COMMERCE

Sitemap Generator Developer's Guide

Version 11.2
October 2015

Sitemap Generator Developer's Guide

Product version: 11.2

Release date: 10-22-15

Table of Contents

Preface vii
About this guide vii
Who should use this guide vii
Conventions used in this guide viii
Contacting Oracle Support viii
1. Introduction 1
About sitemaps 1
About the Oracle Commerce Guided Search Sitemap Generator 1
Reference implementation 2
2. Configuring the Sitemap Generator 3
Configuration files 3
The main configuration file 4
MDEX Engine configuration 5
The navigation page spec list 6
The query field list 9
The template configuration file 9
About tag replacement 11
Replacement tags for INDEX_LINK 11
Replacement tags for DETAIL_LINK 12
Replacement tags for NAVIGATION_LINK 13
Replacement tags for SEARCH_LINK 13
Replacement tags for STATIC_PAGE_LINK 14
The search terms configuration file 14
The static pages configuration file 15
The URL formatting configuration file 15
Copying URL configuration settings from an Assembler application to the Sitemap Generator 16
3. Running the Sitemap Generator 17
Running the Sitemap Generator from the command line 17
Standard output 18
File outputs 18
Index file 18
Detail files 19
Navigation files 19
Search term files 20
Static pages files 21
4. Generating Sitemaps for Multiple Sites 23
Configuring the Sitemap Generator for specific sites 23
Running the Sitemap Generator 23
5. Troubleshooting and Performance 27
Common errors 27
Performance information 29
Runtime performance 29
Production vs. dedicated MDEX Engine 29
Combinatoric navigation pages 30
6. Implementing the Sitemap Generator in Production 31
About validating sitemaps 31
About moving sitemap files to production 31
Notifying search engines of a sitemap location 32
Notifying search engines of an updated sitemap 32

List of Examples

2.1. Example 8
2.2. Example 9

Preface

Oracle Commerce Guided Search is the most effective way for your customers to dynamically explore your storefront and find relevant and desired items quickly. An industry-leading faceted search and Guided Navigation solution, Guided Search enables businesses to influence customers in each step of their search experience. At the core of Guided Search is the MDEX Engine™, a hybrid search-analytical database specifically designed for high-performance exploration and discovery. The Oracle Commerce Content Acquisition System provides a set of extensible mechanisms to bring both structured data and unstructured content into the MDEX Engine from a variety of source systems. The Oracle Commerce Assembler dynamically assembles content from any resource and seamlessly combines it into results that can be rendered for display.

Oracle Commerce Experience Manager enables non-technical users to create, manage, and deliver targeted, relevant content to customers. With Experience Manager, you can combine unlimited variations of virtual product and customer data into personalized assortments of relevant products, promotions, and other content and display it to buyers in response to any search or facet refinement. Out-of-the-box templates and experience cartridges are provided for the most common use cases; technical teams can also use a software developer's kit to create custom cartridges.

About this guide

This guide describes the Oracle Commerce Guided Search Sitemap Generator and provides instructions for using it to generate sitemaps for an Oracle Commerce application.

It assumes that you are familiar with Oracle Commerce terminology and basic concepts. This guide covers only the features of the Sitemap Generator, and is not a replacement for the available material documenting other Oracle Commerce products and features.

Who should use this guide

This guide is intended for developers who wish to create sitemaps for their cross-channel and Web applications.

This document assumes that the reader has a working knowledge of the following software and concepts:

- Basic concepts such as dimensions, dimension values, refinements, ancestors, records, aggregate records, and so on
- Configuring dimensions using Developer Studio

- The Assembler API, especially:
  - The URL formatter classes contained in the `com.endeca.soleng.urlformatter` packages
  - The Guided Navigation classes contained in the `com.endeca.infront.navigation` package

Conventions used in this guide

This guide uses the following typographical conventions:

Code examples, inline references to code elements, file names, and user input are set in `monospace` font. In the case of long lines of code, or when inline monospace text occurs at the end of a line, the following symbol is used to show that the content continues on to the next line: ¬

When copying and pasting such examples, ensure that any occurrences of the symbol and the corresponding line break are deleted and any remaining space is closed up.

Contacting Oracle Support

Oracle Support provides registered users with answers to implementation questions, product and solution help, and important news and updates about Guided Search software.

You can contact Oracle Support through the My Oracle Support site at https://support.oracle.com.

1 Introduction

This section provides an introduction to the Sitemap Generator and its capabilities.

Related links

- About sitemaps (page 1)
- About the Oracle Commerce Guided Search Sitemap Generator (page 1)

About sitemaps

A sitemap provides search engine spiders with information about all of the content available on a site, and allows them to access every specified page without crawling the site links.

Ensuring that site content is included in Web search indices, such as Google and Yahoo, is important in raising a site's search ranking. A sitemap serves as a starting point that ensures that important locations in your application are possible for crawlers to locate, and are properly ranked.

Additionally, pages within commerce applications aren't typically linked by outside sites. Listing a set of representative product pages in your sitemap helps present an accurate portrayal of your application, which leads to more hits when users enter a relevant search query.

The Sitemap Generator creates files that use the XML Sitemap protocol, which Google, Yahoo!, Microsoft, and Ask.com have agreed to support. You can find out more about sitemaps and the sitemap protocol at http://www.sitemaps.org.

About the Oracle Commerce Guided Search Sitemap Generator

The Sitemap Generator is a standalone Java application that builds a set of index pages containing links to all record detail pages in an application, as well as static pages, selected navigation pages, and selected search results pages. It is included with the Oracle Commerce Tools and Frameworks installation package.

The Sitemap Generator retrieves the necessary record and dimension data by issuing a single bulk-export query against an MDEX engine. It then creates index pages using customizable templates to support different

page formats (such as the Sitemap protocol XML format). In situations where one page can be referenced from multiple URLs, such as pages with dynamic URL parameters that can be transposed, the Sitemap Generator uses the same `UrlFormatter` configuration as the Assembler API to create a single canonical URL. This avoids the possibility of presenting duplicate content to search engine indices.

To ensure that search engines properly index important content in your application, you should provide links to search results pages for common searches. You can supply a list of common search terms to the Sitemap Generator in a search term configuration file.

The Sitemap Generator can also create links to static pages within an application. This allows Web crawlers to access static page URLs as well as URLs that link to dynamic Assembler-driven content.

Reference implementation

The sample configuration files provided with the Sitemap Generator are based on the Discover Electronics reference application. Oracle recommends that you have an MDEX Engine configured and running the associated data set before you first configure and run the Sitemap Generator. Please see the *Oracle Commerce Guided Search Getting Started Guide* for documentation on setting up an MDEX Engine with this data set.

2 Configuring the Sitemap Generator

This section provides a general overview of the Sitemap Generator files and describes how to customize the tool to meet your requirements. Before modifying any of the configuration files, Oracle recommends that you first make a backup of the original.

Related links

- Configuration files (page 3)
- The main configuration file (page 4)
- The template configuration file (page 9)
- The search terms configuration file (page 14)
- The static pages configuration file (page 15)
- The URL formatting configuration file (page 15)

Configuration files

This section provides an overview of the configuration files used by the Sitemap Generator.

The `sitemap_generator\conf` directory contains the following files:

File	Description
`conf.xml`	A sample main configuration file, which specifies settings to use when running the Sitemap Generator, such as which template configuration file to use, and where to generate output files.
`xml_tempate.xml`	A sample template configuration file that can be used to create Sitemap protocol XML files.
`html_template.xml`	A sample template configuration file that can be used to create a generic HTML sitemap file.
`searchterms.xml`	A sample search terms configuration file. The Sitemap Generator creates a link for each of the search term parameters in the list.

File	Description
staticpages.txt	A sample static pages configuration file. The Sitemap Generator creates a link for each of the pages in the list.
urlconfig.xml	A sample URL formatting configuration file, which specifies the settings for all URL formatting done by the Sitemap Generator.

The main configuration file

The main configuration file for the Sitemap Generator is located in ToolsAndFrameworks\<version>\sitemap_generator\conf\conf.xml.

This configuration file contains the following elements:

Element	Description
TEMPLATE_FILE	Specifies the template configuration file. The template configuration file customizes the formatting of each of the links generated. The location of this file can either be set using an absolute path, or a relative path from the main configuration file.
INDEX_FILE	Specifies the name of the index file that the Sitemap Generator creates. This setting is also used to determine the relative path and extension for other output files. The location of this file can either be set using an absolute path, or a relative path from the main configuration file.
MDEX_ENGINES	Specifies the MDEX Engine or Engines to query when generating a sitemap. Oracle recommends that you use a dedicated staging index for generating sitemaps to minimize impact on production query performance.
QUERY_FIELD_LIST	(Optional) Specifies dimension and property names available for tag replacement in the template configuration files. This list should include all dimensions and/or properties used in the NAVIGATION_PAGE_SPEC_LIST and in the configuration file for the Assembler seoUrlFormatter. If no fields are specified, only properties and dimensions configured in Developer Studio for display with the results list are available for tag replacement. If values are specified, they override the Developer Studio configuration.
SEARCH_TERMS_FILE	(Optional) Specifies the search terms configuration file to use for building search term pages and links. The location of this file can be set by using either an absolute or relative path.

Element	Description
STATIC_PAGES_FILE	(Optional) Specifies the static pages configuration file to use for building static page links. The location of this file can be set by using either an absolute or relative path.
URL_FORMAT_FILE	(Optional) Specifies the URL formatting configuration file. The URL formatting configuration file customizes the text inserted each time the FORMATTED_URL tag is encountered in the template configuration file. The location of this file can be set by either using an absolute or relative path.
URLFORMATTER_COMPONENT	Specifies the top level component (or bean) in the URL formatting configuration file, which is used each time a FORMATTED_URL tag is encountered in the template configuration file.
URL_ENCODING	(Optional) Specifies the encoding used for formatting URLs. Defaults to UTF-8. You can override this setting for replacement tags in the URL templates by using a pipe | character followed by either XmlEscape or HtmlEscape, depending on the output format you wish to use for sitemap files.
URL_INCLUDE_AGGR_REC_PARAMS	(Optional) Specifies whether to include Nu, Au, and An aggregate record query parameters in URLs. Defaults to false.
LINKS_PER_FILE	Specifies the maximum number of links to include in a sitemap file before rolling to a new output file. Applies to all output files, including detail files, navigation files, search term files, and static pages files.
MAX_RECS	Specifies the maximum number of records to return for the bulk export query. This is useful for debugging and testing against large indices. ALL_RECS indicates that all records should be returned.
NAVIGATION_PAGE_SPEC_LIST	Specifies combinations of dimensions with which to create navigation page links.

MDEX Engine configuration

Each <MDEX_ENGINES> element in the main configuration file can have one or more <ENGINE> elements.

The <ENGINE> element contains the following children:

Element	Description
HOST	Specifies the host of an MDEX Engine to be queried by the Sitemap Generator.
PORT	Specifies the port of an MDEX Engine to be queried by the Sitemap Generator.

Element	Description
ROOT_QUERY	Specifies the query string for the Sitemap Generator to use when submitting the bulk export query against the specified index. Typically, this is left as a root query (N=0) to retrieve all records from an index. However, this element can also be modified to specify a subset of records to retrieve from the index. All queries should be entered in unencoded format (e.g. N=8021).
ROLLUP_KEY	(Optional) If your Assembler application uses aggregate record queries, this value specifies the record property on which to aggregate record results in the MDEX Engine. Results returned from the Engine are grouped according to this key. Alternately, you can omit this parameter and specify the rollup key within the ROOT_QUERY value.

The following example shows the configuration of an MDEX Engine with a rollup key:

```
<MDEX_ENGINES>
  <ENGINE>
      <HOST>localhost</HOST>
      <PORT>15000</PORT>
      <ROOT_QUERY><![CDATA[N=0]]></ROOT_QUERY>
      <ROLLUP_KEY>product.code</ROLLUP_KEY>
  </ENGINE>
</MDEX_ENGINES>
```

The navigation page spec list

Navigation links are created by examining the dimension values tagged to each record processed.

The NAVIGATION_PAGE_SPEC_LIST element is used to configure which unique combinations of dimensions are included in the query string portion of an Assembler driven URL, for example:

```
<NAVIGATION_PAGE_SPEC_LIST>
  <NAVIGATION_PAGE_SPEC>
    <DIMENSION_NAME FULL_HIERARCHY=”True”>product.category</DIMENSION_NAME>
  </NAVIGATION_PAGE_SPEC>
  <NAVIGATION_PAGE_SPEC>
    <DIMENSION_NAME FULL_HIERARCHY=”False”>product.brand.name</DIMENSION_NAME>
  </NAVIGATION_PAGE_SPEC>
  <NAVIGATION_PAGE_SPEC>
    <DIMENSION_NAME FULL_HIERARCHY=”True”>product.category</DIMENSION_NAME>
    <DIMENSION_NAME FULL_HIERARCHY=”False”>product.brand.name</DIMENSION_NAME>
  </NAVIGATION_PAGE_SPEC>
</NAVIGATION_PAGE_SPEC_LIST>
```

Each <DIMENSION_NAME> element includes a required FULL_HIERARCHY attribute that designates whether intermediate navigation page links should be generated for hierarchical dimensions. For example, consider the following dimension with hierarchical values:

```
product.category
```

```
camera
 digital_camera
 film_camera
```

If the FULL_HIERARCHY attribute is set to "False", then URLs are only generated for the "digital_camera" and "film_camera" dimension values. However, if FULL_HIERARCHY is set to "True", then a link for the intermediate value "camera" is also generated.

Example 2.1. Example

Consider three records that are tagged with the following values from the listed dimensions:

Record	product.color	product.category	product.brand.name
Rec1	black	digital_camera	Canon
Rec2	silver	film_camera	Kodak
Rec3	red	digital_camera	Kodak

The first NAVIGATION_PAGE_SPEC element in the example presented earlier in this topic only includes the product.category dimension:

```
digital_camera
film_camera
```

Additionally, since the value of the FULL_HIERARCHY attribute for this dimension is "True", any generated URL must include all ancestors for product.category:

```
camera
digital_camera
film_camera
```

The second NAVIGATION_PAGE_SPEC element is similar, but includes dimension values for the product.brand.name dimension, which is not hierarchical:

```
Canon
Kodak
```

The third NAVIGATION_PAGE_SPEC element is processed by iterating over all three records and creating a hash of all unique combinations of values from these dimensions results in the following navigation links:

```
digital_camera + Canon
film_camera + Kodak
digital_camera + Kodak
```

Additionally, it includes all possible ancestors for the product.category dimension values:

```
camera + Canon
camera + Kodak
```

The complete list of generated URLs is the set of all possible links resulting from the NAVIGATION_PAGE_SPEC elements within NAVIGATION_PAGE_SPEC_LIST:

```
camera
digital_camera
film_camera
Canon
Kodak
digital_camera + Canon
film_camera + Kodak
digital_camera + Kodak
camera + Canon
camera + Kodak
```

It is important to realize that the creation of navigation page links behaves in a very combinatoric fashion. Oracle strongly recommends specifying only **one or two** dimensions for any given <NAVIGATION_PAGE_SPEC>. Otherwise, it is easy to generate millions of navigation links. Creating a sitemap with such a large number of links can lead search engines to classify your site as a link farm, which in extreme cases can result in your site being removed from search engine indices.

The query field list

In order to include dimension and property names in tag replacements, you must include them in the QUERY_FIELD_LIST in the main configuration file.

The values included in the nested QUERY_FIELD elements are available for tag replacement in the template configuration files when generating sitemap URLs. To ensure that these URLs are identical to those generated by the Assembler, you must include all of the property and dimension names from your endeca-seo-url-config configuration file, as well as all of the record properties or dimension values that appear in the main Sitemap Generator configuration file.

Any properties or dimensions that are not included in the QUERY_FIELD_LIST are not available for tag replacement.

For improved performance, Oracle recommends using the QUERY_FIELD_LIST configuration. If no QUERY_FIELD_LIST is specified, only those properties and dimensions that are enabled for display with the record list are available for tag replacement.

Note

The Sitemap Generator only accepts one QUERY_FIELD_LIST. If you create more than one, only the first QUERY_FIELD_LIST in the conf.xml file is read.

Example 2.2. Example

The configuration for the seoUrlFormatter object in the Discover Electronics reference application uses the product.name record property and product.category and product.brand.name dimension value information to create optimized URLs. To create identical links in the Sitemap Generator, these fields must be available for tag replacement:

```
<QUERY_FIELD_LIST>
   <QUERY_FIELD>product.name</QUERY_FIELD>
   <QUERY_FIELD>product.category</QUERY_FIELD>
   <QUERY_FIELD>product.brand.name</QUERY_FIELD>
</QUERY_FIELD_LIST>
```

Related links

- About tag replacement (page 11)

The template configuration file

The template configuration file is an XML file that defines the format of pages and links created by the Sitemap Generator.

To run the Sitemap Generator, you must specify a template file. You can do this by setting the <TEMPLATE_FILE> tag in the main configuration file. Two sample templates are provided in the sitemap_generator\conf directory:

File name	Description
`xml_template.xml`	This file is used to create Sitemap protocol XML pages. See http://www.sitemaps.org/ for more details. **Note** Google, Yahoo, Microsoft, and Ask.com have agreed to support this Sitemap protocol.
`html_template.xml`	This file is used to create simple HTML sitemap pages. These pages include the appropriate "robots" meta tags that allow most spiders to successfully crawl these pages.

The table below lists the standard elements of a template configuration file:

XML Element	Usage
`INDEX_LINK`	Specifies the link format in the index file.
`DETAIL_LINK`	Specifies the link format used in record detail pages.
`NAVIGATION_LINK`	Specifies the link format used in navigation pages.
`SEARCH_TERM_LINK`	Specifies the link format used in search term pages.
`STATIC_PAGE_LINK`	Specifies the link format used in static pages.
`INDEX_HEADER`	Specifies the header used in the index file.
`INDEX_FOOTER`	Specifies the footer used in the index file.
`PAGE_HEADER`	Specifies the header used in the record detail, navigation, search terms, and static pages files.
`PAGE_FOOTER`	Specifies the footer that is used in the record detail, navigation, search terms, and static pages files.

The Sitemap Generator uses the contents of the above elements to create pages. For example, to create a record detail page, the Sitemap Generator uses the `PAGE_HEADER`, `DETAIL_LINK`, and `PAGE_FOOTER` XML elements.

Note

If you have the `CanonicalLinkBuilder` enabled in your Assembler Application, your `DETAIL_LINK`, `NAVIGATION_LINK`, and `SEARCH_TERM_LINK` templates should include only those URL parameters specified in the `<bean class="com.endeca.infront.navigation.url.event.CanonicalLinkBuilder">` bean in the Assembler context file.

About tag replacement

When outputting the content of each element, the Sitemap Generator replaces any **-enclosed text in the template configuration file with dynamic values.

For example, if the DETAIL_LINK template is configured as:

```
<DETAIL_LINK><![CDATA[
<url>
   <loc>http://localhost:8006/discover/detail?ID=**RECID**</loc>
   <lastmod>**TIMESTAMP**</lastmod>
</url>
]]></DETAIL_LINK>
```

And if one of the records resulting from the query (set in the main configuration file) has an ID of 53, then one of the links in the detail page becomes:

```
<url>
   <loc>http://localhost:8006/discover/detail?ID=53</loc>
   <lastmod>2007-10-17</lastmod>
</url>
```

About URL encoding

To ensure that the replacement text in a tag is escaped when it is substituted into an XML or HTML template, you can override the default URL encoding settings in the `conf.xml` file by specifying encoding on a per-tag basis. To do this, append one of the following within a tag:

- `|XmlEscape` — escapes `>`, `<`, `&quot;`, `&`, and `&apos;`
- `|HtmlEscape` — escapes all known HTML 4.0 entities

The pipe character signals an override of the default encoding setting. Both settings use the `org.apache.commons.lang.StringEscapeUtils` class to encode the special characters in URLs.

For example, in the included `sitemap_generator\conf\xml_template.xml` file, the URL formatting for sitemap links from the index page is escaped as follows:

```
<INDEX_LINK><![CDATA[
<sitemap>
    <loc>**FILE_NAME|XmlEscape**</loc>
</sitemap>
]]></INDEX_LINK>
```

Replacement tags for INDEX_LINK

The INDEX_LINK section of the template configuration file has a set of valid replacement tags.

The following table contains all the replacement parameters that can be used in the index link:

Parameter	Description	Example
FILE_NAME	Replaced with each of the file names created by the Sitemap Generator.	http://localhost:8006/ discover/**FILE_NAME**

Replacement tags for DETAIL_LINK

The DETAIL_LINK section of the template configuration file has a set of valid replacement tags.

The following table contains all the replacement parameters that can be used in the detail link:

Parameter	Description	Example
FORMATTED_URL (Preferred)	Replaced with record-related information in search-engine optimized format, as specified in the configuration file for the Assembler URL formatter object (typically an instance of seoUrlFormatter). This setting applies to both aggregate and non-aggregate queries.	http://localhost:8006/ discover/ browse**FORMATTED_URL**
RECID	Replaced with the record ID of each Record.	http://localhost:8006/ discover/ browse?R=**RECID**
[Any Record Property]	Replaced with the value of the Property for each Record. **Note** The Sitemap Generator can use only properties enabled for display in the results list of an query, or those properties and dimensions included in the QUERY_FIELD_LIST configuration. Properties that are enabled for use only on record detail requests cannot be displayed.	http://localhost:8006/ discover/ browse?R=**product.brand**
ROLLUP_KEY	Replaced with rollup key of the aggregated record query.	http://localhost:8006/ discover/ browse/A=**RECID**& Au=**ROLLUP_KEY|UrlEncode**

Replacement tags for NAVIGATION_LINK

The NAVIGATION_LINK section of the template configuration file has a set of valid replacement tags.

The following table contains all the replacement parameters that can be used in the NAVIGATION_LINK:

Parameter	Description	Example
FORMATTED_URL (Preferred)	Replaced with the navigation information in search-engine optimized format, as specified in the URL formatting configuration file. This setting is applicable for both aggregate and non-aggregate queries.	http://localhost:8006/discover/ browse**FORMATTED_URL**
DIMVAL_IDS	Replaced with corresponding dimension value IDs of each navigation page.	http://localhost:8006/discover/ browse?N=**DIMVAL_IDS**
DIMVAL_NAMES	Replaced with dimension values of each navigation page. These are useful when tailoring sitemaps for HTML crawlers (using the HTML template, for example).	 **DIMVAL_NAMES**
ROLLUP_KEY	Replaced with rollup key of the aggregated record query.	http://localhost:8006/discover/ browse? N=**DIMVAL_IDS**&Nu=**ROLLUP_KEY **

Note

The exact format of the URLs in your application may vary based on your URL configuration settings in the Assembler.

Replacement tags for SEARCH_LINK

The SEARCH_LINK section of the template configuration file has a set of valid replacement tags.

The following table contains all the replacement parameters that can be used in the search link:

Parameter	Description	Example
FORMATTED_URL	Replaced with search parameters from the search terms configuration file, in Guided Search URL format.	http://localhost:8006/discover/ browse**FORMATTED_URL**

Replacement tags for STATIC_PAGE_LINK

The STATIC_PAGE_LINK section of the template configuration file has a set of valid replacement tags.

The following table contains all the replacement parameters that can be used in the static page link:

Parameter	Description	Example
STATIC_PAGE	The name of the static page. Obtained from each word in the static pages configuration file.	http://localhost:8006/discover/**STATIC_PAGE**

The search terms configuration file

In order for the Sitemap Generator to create links to pages resulting from common search terms and navigation queries, you must supply these terms and queries in the search terms configuration file.

The location of the search terms configuration file is specified in the main configuration file. At the top of the file, specify a default host, port, and query using <MDEXHOST>, <MDEXPORT>, and <DEFAULTQUERY> tags. Below the default tags, each <URL> tag describes a unique search result link. Each tag contains a set of Guided Search URL parameters in <PARAM> tags. An optional special parameter named <QUERY> overrides the default query for that search result link.

Here is an example of a search terms configuration file:

```
<xml version=”1.0” encoding=”UTF-8”>
<!--searchterms.xml
Configuration of terms from which URLs can be generated -->

<URLS>
  <MDEXHOST>localhost</MDEXHOST>
  <MDEXPORT>15000</MDEXPORT>
  <DEFAULTQUERY><![CDATA[N=0]]></DEFAULTQUERY>
  <URL>
    <PARAM NAME="QUERY"><![CDATA[N=18832]]></PARAM>
    <PARAM NAME="Ntt">DSLR</PARAM>
  </URL>
  <URL>
    <PARAM NAME="Ntt">Digital SLR</PARAM>
  </URL>
  <URL>
    <PARAM NAME="Ntt">PowerShot</PARAM>
  </URL>
</URLS>
```

The Sitemap Generator creates a link based on the parameters within each <URL> tag. The formatting of the base portion of the link (e.g. "localhost:8006/discover") is configured in the template configuration file. You can configure parameter formatting in the URL formatting configuration file.

For example, if the SEARCH_TERM_LINK were defined as localhost:8006/discover in the template configuration file, the Sitemap Generator might generate the following link from the first URL in the above example: http://localhost:8006/discover/browse?Ntt=DSLR.

Note

If you have the CanonicalLinkBuilder enabled in your Assembler Application, the <PARAM> elements for each URL should include only those parameters specified in the <bean class="com.endeca.infront.navigation.url.event.CanonicalLinkBuilder"> bean in the Assembler context file.

Note that the un-encoded N values should be specified for <QUERY> values (e.g. N=18832).

Related links

- The URL formatting configuration file (page 15)
- The template configuration file (page 9)

The static pages configuration file

In order for the Sitemap Generator to create links to existing static pages, you must specify custom static URLs in the static pages configuration file. Note that this is a text file, rather than an XML file.

The location of the static pages configuration file is specified in the main configuration file. The static pages configuration file is carriage-return delimited and plain text, where each line designates a separate page. You can specify either absolute paths or relative paths. For example:

```
browse
contact-us
about-us
mobile/browse
```

The Sitemap Generator creates a link for each URL using the formatting specified in the template configuration file.

For example, if the STATIC_PAGE_LINK were defined as localhost:8006/discover/ in the template configuration file, the Sitemap Generator would generate the following link from the first URL in the above example: localhost:8006/discover/browse.

Related links

- The template configuration file (page 9)

The URL formatting configuration file

The URL formatting configuration file controls the format of the URL parameters that are substituted for the **FORMATTED_URL** tag specified in the template configuration file.

The Sitemap Generator uses the settings specified in the URL formatting configuration file in conjunction with the Assembler API to produce search-engine optimized URLs. By using the same URL configuration for your Assembler application and the Sitemap Generator, you can ensure that your sitemaps generate with the links that are true to those in your application. For details, see Copying URL configuration settings from an Assembler application to the Sitemap Generator (page 16).

The URL formatting configuration file uses Spring Framework syntax. If you need further information about this format, please consult the documentation provided with the Spring Framework. The Sitemap Generator includes a sample configuration file (`sitemap_generator\conf\urlconfig.xml`), which is tailored for the Discover Electronics data set.

Copying URL configuration settings from an Assembler application to the Sitemap Generator

To ensure that the URLs generated by the Sitemap Generator are identical to those created in your application, you should copy the settings from your Assembler application URL configuration file to the `ToolsAndFrameworks\<version>\sitemap_generator\conf\urlconfig.xml` file.

To properly generate search engine optimized URLs, you should use the `SeoUrlFormatter` class in your Assembler application.

To copy URL configuration settings from an Assembler application to the Sitemap Generator:

1. Navigate to the URL formatter configuration file for your application.

   In the Discover Electronics reference application, this is `ToolsAndFrameworks\<version>\reference\discover-electronics\WEB-INF\endeca-seo-url-config.xml`.

2. Copy the file to your Sitemap Generator URL configuration directory and rename it to replace the Sitemap Generator URL configuration file.

   By default, this is `ToolsAndFrameworks\<version>\sitemap_generator\conf\urlconfig.xml`.

   Note

   The URL formatter configuration file includes a `<property>` element that enables the `NavStateCanonicalizer`. Regardless of the configuration in your Assembler application, Oracle recommends setting this property to `true` in the configuration file for the Sitemap Generator to generate canonical links.

3. Confirm that the `<URLFORMATTER_COMPONENT>` setting in the main configuration file for the Sitemap Generator corresponds to the `urlformatter` implementation specified in your application configuration.

   Typically, this is the `seoUrlFormatter`.

3 Running the Sitemap Generator

This section describes the process for running the Sitemap Generator and provides information about the various output types.

Related links

- Running the Sitemap Generator from the command line (page 17)
- Standard output (page 18)
- File outputs (page 18)

Running the Sitemap Generator from the command line

The Sitemap Generator can be run from the command line using `RunSitemapGen.bat` (on Windows) or `RunSitemapGen.sh` (on UNIX).

These scripts take the location (absolute or relative) of the main configuration file, as a single argument. For example, when running on Windows:

```
cd C:\Endeca\ToolsAndFrameworks\<version>\sitemap_generator\bin
RunSitemapGen.bat ..\conf\conf.xml
```

Note that the above example changes to the `ToolsAndFrameworks\<version>\sitemap_generator\bin` directory before running the script. While this is convenient for specifying the `conf.xml` file via a relative path, the `bin` directory is not required as a working directory.

These scripts rely on the `ENDECA_ROOT` environment variable in order to locate the appropriate Java libraries.

Related links

- The main configuration file (page 4)
- About validating sitemaps (page 31)

Standard output

This section provides an example of successful standard output.

A successful run of the Sitemap Generator produces standard output similar to the following:

```
C:\Endeca\ToolsAndFrameworks\3.1.2\sitemap_generator\bin> RunSitemapGen.bat ..\conf
\conf.xml
Jan 16, 2013 9:38:56 AM com.endeca.soleng.sitemap.SitemapMain LoadConfiguration
INFO: Loading config file...
Jan 16, 2013 9:38:56 AM com.endeca.soleng.sitemap.SitemapMain LoadConfiguration
INFO: Clearing old sitemap files...
Jan 16, 2013 9:38:56 AM com.endeca.soleng.sitemap.SitemapMain LoadConfiguration
INFO: Loading templates...
Jan 16, 2013 9:38:56 AM com.endeca.soleng.sitemap.SitemapMain LoadUrlFormatSetti
ngs
INFO: Load URL Formatting Settings...
Jan 16, 2013 9:38:57 AM org.springframework.beans.factory.xml.XmlBeanDefinitionR
eader loadBeanDefinitions
INFO: Loading XML bean definitions from file [C:\Endeca\SitemapGenerator\3.1.2\b
in\..\conf\urlconfig.xml]
Jan 16, 2013 9:38:57 AM com.endeca.soleng.sitemap.SitemapMain runQueries
INFO: Querying engine...
Jan 16, 2013 9:38:59 AM com.endeca.soleng.sitemap.SitemapMain execute
INFO: Writing detail links...
Jan 16, 2013 9:39:00 AM com.endeca.soleng.sitemap.SitemapMain execute
INFO: Writing navigation links...
Jan 16, 2013 9:39:00 AM com.endeca.soleng.sitemap.SitemapMain execute
INFO: Writing search term links...
Jan 16, 2013 9:39:00 AM com.endeca.soleng.sitemap.SitemapMain execute
INFO: Writing static page links...
Jan 16, 2013 9:39:00 AM com.endeca.soleng.sitemap.SitemapMain execute
INFO: Writing index file...
Jan 16, 2013 9:39:00 AM com.endeca.soleng.sitemap.SitemapMain writeIndexFile
INFO: Sitemap files output here: C:\Endeca\ToolsAndFrameworks\3.1.2\sitemap_generator
\sitemap\*.xml
Script completed successfully.
```

File outputs

The following sections discuss each of the files that are output by the Sitemap Generator. By default, these files are saved in the `sitemap_generator\sitemap` directory, but this location can be configured in the main configuration file.

Index file

The Index file contains links to each of the other files that the Sitemap Generator produces.

For example:

```
<?xml version="1.0" encoding="UTF-8" ?>
```

```
  <sitemapindex xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
  <sitemap>
    <loc>http://localhost:8006/discover/detail0.xml</loc>
  </sitemap>
  <sitemap>
    <loc>http://localhost:8006/discover/navigation0.xml</loc>
  </sitemap>
  <sitemap>
    <loc>http://localhost:8006/discover/searchterm0.xml</loc>
  </sitemap>
  <sitemap>
    <loc>http://localhost:8006/discover/staticpage0.xml</loc>
  </sitemap>
</sitemapindex>
```

Detail files

The detail files contain links to individual record detail pages. Based on the query specified in the main configuration file, a link to a record details page is generated for each record returned by that query.

The following is a sample detail file that might be generated using the default settings:

```
<?xml version="1.0" encoding="UTF-8" ?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">

<url>
   <loc>http://localhost:8006/discover/detail/Logitech/QuickCam-Chat-for-Skype/_/
A-1000751</loc>
   <lastmod>2013-01-16</lastmod>
</url>

<url>
   <loc>http://localhost:8006/discover/detail/Logitech/QuickCam-Communicate-Deluxe/_/
A-1000760</loc>
   <lastmod>2013-01-16</lastmod>
</url>

<url>
   <loc>http://localhost:8006/discover/detail/Logitech/QuickCam-Communicate-STX/_/
A-1000761</loc>
   <lastmod>2013-01-16</lastmod>
</url>
...
</urlset>
```

Related links

- The main configuration file (page 4)

Navigation files

For each record returned by the query specified in the main configuration file, the Sitemap Generator creates navigation links based on the dimension values (i.e. 'Camera', or 'Auto-Focus') associated with that record, as well as dimension settings in the main configuration file.

When a user browses an Oracle Commerce application, they reach record pages through intermediate navigation pages. For example, if the record is an auto-focus camera, a user might reach this record by selecting 'Cameras', and then selecting 'Auto-Focus.' The 'Cameras' page and the 'Cameras > Auto-Focus' page are navigation pages.

The following is a sample navigation file that might be generated using default settings:

```
<?xml version="1.0" encoding="UTF-8" ?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
<url>
   <loc>http://localhost:8006/discover/browse/cameras/_/N-25y6Z1z141xx</loc>
   <lastmod>2013-01-11</lastmod>
</url>

<url>
   <loc>http://localhost:8006/discover/browse/bags-cases/_/N-1z141xuZ25xwZej3</loc>
   <lastmod>2013-01-10</lastmod>
</url>
...
</urlset>
```

Related links

- The main configuration file (page 4)

Search term files

The Sitemap Generator creates URLs for search results pages for the most popular search terms, as specified in the search terms configuration file.

The base portion of these search page links can be customized in the template configuration file, and the query string parameters can be customized in the URL formatting configuration file for the Assembler. Typically, you generate the search terms configuration file based on a list of the application's most commonly searched terms.

The following is a sample search terms file that might be generated using default settings:

```
<?xml version="1.0" encoding="UTF-8" ?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">

<url>
   <loc>http://localhost:8006/discover/browse/_/N-ej4?Ntt=DSLR</loc>
   <lastmod>2013-02-19</lastmod>
</url>

<url>
   <loc>http://localhost:8006/discover/browse?Ntt=Digital+SLR</loc>
   <lastmod>2013-02-19</lastmod>
</url>

<url>
   <loc>http://localhost:8006/discover/browse?Ntt=PowerShot</loc>
   <lastmod>2013-02-19</lastmod>
</url>
```

```
</urlset>
```

Related links

- The search terms configuration file (page 14)

Static pages files

The static pages files contain links to pages with static URLs. These can be pages created in Experience Manager, or pages that are managed in another system. The configuration details for these page URLs are specified in the static pages configuration file.

The base section of these URLs can be modified using the template configuration file.

The following is a sample static page that might be generated using default settings:

```
<?xml version="1.0" encoding="UTF-8" ?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">

<url>
   <loc>http://localhost:8006/discover/browse</loc>
   <lastmod>2013-01-16</lastmod>
</url>

<url>
   <loc>http://localhost:8006/discover/contact-us</loc>
   <lastmod>2013-01-16</lastmod>
</url>

<url>
   <loc>http://localhost:8006/discover/about-us</loc>
   <lastmod>2013-01-16</lastmod>
</url>

<url>
   <loc>http://localhost:8006/discover/mobile/browse</loc>
   <lastmod>2013-01-16</lastmod>
</url>

</urlset>
```

Related links

- The static pages configuration file (page 15)

4 Generating Sitemaps for Multiple Sites

This section describes the processes for configuring and running the Sitemap Generator in an implementation with multiple sites.

Related links

- Configuring the Sitemap Generator for specific sites (page 23)
- Running the Sitemap Generator (page 23)

Configuring the Sitemap Generator for specific sites

You must make some modifications to your sitemap configuration to produce sitemap files for a specific site in a multiple site EAC application.

Configure the Sitemap Generator by following the instructions in *Configuring the Sitemap Generator* (page 3). To configure the Sitemap Generator for a specific site, you must make the following additional changes:

- In the `sitemap_generator/conf.xml` file, specify a `ROOT_QUERY` element that corresponds to the site-based filter defined in the site's `filterState.xml` file.
- Specify a site-specific `index.html` file.
- In the template configuration file, update the link configurations to include the site-specific URL.
- Optionallly, configure search terms for your site.

Running the Sitemap Generator

You can run the Sitemap Generator from the command line using `RunSitemapGen.bat` (on Windows) or `RunSitemapGen.sh` (on UNIX).

The Sitemap Generator runs using the configuration files stored in the `sitemap_generator` directory. To create sitemap pages for each site in an EAC application, you need to set up configurations for each site. For example, if

you are creating sitemaps for a Canon and Sony site in the Discover reference application, you would make two copies of the sitemap_generator/conf directory and give the new `conf` directories unique names.

```
sitemap_generator/conf
sitemap_generator/conf_canon
sitemap_generator/conf_sony
```

1. Edit the `conf.xml` file in each site configuration directory.
   a. Specify the type of template file, *`html_template.xml`* or *`xml_template`*`.xml`, in the `TEMPLATE_FILE` element.
   b. In the MDEX engines section, specify the `host` and `port` of the machine that the MDEX engine is running on.
   c. Specify a `ROOT_QUERY` element that restricts the search and navigation query to the records filtered by site. You must specify a `ROOT_QUERY` element that matches the filter parameters specified in the `filterState.xml` file.

      The following example shows the `filterState.xml` file and the corresponding Sitemap Generator `conf.xml` file for a site that restricts itself to only display Canon-branded items.

      The `filterState.xml` file shows a configuration with a record filter that only includes records with the Canon brand.

```
<Item class="com.endeca.infront.navigation.model.FilterState" xmlns="http://
endeca.com/schema/xavia/2010">
 <Property name="recordFilters">
 <List>
 <String>product.brand.name:Canon</String>
 </List>
 </Property>
</Item>
```

      The MDEX engine section of the `conf.xml` file has the host and port of the MDEX engine and a `ROOT_QUERY` element that has been updated to reflect a site restricted to the Canon brand. The `ROOT_QUERY` element specifies the query string for the Sitemap Generator to use when submitting the bulk export query against the specified index.

```
<MDEX_ENGINES>
  <ENGINE>
      <HOST>myhost.example.com</HOST>
      <PORT>15000</PORT>
      <ROOT_QUERY><![CDATA[N=4294967266]]></ROOT_QUERY>
      <ROLLUP_KEY>product.code</ROLLUP_KEY>
  </ENGINE>
</MDEX_ENGINES>
```

      See the *Oracle Commerce Administrator's Guide* for more information on site-based filters.

   d. Specify an output directory in the `INDEX` element so that it specifies the name of the site index file that the Sitemap Generator creates. This setting determines the relative path and extension for other output files for the site as well.

For example, the index element for a Canon site could look like this:

```
<INDEX_FILE>../sitemap_canon/index.xml</INDEX_FILE>
```

2. Edit the template configuration files in each site configuration directory. These configuration files are named `html_template.xml` or `xml_template.xml`.

   a. Specify the `localhost` with the hostname of the server, or the domain pattern information for each site if your site uses a domain pattern.

   b. If your site uses URI patterns, specify the site URI pattern for navigation links, detail links, static pages links, and search term links.

      In the following examples, the URI pattern is */Canon*.

      Here is the URI pattern in a navigation link for navigation pages:

```
<NAVIGATION_LINK><![CDATA[
<url>
   <loc>http://localhost:8006/discover/Canon/browse**FORMATTED_URL|XmlEscape**</
loc>
   <lastmod>**TIMESTAMP**</lastmod>
</url>
]]></NAVIGATION_LINK>
```

      Here is the URI pattern in a link for record detail pages:

```
<DETAIL_LINK><![CDATA[
<url>
   <loc>http://localhost:8006/discover/Canon/detail**FORMATTED_URL|XmlEscape**</
loc>
   <lastmod>**TIMESTAMP**</lastmod>
</url>
]]></DETAIL_LINK>
```

      Here is the URI pattern in a link in for a static page. This example assumes that the static pages are site specific.

```
<STATIC_PAGE_LINK><![CDATA[
<url>
   <loc>http://localhost:8006/discover/Canon/**STATIC_PAGE|XmlEscape**</loc>
   <lastmod>**TIMESTAMP**</lastmod>
</url>
]]></STATIC_PAGE_LINK>
```

      Here is the URI pattern in a link for search terms:

```
<SEARCH_TERM_LINK><![CDATA[
<url>
   <loc>http://localhost:8006/discover/Canon/browse**FORMATTED_URL|XmlEscape**</
loc>
```

```
   <lastmod>**TIMESTAMP**</lastmod>
</url>
]]></SEARCH_TERM_LINK>
```

3. If you want to add additional search terms for your sites, edit the `searchterms.xml` file in each site configuration directory.

4. Go to the `sitemap_generator/bin` directory and run `RunSitemapGen.bat` (on Windows) or `RunSitemapGen.sh` (on UNIX) for each site, specifying the unique `conf.xml` for that site. For example:

```
RunSitemapGen.bat ..\conf_canon\conf.xml
RunSitemapGen.bat ..\conf_sony\conf.xml
```

   The output appears in the `sitemap_generator/sitemap_canon` and `sitemap_generator/sitemap_sony` folders.

5 Troubleshooting and Performance

This section provides information about performance and troubleshooting common problems encountered while running and configuring the Sitemap Generator.

Related links

- Common errors (page 27)
- Performance information (page 29)

Common errors

This section addresses the most common errors encountered while running the Sitemap Generator.

Error message or problem	Description and suggested solution
`Unable to Locate Config File Specified`	This error is generated by the `.sh` or `.bat` scripts if they cannot locate the main configuration file specified as an input parameter. Make sure to specify either an absolute path, or a path relative to the directory from which you are running the script.
`ENDECA_ROOT Is Not Set`	This error is generated by the `.sh` or `.bat` scripts if the `ENDECA_ROOT` environment variable is not set. These scripts use `ENDECA_ROOT` to locate `commons-logging.jar` and `servlet-api.jar`.
`Fatal Error in Running Queries: Error Establishing Connection to Retrieve Navigation Engine`	This generally means you are trying to connect to an MDEX Engine that is not running. Check the main configuration file (`conf.xml`) and make sure that the host and port of the MDEX Engine are correct.

Error message or problem	Description and suggested solution
`Cannot find endeca-sitemapgen jar file under <path>`	This error is generated by the `.sh` or `.bat` scripts if they are unable to locate the `sitemap_generator\lib\endeca-sitemapgen-<version>.jar` file. The location of this file is computed from the location of the `.sh` or `.bat` scripts, with the assumption that those scripts have not been moved from their default locations. This file also needs to be included in the classpath when running the Sitemap Generator.
`Unable to Locate Template Configuration File`	The Sitemap Generator throws this exception if it is unable to find the template configuration file specified in the main configuration file. Remember to use either an absolute path, or a path relative to the location of the main configuration file, when specifying the template configuration file to use.
`Can't Find Output Files`	Remember to use either an absolute path or a path relative to the location of the main configuration file, when specifying the target location for output files. If you are still unable to find the output files, the last line of the standard output should indicate the location where the files were written, such as: `INFO: Sitemap files output here: C:\Endeca \ToolsAndFrameworks\<version>\sitemap_generator\sitemap \*.xml`
`Sitemap Files Missing Values / Name of the Tag Output`	If sitemap files that are output are missing values that were specified in the template configuration files, or contain the Name of the tag used in the template configuration files, such as: `<a href="http://localhost:8006/discover/detail?id=">...</a>` Or: `<a href="http://localhost:8006/discover/detail?id=id">...</a>`, those keys did not have corresponding values for that given record. In the above example, there is no property "id" specified in the `QUERY_FIELD_LIST`. Make sure such a property exists, and that it is included in the `QUERY_FIELD_LIST` in the main configuration file.
`No Navigation Pages Created`	If no navigation pages were created, then the dimensions listed in the `NAVIGATION_PAGE_SPEC_LIST` parameter in the main configuration file may not be included in the `QUERY_FIELD_LIST`. Make sure to add the dimensions to the `QUERY_FIELD_LIST` in the main configuration file. Another possibility is that no records exist with the combination of dimensions that you have specified.

Error message or problem	Description and suggested solution
`Errors Reading Search Term Queries`	If errors occurred running search term queries, the problem may be that the search terms configuration file is not configured appropriately. Please check your: • `MDEXHOST` • `MDEXPORT` • `DEFAULTQUERY` as well as the parameters for each query that failed.

Related links

- The main configuration file (page 4)
- The template configuration file (page 9)
- The search terms configuration file (page 14)

Performance information

Depending on your application, using the Sitemap Generator may have performance implications. This section addresses those possible implications and provides options for resolving them.

Runtime performance

When run locally on the same server as the MDEX Engine being queried, the Sitemap Generator is able to create more than 50,000 detail and navigation links in less than 30 seconds. When run remotely, performance varies based on network congestion.

Production vs. dedicated MDEX Engine

Since the bulk-export query can be expensive, it is advisable for sites with high throughput requirements to either schedule the Sitemap Generator for off-peak hours, or even run the Sitemap Generator against a dedicated staging index. When running the Sitemap Generator against a dedicated index, you can gain an additional performance advantage by using the `QUERY_FIELD_LIST` to specify the properties and dimensions that are available for display in URLs rather than enabling the properties and dimensions to display with record list.

Combinatoric navigation pages

Due to the method in which navigation links are defined, it is important to avoid creating complex navigation page specs that would result in an overwhelming number of unique links. Such a scenario may cause excessive memory usage by the Sitemap Generator, as well as long runtimes to write all of the specified links. Oracle strongly recommends that only two or fewer dimensions be specified in the main configuration file for any given <NAVIGATION_PAGE_SPEC>. Otherwise, it is possible to generate millions of navigation links and reduce the site's search engine ranking by making it appear to be a link farm.

Related links

- The navigation page spec list (page 6)

6 Implementing the Sitemap Generator in Production

This section describes the steps necessary to move your Sitemap Generator files into production and update search engines.

Related links

- About validating sitemaps (page 31)
- About moving sitemap files to production (page 31)
- Notifying search engines of a sitemap location (page 32)
- Notifying search engines of an updated sitemap (page 32)

About validating sitemaps

Before deploying your sitemaps, you should ensure that all the generated links are valid.

Oracle recommends that you write a script to test each URL in the sitemap against your application server and confirm that each link returns an HTTP 200 (OK) code. Broken links may indicate a mismatch between the URL formatting configuration file in your Assembler application, and the Sitemap Generator URL configuration file.

In addition to validating that links resolve to the same destination, confirm that the links created by the Sitemap Generator correspond to the canonical URLs created by the `CanonicalLinkBuilder` class in your Assembler application. You can compare the Sitemap Generator links to the links found in the `<link rel="canonical" ... />` tag of each corresponding application page `<head>` element.

Related links

- The URL formatting configuration file (page 15)

About moving sitemap files to production

Once you have created and validated the sitemap files, you should move them to the production Web servers. You can either do this manually, or via operations scripts.

It is also important to note that when the Sitemap Generator runs, it deletes any previously existing files from the target output directory. Therefore, if you want to do any sitemap archiving, you must handle this separately.

Notifying search engines of a sitemap location

Oracle recommends that you actively register your sitemap files with the major Web search engines.

The procedures for registering your sitemap directly with a search engine vary for each, and documentation is available online.

You can also notify search engines of the sitemap location through the `robots.txt` file. To do so, add a line to your site's `robots.txt` file similar to the following:

```
Sitemap: http://localhost:8006/discover/sitemap.xml
```

The `localhost:8006/discover` portion of the URL should be replaced with the absolute URL for your sitemap index file. For more information on this notification method please visit http://www.sitemaps.org/protocol.php#informing.

Notifying search engines of an updated sitemap

You can ping search engines to let them know that your sitemap has been updated. For more information and instructions for each search engine, please see the online documentation posted by the search engine you wish to notify.