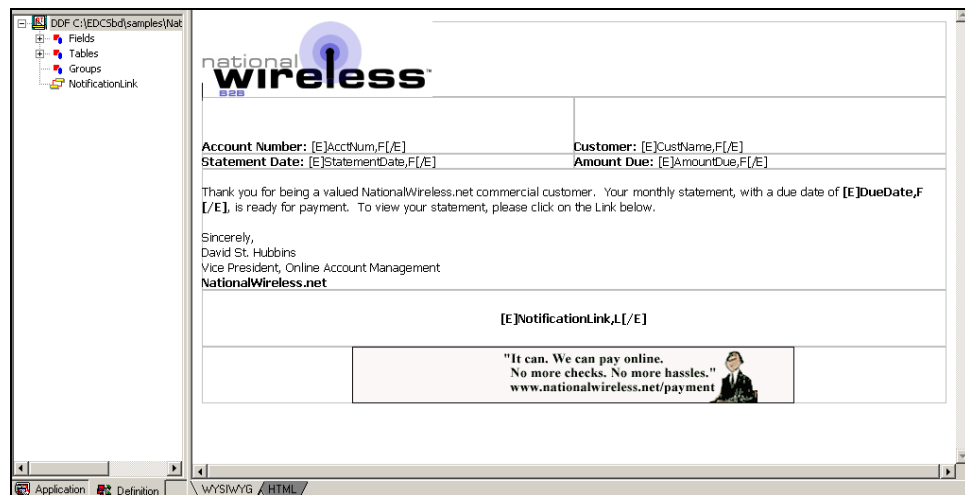
1. With the ALF open in Composer, right-click the  Content node in the Application Tree. (Click the  Table Content node to view a list of table content only.)
2. Select **Content List** from the right-click menu. The Content List dialog appears.
3. Select the unused content item you want to delete and click **Delete**.
4. Click **Close**.


Mapping a Notification Link

You can map a notification link into an HTML email template. eaDirect adds a direct link to the production web server, based on the configuration settings in the eaDirect Command Center. In production, this link is embedded in the delivered email and provides the recipient with a convenient and customized link to their login page.

To map an email notification link into the email template:

1. With the ALF and email template open in Composer, click the  Definition tab in the Application Tree area.
2. Click the Notification Link icon  and drag and drop the link where you want it to appear in the template. (The email notification link appears in the template as [E]NotificationLink, L [/E]. The “L” indicates this is a link.)




3. If the HTML template uses placeholders, delete them.
4. Click the **Save Template**  icon.

Saving the ALF, HTML Template Files

Whenever you make changes to the composition rules in an ALF, you must save the ALF. Whenever you save the ALF, Composer automatically saves the open template file as well. You can save the template independently, however.

See “Completed ALF and HTML Application Files” on Page 225 for details about the features saved in each file.

To save the ALF (and HTML template):


- Select **File>Save Alf** or click  on the ALF toolbar.



Tip

Since the ALF is an XML file, you can edit it in a text or XML editor. However, make sure syntax is correct, or you won't be able to open the an ALF in Composer and problems can arise during production.

To save the HTML template only:

- Click  on the Template toolbar. (Be careful not to click the ALF save icon, shown above; the Save Template icon has the letter “T” on it.)



Tip

Giving each template revision a “template ID” comment in the HTML template that indicates (matches) the dated directory in which the template resides can be helpful when troubleshooting.

Mapping Text Tables and Configuring Elements Using Composer

4

Mapping Tables and Groups

To present the groups and tables you defined in the DDF in your statement, you must map them to the location in the template where you want them to appear. Composer lets you click-and-drag tables and groups to the template.

You can present tables in one of two formats:


- As text in an HTML table
- As a chart that you design when mapping

A group consists of one or more tables that recur in a visual pattern. The simplest group can consist of one table of which there are multiple occurrences. Groups can be used also display information that repeats together. Each set of tables in a pattern is a group occurrence.

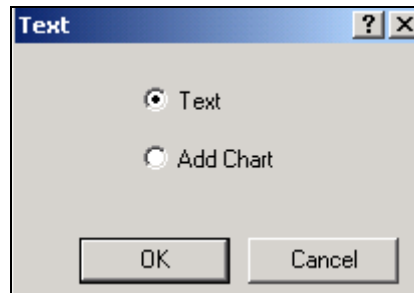
For details about mapping and formatting tables as charts, see Chapter 5.


Mapping a Table as Text

To map a table as text in a statement template:

1. With the ALF open in Composer, add an HTML table to the statement HTML template (if you haven't already); select **Table>Insert Menu**.
2. Click the  **Definition** tab in Tree area and expand the list of DDF tables by clicking on the + sign next to "Tables."

3. Drag and drop the table you want to map to the location in the HTML table where you want it to appear.



4. Select **Text** and click **OK**. “[E] *TableName*, R[/E]” appears in the template.
5. If your HTML template used placeholders, delete them.
6. Click  to save the template.



Tip

If you delete a field, table, or group from the DDF, you must manually delete the associated tag from the HTML template.



Mapping a Group

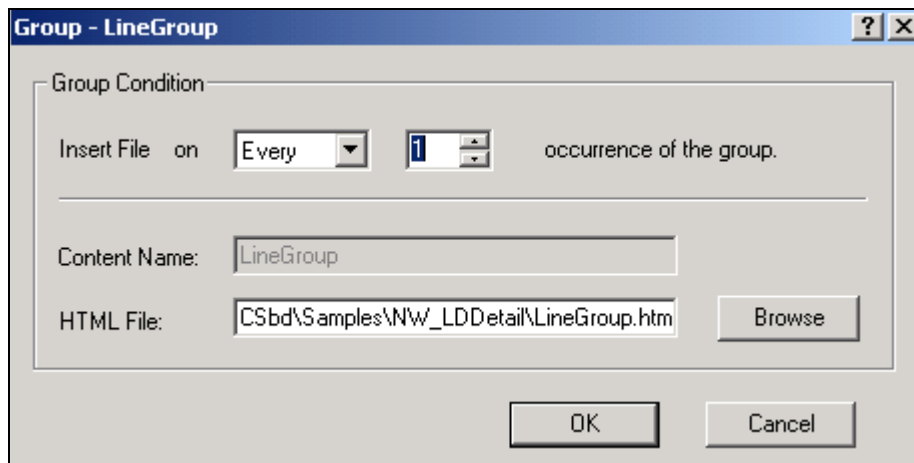
To map a group to a statement template, you drag-and-drop the group to the template, identify the group template to use for display, and specify the condition for using this template (every occurrence or some other basis).

Because you specify an external group template for presenting the group, you don't need to create an HTML table for the group; you don't need to encapsulate the group with any HTML tags. The HTML group template contains the [E] tags for the group's components, and the ALF contains the formatting rules for all tables and groups.

Correct	Incorrect
[E] Group,G [/E]	<table width="100%"> <tr> <td> [E] Group,G [/E] </tr> </td> </table>
<table width="100%"> [E] Table,R [/E] </table>	[E] Table,R [/E]

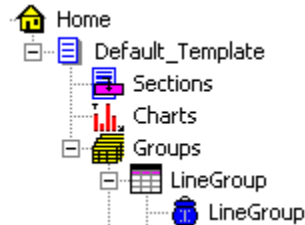
To map a group to a statement template:


1. Click the  Definition tab in the Tree area and double-click the  Groups node to expand the list of groups defined in the DDF.
2. Drag-and-drop the group you want to map to the HTML template. The Group dialog appears.



3. Select the “condition” for inserting the group template; in most cases you want to specify **Every** and **1** to insert the group template for each occurrence. (Specify a different frequency and number of times to insert the group template if required.)
4. Specify the group template or click **Browse** and select a template.

5. Click **OK**. The group HTML appears with a blue “content” icon in the Application Tree under the template where you mapped the group:





6. Click  to save the ALF.

Modifying the Group Template

You can change the group HTML template used to display a group in the statement or modify the “condition,” or frequency to apply the group template.

To modify the group template or condition:

1. With the ALF open in Composer, expand the  groups icons in the Application Tree to display the group you want to modify.
2. Right-click the  icon for the group and select **Modify** from the right-click menu. Composer displays the Group dialog.
3. Make the necessary modifications.
4. Click **OK**.

Deleting a Group

To delete a group from a statement you should delete both the group tag and the group HTML association.

To delete a group from a statement:

1. With the ALF open in Composer, select the group tag in the statement and delete it.
2. In the Application Tree, right-click the group name and select **Delete** from the right-click menu. Composer asks if you're sure you want to delete the group.
3. Click **yes**.

Formatting Tables

You can apply various formatting to enhance the appearance and readability of tables and to customize customer content.



You can apply formatting or add content to:

- **An entire table** – Use the Composer Format menu or Formatting toolbar to apply font and paragraph formatting such as styles, bolding, indentations, alignment, etc. eaDirect applies the formats to all columns and rows in the table. You apply table-wide formatting in the HTML template.
- **Individual columns** – In the temporary HTML displayed for configuring a table, use the Format menu or the Formatting toolbar to apply formatting to columns.
- **Customer-specific rows based on their data (using conditions)** – In the temporary HTML displayed for configuring a table, you create a section and specify the data-driven condition that (a column of) the customer's data must meet for eaDirect to apply the formatting or content. Conditional table formatting is useful, for example, to highlight a row containing a negative amount in red, add an advertising image, text, or link in a cell, or to remove a link to a detail drill-down view if a data amount equals zero.

- **A patterns of rows** – In the temporary HTML displayed for configuring a table, you create a section and define a pattern of lines and the table formatting (or other content, if useful) you want to apply to the pattern. For example, after every third row you could highlight two rows in blue for readability.


Formatting an Entire Table


To apply formatting to an entire table:

1. With the ALF open in Composer, select the table and use the Format menu or the Formatting toolbar to choose formatting. (If a table is part of a group, first click the group name shown under the  Groups node in the Application Tree to list the tables in the group.)
2. Click  to save the formatting to the template (or wait and save the entire ALF at the end of your editing session, if you prefer).

Formatting Individual Columns

To format table columns:

1. With the ALF open in Composer, select the table you want to edit. Be sure to select the start and ending E tags with the table name. (If a table is part of a group, first click the group name shown under the  Groups node in the Application Tree to list the tables in the group.)
2. Right click on the selection, and select **Configure Table** from the right-click menu. A new window opens with a temporary HTML page.
3. Select individual columns to format in either WYSIWYG or HTML mode. In WYSIWYG mode, use the Format menu or the Formatting toolbar to add formatting. In HTML mode, you can enter HTML formatting code directly. (You must add all structural formatting, such as modifications to td width, height, or vAlign in HTML mode.)

4. Click  to save the ALF.

Creating Conditions on a Table (Line Item)

Conditions on a table let you tailor advertising or marketing messages based on the data in the individual lines of a customer's statement, or format the line based on a key data value.

You can create data-driven conditions on a table or selected columns instructing eaDirect when to apply table content to that selection. eaDirect checks the condition for each line of data and, if true, applies the content to that line only. Table content can include any combination of font and cell formatting (color, bold, italics, alignment, etc.), or an image, text, or hyperlink that eaDirect adds to the section when composing the statement. You can optionally specify different content for each column in a condition. The data driving a table condition must be within the same table.

For example, you could create a condition on a table to:

- Highlight a row containing a negative amount in red (content can include or be limited to font or cell formatting).
- Add a targeted advertising image to a line item in a credit card statement based on the nature of the purchasing data, such as the name of the clothing retailer, type of clothing purchased, or the purchase amount. (You can optionally *replace* the section with an image using reusable table content in the condition.)
- Add a hyperlink, such as one offering a discount purchase coupon for the linked retail Web site.
- Remove a link to a detail drill-down view if a data amount equals zero
- Add a text message with targeted offers or information to the customer based on the transaction. (You can optionally *replace* the section with text using reusable table content in the condition.)

- Replace the section with a null. (You must use reusable table content in the condition to replace a section with a null.)
- Suppress section display. (You must use reusable table content in the condition to suppress display.)


When creating a condition on a table, you must define the content specifically for that condition, called “table content.” You cannot use content created for non-table sections in a table condition and vice-versa. You can, however, create reusable table content and apply it to multiple conditions. You must create and apply reusable table content in the condition if you want to insert a null or suppress section display; regular table content cannot do this. Also, if you want to replace a section with an image or text instead of adding it before or after existing section content, you must create and apply it as reusable table content in the condition. See “Creating Table Content” on Page 291 for details.

Each condition on a table requires one or more logic elements. A logic element is an expression that, in a table condition, can compare:

- The values in two columns
- The value in a table column to a specific value such as a date, time, amount, NULL or EMPTY, or a regular expression
- A table or group with NULL or EMPTY (to test for existence)

You can create logic elements while defining a condition or create logic elements first, making them available to select from a drop-down list when creating the condition. See “Creating Conditions” on Page 244 for details on condition syntax and creating logic elements.

To create a condition on a table:

1. With the ALF open in Composer, select the table on which you want to create a condition. (If a table is part of a group, first click the group name shown under the  Groups node in the Application Tree to expose the tables in the group.)
2. Right click and select **Configure Table** from the right-click menu.

3. Select the section you want the condition to apply to (one or more, or all columns), then right-click and select **Add Section** and **Conditional Table Format** from the right-click menu. The Create Section dialog appears.

Create Section [?] [X]

Section Name: eaMarket enabled

Build Condition

Condition Operators Check Add to List

Condition Name: Save

Select Logic Element

Name: Create...

Details: Add

Select Content

Name: Create...

Reusable...

Preview

Add

Condition List

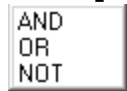
Remove

OK Cancel



4. Enter a name for the section. Names cannot contain spaces or special characters, and must start with an alpha character. Maximum length is 30 characters. Use meaningful names; “BigSpenderInsertAirlineOffer” is more descriptive and easier to identify than “ALFsection1”.
5. Use the options on this dialog to build a conditional statement in the **Build Condition** field. You can manually enter the text in the Build Condition field, but choosing from the menus reduces typing and syntax errors. (You can optionally choose a previously saved condition from the drop-down list if you’ve already created a condition you want to reuse in this section. If so, skip to Step 13.)



6. Click **Condition** and select **IF** from the drop-down list.
7. Under Select Logic Element, click **Create** to create a logic element for the condition; see Creating a Logic Element on Page 245 for details. If you have already created the logic element, choose it from the Name drop-down list and click **Add** to add it to the condition.
8. Click **Operators** to select any operators your conditional statement requires



9. Add any additional logic elements the conditional statement requires.
10. Click **Condition** again and select the right parenthesis from the drop-down list. One set of parenthesis must enclose all logic elements in a condition.
11. Click **Condition** and select **THEN**.
12. Do one of the following:
 - To create content for this table, click **Create**. (See the instructions for creating table content in “Creating Table Content” on Page 291.)
 - To use an existing reusable table content, select it from the Name drop-down list under Select Content.

- To create new, reusable table content, click **Reusable**. (See the instructions for creating reusable table content in “Creating Table Content” on Page 291.)
13. Click **Add** to insert the content name in the condition.
 14. When the condition is complete, click **Check** to verify it is valid. Correct or complete the statement as necessary, checking the condition again.
 15. If you want this condition to be available in eaMarket, click the **eaMarket enabled** checkbox. (If you eaMarket enabled a section, it appears with the eaMarket enabled section  icon.)
 16. When the condition is complete and valid, click **Add to List**. Composer displays the condition in the Condition List at the bottom of the dialog.
 17. If you want to save the condition for possible reuse in another section, click the condition in the Condition List and type a name in the Condition Name drop-down box (next to the Add to List button). Click **Save**.
 18. If necessary, build any additional conditions to create a compound condition on the section. Begin subsequent conditions with ELSE IF.
 19. Click **OK**. The condition appears in the  **Sections** node for the template. Composer also adds a number to the table tag in the template, as in [E]LineCallList_0,R[/E], indicating a condition is defined on the table.

Creating Table Content

When creating a condition on a table, you must define new table content for use with the condition. You cannot use content created for non-table sections of a template in a table condition. In addition, the table content you create does not automatically become available for use in other table conditions. Composer ties table content directly to a particular table condition. You can create reusable table content, however, to define certain types of content, and these are available for reuse in other table conditions.

Table content can specify:

- Font formatting - Bold, italics, underlining, and text color.
- Cell formatting – Alignment (left, right, center) and cell background color.
- Text and an optional hyperlink on the text.
- An image and an optional hyperlink on the image.
- Hyperlinks

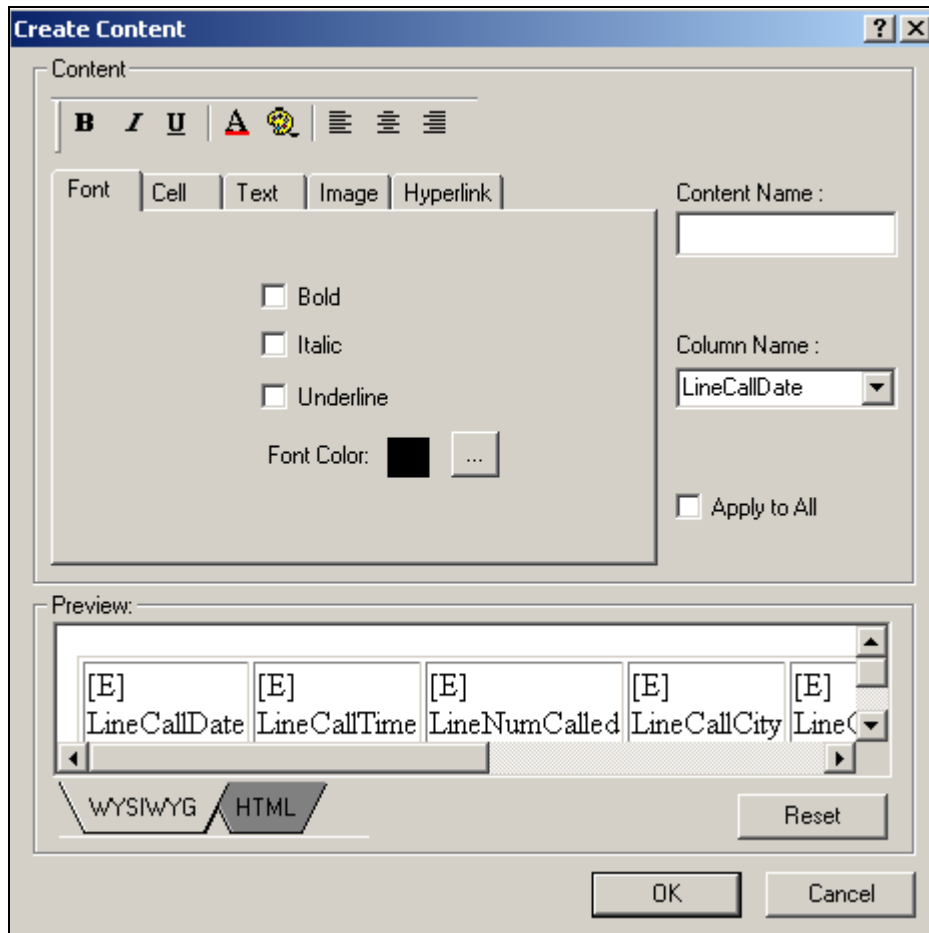
In addition, you can specify different content and format for individual columns in the condition (and these specifications are kept together as table content for the condition).

You must use reusable table content to:


- Suppress display
- Insert a null
- Replace a section with text (use regular table content to *add* text)
- Replace a section with an image (use regular table content to *add* an image)

To create table content:

1. You create table content by clicking the Content button on the Create Section dialog when creating a condition on a line item or on a pattern of rows. (eaDirect displays the Create Content dialog.) You cannot create table content independently of a table condition, except to create reusable content (see below). On the Create Content dialog, enter a name for the content in the Content Name field.



2. If you want the table content you specify to apply to all columns selected for the condition, click the Apply to All option. If you want to apply different content to individual columns, select the column names one at a time and proceed to specify the content.
3. Specify content use the tab options and/or the formatting toolbar(you can optionally click the HTML tab to edit the HTML directly):
 - **Font formatting** – For bold, italics, underlining, and font color, use the Font tab and/or the **B I U A** toolbar buttons.


- **Cell formatting** – To align left, center, or right, and apply background color to cells, use the Cell tab and/or the  toolbar buttons.
 - **Text** – Use the Text tab to define a block of text to add and optionally specify a hyperlink to use on the text. You must specify whether you want eaDirect to insert the text before or after the existing section content; select either the **Prefix** or **Suffix** option on the Text tab.
 - **Image** – Use the Image tab to specify the path of an image file to insert and optionally specify a hyperlink to use on the image. You must specify whether you want eaDirect to insert the image before or after the existing section content; select either the **Prefix** or **Suffix** option on the Image tab.
 - **Hyperlink** – Use the Hyperlink tab to specify an address to add as a link.
4. If you specify content in the Text, Image, or Hyperlink tabs, click **Apply** on that tab.
 5. Use the Reset button if you want to clear any formatting you've selected.
 6. Click **OK**. The name of the table content appears with a blue icon under the Table Content node in the Application Tree.

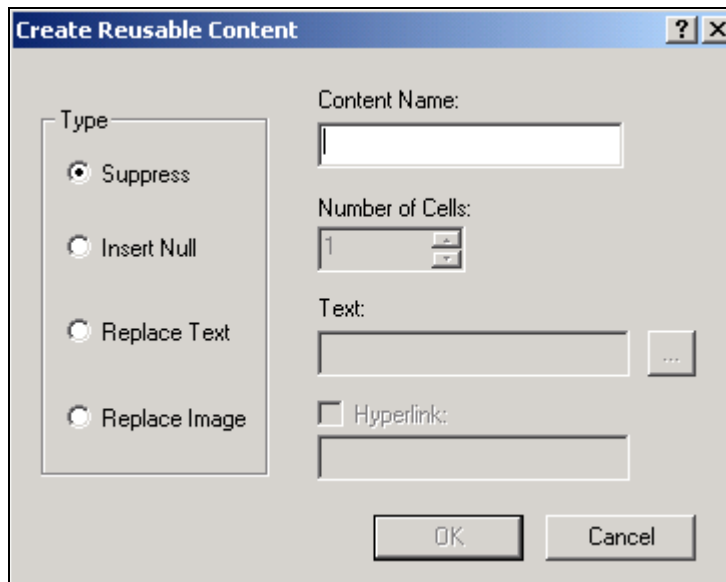


To create reusable table content:

You can create reusable table content while creating a condition on a table or at any time.

1. If you clicked the Reusable button on the Create Section dialog when creating a condition on a table (including on a pattern of rows), eaDirect displayed the Create Reusable Content dialog.

If not, with the ALF open in Composer, right-click the  Table Content node in the Application Tree and select **Create Reusable**. Composer displays the Create Reusable Content dialog.



2. Specify a name for the reusable content.
3. Click the type of content you want to define: Suppress (to suppress display), Insert Null, Replace Text, or Replace Image.
4. For a null, text, or image, specify the number of cells.
5. To add a hyperlink to text or an image, click the **Hyperlink** option and specify a link.
6. Click **OK**. The name of the reusable content appears with a yellow icon under the Table Content node in the Application Tree:



To view a list of table and reusable content:

- See “Viewing ContentDeleting Unused Content” on Page 276.

To change table content:




- See “Changing Content” on Page 278.

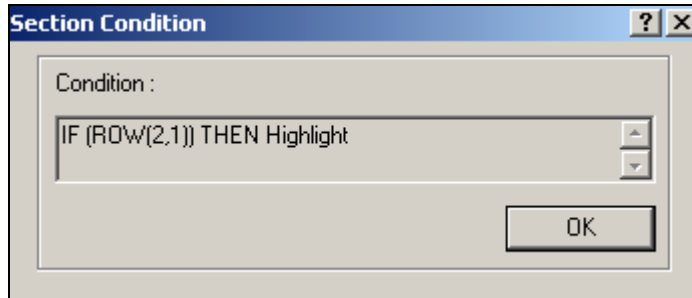
To delete (unused) reusable table content:

- See “Deleting Unused Content” on Page 278.

Deleting, Changing, or Viewing Table Conditions

To view the properties of a table condition:



1. With the ALF open in Composer, double-click the  Sections node (below the  Home node). Composer displays the list of conditions (sections).
2. Double-click the name of the condition (section) you want to view.
3. Right-click on the  Condition node and select **View Properties** from the right-click menu. Composer displays the Section Condition dialog with the conditional statement.





4. Click **OK**.

To modify a condition on a table:

You can modify a table condition or add one or more ELSE IF conditions to a section (creating compound conditions). You can also modify the condition on a pattern of rows.

1. With the ALF open in Composer, double-click the  **Sections** node (below the  **Home** node) for the table.
2. Right-click the name of the condition you want to modify.
3. Select **Modify Section** from the right-click menu. The Modify Section dialog appears.
4. For a regular table condition, double-click on the condition and edit as needed. To add a new condition to the table section, follow the procedures above to add an ELSE IF condition. For a condition on a pattern of rows, edit the condition as needed.
5. Click **OK**.

To delete a table condition:

1. With the ALF open in Composer, double-click the  **Sections** node (below the  **Home** node) for the table.
2. Right-click the name of the condition you want to delete.

3. Select **Delete Section** from the right-click menu.
4. Composer asks if you're sure you want to delete the section. Click **Yes**.
Composer updates the table tag on the template (removes “_0,” for example).

Applying Formatting or Adding Content to a Pattern of Rows

You can apply table content to a defined pattern of rows, letting you apply color highlight to alternating rows in a statement, for example. Color or other formatting can often make the lines of a statement easier to read.

You can apply the same types of table formatting and content to a row pattern as you can to regular table conditions on a line item. The “condition” in this case is simply the pattern of rows you define, and is not based on extracted data. For details on the types of formatting and content you can add to a pattern of rows, see “Creating Table Content” on Page 291.

You specify a pattern by the starting row number and the number of rows to repeat. For example, if you specify the row number to start at as 3 to repeat for 2 rows, and define the content as font color highlighting in yellow, then eaDirect composes every other two rows in yellow:


Row 1: Default color
Row 2: Default color
Row 3: Yellow
Row 4: Yellow
Row 5: Default color
Row 6: Default color

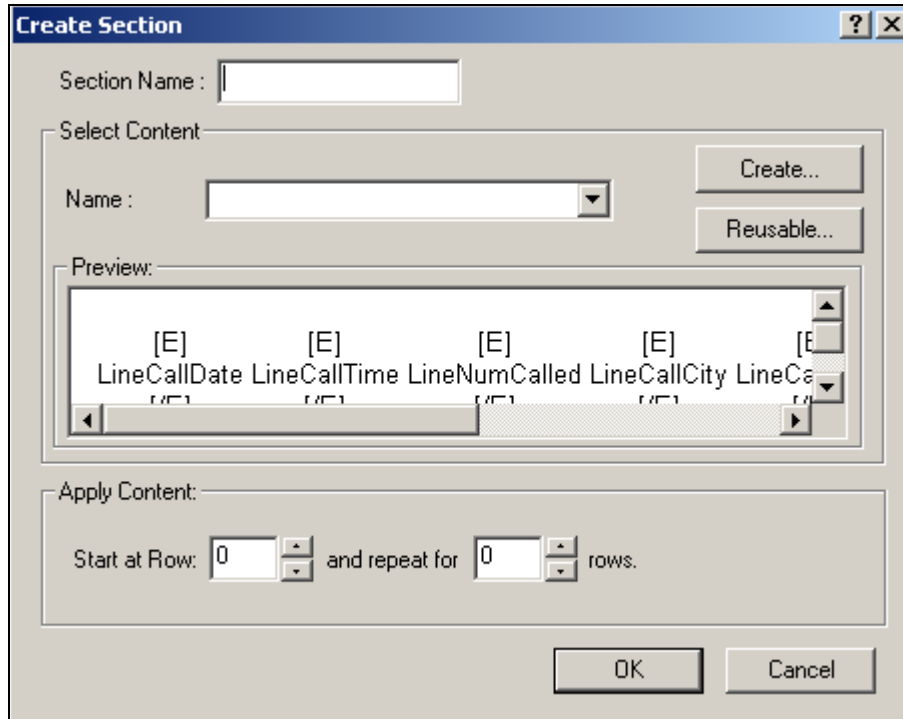
To apply a somewhat more complex highlighting, you can define multiple patterns for a table. For example, you would need to create two separate sections to achieve the following color highlight pattern:

Row 1: Yellow
Row 2: Green
Row 3: Blue
Row 4: Yellow
Row 5: Green
Row 6: Blue


In this example, you apply yellow as the default table color in the default template. Then you would create one section on the table with a pattern that starts on row 2 and repeats for 1 row in green, and another section on the table with a pattern that starts on row 3 and repeats for 1 row in blue.

To format a pattern of rows in a table:

1. With the ALF open in Composer, select the table you want to edit. (If a table is part of a group, first click the group name shown under the  Groups node in the Application Tree to expose the tables in the group.)
2. Right click on the selection, and select **Configure Table** from the right-click menu. A new window opens with a temporary HTML page.
3. Select the table you want to format, and right click to select **Format Table** from the right-click menu. The Create Section dialog appears.



4. Enter a name for the section, such as HighlightEveryOtherRow.
5. Do one of the following:
 - To create content for this table, click **create**. (See the instructions for creating table content in “Creating Table Content” on Page 291.)
 - To use an existing reusable table content, select it from the Name drop-down list under Select Content.
 - To create new, reusable table content, click **Reusable**. (See the instructions for creating reusable table content in “Creating Table Content” on Page 291.)
6. Specify the row number where you want the pattern to begin.
7. Specify the number of rows you want to apply the table content to.

8. Click **OK**. The condition appears in the  **Sections** node for the template. Composer also adds a number to the table tag in the template, as in [E]LineCallList_0,R[/E], indicating a condition is defined on the table.

To delete, modify, or view a condition on a pattern of rows:

- See “Deleting, Changing, or Viewing Table Conditions” on Page 296.

Sorting Tables and Groups

You can enable Web users to sort the data in their statements by creating sort elements and adding a link to the element in an appropriate place in the statement. You can sort on a table or a group.

You can create multiple sort elements within a statement ALF; you can also use the same link for multiple sort (or filter) elements.

The Simulator lets you validate the format of sort elements; see “Simulating Sorting and Filtering Elements” on Page 345.

Table Sorting

You can enable Web users to sort rows in a table in ascending or descending order by the values in a particular column. To do this, you create a sort element in table mode identifying the table and column to sort on, specify whether to sort in ascending or descending order, then create a sort element link and add it to the template.

When a Web user clicks a sort element link, the statement page redisplay with the table rows reordered according to the specifications of the sort element. When the user clicks the sort element again, sorting order automatically reverses.

Example - View of unsorted table (rows of a credit card statement ordered by Posting Date):

Posting Date	Transaction Date	Reference Number	Transactions	Charges
5/20	5/19	10384	Illuminations Lighting	166.79
5/23	5/22	10298	Springfield Getty	18.50
5/25	5/24	10273	American Airlines	1680.00
5/30	5/29	00578	Ocean Grill	65.00
6/06	6/04	20182	Harbor Line Cruises	126.06

Above table sorted by “Charge” column:

Posting Date	Transaction Date	Reference Number	Transactions	Charges
5/23	5/22	10298	Springfield Getty	18.50
5/30	5/29	00578	Ocean Grill	65.00
6/06	6/04	20182	Harbor Line Cruises	126.06
5/20	5/19	10384	Illuminations Lighting	166.79
5/25	5/24	10273	American Airlines	1680.00

If a table is part of a group, eaDirect sorts within each occurrence of the table. To enable sorting across occurrences of a table or group, create a combine element on the table or group in addition to the sort element. Creating both a sort and combine element on a table or group sorts and displays the rows from all table or group occurrences in sequence according to the sort column value. To create a combine element, see “Using Combine Elements to Page, Sort, & Filter Across Table or Group Occurrences” on Page 333.

Group Sorting

You can enable Web users to sort occurrences of group data in ascending or descending order. To sort a group, you create a sort element in group mode identifying the table and column (or the group field), specify whether to sort in ascending or descending order, then create a sort element link and add it to the template. When a Web user clicks the link, the statement page redisplay with the group occurrences reordered according to the specifications of the sort element. When the user clicks the link again, sorting order automatically reverses.

eaDirect sorts group occurrences according to the sort column value in the **first row of the first occurrence of the table** in the group. The first occurrence of the table on which you define the sort element acts as the “lead table,” and the first row of this table occurrence as a “lead line” for sorting the group.

This can be useful, for example, when the tables in a group contain only one row, such as the lines of an address, or wherever it makes sense to sort on just one column and the first row of a table.

Example - View of unsorted group (tables of calling card line item data):


				Duration	Charge
G1 ₍₀₎	T1 ₍₀₎	10/21/02	12:35 pm	0:24:18	10.00
	T2 ₍₀₎	Rye, NY	Atlanta, GA		
G1 ₍₁₎	T1 ₍₀₎	10/22/02	4:08 pm	0:05:46	1.60
	T2 ₍₀₎	Atlanta, GA:	Chicago, IL		
G1 ₍₂₎	T1 ₍₀₎	10/24/02	6:00 am	1:59:04	34.90
	T2 ₍₀₎	Chicago, IL	Rye, NY		

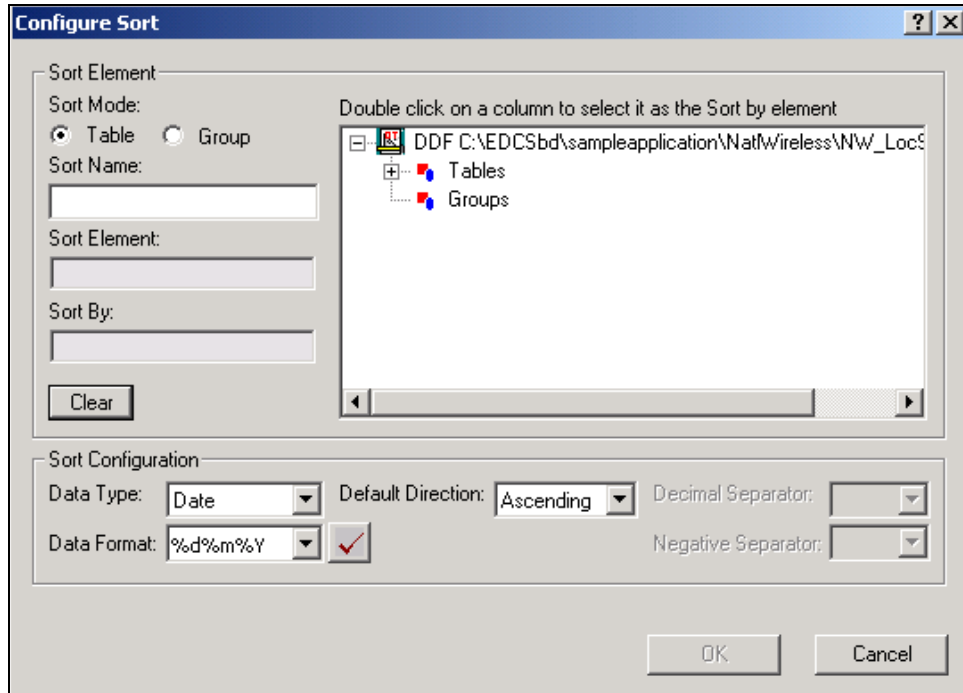
Above group sorted by call charge column:

				Duration	Charge
G1 ₍₁₎	T1 ₍₀₎	10/22/02	4:08 pm	0:05:46	1.60
	T2 ₍₀₎	Atlanta, GA:	Chicago, IL		
G1 ₍₀₎	T1 ₍₀₎	10/21/02	12:35 pm	0:24:18	10.00
	T2 ₍₀₎	Rye, NY	Atlanta, GA		
G1 ₍₂₎	T1 ₍₀₎	10/24/02	6:00 am	1:59:04	34.90
	T2 ₍₀₎	Chicago, IL	Rye, NY		




Creating a Sort Element

To create a sort element:

1. With the ALF open in Composer, right-click  **Sort Elements** in the Application Tree. (Open NW_LocSummary.alf.)
2. Select **Create Sort Element** from the right-click menu. The Configure Sort dialog appears.



3. Under Sort Mode, click **Table** to create a sort on a table, or **Group** to create a sort on the occurrences of a group.
4. Enter a descriptive name for the sort. The name cannot contain spaces or special characters, must start with an alpha character, and cannot exceed 30 characters in length. (Enter `SortLocChargeDesc`.)
5. On the tree of table and groups, expand the list and select the table or group you want to sort. (Expand the `LocalChargeSummary` table.)
6. Double-click the table and column you want to sort by. If sorting a group, you can select a group field. If you make a mistake, click **Clear** and reselect the table and column. (Select `LocalChargeDesc`.)
7. Select the data type of the column you're sorting. (Select `String`.)

8. If the data type is date or time, select the appropriate sort format from the `Data Format` drop-down list or type one of your own. Manually enter any separators within the data format, if necessary. (You can copy, cut, and paste selected values from the list of formats to build the one you want. Right click the format to display the Edit menu.)
9. In the `Default Direction` field, choose the direction to sort: ascending or descending.
10. If the data type is currency or decimal, specify a separator (decimal point or comma) in the `Decimal Separator` field.
11. If the data type is currency, you can optionally select a symbol to represent a negative value: -, CR, cr, DB, db, or ().
12. If the data type is a date or time, click the `Validate` button  to verify that the format you've entered complies with the data type.
13. Click  to create the sort element.
14. Click  to save the ALF.
15. Follow the instructions in the next section to create a link to the sort element.

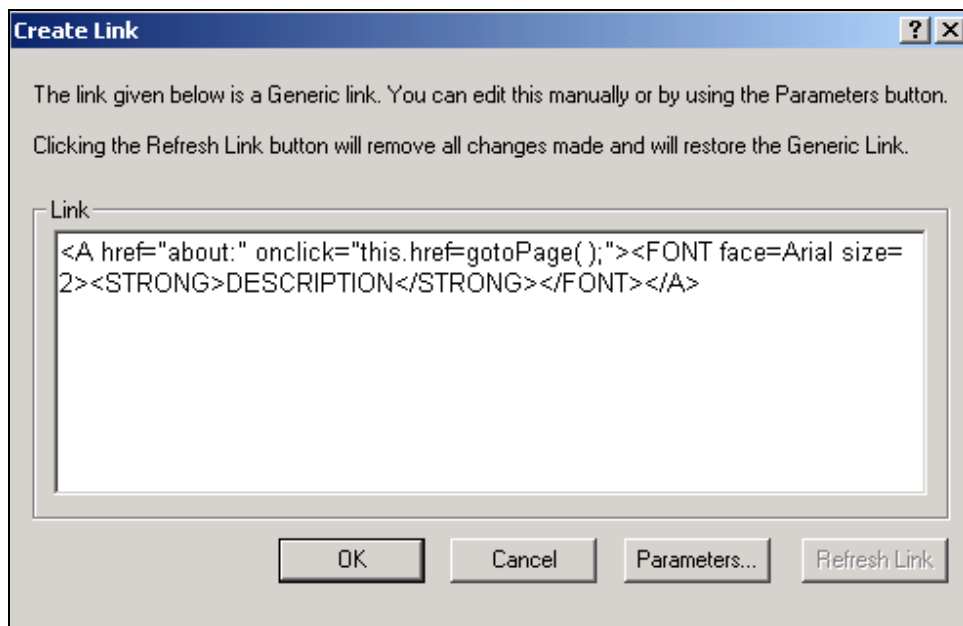
Creating a Link to a Sort Element

After creating a sort element, you must create a link users can click to perform the sort. You can create a link on static text, an image, or an `[E]` tag.

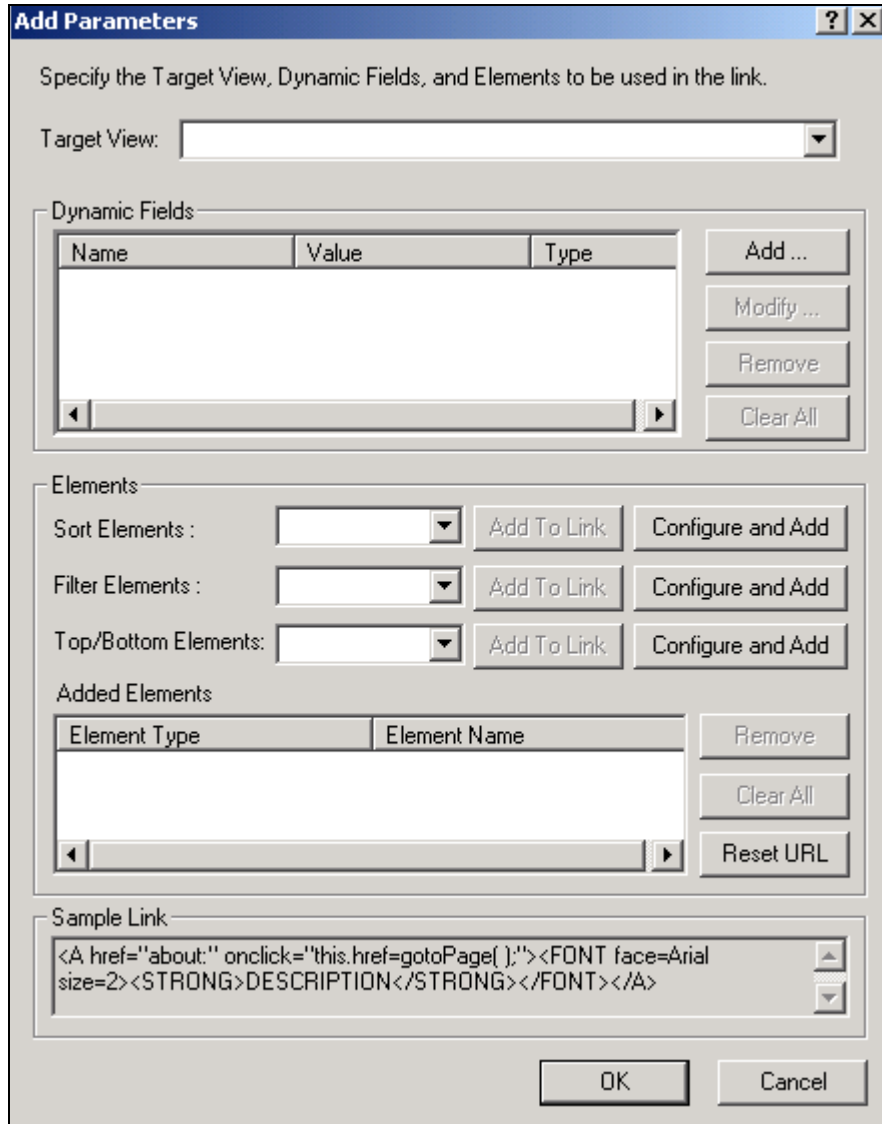
You can optionally choose to add a `Reset URL` function to the link. By default, the sort would be cumulative on any previous URLs the user chose in the session. Adding this function enables you to clear any previous URL actions the Web user took, such as filtering data, before applying the sort (except if you sort the same table or group using a different column from a previous sort, eaDirect removes the previous sort and applies the new one).

To create a link to a sort element:



1. With the ALF open, select the item you want to use as a link (be sure to completely highlight the item). (Highlight the word “Description” above the LocalChargeSummary table.)
2. Right-click and select **Create Link** from the right-click menu. The Create Link dialog appears.

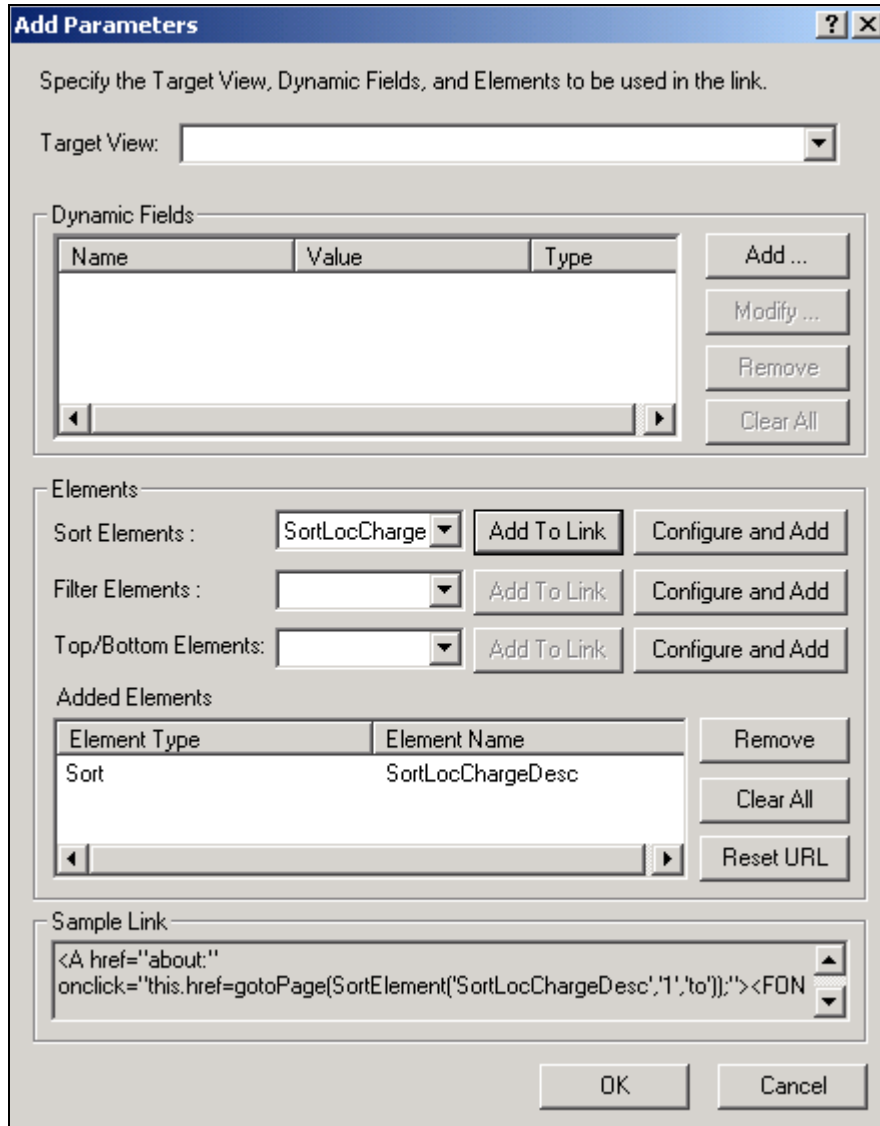


3. Click **Parameters**. The Add Parameters dialog appears.

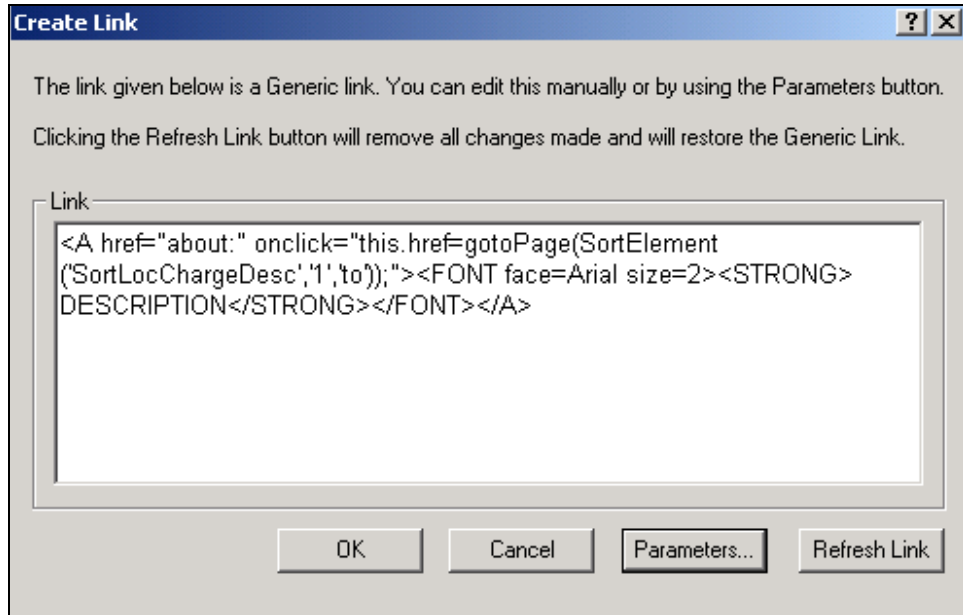





4. If the link is to a sort that appears in a different view, enter the name of the target view. (For NatlWireless, leave this blank.)
5. Click the **Sort Elements** drop-down list and select the sort element you want to link. (Select SortLocChargeDesc.)

6. Click . The sort element appears in the Added Elements area. (You can use the Remove and Clear All buttons on items in this list.)
7. If you want to clear the actions of any previous URLs before displaying this sort, click  to add the Reset URL function to the link.




8. Click . Composer displays the view name and sort element in the Create Link dialog.



9. Click  to create the link. Composer adds a blue hyperlink to the statement.
10. Click  to save the template.
11. If you added a link to a table column, click  to save the ALF.



You can create a sort or filter element from the Add Parameters dialog; click . See “Sorting Tables” on Page 301 or “Filtering Data” on Page 317 for details.

Modifying a Sort Element

To modify a sort element:

1. With the ALF open in Composer, right-click the sort element and select **Modify Sort Element** from the right-click menu. Composer displays the Configure Sort dialog.
2. Edit the sort element.
3. Click **OK**.

Deleting a Sort Element

To delete a sort element:

1. With the ALF open in Composer, right-click the sort element and select **Delete Sort Element** from the right-click menu. Composer asks if you're sure you want to remove this sort element.
2. Click **Yes**. Composer removes the sort element from the ALF.
3. You must manually remove the link to the sort element in the HTML template.

Paging Statements

Creating a page element lets you present large documents in multiple browser pages each with a fixed (configurable) number of rows. You can configure paging on a group or a table. You can also define page sets to display a fixed (configurable) number of pages in each set.

Using a paging element and page set can speed up the display of statement data to the user. Paging uses progressive data extraction so the user does not have to wait for all data to extract before starting to view it.

Once you create a page element and any page sets you must add one or more links to the statement for navigating between pages and page sets.

By default, eaDirect starts a new page for each occurrence of a table or group, regardless of how many rows there are. To eliminate the new page start and display table or group information contiguously, define a combine element for the table or group. Web users can then view the same amount of information in fewer (full) pages. See “Using Combine Elements to Page, Sort, & Filter Across Table or Group Occurrences” on Page 333.

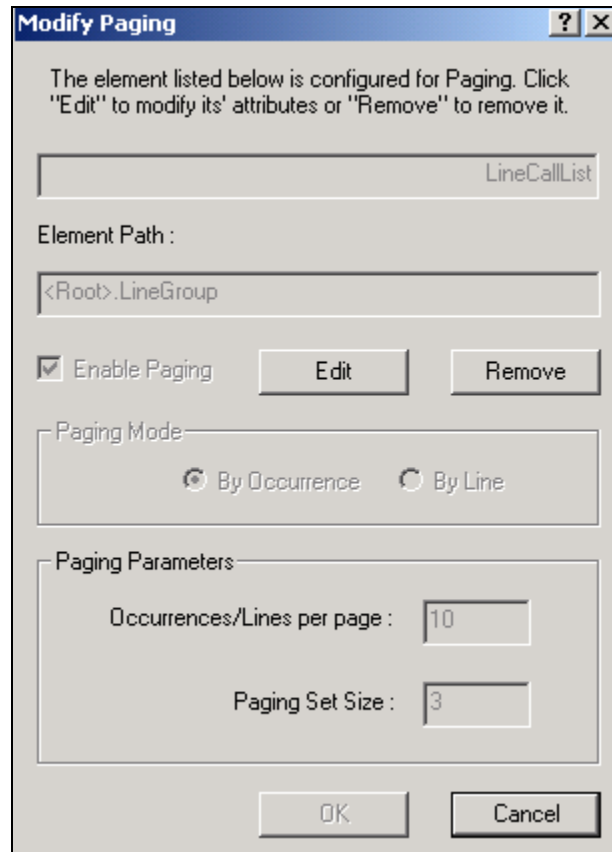
Note that any lines containing group fields appear in addition to the number of lines, or rows, you specify for paging.

Also note that you cannot simulate paging in Composer; a simulation returns the first page of a document only. Also, paging links call a JavaScript function that is only available in a fully installed live environment.

Creating a Page Element

To create a page element:

1. With the ALF open in Composer, select the table or group to use for the paging element. Be sure to select the entire tag.
2. Right click on the tag and select **Paging**, then **Configure Paging Element** from the right-click menus. The Configure Paging dialog appears.



3. Select a paging mode, either **By Occurrence** (for a group) or **By Line** (for a table or a group). Paging groups by line counts all the contents of the group by the number of lines (or rows) returned. Paging groups by occurrence displays the entire group, regardless of the number of lines returned.
4. Enter the number of occurrences (for a group) or lines that will be shown on each HTML browser page (for tables and groups). Note that any lines containing group fields are not included in this count, but appear in addition to the number of rows you specify here. You may want to take this into account when planning your page size.
5. Enter the number of pages to extract at a time (page set) in **Paging Set Size**.

6. Click **OK**.

Creating a Page Set

To create a page set:

You must create a paging element before you can create a page set.

1. Enter text in the template to serve as a placeholder for the paging set tag.
2. Select the text.
3. Right-click and select **Paging** and **Insert Paging Set** from the right-click menu. Composer inserts the paging set tag.
4. Delete the placeholder text and “%P” from the template. The remaining tag should look something like: `[E] PageSet [/E]`.

Inserting Navigational Paging Links


You can create navigation links in the template to move to a previous or next extracted page, to extract the previous or next page set, and to go to the first page.

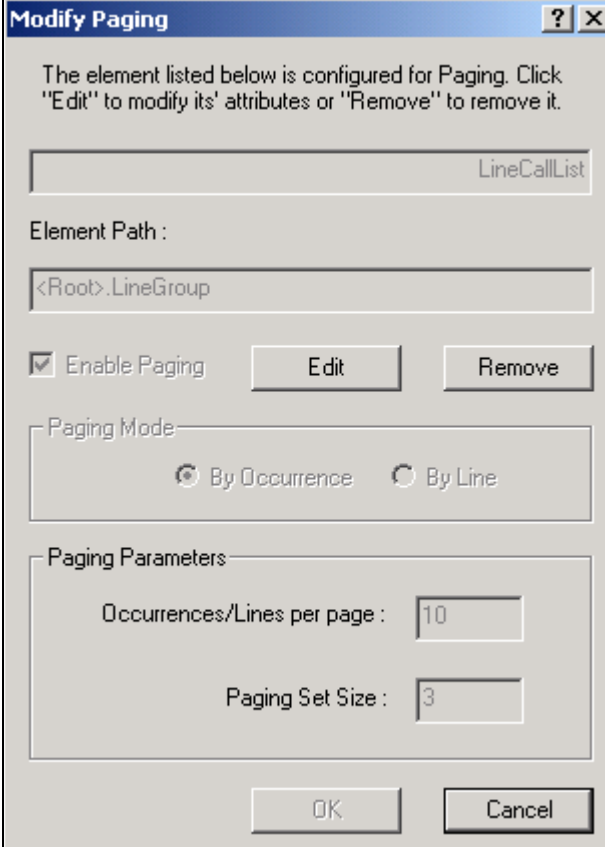
To insert page navigational links:

1. Insert text (such as “Next”) or a graphic to use as a linking object.
2. Select the object.
3. Right-click and select **Paging** and **Paging Links** from the right-click menu. The Select Paging Link Type dialog appears.
4. Select the appropriate radio button for the paging link type.
5. Create any additional links.
6. Save your work.

Modifying a Page Element

To modify a page element:

1. With the ALF open in Composer, double-click the  Page Element node in the Application Tree. Right-click on the page element and select **Modify Page Element** from the right-click menu. eaDirect displays the Modify Paging dialog.



Modify Paging ? X

The element listed below is configured for Paging. Click "Edit" to modify its' attributes or "Remove" to remove it.

LineCallList

Element Path :
<Root>.LineGroup

Enable Paging Edit Remove

Paging Mode
 By Occurrence By Line

Paging Parameters
Occurrences/Lines per page : 10
Paging Set Size : 3

OK Cancel

2. Click **edit** and modify the page element.
3. Click **OK**.

Removing a Page Element

To remove a page element:

1. Either:
 - Right-click on the page element in the Application Tree and select **Modify Page Element**. Click **Remove** on the Modify Paging dialog.
 - Right-click anywhere in the template and select **Paging** and **Remove Paging Element** from the right-click menu.
2. Click **OK** when prompted to remove the existing page element.

Filtering Data

You can let Web users view a narrow extraction of data based on filtering criteria you define in a filter element. You must create a filter on a particular table column then add a link to the statement template, which users click to compose the filter. You can filter table or group data.

You can create two types of filter elements:

- **General content-based** – To apply a filter to extracted data, producing a narrowed view of the data set. You could create a filter to display only those line items past 60 days due, over \$1,000, etc.
- **Top/bottom** – To create a counted subset of filtered data based on a configurable number of items, from either the top or bottom of the filtered data set. For example, you could create a top/bottom filter element to display the top 10 most expensive phone calls in a telecomm statement.

For each filter element (except top/bottom) you must define an expression that compares the value in a particular column to either a hard value (date, integer, time, currency amount, decimal value, or string), another table column, or a field. For example, a simple filter expression for string data consists of a column name followed by an operator (**==**, **<**, **>**, **<=**, **>=**, etc.) and the comparator, for example:

```
AreaCode == "MA"
```

You can create a compound filter expression using AND, OR, and NOT operators. For details and examples of data filtering expressions, see Appendix A.

The Simulator lets you validate the format of sort elements; see “Simulating Sorting and Filtering Elements” on Page 345.

Table Filtering

You can enable Web users to view a filtered subset of table rows using either a general content-based filter or a top/bottom counted subset filter. To filter a table, you must create a filter element (in table mode) that defines the table and column to filter along with filter extraction criteria, then create a filter element link and add it to the statement template.

When a Web user clicks the link, the statement page redisplay with only the table rows that satisfy the specifications of the filter element expression.

You can create a filter on an individual table or on each occurrence of a table in a group (or across group occurrences). To enable filtering across occurrences of a table, create a combine element on the table in addition to the filter element. To create a combine element, see “Using Combine Elements to Page, Sort, & Filter Across Table or Group Occurrences” on Page 333.

Group Filtering


You can enable Web users to view only the **occurrences** of a group that satisfy a general content-based filter or a top/bottom counted subset filter. To filter a group, you create a filter element (in group mode) identifying the group to filter, along with the filter extraction criteria, then create a filter element link and add it to the template. When a Web user clicks the link, the statement page redisplay with only the group occurrences that meet the specifications of the filter element expression.

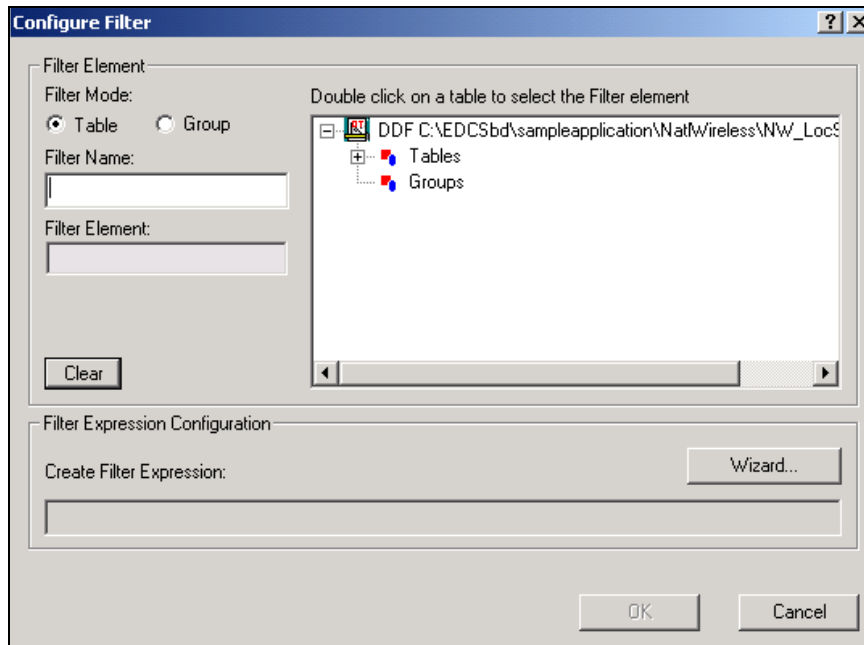
eaDirect filters group occurrences based on the filter column value in the **first row of the first occurrence of the table** in the group. The first occurrence of the table on which you define the filter element acts as a “lead table,” and the first row of this table occurrence acts as a “lead line” for filtering the group.

To enable filtering across occurrences of a table and/or group, create a combine element on the table and/or group in addition to the filter element. To create a combine element, see “Using Combine Elements to Page, Sort, & Filter Across Table or Group Occurrences” on Page 333.


Creating a General Filter Element

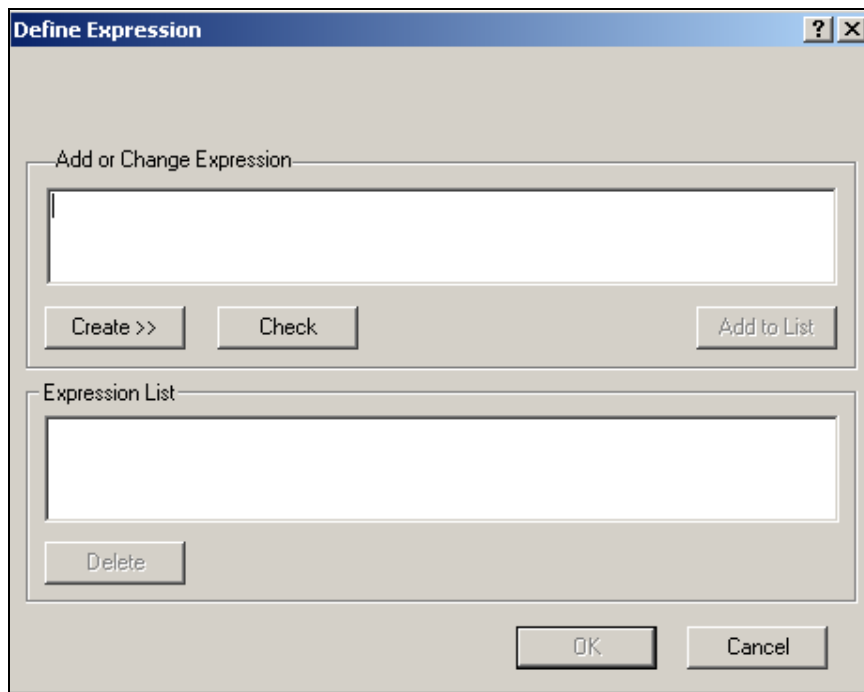
To create a general filter element:

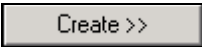
1. With the ALF open in Composer, right-click the  Filter Elements node in the Application Tree. (Open NW_LocSummary.alf.)
2. Select **Create Filter Element** from the right-click menu. The Configure Filter dialog appears.

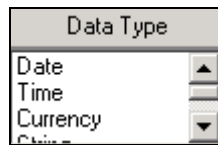
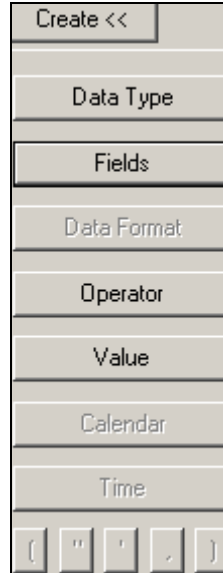


3. Select the filter mode to indicate whether the filter is on a table or group. (Keep Table node selected for National Wireless.)

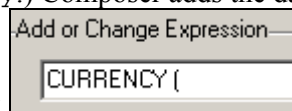
4. Enter a descriptive name for the filter. The name must not contain spaces or special characters, must start with an alpha character, and cannot exceed 30 characters in length. (Enter LocalOverFiftyDol.)
5. Double-click the table (or group, if in group mode) you want to filter. If you make a mistake, click **Clear** and reselect the table. (Select the LocalLineSummary table.)
6. Click  to build a filter expression. The Define Expression dialog appears.



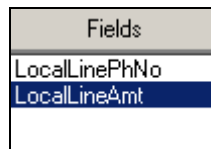
7. Click  to display the buttons for selecting fields, operators, dates, etc. to build an expression in the **Add or Change Expression** box. Select or enter items in the sequence you want them to appear in the expression. (For detailed information on Data Filtering Expressions, see Appendix A.) A drop-down menu with button selections appears.



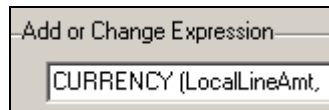
8. Click **Data Type** and select the type of data in the column or field you want to compare. You do not need to select a data type for a string. (Select *Currency*.) Composer adds the data type and an open parenthesis to the expression:



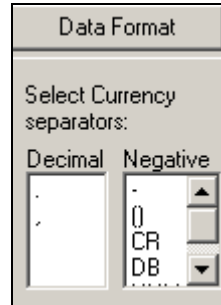
parenthesis to the expression:



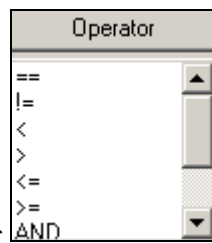
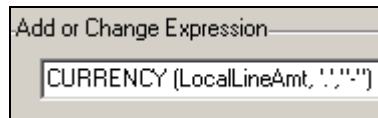
9. Click **Fields** and select the column or field you want to compare. (Select *LocalLineAmt*.) Composer adds the column name (and a comma, if data type is not string) to the expression.



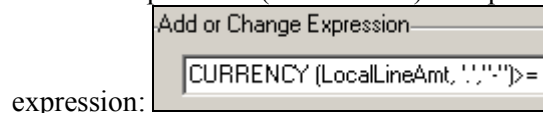
- If you are filtering on a date, time, decimal, or currency value, click **Data Format**.



- Select the format of the data as stored in the input file. (For additional information about valid format strings you can enter, see Appendix A.) Composer adds the format and a close parenthesis to the expression. Manually add any required separators, as in `DATE (DueDate, "%m/%d/%Y")`. (For NatlWireless, select a period “.” for the decimal separator, and a minus sign “-” for the negative separator.)



- Click **operator** `AND` and select the appropriate operator for the filter expression. (Select “>=”) Composer adds the operator to the



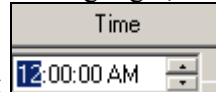
- Click **Data Type** and select the same data type you selected in the first part of the filter expression. (Select data type Currency.) Composer adds the data type and an open parenthesis to the second part of the expression.

14. Specify the field, column, date, time, or value for the filter comparator:

- **To specify a date for comparison**, click **Calendar** and choose a specific month, year, and day. Composer adds the date in quotation marks, followed by a comma, to the expression.

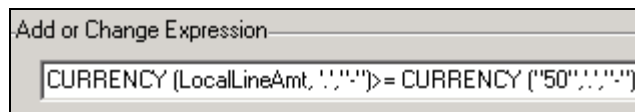
Sun	Mon	Tue	Wed	Thu	Fri	Sat
25	26	27	28	29	30	31
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5


- **To specify a time for comparison**, click **Time**. Click the hour, minutes, seconds, or AM/PM to highlight, and use the up and down



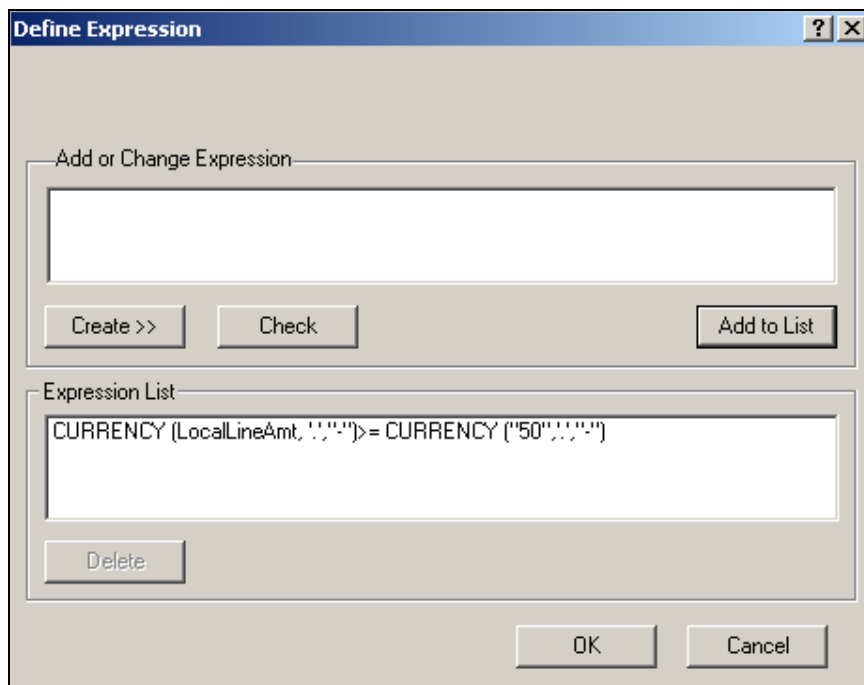
arrows to specify the value: `12:00:00 AM`. Composer adds the time in quotation marks, followed by a comma, to the expression.

- **To specify a currency, decimal, or integer value**, enter the value between quotation marks followed by a comma. Add a close parenthesis after an integer value. (For NatlWireless, enter “50” followed by a comma, then select “.” and “-“ separators.)

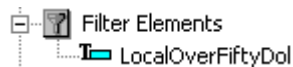



- **To specify a string value**, enter the comparator string in quotation marks, as in “Massachusetts”. Type quotation marks directly or click .
- **To compare a string column to a NULL or EMPTY**, click Value and select NULL or EMPTY. Composer adds the word NULL or EMPTY to the expression.

- **To compare the value to a column or field**, click **Fields** and choose the column or field. Note that you can compare a column to a column, pure field, or dynamic field. In group mode, you can compare a group field to a group, pure, or dynamic field, but not to a column. Composer adds the column or field name in quotation marks, followed by a comma, to the expression.
15. If the data type is a date, time, currency, or decimal, click **Data Format** and select the same format you selected in the first part of the expression. Composer adds the format to the expression followed by a close parenthesis. Manually add any separators to the second part of the expression, as in:
DATE ("10/15/2002", "%d/%m/%Y").
 16. When the filter element expression is complete, click **Check** to verify the integrity of the expression.
 17. When the expression is correct, click **OK**.
 18. Click **Add to List**. The expression appears in the Expression List.



19. To create a compound expression, click **operator** and select AND, OR, or NOT, and creating another expression.
20. Click **OK**.
21. Click **OK** on the Configure Filter screen. Composer displays the filter name in the Application Tree. The “T” next to the filter name indicates it is on a table.




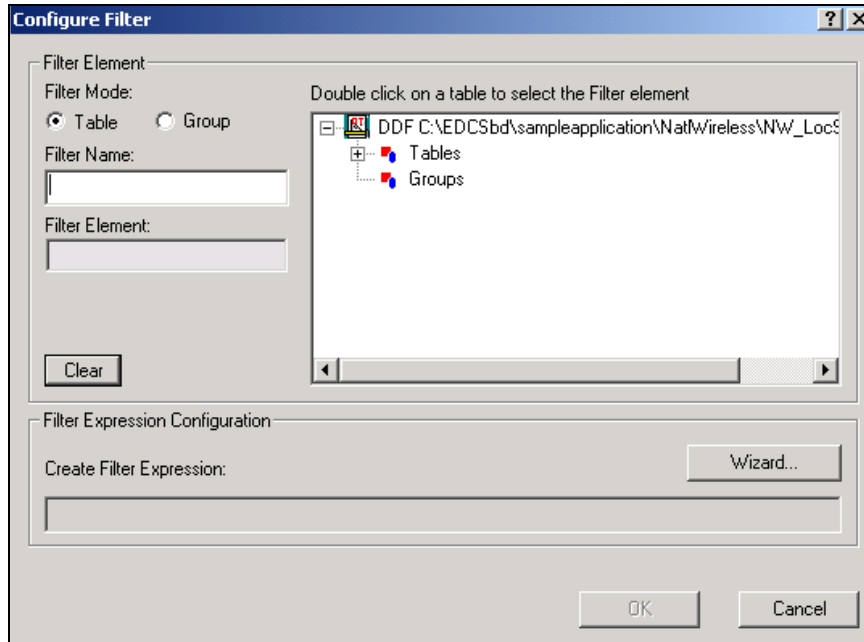
22. Click  to save the ALF.

Creating a Top/Bottom Filter


Top/bottom filters let you create a counted subset of filtered data from either the top or bottom of the filtered data set. For example, you could create a top/bottom filter element to display the top ten most expensive phone calls in a telecomm statement.

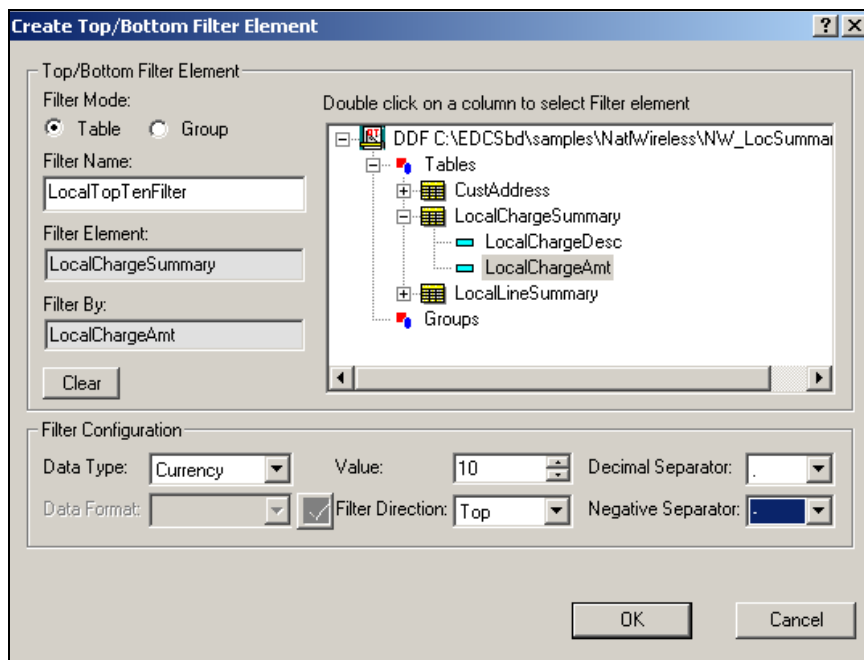
To create a top or bottom filter:

1. With the ALF open in Composer, right-click the  **Top/Bottom Elements** node. (Open `NW_LocSummary`.)
2. Select **Create Top/Bottom Element** from the right-click menu. The Configure Top/Bottom Filter dialog appears.

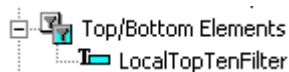


3. Select the filter mode to indicate whether the top/bottom filter is on a table or group.
4. Enter a descriptive name for the top/bottom filter. The name must not contain spaces or special characters, must start with an alpha character, and cannot exceed 30 characters in length. (Enter `LocalTop10Filter`.)
5. Select the group or table you want to filter. If you make a mistake, click **clear** and reselect the table. (Open the `LocalChargeSummary` table.)
6. Select the column you want to filter. If creating a filter on a group, you can select a group field. (Select `LocalChargeAmount`.)
7. Select a data type. (Select `Currency`.)
8. If the data type is a date or time, then select the appropriate format in the `Data Format` field. See Appendix A for more information about data definitions.

9. If the data type is date or time, click the Validate button  to verify that the format in the Data Format field complies with the data type. (See Appendix A for more information about data definitions.)
10. Select a value for the number of data entries you wish to present. (Select 10.)
11. Select the direction of the filter; top or bottom of the data. (Select Top.)
12. If the data type is currency or decimal, select a decimal separator. (Select a decimal point “.”)
13. If the data type is currency, select a negative separator. (Select the minus sign.)



14. Click **OK**. The top/bottom filter appears under Top/Bottom Elements of the Application Tree. The “T” next to the filter name indicates it is on a table.



Linking a Filter Element to a Statement

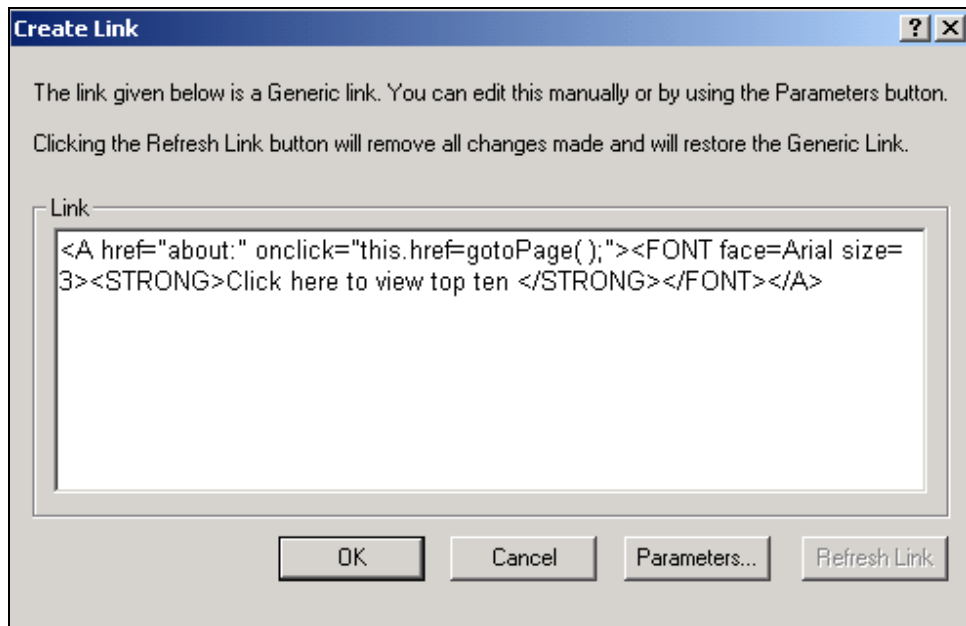
After creating a filter element, you must add a link to the statement for users to view the filtered data. (You can optionally create a filter or sort element while creating a link.) You can create a link on static text, an image, or on an **[E]** tag.

You can have a combination of link to view, sort, and filter links in a statement.

You can optionally choose to reset the URL as part of the link. This enables you to clear any previous URL actions the Web user took, such as sorting or filtering data, before applying the filter. If you do not choose this option, the filter would be cumulative on any previous URLs the user chose in the session.

To create a link to a filter element:

1. With the ALF open, select the item you want to use as a link (be sure to completely highlight the item).
2. Right-click and select **Create Link** from the right-click menu. The Create Link dialog appears.



3. Click **Parameters**. The Add Parameters dialog appears.

Specify the Target View, Dynamic Fields, and Elements to be used in the link.

Target View:

Dynamic Fields

Name	Value	Type

Buttons: Add ... Modify ... Remove Clear All

Elements

Sort Elements : Add To Link Configure and Add

Filter Elements : Add To Link Configure and Add

Top/Bottom Elements: Add To Link Configure and Add

Added Elements

Element Type	Element Name

Buttons: Remove Clear All Reset URL

Sample Link


```
<A href="about:" onclick="this.href=gotoPage( );"><FONT face=Arial size=3><STRONG>Click here to view top ten </STRONG></FONT></A>
```


Buttons: OK Cancel

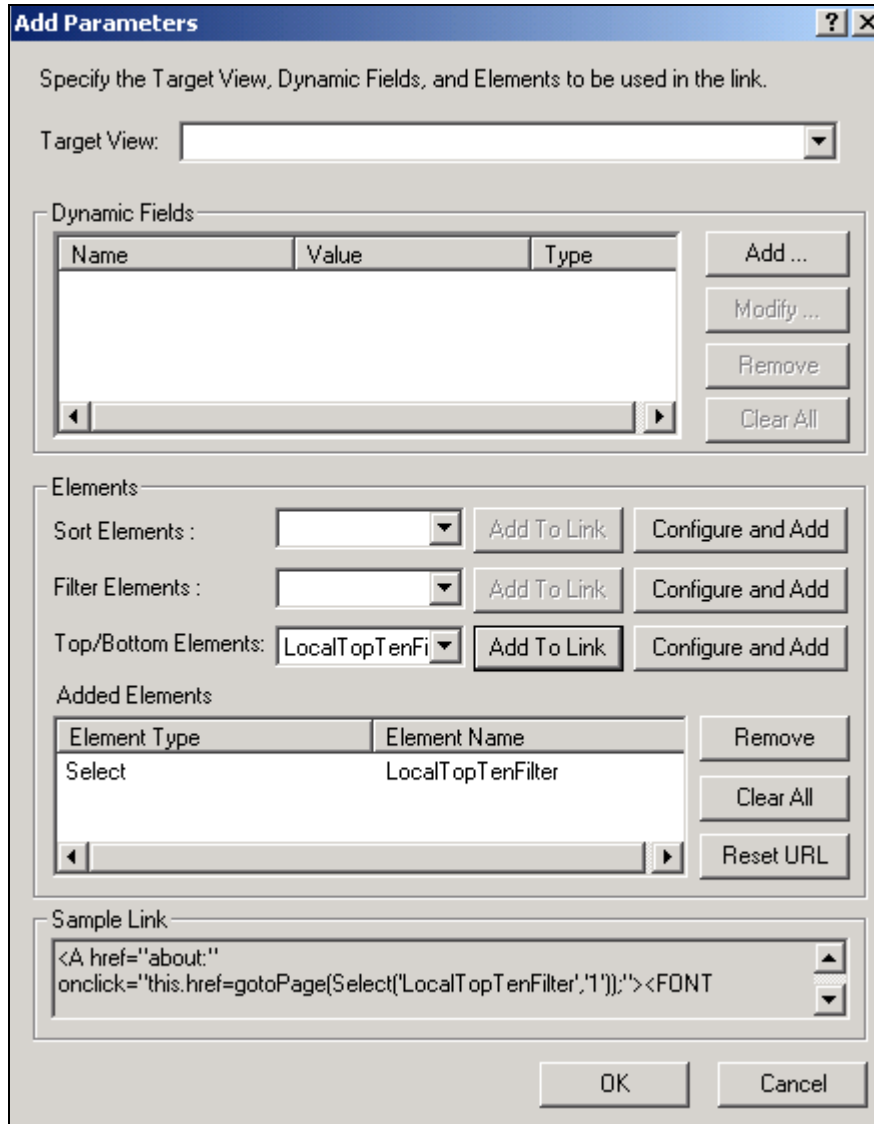
4. If you want to display the filtered data in a different view, enter the name of the **Target view**. (You can also add a filter element, sorting element, or a name/value pair from dynamic pattern matching to the link.)

5. Click the **Filter Elements** or **Top/Bottom Elements** drop-down list and select the filter element you want to link.

Tip

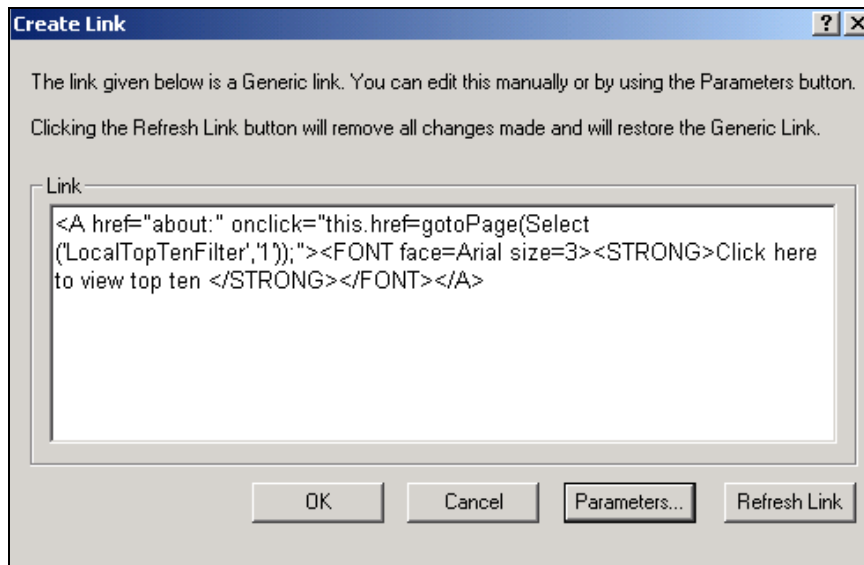
You can optionally create a sort or filter element from this dialog by clicking . See the appropriate procedure in this guide for creating the particular element.



6. Click . Composer adds the filter element to the Added Elements list.



7. If you are using more than one sort or filtering function, click **Reset URL** if you want to add the Reset URL function to the link (this clears the history of the previous URL before applying this filter).

8. Click **OK**. The Create Link dialog appears showing the view name and filter element.



9. Click **OK** to save the link to the HTML Template. Composer adds a blue hyperlink to the statement.
10. Save the template by clicking .
11. Click  to save the ALF if you added a link to a table column.

Modifying a Filter Element

To change the properties of a filter element:

1. With the ALF open in Composer, right-click the filter element (general or top/bottom) and select **Modify Filter Element** (or **Modify Top/Bottom Element**) from the right-click menu. Composer displays the Configure Filter (or Configure Top/Bottom Filter) dialog.

2. Edit a top/bottom filter as needed, or click **Wizard** to edit a general filter. To delete one or more expressions from a compound filter element, select the expression and click **Delete Expression**.
3. Click **OK**. (Click **OK** again if editing a general filter.)

Deleting a Filter Element

To delete a filter element:

- With the ALF open in Composer, right-click the filter element (general or top/bottom) and select **Delete Filter Element** (or **Delete Top/Bottom Element**) from the right-click menu.

Using Combine Elements to Page, Sort, & Filter Across Table or Group Occurrences

Creating a combine element on a group or table lets you to “combine” occurrences of groups or tables when paging, sorting, or filtering a statement.

Defining a table or group as a combine element eliminates breaks across occurrences of the group or table for *all* page, sort, and filter (general or top/bottom) elements defined for the group or table.

Creating a combine element lets the Web user:

- **Click through fewer pages to view a statement.** By default, paging elements start a new Web page for each new table or group occurrence, regardless of how many rows are in each table, often not filling the entire page. Creating a combine element on a table or group eliminates page breaks at the end of the group or table occurrence, displaying them contiguously, requiring fewer pages. Paging lets you display a consistent number of HTML lines per page. (You must create a page element on the table or group to configure the paging feature; see “Paging Statements” on Page 312.)

- **Sort or filter (general or top/bottom) statement data across all table or group occurrences.** By default, eaDirect sorts or filters each occurrence of a table. Creating a combine element on a table or group lets users sort or filter data across all occurrences of the table or group. (You must create a sort or filter element on the table or group to configure these features; see “Sorting Tables” on Page 301 or “Filtering Data” on Page 317.)

Create a combine element wherever it makes sense to remove the boundaries of table or group occurrences and let users page, sort, or filter across this statement data.

Unlike other element types, you do not create a link for a combine element.


You define combine elements in the default ALF only, however eaDirect applies the combined property to any elements that appear in any switch ALFs.

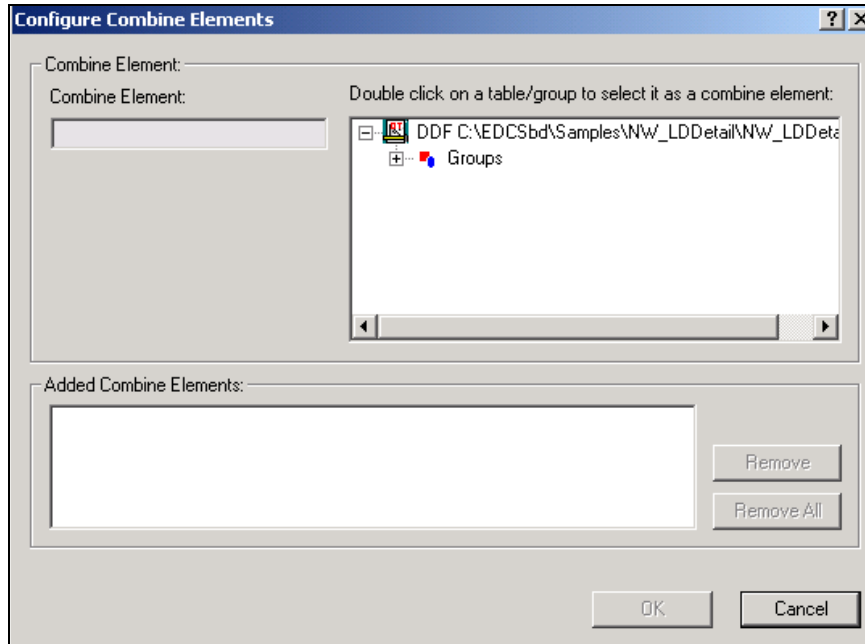


If you create a combine element on a group that has group fields at that level, eaDirect displays the value in the first occurrence of the group only. The data associated with the group field in subsequent occurrences of the group does not appear, and is effectively lost to the viewer. (Group fields below that level still appear with each occurrence.)

Creating a New Combine Element

To create a new combine element:


1. Creating With the default ALF open in Composer, right-click the  **Combine Elements** node in the Application Tree.
2. Select **Configure Combine Elements** from the right-click menu. Composer displays the Configure Combine Elements dialog.



3. Select the group or group table on which you want to combine occurrences.
4. Select any additional groups or group tables to create additional combine elements (if any).
5. Click **OK**.

Viewing and Deleting Combine Elements

To view a list of or delete combine elements:

1. With the default ALF open in Composer, right-click  **Combine Elements**.
2. Select **Configure Combine Elements** from the right-click menu. Composer displays the Configure Combine Elements dialog.
3. You can see the complete list of combine elements for the view listed under **Added Combine Elements**.

To delete a combine element, select the element from the list and click **Remove**. (To remove all combine elements, click **Remove All**.)

4. Click **OK**.

Totaling with Arithmetic Elements

You can create an arithmetic element to calculate and display the sum of all values in a table column or the total number of rows in a table (line count). Arithmetic elements let you create subtotals or totals not provided in your application's data input file. You can also calculate totals of sorted or filtered data. You can create an arithmetic element on a group field and place it anywhere on a template, on any hierarchy.

You can use arithmetic elements with paging, however, *if you do not also sort or filter the data, you can only display a total at the end*. (Paged data extracts progressively, and is therefore not available for summing until the last set displays. If you sort or filter the data when paging, however, data extraction is not progressive, and is extracted and available for subtotaling individual tables in a group for display on each page.)

You can configure an arithmetic element to create a total for:

- A single occurrence of a table (one total for one occurrence of a table, also called a “pure table”)
- Each occurrence of a table (a total for each table occurrence in a group)
- Each occurrence of a group (one total for all tables in a group occurrence)
- All occurrences of a table across all occurrences of a group (one total for the statement)

Once you configure the arithmetic element for a table, you must drag and drop the arithmetic element to the appropriate place in the default HTML template or in the group template. The value eaDirect calculates depends on where you place the arithmetic element in the statement:

- **In the default HTML template** - To total a column in a single occurrence of a table or to generate one grand total of column data across all occurrences of a table, add the arithmetic element directly to the default HTML template.
- **In the group HTML** - To calculate a total on each occurrence of a group, place the arithmetic element in the group template. To calculate a total for each table occurrence in a group, add an HTML table to the group template and place the element inside the table. (This behaves like a footer to the table section in the ALF and calculates a total for each group occurrence.)

You can total the values in a column of data that contains a time, currency, integer, or decimal value only. You can calculate a line count on any type of column.

For time, currency, and decimal values, you must specify how the data is stored in the input file (time requires a strptime data format). You can also specify the output format, as well as any text or characters to display with the output, such as “Total:” and a currency symbol (\$, £, etc.).


Specifying output format with the arithmetic element (vs. directly on the template) is necessary when calculating a subtotal for each table in a group or when you apply paging without also sorting or filtering. eaDirect can’t compose a sum until all data has been extracted with the final page, and you wouldn’t want the words “Total:” or other output formatting to appear on every page without the sum or count.

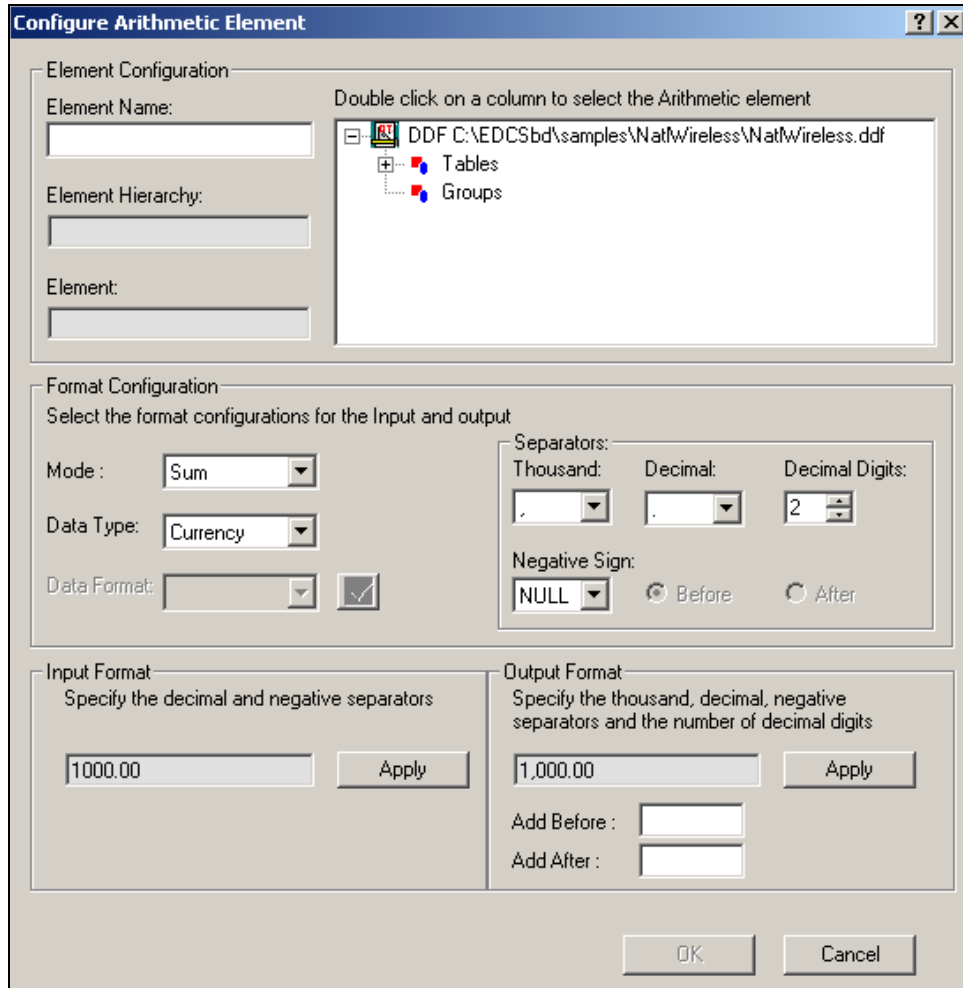
**Caution**

Be sure to specify the correct data format in the arithmetic element. If some rows of data match the specified data format and others do not, eaDirect includes only matching rows in the total. When you validate elements, Composer reports the non-matching rows.

Creating a New Arithmetic Element


To create an arithmetic element and add it to your statement:

1. With the ALF open in Composer, right-click the  **Arithmetic Elements** node in the Application Tree, and select **Create Arithmetic Element**. (For NatlWireless, open NW_LocSummary.alf.) Composer displays the Configure Arithmetic Element dialog:



2. Enter a meaningful name for the arithmetic element. (Enter **PhoneLineCount**.)
3. Expand the **Tables** or **Groups** node and select the table you want to create an element on (element hierarchy). (In NW_LocSummary.alf, expand the LocalLineSummary table.)
4. Double-click on the column or field you want to total or count. The element can be a group field. (Select LocalLinePhNo.)

5. Select the type of arithmetic element you want to create, either sum or count, from the Mode drop-down list under Format Configuration. (Select Count.)
6. If the arithmetic element is to sum values, select the type of data in the column: time, currency, integer, or decimal, then specify the following data input and output information.

Time: Select or enter the strptime format used in the input file. Click the  button to verify its validity. Click the **Apply** button under Input Format, then select or enter the format you want to use to display the output, and click the **Apply** button under Output Format.

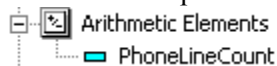
Currency: Specify the decimal and negative separators used in the input file, and click the **Apply** button under Input Format. Specify the thousand, decimal, and negative separators and the number of decimal digits to use to format the output, and whether you want it to appear before or after the number, and inside or outside of brackets () (if you selected brackets as the negative sign). Click the **Apply** button under Output Format

Decimals: Specify the decimal separator used in the input file, and click the **Apply** button under Input Format. Then specify the thousand and decimal separators and number of decimal digits to use for the output format, and click the **Apply** button under Output Format.




Be sure to choose a data output format with enough digits to display the potential sum value (otherwise digits could be cut off).

7. Click **OK**. Composer displays the arithmetic element in the Application Tree:




8. Add the arithmetic element to the statement:
 - **To total a column in a single occurrence of a table or to generate one grand total of column data across all occurrences of a table,** click and drag the arithmetic element from the Application Tree to the default HTML template.

- **To calculate a total on each occurrence of a group**, place the element in the group HTML template; select the table, right-click and select **Configure Table** from the right-click menu. Click-and-drag the element from the Application Tree to the group HTML template.
 - **To calculate a total on each occurrence of a table**, place the element in the group HTML template; select the table, right-click and select **Configure Table** from the right-click menu. Add an HTML table to the group template and drag-and-drop the arithmetic element inside the table. (This behaves like a footer to the table section in the ALF and calculates a total for each table in the group.)
9. Click  to save the ALF.


Modifying an Arithmetic Element

To modify an arithmetic element:

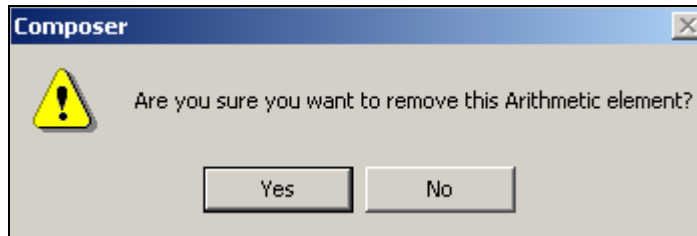
1. With the ALF open in Composer, double-click the  **Arithmetic Elements** node to display a list of arithmetic elements in the Application Tree. Right-click the arithmetic element you want to edit.
2. Select **Modify Arithmetic Element** from the right-click menu. The **Configure Arithmetic Element** dialog appears.
3. Edit the properties of the element as you did when adding the new element, and click **OK**.

Deleting an Arithmetic Element

To delete an arithmetic element:

1. With the ALF open in Composer, double-click the  **Arithmetic Elements** node to display a list of arithmetic elements in the Application Tree. Right-click the arithmetic element you want to delete.

2. Select **Delete Arithmetic Element** from the right-click menu.
Composer asks if you're sure you want to delete the element:



3. Click **Yes**. Composer removes the arithmetic element from the ALF.
4. Manually remove the `[E]ElementName,A[\E]` tag from the correct HTML template.


Using the Simulator

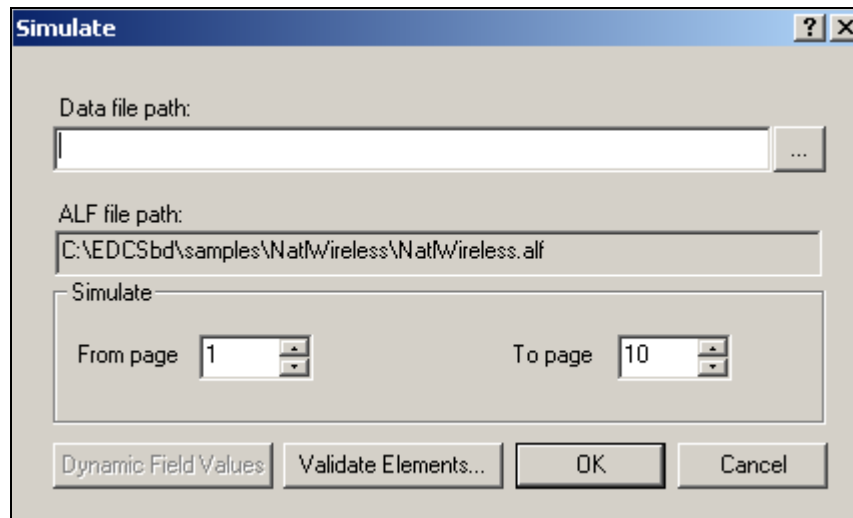
Use Composer's Simulator feature to view the customized presentment of statement data as it would appear live in a browser. Simulator uses the DDF, ALF, and HTML templates to deliver a composed HTML version of the statements using a sample data input file. Simulator prevents the "blind design" technique often associated with application development. You can run the Simulator at any time during the development process.

You can enter values and check individual dynamic fields. Simulator also lets you select sort and filter elements to validate. You must select and validate each sort element define on the same table one at a time. By default, Simulator verifies all arithmetic elements used in the templates, reporting the results in the Data Type Report. (You can turn this option off.)

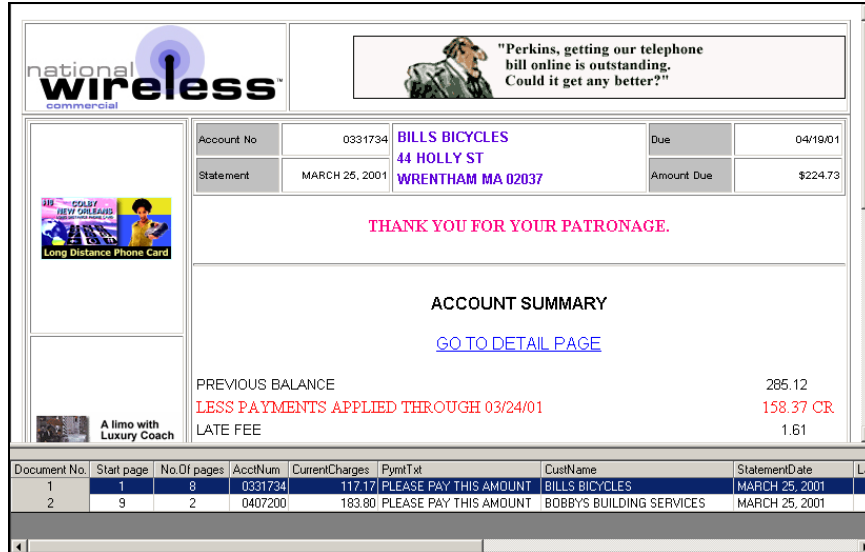
When you select elements to validate, Simulator generates the Data Type Report with the simulation results. You can view the report online, print, and/or save the log to a file.



To run a simulation:

1. Select **View>Simulate** or click  on the Simulator toolbar. The Simulate dialog appears.



2. Select a data file. (For example, select NatlWireless.txt. for the NatlWireless.alf and the NatlWireless.htm template.)
3. Specify the page range to include in the simulation. (Enter 1-110 pages.)
4. If you are simulating using dynamic field values, see Simulating Dynamic Field Values.
5. If you are simulating sort or filter elements, see Simulating Sorting and Filtering Elements.
6. Click **OK**. The Simulate Bar appears displaying a composed HTML view of each statement appears in the right-pane and a simulated listing of the data.





7. To view the statements, click the navigation icons   on the Simulator toolbar or click on the pages in the simulated document.



Tip

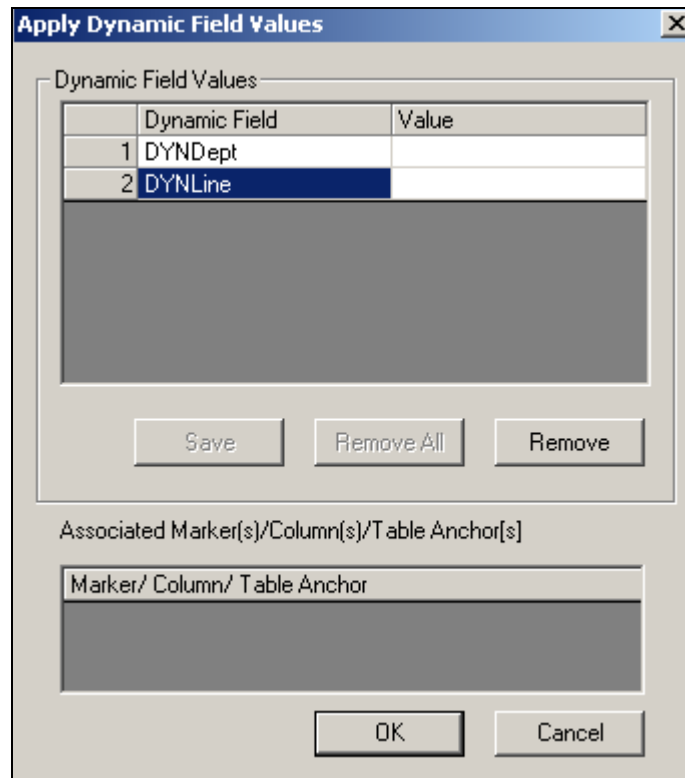
You can drag and drop columns to change their position in the “hit list,” which is useful for comparing columns.

8. To return to the uncomposed HTML statement work area, click the toggle icon .
9. After editing, click the refresh icon  to refresh the simulation without navigating to a sample data source.

Simulating Dynamic Field Values

To simulate dynamic field values:

1. While simulating the statement, from the Simulate dialog, click **Dynamic Field Values**. The Apply Dynamic Field Values dialog appears.



2. Click to select the dynamic field for simulation. Selecting a dynamic field will list the associated names where the dynamic field is used in the DDF data elements in the Associated Marker(s)/Columns(s)/Table Anchors(s)” List Box. (In this example, the **DYNLocalLinePhone** field is selected.)
3. Double-click in the Value field to enter the dynamic value. (Enter **202**.)
Click **Save** to save the field values to the registry.
Click **Remove All** to remove all dynamic field values from the registry.
Click **Remove** to remove the selected dynamic field value from the registry.
4. Click **OK**. The Simulate dialog appears.

5. Click **OK**. The Simulate Bar appears with a composed HTML view of each statement, with the dynamic field values in the right-pane.

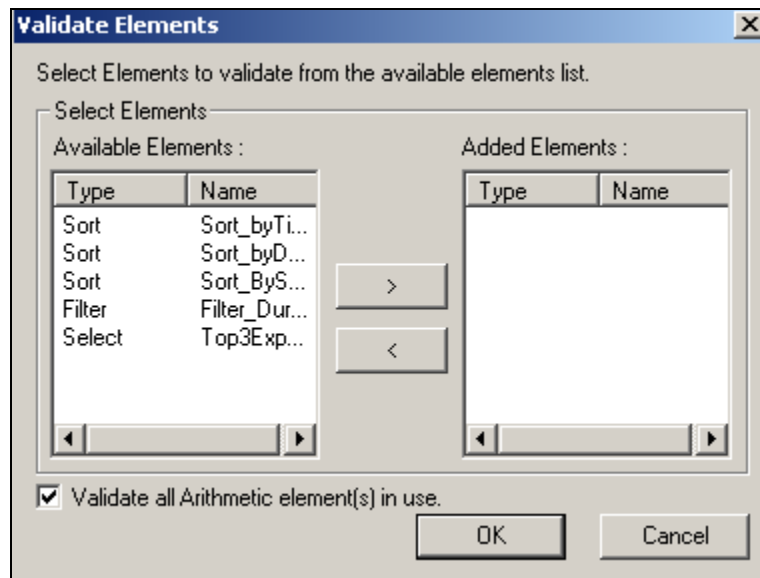



Simulator continues to apply the dynamic field value for subsequent simulations. To remove the value, double-click in the Value field on the Apply Dynamic Field Values dialog and remove the value.

Simulating Sorting and Filtering Elements

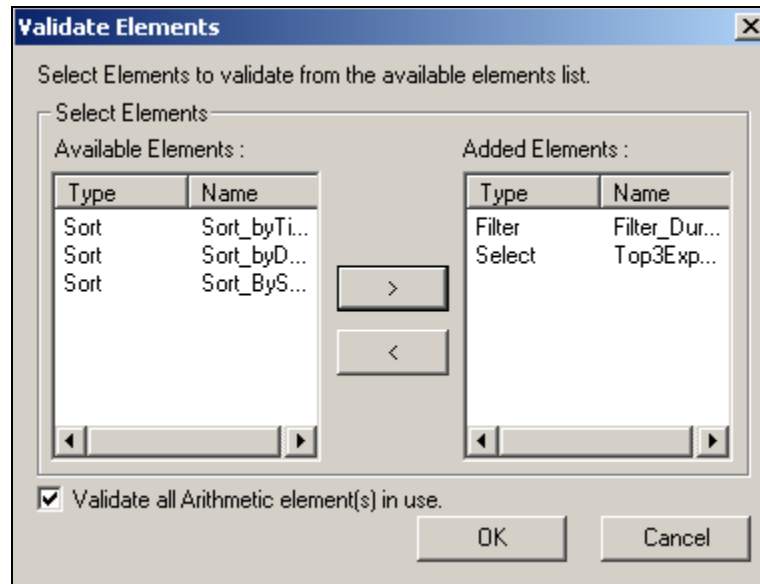
To simulate sorting and filtering:


1. While simulating the statement, from the Simulate dialog, click **Dynamic Field Values**. The Validate Elements dialog appears.

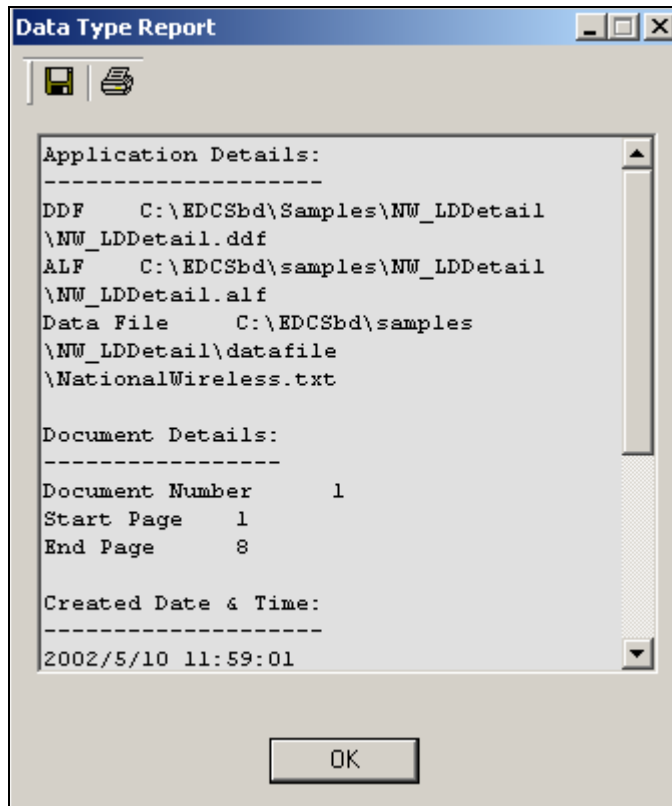




2. Select the filters and/or sort elements you want to sort from the Available Elements list and click  to place it in the Added Elements list. (You can also double-click on an element to move it from one window to the other.) You can only select and validate one sort element defined on table at a time.

3. If you want Simulator to validate all arithmetic elements placed in the templates, check the “Validate all Arithmetic element(s) in use” option.



4. Click **OK**. The Simulate dialog appears.
5. Click **OK**. The Simulate Bar appears with a composed HTML view of each statement with the sorted and filtered data only appearing in the right-pane.
6. Click the Data Type Report icon  in the Simulate tool bar. Composer displays the Data Type Report dialog. Scroll down the report to view element details.



7. To save the Data Type Report information as a text document with a time stamp as the filename, click .
8. To print the Data Type Report, click .
9. Click **OK**. Composer prompts you to save the log if you have not.

Caution

Composer continues to apply the added elements to subsequent simulations. To remove the elements, enter the Apply Dynamic Field Values dialog, select the element in the Added Elements list and click



(or double-click on the element).

Mapping Tables as Charts Using Composer

5

Chart Information in the ALF


You can drag and drop the tables defined in the DDF into the HTML template using the WYSIWYG or the HTML editor. You can represent a table as either a text table or as a chart in the HTML template.

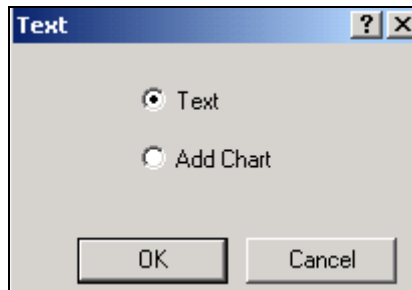


Tip

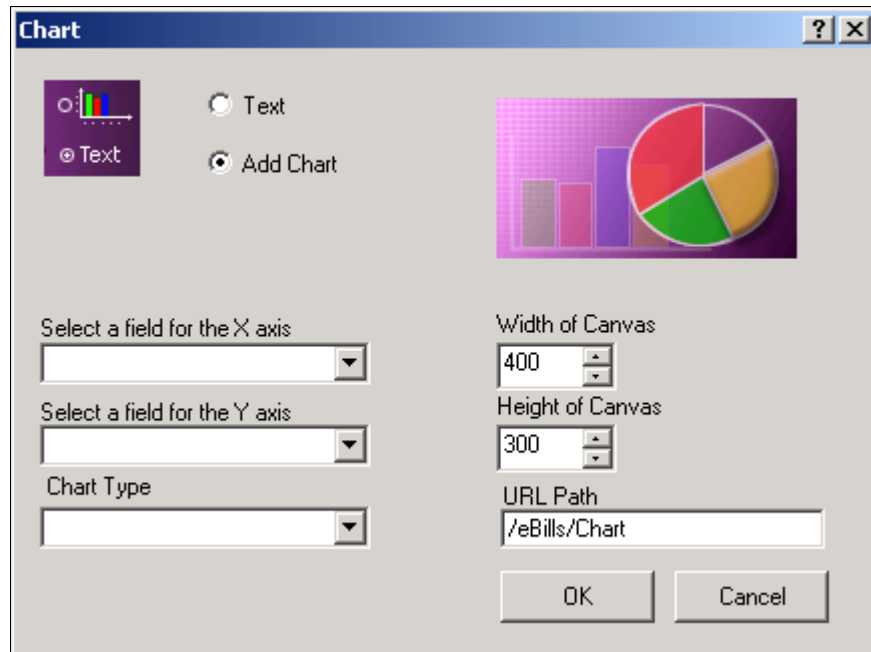
The charting option is NOT available for a table with only one field.

To create a chart:

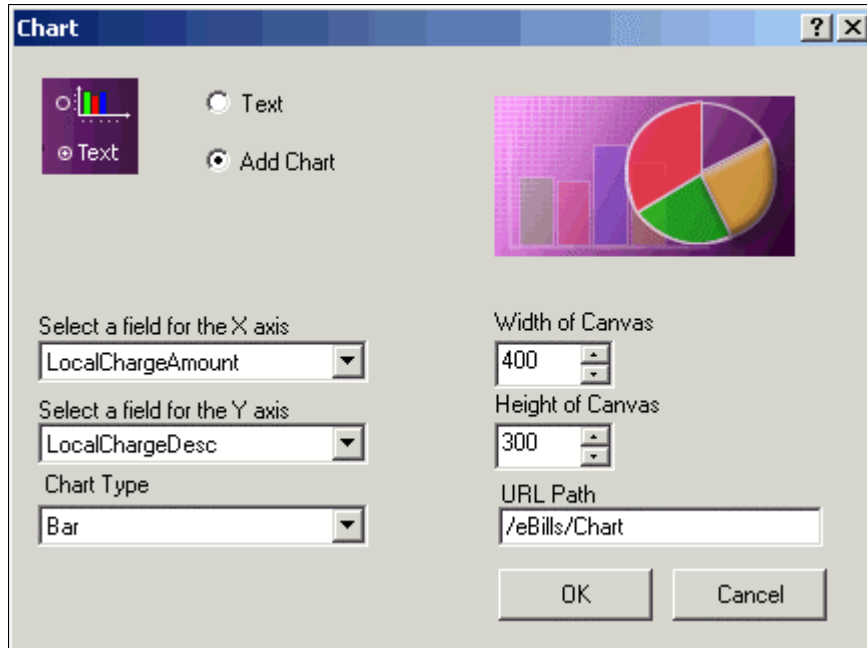
1. Click the **WYSIWYG** tab or **HTML** tab.
2. Click the **Definition** tab in the Tree.
3. Click to open **Tables** in the Tree.
4. Drag and drop the table  definitions into the HTML template. The table assumes the properties of the area in which it is placed. (Drag and drop the LocalChargeSummary table to the HTML Template.)



5. Select **Chart**.



6. Select a field for the X axis of the chart. (Select `LocalChargeAmount`.)
7. Select a field for the Y axis of the chart. (Select `LocalChargeDesc`.)
8. Select the type of chart. (Select `Pie`.)
9. Select the width and height settings for the chart. (Leave at 400 and 300 respectively.)
10. Enter the URL path. (Enter the path to the Web Server chart storage area.)




11. Click **OK**. (The tag `[E]LocalChargeSummary_0,U[/E]` appears.)

At this point a place holder for the chart is added to the template and a properties file is created. Example: If the table name is **Test** then a `Test_0.Properties` file is created in the `AppProfiles\DDN Name` directory (Where the ALF and Templates are located). When creating a chart, make sure that there is more than one field in the table and that there are numeric values included. Even though the field type is not numeric, the values are converted into a numeric format by the Chart engine.

12. If there are placeholders in the HTML template, delete them. (Delete the “XX” placeholders.)

13. Click the **Save Template** icon .

14. Save the ALF by clicking the **Save ALF** icon .



Tip

When mapping a table to a template in the Composer, it is not necessary to encapsulate the table with HTML table row `<TR>` and table data `<TD>` (cell) tags. The table rows and table data (cells) are generated when the data is dynamically pulled from the data file and passed to the edocs WebComposer object. This object formats the table rows, cells, and font characteristics of the data based on the settings defined in the Composer.

Charting Attributes

For an eaDirect view to display charts, an ALF must be constructed and placed in the view directory. The ALF will provide the document type definition (DTD) in the ALF (usually at the beginning of the document).

The DTD will have the attributes (described here) assigned to the chart.

Attribute Name	Meaning
Height	Total height of the canvas
Width	Total width of the canvas
HidePieLegend	Toggles the display of legends for Pie charts
AddValueToLegend	Displays the percentage in the chart legend
BaseURL	Points to the location where the chart images are stored
UnixChart	Chart types in a UNIX platform

You will notice other attributes listed in the ALF. These attributes are placed in the ALF for backward compatibility with previous versions of eaDirect and have no effect on the current version of eaDirect.

Chart Information Contained Within the ALF

You can see the HTML for the chart. For example, after defining your table properties and saving your ALF, open the *NatlWireless.ALF* in Notepad and scroll down to the [CHARTS] section. There you will find the HTML that will be pulled into the Statement Template during assembly (when the chart tags are found). An example of chart information follows:


```

<CHARTS><Chart Name = "LocalChargeSummary_0"RecordName =
"LocalChargeSummary"TopTitle = ""BottomTitle =
""LeftTitle = ""RightTitle = ""XField =
"LocalChargeDesc"YField = "LocalChargeAmount"Key =
"1"StackedStyle = "0"ColorScheme = "0"GridLines =
"3"Full3D = "1"AngleX = "45"AngleY = "0"Attribute =
"109577474"MarkerVolume = "3"Shadow = "1"MultiShape =
"11"Dimension_3D = "1"View3DDepth = "80"Type =
"0"CGITimeSpan = ""BackgroundColor = ""ForegroundColor =
""Height = "300"Width = "400"LegendShow =
"1"LegendToolSize = "100"LegendToolStyle =
"167116800"HidePieLegend = "0"SeriesColor = ""LeftGap =
"40"RightGap = "40"ImgQuality = "75"ImgSmooth =
"0"AddValueToLegend = "0"BaseURL = ""UNIXChart =
"" ></Chart></CHARTS>

```

Chart Properties

The chart properties file is a list of name-value pairs that control the type of chart, location of the axes, fonts, colors, whether to use three-dimensional effects, etc. To change the display of some aspect, an alternate value for the controlling property must be provided. If there is a conflict between a chart attribute defined in the ALF and a value defined in the chart property, it will override the chart attribute.

The chart properties file is included as part of your published application view using eaDirect's Command Center. Publishing views are described in detail in the *Production Guide*. Publishing chart views follows the same general process. The following summarizes the parts specific to charts:

- In the Publisher window, select the **Create** button. Set values for "ddn" (e.g., NationalWireless), job type = "HtmlDetail", and viewName (e.g. "Cell_0").
- In the next window that pops up, in the Web-Views section, which is also the upper portion of the page, select the option "Chart 0". You are next requested to specify, through your file system, the chart properties file. A file extension of *.properties* is required.

The following sections describe the types of properties included in the file.

Chart Type

The primary chart property is Type. By setting Type = Pie, one obtains a Pie chart, using “Bar” instead, results in a bar chart. The table below itemizes the various chart types currently available to eaDirect running on Unix platforms. Note that the “Type” property and its value are both case-sensitive. The remainder of the properties and their values are not so limited.

Bar	HiLoBar	HorizBar	HorizHiLoBar
IndBar	LabelLine	Line	Pie
Polar	StackBar	StackColumn	Speedo
Stick	StickBar	Regress	

The following are issues related to the chart types:

- Do not set properties that are not applicable to a chart type. For example, do not set Axis properties when requesting a Pie chart.
- Do not set Bar properties while rendering a Pie chart. For charts created using the Composer tool, the chart types: HiLobar, HorizHiLoBar, and Speedo are not available.
- If you want a Bar chart that displays elements of individual data series in different colors, select the IndBar type.

Chart General Properties

General chart properties determine whether the legend should be displayed, whether to provide three-dimensional effects, and finally whether to display the axis. Note that for Pie type charts, the convention is not to display the axis. For example:

```
Chart.LegendVisible=false
Chart.Name=MyChart
Chart.XAxisVisible=true
Chart.YAxisVisible=true
Chart.ThreeD=false
```

```

## the XOffset and YOffset determine the amount of three
## dimensional effect along the X and Y axes
## respectively.
Chart.XOffset=0
Chart.YOffset=0
Chart.Quality=1

```

The `Quality` property defaults to 1 for the highest quality image possible. A value of 0 sets it to a lower quality image.

Background Properties

Titles and sub-titles are elements of the chart background. Their color, font and string value are controlled by appropriately setting the following background properties:

```

## Red, Green and Blue values, valid values 0-255
##
Background.Gc.FillColor.Red=0
Background.Gc.FillColor.Blue=0
Background.Gc.FillColor.Green=0

Background.Gc.LineColor.Red=0
Background.Gc.LineColor.Blue=0
Background.Gc.LineColor.Green=0

Background.Gc.LineWidth=1

Background.Gc.MarkerColor.Red=0
Background.Gc.MarkerColor.Blue=0
Background.Gc.MarkerColor.Green=0

Background.SubTitleColor.Red=0
Background.SubTitleColor.Blue=0
Background.SubTitleColor.Green=0

Background.SubTitleFont.name=TimesRoman
Background.SubTitleFont.style=plain
Background.SubTitleFont.size=12

Background.SubTitleString=null

Background.TitleColor.Red=0
Background.TitleColor.Blue=0
Background.TitleColor.Green=0

```

```
Background.TitleFont.name=TimesRoman
Background.TitleFont.style=plain
Background.TitleFont.size=12

Background.TitleString=null
```

Plot Area Properties

The plot area is the region bounded by the axis; where the data is actually plotted. The fill color for this region, in addition to the marker and grid line particulars, may be specified. For example:

```
Plotarea.Gc.FillColor.Red=0
Plotarea.Gc.FillColor.Blue=0
Plotarea.Gc.FillColor.Green=0

Plotarea.Gc.LineColor.Red=0
Plotarea.Gc.LineColor.Blue=0
Plotarea.Gc.LineColor.Green=0

Plotarea.Gc.LineWidth=1

Plotarea.Gc.MarkerColor.Red=0
Plotarea.Gc.MarkerColor.Blue=0
Plotarea.Gc.MarkerColor.Green=0
```

Axis Properties

The axis properties control the location of the axis on the canvas, its major and minor ticks, title, grid, and labels. Note that while only the X axis properties are presented below, an identical set of properties exist for the Y axis that starts with “Y.” instead of “X.” For example:

```
## to skip labels that collide
## meaningful if the user can guess the missed labels
## generates a more legible axis without having to
reduce
## font size or increase the chart height or width.
##
X.Axis.CullingLabel=false

## Useful most times
##
X.Axis.AutoScale=true
```

```
## ending point for the axis. For AUTO_SCALE, the axis
## end will be automatically determined. For log-scaled,
it is
## a power of ten
##
X.Axis.AxisEnd=6.0

## starting point for the axis. For AUTO_SCALE,
## the axis start will be automatically
## determined. For log-scaled, it is a power of ten
##
X.Axis.AxisStart=0.0

## place bar within axis, set true for bar charts
##
X.Axis.BarScaling=true

## Display grid lines
##
X.Axis.GridVis=false

## Tilts the label by amount specified, integer value
##
X.Axis.LabelAngle=0

## Axis label color
##
X.Axis.LabelColor.Red=0
X.Axis.LabelColor.Blue=0
X.Axis.LabelColor.Green=0

## Axis Label Font
##
X.Axis.LabelFont.Name=TimesRoman
X.Axis.LabelFont.Style=plain
X.Axis.LabelFont.Size=12

## For double quantities such as currency
##
X.Axis.LabelPrecision=2

## Determines whether the label is visible
X.Axis.LabelVis=true

## Determines whether the axis line is visible
X.Axis.LineVis=true
```

Mapping Tables as Charts Using Composer

```
## Determines whether the scale is log based
##
X.Axis.LogScaling=false
X.Axis.MajTickLength=5
X.Axis.MajTickVis=true
X.Axis.MinTickLength=2
X.Axis.MinTickVis=false
X.Axis.NumGrids=5
X.Axis.NumLabels=5
X.Axis.NumMajTicks=5
X.Axis.NumMinTicks=10

## reduce LlX to left shift axis position on canvas
## increase for right shift, that is along the X
direction
##
X.Axis.Plotarea.LlX=0.2

## reduce LlY to move lower on axis on location
## increase LlY to raise axis, that is along Y direction
X.Axis.Plotarea.LlY=0.2

## similar to LlX and LlY, but controls upper right
corner
##
X.Axis.Plotarea.UrX=0.8
X.Axis.Plotarea.UrY=0.8

## places axis on left, right, top or bottom
##
X.Axis.Side=1

X.Axis.TitleColor.Red=0
X.Axis.TitleColor.Blue=0
X.Axis.TitleColor.Green=0

X.Axis.TitleFont.Family=Times New Roman
X.Axis.TitleFont.Name=TimesRoman
X.Axis.TitleFont.Style=plain
X.Axis.TitleFont.Size=12

X.Axis.TitleString=Remember to set X axis title!
```

```
X.Axis.GridGc.FillColor.Red=0
X.Axis.GridGc.FillColor.Blue=0
X.Axis.GridGc.FillColor.Green=0

X.Axis.GridGc.LineColor.Red=0
X.Axis.GridGc.LineColor.Blue=0
X.Axis.GridGc.LineColor.Green=0

X.Axis.GridGc.LineWidth=1

X.Axis.GridGc.MarkerColor.Red=0
X.Axis.GridGc.MarkerColor.Blue=0
X.Axis.GridGc.MarkerColor.Green=0

## valid values = Left, Right, Top, Bottom
##
X.Axis.Side=Left

X.Axis.TickGc.FillColor.Red=0
X.Axis.TickGc.FillColor.Blue=0
X.Axis.TickGc.FillColor.Green=0

X.Axis.TickGc.LineColor.Red=0
X.Axis.TickGc.LineColor.Blue=0
X.Axis.TickGc.LineColor.Green=0

X.Axis.TickGc.LineWidth=1

X.Axis.TickGc.MarkerColor.Red=0
X.Axis.TickGc.MarkerColor.Blue=0
X.Axis.TickGc.MarkerColor.Green=0

X.Axis.LineGc.FillColor.Red=0
X.Axis.LineGc.FillColor.Blue=0
X.Axis.LineGc.FillColor.Green=0

X.Axis.LineGc.LineColor.Red=0
X.Axis.LineGc.LineColor.Blue=0
X.Axis.LineGc.LineColor.Green=0

X.Axis.LineGc.LineWidth=1

X.Axis.LineGc.MarkerColor.Red=0
X.Axis.LineGc.MarkerColor.Blue=0
X.Axis.LineGc.MarkerColor.Green=0
```

If your labels appear too close together on the X-axis, there is an alternative. You can place legends on the side, top, or bottom by specifying the chart property `Chart.Legend.Visible=true` and by specifying the values for height, width, and colors for the icons. Turn off the parts of the X-axis labels with the `LabelVis` property.

Legend Properties

The legend or a key for a chart are controlled by the following properties:

```
Legend.BackgroundVisible=true
## separation between rows of the legend
##
Legend.IconGap=0.02
## Legend icon height 0 < k < 1.0,
## where 1.0 = full height of canvas
##
Legend.IconHeight=0.05
## Legend icon width 0 < k < 1.0,
## where 1.0 = full width of canvas
Legend.IconWidth=0.07
## Red, Green and Blue values, valid values 0-255
Legend.LabelColor.Red=0
Legend.LabelColor.Blue=0
Legend.LabelColor.Green=0
## Legend Font
##
Legend.LabelFont.Name=TimesRoman
Legend.LabelFont.Style=plain
Legend.LabelFont.Size=12
## Legend lower x left corner 0 < y < 1.0,
## 1.0 = full width
Legend.LlX=0.0
## Legend lower y left corner 0 < y < 1.0, 1.0 = full
height
##
Legend.LlY=0.0
```



```

## Legend upper x right corner 0 < y < 1.0, 1.0 = full
width
##
Legend.UrX=0.0

## Upper y right corner 0 < y < 1.0, 1.0 = full height
##
Legend.UrY=0.0

## true for Legend below chart
##
Legend.VerticalLayout=false

## The following properties change a Legend's graphic
## component.
##
Legend.BackgroundGC.Gc.FillColor.Red=0
Legend.BackgroundGC.Gc.FillColor.Blue=0
Legend.BackgroundGC.Gc.FillColor.Green=0

Legend.BackgroundGC.Gc.LineColor.Red=0
Legend.BackgroundGC.Gc.LineColor.Blue=0
Legend.BackgroundGC.Gc.LineColor.Green=0

Legend.BackgroundGC.Gc.LineWidth=1

Legend.BackgroundGC.Gc.MarkerColor.Red=0
Legend.BackgroundGC.Gc.MarkerColor.Blue=0
Legend.BackgroundGC.Gc.MarkerColor.Green=0

```

Favorite Colors Properties

The properties file allows you to specify your favorite colors, to use as fill color for data slices. You can specify favorite colors to ensure a pleasing overall color scheme, in particular you may want your chart colors not to clash with the colors used in the page embedding the chart. The total number of favorite colors and their order of specification are important to bear in mind for the following reasons:

- **Number:** Favorite colors become active if there are enough of them to meet display needs. For example, if a Pie has “n” distinct slices, where n is greater than the number of favorite colors specified, the favorite colors will not be used. This is because there is no way to guess which colors would go well with those already specified.

- **Order:** The favorite colors are used in the order specified, so it is your responsibility to ensure that a color is distinct and distinguishable from its adjacent colors. Or else as in the case of a Pie chart, the resulting chart may not provide enough contrast

For example, the following defines the first and second favorite colors:

```
Favorite.1.Color.Red=201
Favorite.1.Color.Blue=92
Favorite.1.Color.Green=132

Favorite.2.Color.Red=51
Favorite.2.Color.Blue=52
Favorite.2.Color.Green=53
```

Pie Properties

The pie properties are available only when the selected chart type is Pie. These control the pie height, pie width, location of pie center on the canvas, and how the labels and legends must appear. For example, should the percentage sign be tacked on? Should the pie slice size be shown? Below is the list of various Pie properties along with their default values. Note that color and font related properties are broken into their primitive constituents.

```
Pie.LabelPosition=2
Pie.StartDegrees=0
## true displays pie slice name, e.g., "College Fund"
Pie.TextLabelsOn=false
## true displays the size of pie slice numerically,
e.g., 30.5
Pie.ValueLabelsOn=false
## true turns on the % sign after the pie slice size,
e.g., 30.5%
Pie.PercentLabelsOn=true
## Pie Label Color
##
Pie.LabelColor.Red=0
Pie.LabelColor.Blue=0
Pie.LabelColor.Green=0
```

```

## Pie Label Font,
## font names and style are the same as in java.awt.font
##
Pie.LabelFont.Name=TimesRoman
Pie.LabelFont.Style=plain
Pie.LabelFont.Size=12

## To obtain a circular Pie, scale height and width
using
## true canvas height and width to be equal in pixels.
## For width = 300 and height = 200,
## to obtain a Pie radius of 100, use:
##
Pie.Height=0.5
Pie.Width=0.33

## Locate Pie center. Take into consideration length of
labels
## in legend and whether the legend will be
## displayed horizontally or vertically
##
Pie.XLoc=0.5
Pie.YLoc=0.5

```

If your labels appear too close together, there is an alternative. You can place legends on the side, top, or bottom by setting the chart property `Chart.Legend.Visible=true` and by specifying the values for height, width, and colors for the icons (see the *Legend Properties* section above). Turn off the label by setting the `TextLabelson`, `ValueLabelsOn`, and `PercentLablesOn` properties to false.

Bar Properties

The following properties control how bars are displayed in Bar Charts:

```

## Set the Bar baseline.
##
Bar.Baseline=0.0

## sets the cluster width
##
Bar.ClusterWidth=0.8

```

```
## Set to true to clip bars at Plotarea boundaries.  
##  
Bar.DoClip=false
```

Line Properties

Set to true to clip lines at the plot area boundary:

```
Line.Clip=false
```

Example:



The following is an example of a Pie chart and its non-default property settings:

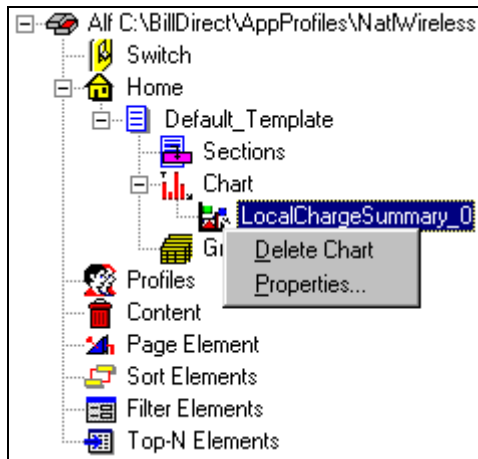
```
Pie.Height=0.5  
Pie.Width=0.33  
  
Chart.LegendVisible=true  
Legend.IconGap=0.02  
Legend.IconHeight=0.05  
Legend.IconWidth=0.07  
Legend.LabelColor.Red=123  
Legend.LabelColor.Blue=126  
Legend.LabelColor.Green=129  
Legend.LlX=0.7  
Legend.LlY=0.4  
Legend.VerticalLayout=false  
Pie.PercentLabelsOn=true  
Pie.TextLabelsOn=false  
Pie.ValueLabelsOn=true  
Pie.XLoc=0.5  
Pie.YLoc=0.5
```

Changing Chart Properties

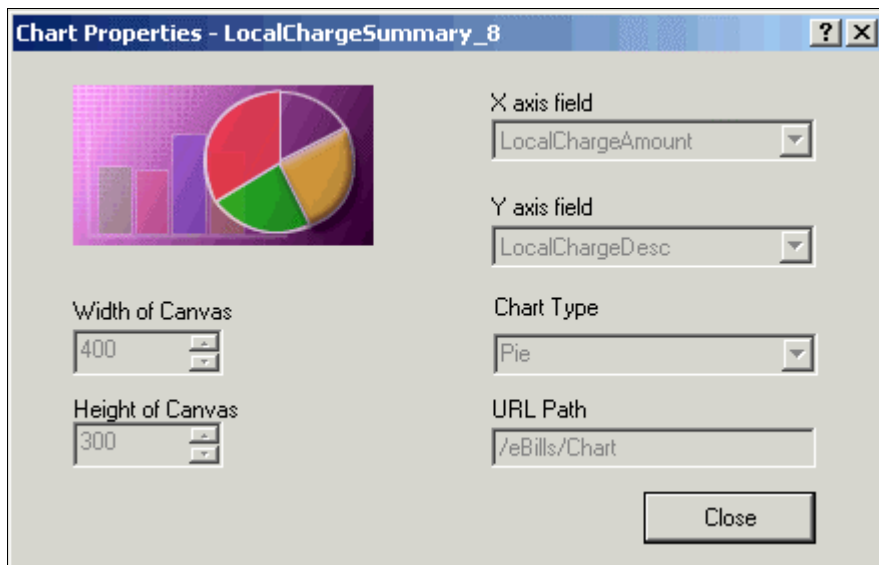
Changing chart properties is similar to the initial creation of a chart.

To display the properties of a chart:

1. In the Application Tree click to open the Charts  , and right-click on the chart  you want to modify.





2. Select **Properties** from the right-click menu. The Chart Properties dialog box appears.



Deleting Charts

To delete a chart:

1. In the Application Tree click to open the Charts , and right-click on the chart  you want to delete.
2. Select **Delete Chart** from the right-click menu.
3. A message appears asking you to confirm the delete operation.
4. Click **Yes**.

Appendix A: Data Definitions

EBNF Description

EBNF (Extended Backus-Naur Form) is a language described by sets of rules, syntax, tokens etc. Specify one or more syntactic groupings and give rules for constructing the language. For example, in C language, one kind of grouping is called an 'expression.' One rule for making an expression might be, "An expression can be made of a minus sign and another expression".

```
Expression ::= - Expression
```

Rules are often recursive, but there must be at least one rule that leads out of the recursion. EBNF's repetition is constructed so that an expression has only one derivation.

The most common formal system for presenting such rules that people can read is called EBNF — a context-free grammar.

Data Definitions

Data definitions supported by eaDirect for use with the EBNF expression language are:

- **String** — Delimit by double quotes.
Example: "foo", "00012", "508-467-9887"
- **Integer** — Format (+|-|)(0-9)(0-9)*
Grouping separator for thousands can be a DCHAR (. or ,)
Example: 10, 35, 10034, -3462 etc.

- **Double/Float** — Format of (+ | - |)(0-9)(0-9)*. (0-9)(0-9)*
This is a double precision value, i.e. a decimal value.
Example: 0.0, -10.23, 10089.4658
- **Date** — Format where DD is the day, MM is the month, YY/YYYY is the year of format (0-9) and Mmm/ Mmmmm is the three-character month name and the full name of the month respectively.
- **Time** — HH stand for hours, MM for minute, and SS for seconds of format (0-9) [AM|PM].
- **Currency** — According to the following format:
(+ | - | (|) (0-9)(0-9)*. (0-9)(0-9)* (- |) | CR | | DB |)
In addition, the currency symbol can appear in the data.
The currency format can change according to the location.

INT(*data*) — Convert the parameter passed as a string to an integer. The characters in the string must all be digits and spaces.

DOUBLE(*data*, “*decimal separator sign*”) — Convert the parameter passed as a string to double/float value. The characters in the string must all be digits, spaces plus the decimal separator sign.

CURRENCY(*data*, “*decimal separator sign*”, “*negative format string*”) — Convert the parameter passed as a string to double/float value. The *negative format string* can be “CR”, “DB” etc., and the *decimal separator sign* can be “.”, “;” etc. Data could also have other characters in the string, such as: “\$ 1, 000, 000 .00 CR”.

Currency Example	Notes
4,123,978.999-	Trailing negative + three decimals
4,123,978.99-	Trailing negative
-4,123,978.99	Leading negative
-4,123,978.999	Leading negative + three decimals
4,123,978.99CR	Trailing CR (credit)
4,978.99 CR	Trailing CR separated by space
4,123,978.999CR	Three decimals, trailing CR (credit)
4,978.999 CR	Three decimals, trailing CR separated by space
4,123,978.99DB	Trailing DB (Debit)
4,978.99 DB	Trailing DB separated by space
4,123,978.999DB	Three decimals, trailing DB (Debit)
4,978.999 DB	Three decimals, trailing DB separated by space

DATE(*data, format_string*) — Convert the parameter passed as a string to Date value.

TIME(*data, format_string*) — Converts the parameter passed as a string to the Time value.

Date/Time Format String Components			
Format	Description	English	French
%%	Parse literal '%'		
%a	Day of week using the locale's weekday names (abbreviated name)	Fri	ven
%A	Day of week using the locale's weekday names (long name)	Friday	vendredi
%b	Month using the locale's month names (abbreviated name)	Jun	jun
%B	Month using the locale's month names (full name)	June	juin
%c	Locale-appropriate date and time representation (abbreviated form)	Fri Jun 08 17:53:28 2002	ven 08 jun 2002 17:50:58 EDT
%C	Locale-appropriate date and time representation (full form)		
%d	Day of month [1-31]; leading zero is permitted but not required	8 08	
%D	Date as %m/%d/%y.		
%e	Same as %d.		
%G	One digit of fractional seconds. Leading zero is permitted but not required. Fractional second value increments in six second intervals; e.g., .1=6 seconds, .7=42 seconds	.6 or 0.6 both equivalent to 0:36	
%h	Same as %b.	Jun	jun

Date/Time Format String Components			
Format	Description	English	French
%H	Hour (24-hour clock) [0-23]. Leading zero is permitted but not required.		
%I	Hour (12-hour clock) [1-12]. Leading zero is permitted but not required.		
%j	Day number of the year [1-366]. Leading zeros are permitted but not required.		
%K	Time duration as HHHH. Leading zeros are permitted but not required.	1324 0123	
%m	Month number [1-12]. Leading zero is permitted but not required.		
%M	Minute [0-59]. Leading zero is permitted but not required.		
%p	Locale's equivalent of either a.m. or p.m.	For U.S. locale: AM, PM, am, pm	
%Q	Time duration as MMM. Leading zeros are permitted but not required.	324 12 012	
%r	Appropriate time representation in the 12-hour clock format with %p.		
%R	Time as %H:%M.		

Date/Time Format String Components			
Format	Description	English	French
%S	Seconds [0-61]. Leading zero is permitted but not required. The range of values is [00-61] rather than [00-59] to allow for the occasional leap second and even more infrequent double leap second.		
%T	Time as %H:%M:%S.		
%w	Weekday as a decimal number [0-6], with 0 representing Sunday.		
%x	Locale-appropriate date representation.	06/08/02	08.06.2002
%X	Locale-appropriate time representation.	17:50:03	17:50:03
%y	The year within a century [00-99]. When a century is not otherwise specified, values in the range 69-99 refer to years in the twentieth century (1969 to 1999, inclusive). Values in the range 00-68 refer to years in the twenty-first century (2000 to 2068 inclusive). Leading zeros are permitted but not required.	02	02
%Y	Year including the century (for example 1993) [0001-9999].	2002	2002

Date String Formats		
Input Format	Examples	Format String
DDMMYYYY	31012002	%d%m%Y
YYYYMMDD	20021130	%Y%m%d
YYYY	2002	%Y
MMDD	1130	%m%d
YYYY-MM	2002-06 2002/06 2002.06	%Y-%m %Y/%m %Y.%m
YYYY-MM-DD	2002-06-30 2002/06/30 2002.06.30	%Y-%m-%d %Y/%m/%d %Y.%m.%d
DD.MM.YY	30.06.02	%d.%m.%y
DD.MM.YYYY	30.06.2002	%d.%m.%Y
DD-Mmm-YYYY	30-Jun-2002 30.Jun.2002 30 Jun 2002	%d-%b-%Y %d.%b.%Y %d %b %Y
DD-Mmm-YY	30-Jun-02 30.Jun.02 30 Jun 02	%d-%b-%y %d.%b.%y %d %b %y
DD Mmm Space between day and month	30 Jun	%d %b
DD/MM/YY	30/06/02	%d/%m/%y
DD/MM/YYYY	30/06/2002	%d/%m/%Y

Appendix A: Data Definitions

Date String Formats		
Input Format	Examples	Format String
DD Mmmmm YYYY Includes full name of month, space between day and month, month and year	30 April 2002	%d %B %Y

Time String Formats		
Input Format	Examples	Format String
HHMM	2344	%H%M
HH:MM	23:44	%H:%M
HHMM PM Space before meridian indicator	1144 PM 1144 AM 1144 am	%H%M %p
HHMMPM	1144PM 1144AM 1144am	%H%M%p
HH:MM PM Space before meridian indicator	11:44 PM 11:44 AM 11:44 am	%H:%M %p
HH:MMPM	11:44PM 11:44AM 11:44am	%H:%M%p
HHMMSS	234405	%H%M%S
HH:MM:SS	03:44:05 22:10:36	%H:%M:%S
HHMMSS PM Space before meridian indicator	114405 PM 114405 pm 114405 am	%H%M%S %p
HHMMSSPM	114405PM 114405pm 114405am	%H%M%S%p
HH:MM:SS PM Space before meridian indicator	03:44:05 PM 03:44:05 pm 03:44:05 am	%H:%M:%S %p

Appendix A: Data Definitions

Time String Formats		
Input Format	Examples	Format String
HH:MM:SSPM	03:44:05PM 03:44:05pm 03:44:05am	%H:%M:%S%p

Time Duration String Formats		
Input Format	Examples	Format String
HH:MM.D Duration of call in hours, minutes, and decimal minutes (0.1 = 6 seconds)	13:06.1	%H:%M.%G
MMM:SS	135:05	%Q:%S
HHHH:MM:SS	1233:05:17	%K:%M:%S

Appendix B: How Composer Assembles a Document

How Composer Assembles a Document

Composer assembles an HTML document using the following logic:

1. Identify whether there is a conditional template. If yes, determine which statement template to use for the specific customer. If no, use the default template.
2. Assemble the HTML for the selected template.

Whenever an edocs tag `[E]` or `<edocsSection=NAME>` tag is encountered, the composing engine refers to the ALF for additional instructions.

A set of tags, such as `[E]NameGroup,G[/E]`, indicates to the composing engine that a group template needs to be inserted in statement being generated. The composing engine uses the logic and formatting information in the group section of the ALF.

When a table tag, `[E]Table,R[/E]`, is found, the ALF is referred to for formatting instructions to compose the table.

When a field tag, `[E]Field,F[/E]`, is found, the DDF is referred to via the ALF to extract and present the correct data.

When a chart tag, `[E]ChartTable,U[/E]`, is found, the ALF is referred to by the third party charting software for bitmap composition instructions.

When the link tag, `[E]NotificationLink,L[/E]`, is found, the system looks at the application configurations in the Command Center to build a custom Link to a login page.

3. The group condition calls the HTML template specified to present the group. During development, the Composer assembles that HTML in the places marked by `[E]` tags.
4. If the file includes nested groups, then the group section of the ALF is called again. Another condition calls the specified HTML template. The Composer assembles that HTML Template body tags for the inner group(s) in the places marked by `[E]` tags.
5. Assembly of the HTML document template is complete and the bill is presented to the customer.

Index

A

- Adding
 - content to a pattern of table rows, 298
- AFP font settings, 62
- ALF. *See Application Logic File*
- ALF switch conditions
 - changing, 260
 - creating, 256, 259
 - deleting, 260
 - editing, 260
- ALF Toolbar, 232
- Alignment (left, right, center)
 - adding as conditional table content, 292
- Application Logic File
 - before creating, 223
 - contents, 225
 - creating, 234
 - creating for a sub-account view, 236
 - definition, 19

- saving, 280
- switch, 256

- Application Tree, 227
- Application, sample, 21
- Arithmetic element, 336
 - creating, 337
 - deleting, 340
 - editing, 340

B

- Background color
 - adding as conditional table content, 292
- Bolding
 - adding as conditional table content, 292

C

- Cell formatting
 - adding as conditional table content, 292
- Changing
 - ALF used in a switch condition, 260
 - chart properties, 364
 - table content, 296

- Chart
 - attributes, 352
 - axis, 356
 - background, 355
 - bar properties, 363
 - changing properties, 364
 - creating, 349
 - deleting, 366
 - extracting tables, 128
 - favorite colors, 361
 - general properties, 354
 - information about in the ALF, 352
 - legend, 360
 - line properties, 364
 - pie properties, 362
 - plot area, 356
 - properties, 353
 - type, 354
- Charting, 349
- Checking elements and dynamic fields with Simulator, 341
- Color
 - background, adding as conditional table content, 292
 - text, adding as conditional table content, 292
- Columns
 - formatting, 285, 286
- Combine element, 333
 - creating, 334
 - deleting, 335
 - viewing, 335
- Composer, 223
 - how it assembles a document, 377
 - interface, 226
 - menus, 230
 - toolbars, 231
- Composing with Simulator, 341
- Conditional statement
 - multiple, 246
 - syntax, 245
- Conditional templates
 - adding, 261
 - deleting, 266
- Conditions
 - creating, 244
 - creating on a table (line item), 287
 - deleting, 273
 - modifying, 273
 - on a section, creating, 266
 - on a templates, 261
 - on group fields, 247
 - viewing a list of saved, 271
 - viewing properties of, 272
- Content

- adding to a pattern of table rows, 298
- creating, 274
- deleting, 278
- editing, 278
- table, 291
- viewing, 276

Creating

- ALF switch condition, 256
- Application Logic File, 234, 236
- arithmetic element, 337
- combine element, 334
- conditions, 244
- conditions on a table (line item), 287
- content, 274
- content (table), 274
- general filter element, 319
- link to a sort element, 306
- logic elements, 245
- page element, 313
- page set, 315
- reusable table content, 294
- sort element, 304
- top/bottom filter element, 325

Customer Self Service, 9

D

Data

- definitions, 367
- filtering, 317
- paging, 312
- sorting, 301

Data Definition File, 18, 223

- for DJDE formats, 47
- for line printer formats, 69

Data extraction, 17, 25

- area, 79
- creating rules, 87
- data pattern, 81

Data Type Report, 341

Data-driven conditions on a section, 266

Data-driven conditions on a table (line item), 287

DDF, 223. *See Data Definition Files*

Default template, 238

- changing, 265
- notification mapping, 239

Definition toolbar, 37

Definition Tree, 227

DefTool, 20

- icons, 37
- menu, 29
- simulations, 33
- toolbars, 36

Deleting
ALF switch conditions, 260
arithmetic element, 340
chart, 366
combine element, 335
conditional template, 266
conditions, 273
content, 278
filter element, 333
group, 284
logic elements, 255
page element, 317
sort element, 312
table conditions, 297
unused reusable table content, 296

Development Tools,
eaSuite, 12

Document
define style, 90
recipient, 95
start, 88

Document style, 37

Dynamic field, 39

Dynamic pattern matching, 205
link, 212
set markers, 206
simulating values, 211

E

eaAssist, 11

eaDirect, 11
application files, 21
application process, 18
development process, 18
introduction, 15

eaMarket, 11

eaPay, 12

eaPost, 12

eaSuite, 9

EBNF Description, 367

Editing
ALF switch conditions, 260
arithmetic element, 340
changing default template file, 265
changing template for conditional alternate, 265
conditions, 273
content, 278
filter element, 332
group template, 284
logic elements, 255
page element, 316
sort element, 312
table conditions, 297
template conditions, 265

Electronic statement presentment, 15

Element

- arithmetic, 336
- combine, 333
- creating link to sort, 306
- filter, 317
- paging, 312
- sort, 304

Element IDs, 220

Extraction simulator, 179

F

Field

- anchor, 119
- create anchor, 122
- define, 104, 107, 111, 112, 113
- delete anchor, 127
- edit anchor, 127
- extraction simulation, 182
- properties, 82

Fields

- mapping to the template, 239

Files, saving, 280

Filter element, 317

- creating general, 319
- creating top/bottom, 325
- deleting, 333
- editing, 332
- general content-based, 317

- linking to a statement, 328
- syntax, 317
- top/bottom, 317

Filtering

- across table or group occurrences, 333
- data, 317
- groups, 318
- tables, 318

Font formatting

- adding as conditional table content, 292

Formatting

- adding to a pattern of table rows, 298
- columns, 285, 286
- tables, 285
- Toolbar, 233

G

General filter element

- creating, 319

Group

- components, 150
- define, 147, 153
- define condition, 283
- define end marker, 160
- define properties, 165
- define start marker, 155
- delete, 285
- new, 41
- table columns, 151

- template, editing, 284
- Group fields, 171, 172, 178, 247
- Groups
 - combining, 333
 - deleting, 284
 - filtering, 318
 - mapping, 282
 - mapping to the template, 281
 - sorting, 301, 303

H

- Help
 - documentation, 13
 - technical support, 13
- Highlighting a pattern of table rows, 298
- Highlighting data, 28
- History list
 - sub-account data, 169
- HTML
 - tables, 243, 244
 - tables, inserting, 241
 - text, 244
- HTML code
 - inserting, 241
- HTML files, adding as content, 274
- HTML Mode, 229
- HTML templates, 223
 - saving, 280
- Hyperlinks, 242

- adding as conditional table content, 292
- inserting, 241
- on an image, adding as conditional table content, 292
- on text, adding as conditional table content, 292

I

- Images, 242
 - adding as conditional table content, 292
 - adding as content, 274
 - inserting, 241
- Indexing sub-account data, 169
- Inserting
 - HTML code, 241
 - HTML tables, 241
 - hyperlinks, 241
 - images, 241
 - navigational paging links, 315
 - null, 292
- Italics
 - adding as conditional table content, 292

L

- Line item
 - creating conditions on in a table, 287
- Links

- linking a filter element to a statement, 328
- mapping notification, 279
- navigational paging, inserting, 315

Logic elements

- creating, 245
- deleting, 255
- editing, 255
- syntax, 246
- viewing, 253

M

Mapping

- fields to the template, 239
- groups, 282
- notification link, 279
- tables and groups to the template, 281
- tables as charts, 349
- tables as text, 281

Marker

- new, 40
- properties, 84

Menus

- Composer, 230

Metacode

- settings, 76

N

National Wireless sample application, 23, 170

National Wireless Sample Application, 21

Navigational paging

- links, inserting, 315

New field, 39

Non-data driven

- conditions on a table (pattern of rows), 298

Notification link

- mapping, 279

Null

- inserting, 292

O

Object Toolbar, 233

Occurrences of table or group, combining, 333

P

Page

- delete, 317

Page element

- creating, 313
- editing, 316
- removing, 317

Page set

- creating, 315

Page style, 38, 100

- define, 101

Paging, 312

- across table or group occurrences, 333
- links, inserting, 315

- Pattern of table rows,
 - creating conditions on, 298
- Presentation design, 17
- Presenting sub-account data, 169
- Primary key
 - define, 96

R

- Regular expression, 188
 - currency values, 194
 - longhand dates, 195
 - partially static data, 199
 - shorthand dates, 193
 - static data, 198
 - varying data values, 197
- Removing a page element, 317
- Replacing a section with text or an image, 292
- Reusable table content
 - creating, 294
 - deleting, 296
 - viewing list of, 296

S

- Sample application
 - National Wireless, 21
 - sub-account view, 23, 170
- Saving the ALF, HTML template files, 280
- SDK, 12

- Section
 - replacing with text or an image, 292
- Selective extraction simulation, 185
- Simulating
 - dynamic field values, 35
- Simulator, 341
 - navigation, 180
 - Toolbar, 234
- Sort element, 301
 - creating, 304
 - deleting, 312
 - editing, 312
- Sorting
 - across table or group occurrences, 333
 - creating a link to a sort element, 306
 - creating a sort element, 304
 - groups, 303
 - tables, 301
 - tables and groups, 301
- Specifying a default template, 238
- Statement
 - mastering, 15
 - paging, 312
- Sub-account
 - conditions on, 247
 - creating a DDF, 169

T

- creating an ALF, 236
- inclusion in history list, 169
- Subkey, 171, 172, 178
- Subtotalling, 336
- Summing rows, 336
- Suppressing display, 292
- Syntax
 - filter elements, 317
 - logic elements, 246
- Tab
 - Application, 227
 - Definition, 227
- Table
 - anchor, 85
 - chart extraction, 128
 - column, 86
 - content, changing, 296
 - content, creating, 274, 291
 - define anchor, 146
 - define column, 143
 - define end marker, 137
 - define external anchor, 146
 - define format, 141
 - define internal anchor, 147
 - define start marker, 132
 - definitions, 129
 - external anchor, 86
 - extracting, 128
 - extraction simulation, 184
 - internal anchor, 85
 - new, 40
 - properties, 85
 - rules, 130
 - sort, 304
- Table conditions
 - deleting, 297
 - editing, 297
 - viewing properties of, 296
- Tables
 - combining, 333
 - creating conditions on (line item), 287
 - filtering, 318
 - formatting, 285
 - mapping as text, 281
 - mapping to the template, 281
 - sorting, 301
- Template, 20, 223
 - changing default, 265
 - conditions, 261
 - conditions, editing, 265
 - conditions, viewing, 265
 - deleting conditional, 266

- for conditional
 - alternate, changing, 265
- group, modifying, 284
- mapping a group, 282
- mapping a table as text, 281
- mapping tables and groups, 281
- matching fields to, 239
- Toolbar, 232
- Testing composition with Simulator, 341
- Text
 - adding as conditional table content, 292
 - adding as content, 274
 - color, adding as conditional table content, 292
 - replacing a section with, 292
- Toolbars
 - Composer, 231
- Top/bottom filter element, 317
 - creating, 325

V

- Validating with Simulator, 341
- View, 16
 - sub-account, 169
- Viewing
 - ALF switch conditions, 259
 - combine element, 335
 - conditions, list of saved, 271
 - conditions, properties of, 272
 - content, 276
 - list of reusable table content, 296
 - logic elements, 253
 - properties of table conditions, 296
 - template conditions, 265

W

- WYSIWYG Mode, 229

Z

- Zoom in/out, 28