Information Manager Developer Guide

A Guide to the Information Manager Web Services and Tag Library

InQuira Version 8.2

Document Number IM82-WSR-00

March 9, 2010

InQuira, Inc.

900 Cherry Ave., 6th Floor

San Bruno, CA 94066

Copyright Information

Restricted Rights

Trademarks and Service Marks

Contents

Preface About This Guide

This guide is intended for application developers who need to integrate Information Manager content, content category, and user and security functions with external applications. It contains reference information and examples for all packages, classes, methods, and interfaces of the Information Manager Web Services API.

In This Guide

The *Information Manager Developer's Guide* is divided into the following sections:

Chapter 1, Information Manager Web Services	This section describes the Information Manager Web Services interfaces that support the Information Manager security, content, content category, and user services.
Chapter 2, The Information Manager Tag Library	This section contains reference information and examples for the JSP tags included in the standard Information Manager tag library.
Chapter 3, Creating and Deploying Custom Java Server Pages (JSPs)	This section provides examples of using the Information Manager tag library tags to create JSPs.

Contacting InQuira

You can contact InQuira by mail, telephone, fax, and email.

Address:	InQuira, Inc. 900 Cherry Ave., 6th Floor San Bruno, CA 94066
Telephone:	(650) 246-5000
Fax:	(650) 264-5036
Email:	For sales information, send email to sales@inquira.com. For product support, send email to support@inquira.com.
World Wide Web:	Learn more about InQuira products, solutions, services, and support on the world wide web at: **www.inquira.com**.

InQuira Product Documentation

InQuira documentation is available only to licensed users of our software products and may not be redistributed in any form without express permission from InQuira, Inc.

The InQuira documentation is available in PDF format.Customers can download the PDF files from:

http://documentation.inquira.com/

NOTE: You need a PDF reader application installed on each processor on which you plan to view the InQuira product documentation. The Adobe Acrobat reader is available from Adobe Systems at: http://www.adobe.com.

Detailed information about each product document set is available in:

- *Intelligent Search Documentation* on page 3
- *InQuira Analytics Documentation* on page 5
- *Information Manager Documentation* on page 6
- *Contact Center Documentation* on page 7

If you encounter a problem, need help using the documentation, or want to report an error in the content, please contact InQuira Customer Support.

If you need help obtaining InQuira product documentation, or want to obtain permission to redistribute a portion of the contents, please contact your InQuira account representative.

Intelligent Search Documentation

Intelligent Search is distributed with the following documentation.

Document	Number	Description
Intelligent Search Installation Guide	IS80-IG-00	This guide is intended for technical staff who are responsible for installing InQuira 8.1. It provides detailed information on installing InQuira 8.1 and configuring the application on a single processor using the Installation Configuration Environment facility.

Intelligent Search Administration Guide	IS80-CA-00	This guide is intended for system and application administrators who need to configure an InQuira 8.1 application in an enterprise environment. It describes InQuira 8.1 integration, development, configuration, and maintenance processes and tasks.
Intelligent Search Language Administration Guide	IS80-LA-00	This guide is intended for business users and subject matter experts who need to create and maintain the language processing elements of an InQuira 8.1 application using the System Manager. This book provides usage information about the System Manager, conceptual information about the InQuira 8.1 language objects, and task information about the process of managing the user experience provided by the InQuira 8.1 application.
Intelligent Search Language Tuning Guide	IS80-LD-00	This guide is intended for application developers who need to create and maintain advanced InQuira 8.1 language-processing elements using the Dictionary and other InQuira Language Workbench applications.
Intelligent Search Optimization Guide	IS80-AG-00	This guide is intended for application developers who need to implement InQuira 8.1 advanced features, including Personalized Navigation and Process Wizards.
Intelligent Search Application Development Guide	IS80-API-00	This guide provides information about integrating and customizing the InQuira 8.1 Personalized Response User Interface.

Intelligent Search Language Reference	IS80-LRG-00	This guide is for language developers implementing InQuira 8.1 applications that utilize the intent libraries and advanced language processing functions. These guides are published as separate documents that provide reference information for each industry-specific intent library. Each reference also contains complete descriptions of InQuira Match Language and Variable Instantiation Language.
Intelligent Search User Interface Guide	IS80-UI-00	This guide is intended for application developers who need to customize the InQuira 8.1 Personalized Response User Interface, and integrate it with a production web application. It contains information about the elements and features of the User Interface, and provides guidelines for integrating it into an enterprise web architecture, customizing its appearance and functionality, and implementing various special features.

InQuira Analytics Documentation

InQuira Analytics is distributed with the following documentation.

Document	Number	Description
InQuira Analytics Installation Guide	IA80-IG-00	This guide is intended for technical staff who are responsible for installing InQuira Analytics. It provides detailed information on installing and configuring the InQuira Analytics product for use with an InQuira 8.1 application.
Analytics User Guide	IA80-CA-00	This guide is intended for systems and application administrators who need to configure the Intelligent Search and Information Manager Analytics components to report on InQuira 8.1 application performance.

Information Manager Documentation

InQuira Information Manager is distributed with the following documentation.

Document	Number	Description
Information Manager Installation Guide	IM80-IG-00	This guide is intended for technical staff who are responsible for installing InQuira Information Manager. It provides detailed information on installing and configuring the Information Manager product.
Information Manager Administration Guide	IM80-CA-00	This guide is intended for systems and application administrators who need to configure and administer an InQuira Information Manager application, and integrate it with an InQuira 8.1 application. It also contains information for general business users who need to use the Information Manager to create and manage content.
Information Manager Content Authoring Guide	IM80-AG-00	This guide is intended for technical staff who are responsible for authoring content in InQuira Information Manager. It provides detailed information on creating content and managing workflow tasks in the Information Manager console.
Information Manager Developer's Guide	IM80-WSR-00	This guide is intended for application developers who need to integrate Information Manager content, content category, and user and security functions with external applications. It contains reference information and examples for all packages, classes, methods, and interfaces of the Information Manager Web Services API.

Contact Center Documentation

The InQuira 8.1 contact center products are distributed with the following documentation.

Document	Number	Description
Contact Center Advisor Integration Guide	CA80-IG-00	This guide is intended for application developers and systems administrators who need to plan for and integrate the InQuira Contact Center Advisor with an InQuira application and a supported CRM application.
Intelligent Search Siebel Integration Guide	CAS80-IG-00	This guide is intended for application developers and systems administrators who need to plan for and integrate InQuira 8.1 with Siebel 7 Enterprise Applications using the Siebel Adapter for InQuira 8.1.

Screen and Text Representations

The product screens, screen text, and file contents depicted in the documentation are examples. We attempt to convey the product's appearance and functionality as accurately as possible; however, the actual product contents and displays may differ from the published examples.

References to World Wide Web Resources

For your convenience, we refer to Uniform Resource Locators (URLs) for resources published on the World Wide Web when appropriate. We attempt to provide accurate information; however, these resources are controlled by their respective owners and are therefore subject to change at any time.

Chapter 1 Information Manager Web Services

The following sections describe the Information Manager Web Services:

Introduction to Information Manager Web Services on page 10	This section describes the Information Manager Web Services package, which comprises the interfaces that support the security, content, content category, and user services. It describes the standard input, security requirements, and standard response for the Information Manager Web Services.
SecurityServices Implementation on page 14	This section describes the security interface, and the authenticate method, which are required for all calls to Information Manager Web Services.
Category Services Interface on page 21	This section describes the security interface and the set of public methods available for managing Information Manager category hierarchies.
Content Services Interface on page 32	This section describes the content services interface and the set of public methods available for managing Information Manager content.
User Services Interface on page 53	This section describes the user services interface and the set of public methods available for managing user information in an Information Manager repository.
Channel Services Interface on page 59	This section describes the channel services interface and the set of public methods available for managing Information Manager content channels.

Introduction to Information Manager Web Services

This section describes the Information Manager Web Services package, which comprises the interfaces (see *Information Manager Web Services Interfaces* on page 11) that support the security, content, content category, and user services. It describes the standard input, security requirements, and standard responses for the Information Manager Web Services.

Information Manager Web Services includes the following features:

- Is built using Apache Axis 2.x framework
- Is deployed as a single WAR file in servlet specification 2.2 or higher compliant J2EE containers
- Uses XML based strings to pass parameters for most services
- Is compatible with .Net
- Requires additional folder in $IM_HOME/config directory called IMWEBSERVICES (create or copy the application.properties file from IMADMIN folder)
- Supports a database connection pool. You can add parameters to the application.properties file to set the the minimum and maximum database connections:
  - MINDBConnections – minimum number of database connections to keep open (default is 1)
  - MAXDBConnections – the maximum number of database connections to open (default is 5)
- Includes pre-built Java client stubs available in imwsclient-xxx.jar deployed as part of the imwsclient.war file

---

NOTE: The Information Manager Web Services client files are not deployed by default during installation. All other Information Manager Web Services WAR files are available in the $IM_HOME/wars folder.

- Includes a JSP-based Web Service test application available at:

  `http://<host>:<port>/imws`

Information Manager Web Services Interfaces

The Information Manager Web Services contains the following interfaces:

Security Services Interface on page 18	This interface defines the set of public methods available for authenticating a user in order to use the other IM web services.
Category Services Interface on page 21	This interface defines the set of public methods available for managing Information Manager category trees.
Content Services Interface on page 32	This interface defines the set of public methods available for managing Information Manager content.
User Services Interface on page 53	This interface defines the set of public methods available for managing Information Manager user and security information.
Channel Services Interface on page 59	This interface defines the set of public methods available for managing Information Manager content channels.

Using the Information Manager Web Services Interfaces

Information Manager Web Services calls always require a security token, and typically use an XML-formatted string as an input parameter.

The security token is created by the Security Service using the `authenticate` method see *authenticate* on page 19 (see *Information Manager Web Services Authentication* on page 12 for more information). The structure of the XML input varies, depending on the Information Manager Web Services that is being called.

Each Information Manager Web Services responds in a standard format that contains relevant data about the transaction (see *Information Manager Web Services XML Response* on page 13 for more information).

Information Manager Web Services Authentication

Each Information Manager Web Services requires a security token to be passed in as a parameter. The security token can be obtained from the Security Service using the method *authenticate* on page 19 . Any parameter that requires the security token to be passed in must pass in the entire response generated by the *authenticate* on page 19 method, including the encrypted data.

Security Token Expiration

The security token is valid only for a specified time period, and cannot be re-used after the specified time. The period for which the token is valid is configurable.

Security Token Example

The following is an example of a security token generated by the `authenticate()` see *authenticate* on page 19method:

```
<RESPONSE>
   <STATUS>
      <SERVERTIME><![CDATA[1140031278184]]></SERVERTIME>
      <TYPE><![CDATA[authenticate]]></TYPE>
      <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
      <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
      <VALUE><![CDATA[1]]></VALUE>
      <ERRORS></ERRORS>
   </STATUS>
   <DATA>
      <TOKEN><![CDATA[IT/9+42A/
Qs6FlPSStOXib2eWifayDcMjJCUz1V9wlLmlPEL3Z9DTyGXa4mQIZts0YLNDrUPOm0cKGaTvh
mBtYMAvCxoivFzg4D57FkXvYvnbLBOH/59MN/
pcY3XKxQPClPb5+2ZR766RzBjpy+C6X8tPeH5YnIDbkx5xuDcVxnU+0Wm5fSSgTKa0LiL9Yw
CKa9yKDySNfhHFHnPzHgOaQ==]]></TOKEN>
   </DATA>
</RESPONSE>
```

Information Manager Web Services XML Input

Information Manager Web Services calls typically use an XML-formatted string as an input parameter. The structure of the XML input varies, depending on the Information Manager Web Services that is being called.

Information Manager Web Services XML Response

Each Information Manager Web Services responds in a standard format that contains relevant data about the transaction.

The general format of the response XML is as follows:

```
<RESPONSE>
   <STATUS>
      <SERVERTIME><![CDATA[1139414879065]]></SERVERTIME>
      <TYPE><![CDATA[getCategories]]></TYPE>
      <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
      <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
      <VALUE><![CDATA[1]]></VALUE>
      <ERRORS/>
   </STATUS>
   <DATA>...</DATA>
</RESPONSE>
```

where:

<**SERVERTIME**>	is the server time in milliseconds	
<**TYPE**>	is the name of the web service call	
<**RESPONSE_ VERSION**>	is the version of the response schema	
<**DATA_VERSION**>	is the version of the data response	
<**VALUE**>	is the return value from web service call. Values:	
	> **0**	indicate success
	< **0**	indicate failure
<**ERRORS/**>	is a node containing errors that occurred. If the web service call throws an exception, the ERRORS node will be populated with an error code and a message describing the error.<tbd> No exceptions will be thrown.	
<**DATA**>	is a section that contains the specific response from the web service	

SecurityServices Implementation

java.lang.Object

com.inquira.imws.impl.SecurityServicesImpl

All Implemented Interfaces:

com.inquira.imws.interfaces.SecurityServices

public class SecurityServicesImpl

Extends:

java.lang.Object

Implements:

com.inquira.imws.interfaces.SecurityServices

Description:

This class generates the security token used to authenticate web services.

Field Summary:

static org.apache.log4j.Logger	logger

Constructor Summary:

SecurityServicesImpl()

Method Summary:

See *Security Services Interface* on page 18.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

CategoryServices Implementation

java.lang.Object

com.inquira.imws.impl.CategoryServicesImpl

All Implemented Interfaces:

com.inquira.imws.interfaces.CategoryServices

public class CategoryServicesImpl

Extends:

java.lang.Object

Implements:

com.inquira.imws.interfaces.CategoryServices

Description:

This class provides access to content category hierarchies defined within an Information Manager application repository.

Field Summary:

static org.apache.log4j.Logger	logger

Constructor Summary:

CategoryServicesImpl()

Method Summary:

See *Channel Services Interface* on page 59.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

ContentServices Implementation

java.lang.Object

com.inquira.imws.impl.ContentServicesImpl

All Implemented Interfaces:

com.inquira.imws.interfaces.ContentServices

public class ContentServicesImpl

Extends:

java.lang.Object

Implements:

com.inquira.imws.interfaces.ContentServices

Description:

This class provides access to content stored in an Information Manager application repository.

Field Summary:

static org.apache.log4j.Logger	logger

Constructor Summary:

ContentServicesImpl()

Method Summary:

See *Content Services Interface* on page 32.

Methods inherited from class java.lang.Object:

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

The UserServices Implementation

java.lang.Object

com.inquira.imws.impl.UserServicesImpl

All Implemented Interfaces:

com.inquira.imws.interfaces.UserServices

public class UserServicesImpl

Extends:

java.lang.Object

Implements:

com.inquira.imws.interfaces.UserServices

Field Summary

static org.apache.log4j.Logger	logger

Constructor Summary

UserServicesImpl()

Method Summary:

See *User Services Interface* on page 53.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Security Services Interface

com.inquira.imws.interfaces

Interface SecurityServices

public interface **SecurityServices**

This interface defines the methods available to authenticate users when accessing the content, content category, and user services.

Each web service requires that a security token be passed as a parameter, and any parameter that requires the security token must pass the entire response generated by the `SecurityService.authenticate()` method, including the encrypted data. See *Information Manager Web Services Authentication* on page 12 for more information.

Information Manager Web Services calls typically use an XML-formatted string as an input parameter. The structure of the XML input varies, depending on the Information Manager Web Services that is being called.

Each web service will return a standardized response that contains relevant data about the transaction as described in *Information Manager Web Services XML Response* on page 13.

Method	Description
authenticate on page 19	This method authenticates the user and returns an XML string containing the security token required for all other IMWS methods. NOTE: The token is valid for a configurable period of time. The default value is 3000 seconds (50 min.). To override the default value add token.timeout (time is in seconds) to the application.properties file. The required parameters are case sensitive.
isTokenValid on page 20	This method determines if a security token is valid.

authenticate

Method:

public java.lang.String authenticate(java.lang.String inputXml)

Description:

Method to authenticate user.

Specified by:	authenticate in interface com.inquira.imws.interfaces.SecurityServices	
Parameters:	**inputXml**	XML formatted string containing the required data (userid, password, and repository reference key)
Returns:	An XML string containing the security token required for all other IMWS methods. The token is valid for a configurable period of time (the default value is 600 seconds). To override the default value add token.timeout (time is in seconds) to the application.properties file. The required parameters are case sensitive.	

Sample input:

```
<AUTHENTICATE>
   <USER>USER LOGIN</USER>
   <PASSWORD>USER PASSWORD</PASSWORD>
   <REPOSITORY_REFERENCE_KEY>REPOSITORY REFERENCE KEY</
REPOSITORY_REFERENCE_KEY>
</AUTHENTICATE>
```

Sample Response:

```
<RESPONSE>
<STATUS>
  <SERVERTIME><![CDATA[1139413342126]]></SERVERTIME>
  <TYPE><![CDATA[authenticate]]></TYPE>
  <REPONSE_VERSION><![CDATA[1]]></REPONSE_VERSION>
  <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
  <VALUE><![CDATA[1]]></VALUE>
  <ERRORS/>
</STATUS>
<DATA>
<TOKEN><![CDATA[ENCRYPTED TOKEN DATA]]></TOKEN>
</DATA>
</RESPONSE>
```

isTokenValid

Method:

boolean isTokenValid(java.lang.String token)

Description:

Method to check the validity of a security token prior to use.

Specified by:	authenticate in interface com.inquira.imws.interfaces.SecurityServices	
Parameters:	**token**	Security token
Returns:	True for a valid token or False for an invalid token.	

Sample Response:

```
<RESPONSE>
  <STATUS>
    <SERVERTIME><![CDATA[1140031278184]]></SERVERTIME>
    <TYPE><![CDATA[authenticate]]></TYPE>
    <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
    <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
    <VALUE><![CDATA[1]]></VALUE>
    <ERRORS></ERRORS>
  </STATUS>
  <DATA>
    <TOKEN><![CDATA[IT/9+42A/Qs6FlPSStOXib2eWia9yKDySNfhHFHnPzHgOaQ==]]></
TOKEN>
  </DATA>
</RESPONSE>
```

Category Services Interface

com.inquira.imws.interfaces

Interface CategoryServices

public interface **CategoryServices**

This interface defines the set of public methods available for managing Information Manager category hierarchies.

Each web service requires that a security token be passed as a parameter, and any parameter that requires the security token must pass the entire response generated by the `SecurityService.authenticate()` method, including the encrypted data. See *Information Manager Web Services Authentication* on page 12 for more information.

Information Manager Web Services calls typically use an XML-formatted string as an input parameter. The structure of the XML input varies, depending on the Information Manager Web Services that is being called.

Each web service will return a standardized response that contains relevant data about the transaction as described in *Information Manager Web Services XML Response* on page 13 .

Method	Description
addCategory on page 22	This method adds a new category to the Information Manager category tree. If the parent_reference_key is not specified, the category is created as a root category. Otherwise the category is created as a child of an existing parent. The display name is stored in the default locale of the user.
deleteCategory on page 23	This method deletes a category from the repository. The response contains whether or not the delete was successful.
getCategories on page 24	This method returns a list of categories for the given locale and parent. If no parent category is specified, the root categories are displayed.
getCategory on page 26	This method returns the details for a single category identified by the reference key passed as a parameter.
getChannelCategories on page 28	This method returns the list of categories that are assigned to a specific channel. The display names are based on the locale passed in. If no locale is passed in the default locale for the repository is used.
updateCategory on page 30	This method updates 1 or more categories. The status of the update for each category is displayed in the response.

addCategory

Method:

java.lang.String addCategory(java.lang.String token, java.lang.String inputXml)

Description:

This method adds a category into the system. If a parent reference key is specified, the category is added as a child to the specified parent. The display name is used for the default locale of the repository.

Specified by:	com.inquira.imws.interfaces.CategoryServices	
Parameters:	token	security token
	inputXml	XML formated input (see above) containing the required data
Returns:	An XML-formatted string containing the response structure (see Sample Response below)	

Sample Input:

```
<CATEGORY>
  <REFERENCE_KEY>IMWS12</REFERENCE_KEY>
  <DISPLAY_NAME>imws12</DISPLAY_NAME>
  <CATEGORY_DESCRIPTION>Description of the category</CATEGORY_DESCRIPTION>
  <PARENT_REFERENCE_KEY></PARENT_REFERENCE_KEY>
</CATEGORY>
```

Sample Response:

```
<RESPONSE>
  <STATUS>
    <SERVERTIME><![CDATA[1139416934564]]></SERVERTIME>
    <TYPE><![CDATA[addCategory]]></TYPE>
    <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
    <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
    <VALUE><![CDATA[1]]></VALUE>
    <ERRORS/>
  </STATUS>
  <DATA>
    <CATEGORIES>
      <CATEGORY>
        <NAME><![CDATA[MY CAT]]></NAME>
        <REFERENCE_KEY><![CDATA[IMWS12]]></REFERENCE_KEY>
        <GUID><![CDATA[0049844aeb01094a8aa499007fff]]></GUID>
        <OBJECTID><![CDATA[478.003.001]]></OBJECTID>
        <PARENTID><![CDATA[0049d401c60108e94a569a007fff]]></PARENTID>
        <CHILDCOUNT><![CDATA[0]]></CHILDCOUNT>
      </CATEGORY>
    </CATEGORIES>
```

```
  </DATA>
</RESPONSE>
```

deleteCategory

Method:

java.lang.String deleteCategory(java.lang.String token, java.lang.String inputXml)

Description:

This method delete a category from the system.

IMPORTANT: The system does not prompt for confirmation of deletions.

Specified by:	com.inquira.imws.interfaces.CategoryServices	
Parameters:	**token**	security token
	inputXml	XML-formated input containing the required data
Returns:	XML-formatted string containing response structure (see Sample Response below)	

Sample Input:

```
<CATEGORY>
  <REFERENCE_KEY>IMWS12</REFERENCE_KEY>
</CATEGORY>
```

Sample Response:

```
<RESPONSE>
  <STATUS>
    <SERVERTIME><![CDATA[1139417446792]]></SERVERTIME>
    <TYPE><![CDATA[deleteCategory]]></TYPE>
    <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
    <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
    <VALUE><![CDATA[1]]></VALUE>
    <ERRORS/>
  </STATUS>
  <DATA>
    <CATEGORIES/>
  </DATA>
</RESPONSE>
```

getCategories

Method:

java.lang.String getCategory(java.lang.String token, java.lang.String inputXml)

Description:

This method returns a list of categories for a repository. If a parent reference key is not provided, all of the first branch of categories are returned. If a parent reference key is provided then its direct children are returned. If a locale is not supplied, then the default locale for the repository is used.

Specified by:	com.inquira.imws.interfaces.CategoryServices	
Parameters:	**token**	security token
	inputXml	XML-formated input containing the required data
Returns:	XML-formatted string containing response structure (see Sample Response below)	

Sample Input:

```
--Using reference keys
<CATEGORY>
   <PARENT_REFERENCE_KEY>NEWS</PARENT_REFERENCE_KEY>
   <LOCALE_CODE></LOCALE_CODE>
</CATEGORY>

--Using GUIDS
 <CATEGORY>
  <GUID>NEWS</GUID>
  <LOCALE_CODE></LOCALE_CODE>
</CATEGORY>

--Using OBJECTS ID
 <CATEGORY>
  <OBJECTID>NEWS</OBJECTID>
  <LOCALE_CODE></LOCALE_CODE>
</CATEGORY>
```

Sample Response:

```
 <RESPONSE>
   <STATUS>
      <SERVERTIME><![CDATA[1139416313519]]></SERVERTIME>
      <TYPE><![CDATA[getCategory]]></TYPE>
      <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
      <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
      <VALUE><![CDATA[1]]></VALUE>
      <ERRORS/>
   </STATUS>
   <DATA>
      <CATEGORIES>
         <CATEGORY>
            <NAME><![CDATA[News]]></NAME>
            <CATEGORYDESCRIPTION><![CDATA[This category is about News]]></
CATEGORYDESCRIPTION>
            <REFERENCE_KEY><![CDATA[NEWS]]></REFERENCE_KEY>
            <GUID><![CDATA[192-168-1-9-63e1b8-f19e162ac3--7fd9]]></GUID>
            <OBJECTID><![CDATA[053]]></OBJECTID>
            <PARENTID/>
            <CHILDCOUNT><![CDATA[11]]></CHILDCOUNT>
         </CATEGORY>
      </CATEGORIES>
   </DATA>
</RESPONSE>
```

getCategory

Method:

java.lang.String getCategory(java.lang.String token, java.lang.String inputXml)

Description:

Returns the category for the passed in reference key. Returns the categories assigned to the specified content channel. If a content channel is not specified then all of the categories for the repository are returned. If a locale is not specified then the default locale for the repository is used.

Specified by:	com.inquira.imws.interfaces.CategoryServices	
Parameters:	**token**	security token
	inputXml	XML-formated input containing the required data
Returns:	XML-formatted string containing response structure (see Sample Response below)	

Sample Input:

```
--Using reference keys
 <CATEGORY>
      <REFERENCE_KEY>NEWS</REFERENCE_KEY>
      <LOCALE_CODE></LOCALE_CODE>
 </CATEGORY>

--Using GUIDS
 <CATEGORY>
  <GUID>NEWS</GUID>
  <LOCALE_CODE></LOCALE_CODE>
</CATEGORY>

--Using OBJECTS ID
 <CATEGORY>
  <OBJECTID>NEWS</OBJECTID>
  <LOCALE_CODE></LOCALE_CODE>
</CATEGORY>
```

Sample Response:

```
<RESPONSE>
   <STATUS>
      <SERVERTIME><![CDATA[1139416313519]]></SERVERTIME>
      <TYPE><![CDATA[getCategory]]></TYPE>
      <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
      <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
      <VALUE><![CDATA[1]]></VALUE>
      <ERRORS/>
   </STATUS>
   <DATA>
      <CATEGORIES>
         <CATEGORY>
            <NAME><![CDATA[News]]></NAME>
            <CATEGORYDESCRIPTION><![CDATA[This category is about News]]></
CATEGORYDESCRIPTION>
            <REFERENCE_KEY><![CDATA[NEWS]]></REFERENCE_KEY>
            <GUID><![CDATA[192-168-1-9-63e1b8-f19e162ac3--7fd9]]></GUID>
            <OBJECTID><![CDATA[053]]></OBJECTID>
            <PARENTID/>
            <CHILDCOUNT><![CDATA[11]]></CHILDCOUNT>
         </CATEGORY>
      </CATEGORIES>
   </DATA>
</RESPONSE>
```

getChannelCategories

Method:

java.lang.String getChannelCategories(java.lang.String token, java.lang.String inputXml)

Description:

Returns the list of categories that are available for the passed in content channel.

Specified by:	com.inquira.imws.interfaces.CategoryServices	
Parameters:	**token**	security token
	inputXml	XML-formated input containing the required data
Returns:	XML-formatted string containing response structure (see Sample Response below)	

Sample Input:

```
<CATEGORY>
  <CHANNEL_REFERENCE_KEY>NEWS</CHANNEL_REFERENCE_KEY>
  <LOCALE_CODE></LOCALE_CODE>
  <SORT_ORDER>name</SORT_ORDER>
  <SORT_DIRECTION>desc</SORT_DIRECTION>
</CATEGORY>
```

Sample Response:

```
<RESPONSE>
  <STATUS>
    <SERVERTIME><![CDATA[1139415478107]]></SERVERTIME>
    <TYPE><![CDATA[getChannelCategories]]></TYPE>
    <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
    <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
    <VALUE><![CDATA[1]]></VALUE>
    <ERRORS/>
  </STATUS>
  <DATA>
```

```
      <CATEGORIES>
        <CATEGORY>
          <NAME><![CDATA[News]]></NAME>
          <CATEGORYDESCRIPTION><![CDATA[This category is about News]]></
CATEGORYDESCRIPTION>
          <REFERENCE_KEY><![CDATA[NEWS]]></REFERENCE_KEY>
          <GUID><![CDATA[192-168-1-9-63e1b8-f19e162ac3--7fd9]]></GUID>
          <OBJECTID><![CDATA[053]]></OBJECTID>
          <PARENTID/>
          <CHILDCOUNT><![CDATA[11]]></CHILDCOUNT>
        </CATEGORY>
        <CATEGORY>
          <NAME><![CDATA[Instruments]]></NAME>
          <CATEGORYDESCRIPTION><![CDATA[The description of this directory is Music]]></
CATEGORYDESCRIPTION>
          <REFERENCE_KEY><![CDATA[INSTRUMENTS]]></REFERENCE_KEY>
          <GUID><![CDATA[216-199-173-3-b307f0-f6ef16ad04--6f2a]]></GUID>
          <OBJECTID><![CDATA[103]]></OBJECTID>
          <PARENTID/>
          <CHILDCOUNT><![CDATA[3]]></CHILDCOUNT>
        </CATEGORY>
      </CATEGORIES>
    </DATA>
</RESPONSE>
```

updateCategory

Method:

java.lang.String updateCategory(java.lang.String token, java.lang.String inputXml)

Description:

This method updates a list of categories. Multiple categories can be passed in and will be updated. The response will contain a node of all of the updated categories.

Specified by:	updateCategory in interface com.inquira.imws.interfaces.CategoryServices	
Parameters:	**token**	Security token.
	inputXml	XML formated input containing the required data. See Sample Input.
Returns:	XML formatted string containing response structure (see Sample Response)	

Sample Input:

```
<CATEGORIES>
  <CATEGORY>
    <REFERENCE_KEY>WS1</REFERENCE_KEY>
    <DISPLAY_NAME>MY WS CAT 1</DISPLAY_NAME>
    <CATEGORY_DESCRIPTION></CATEGORY_DESCRIPTION>
    <LOCALE_CODE/>
  </CATEGORY>
  <CATEGORY>
    <REFERENCE_KEY>WS2</REFERENCE_KEY>
    <DISPLAY_NAME>MY WS CAT 2</DISPLAY_NAME>
    <CATEGORY_DESCRIPTION></CATEGORY_DESCRIPTION>
    <LOCALE_CODE/>
  </CATEGORY>

 <CATEGORY>
    <REFERENCE_KEY>WS3</REFERENCE_KEY>
    <DISPLAY_NAME>MY WS CAT 3</DISPLAY_NAME>
    <CATEGORY_DESCRIPTION></CATEGORY_DESCRIPTION>
    <LOCALE_CODE/>
  </CATEGORY>
</CATEGORIES>
```

Sample Response:

```
<RESPONSE>
  <STATUS>
    <SERVERTIME><![CDATA[1139417719578]]></SERVERTIME>
    <TYPE><![CDATA[updateCategory]]></TYPE>
    <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
    <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
    <VALUE><![CDATA[1]]></VALUE>
    <ERRORS/>
  </STATUS>
  <DATA>
    <CATEGORIES>
      <CATEGORY>
        <NAME><![CDATA[MY WS CAT 1]]></NAME>
        <CATEGORYDESCRIPTION><![CDATA[This is the description of the category]]></
CATEGORYDESCRIPTION>
        <REFERENCE_KEY><![CDATA[WS1]]></REFERENCE_KEY>
        <GUID><![CDATA[0049d401c60108e94a569a007fff]]></GUID>
        <OBJECTID><![CDATA[478.003]]></OBJECTID>
        <PARENTID><![CDATA[004928e9fe0108e4de34d8007fff]]></PARENTID>
        <CHILDCOUNT><![CDATA[0]]></CHILDCOUNT>
      </CATEGORY>
      <CATEGORY>
        <NAME><![CDATA[MY WS CAT 2]]></NAME>
        <CATEGORYDESCRIPTION><![CDATA[This is the description of the category]]></
CATEGORYDESCRIPTION>
        <REFERENCE_KEY><![CDATA[WS2]]></REFERENCE_KEY>
        <GUID><![CDATA[0049d401c60108e94a569a007ffd]]></GUID>
        <OBJECTID><![CDATA[478.004]]></OBJECTID>

<PARENTID><![CDATA[004928e9fe0108e4de34d8007fff]]></PARENTID>
        <CHILDCOUNT><![CDATA[0]]></CHILDCOUNT>
      </CATEGORY>
      <CATEGORY>
        <NAME><![CDATA[MY WS CAT 3]]></NAME>
        <CATEGORYDESCRIPTION><![CDATA[This is the description of the category]]></
CATEGORYDESCRIPTION>
        <REFERENCE_KEY><![CDATA[WS3]]></REFERENCE_KEY>
        <GUID><![CDATA[0049d401c60108e94a569a007ffb]]></GUID>
        <OBJECTID><![CDATA[478.005]]></OBJECTID>
        <PARENTID><![CDATA[004928e9fe0108e4de34d8007fff]]></PARENTID>
        <CHILDCOUNT><![CDATA[0]]></CHILDCOUNT>
      </CATEGORY>
    </CATEGORIES>
  </DATA>
</RESPONSE>
```

Content Services Interface

public interface ContentServices

Interface ContentServices

public interface **ContentServices**

This interface defines the set of public methods available for managing Information Manager content.

Each web service requires that a security token be passed as a parameter, and any parameter that requires the security token must pass the entire response generated by the `SecurityService.authenticate()` method, including the encrypted data. See *Information Manager Web Services Authentication* on page 12 for more information.

Information Manager Web Services calls typically use an XML-formatted string as an input parameter. The structure of the XML input varies, depending on the Information Manager Web Services that is being called.

Each web service will return a standardized response that contains relevant data about the transaction as described in *Information Manager Web Services XML Response* on page 13.

Method	Description
createContent on page 35	This method creates a new content record. The security token determines if the user is authorized to create the specified content record. The Boolean flag is used to determine if the record should be published immediately if the channel has a workflow assigned to it. The user must have permission to publish for the channel.
deleteContent on page 39	This method deletes a specified content record from the user's repository. All associated locales and content history for that content record will also be deleted. The user associated with the security token must have permissions and privileges to delete the specified content record or an exception is flagged.

getContentData on page 40	This method retrieves content in a manner similar to the get.content.data tag. The supplied attributes perform the same function as those in the get.content.data tag. The orderings element can contain any column name from the CONTENTTEXTPUB table. This service allows the retrieval of unpublished content by setting the LATEST_VERSION node to TRUE. Note that currently, user group, view and editor group security needs to be passed in by the calling application. This method can also be used to return records by providing a list of content IDs or document IDs. Use one or more CONTENTID nodes to return the records using content IDs; use one or more DOCUMENTID nodes to return the records using document IDs. Only one or the other can be used. If you pass a mix of CONTENTID and DOCUMENTID nodes, only the ones using CONTENTID will be returned. When using this option, only LOCALE_CODE, LATEST_VERSION, and ORDERINGS are valid parameters. All other parameters are ignored.
getContentRecord on page 43	This method returns a content record based on its content ID or document ID and locale. This service looks first for a record ID either as an attribute of the CONTENT node or a node called RECORDID. If no record ID is found, it looks for a DOCUMENTID node. To include staging records, set the LATEST_VERSION node to TRUE (the default behaviour is to return only published records). If the LATEST_VERSION node is set to TRUE the service returns the latest version of the content record in the specified locale. A valid security token and locale code is required. This web service is functionally equivalent to the get.content.record JSP tag.
getContentTemplate on page 46	This method returns an empty content record for the specified channel. The user associated with the security token must have permissions and privileges to access the specified content channel or an exception is flagged. Only the first child node is created for channel attributes that are defined as nodes. Additional nodes can be added as needed.

modifyContent on page 48	This method modifies an existing content record. Some fields cannot be changed and will be ignored by this method (i.e., the resource path, documentid, recordid, and so forth). The user must have permissions to edit content for the specified channel.
removeCaseLink on page 52	This method removes a case number from a content record.

createContent

Method:

java.lang.String createContent(java.lang.String token, java.lang.String inputXml, boolean publish)

Description:

Method to create a new content record. A valid security token is required.

Specified by:	public java.lang.String createContent(java.lang.String token, java.lang.String inputXml, boolean publish)	
Parameters:	**token**	security token
	inputXml	content record XML (see above)
	publish	boolean flag indicating whether the record should be automatically published
Returns:	XML Formatted string containing the web service response (see above)	

Sample Input:

```
<CONTENT RECORDID="0020f259d60108fe459b11007fe8">
    <SERVERTIME><![CDATA[1140035991660]]></SERVERTIME>
    <MASTERIDENTIFIER><![CDATA[test 2]]></MASTERIDENTIFIER>
    <TYPE><![CDATA[BIO]]></TYPE>
    <TYPE_GUID><![CDATA[8010149aafb30fe21075891007e80]]></TYPE_GUID>
    <REPOSITORY><![CDATA[GILTEST]]></REPOSITORY>
    <REPOSITORY_GUID><![CDATA[192-168-1-11-54ea79-f17a0979fe--7ff3]]></
REPOSITORY_GUID>
    <DOCUMENTID><![CDATA[A485]]></DOCUMENTID>
    <LOCALECODE><![CDATA[en_US]]></LOCALECODE>
    <LOCALEVALUE><![CDATA[1033]]></LOCALEVALUE>
    <MAJOR_VERSION><![CDATA[1]]></MAJOR_VERSION>
    <MINOR_VERSION><![CDATA[0]]></MINOR_VERSION>
    <AUTHOR><![CDATA[Super Admin]]></AUTHOR>
    <AUTHORID><![CDATA[1921681235663a2-ed25809901-7f5d]]></AUTHORID>
    <OWNER><![CDATA[Super Admin]]></OWNER>
    <OWNERID><![CDATA[1921681235663a2-ed25809901-7f5d]]></OWNERID>
    <STARTTIMESTAMP><![CDATA[2006-01-24 16:18:00 EST]]></STARTTIMESTAMP>
    <STARTTIMESTAMP_MILLIS><![CDATA[1138137480000]]></
STARTTIMESTAMP_MILLIS>
    <ENDTIMESTAMP><![CDATA[2006-01-28 16:18:00 EST]]></ENDTIMESTAMP>
    <ENDTIMESTAMP_MILLIS><![CDATA[1138483080000]]></ENDTIMESTAMP_MILLIS>
    <LASTMODIFIEDTIMESTAMP><![CDATA[2006-01-24 16:19:04 EST]]></
LASTMODIFIEDTIMESTAMP>
```

```
    <LASTMODIFIEDTIMESTAMP_MILLIS><![CDATA[1138137544000]]></
LASTMODIFIEDTIMESTAMP_MILLIS>
    <METRICS>
       <DEFAULT>
          <VALUE ISNUMBER="Y"><![CDATA[0]]></VALUE>
          <LASTACCESSED><![CDATA[2006-01-24 16:19:01.0]]></LASTACCESSED>
       </DEFAULT>
    </METRICS>
    <RESOURCEPATH><![CDATA[sites/GILTEST/content/live/
0020f259d60108fe459b11007fe8/1033/]]></RESOURCEPATH>
    <PUBLISHEDTIMESTAMP><![CDATA[2006-01-24 16:19:02 EST]]></
PUBLISHEDTIMESTAMP>
    <PUBLISHEDTIMESTAMP_MILLIS><![CDATA[1138137542000]]></
PUBLISHEDTIMESTAMP_MILLIS>
    <CASELINKS REUSECOUNT="1" DOCVALUE="90">
       <CASELINK DATEASSIGNED="2006-02-17 16:13:58.0">
       <CASENUMBER><![CDATA[upstart]]></CASENUMBER>
       <CASEDESCRIPTION><![CDATA[NEW --- Now the description!]]></
CASEDESCRIPTION>
       <CASEINCIDENT><![CDATA[90]]></CASEINCIDENT>
    </CASELINK>
  </CASELINKS>
    <VIEWS>
       <VIEW>
         <NAME><![CDATA[GILTEST]]></NAME>
         <REFERENCE_KEY><![CDATA[GILTEST]]></REFERENCE_KEY>
         <GUID><![CDATA[192-168-1-11-54ea79-f17a0979fe--7ff3]]></GUID>
         <OBJECTID><![CDATA[018]]></OBJECTID>
         <PARENTID></PARENTID>
         <CHILDCOUNT><![CDATA[4]]></CHILDCOUNT>
       </VIEW>
    </VIEWS>
         <CATEGORIES></CATEGORIES>
         <SECURITY>
            <USERGROUP>
               <NAME><![CDATA[EXECUTIVE]]></NAME>
               <REFERENCE_KEY><![CDATA[EXECUTIVE]]></REFERENCE_KEY>
               <GUID><![CDATA[80101466f430fbf03a3ae5007ffc]]></GUID>
               <OBJECTID><![CDATA[193]]></OBJECTID>
               <PARENTID></PARENTID>
               <CHILDCOUNT><![CDATA[0]]></CHILDCOUNT>
            </USERGROUP>
         </SECURITY>
         <EDITORGROUPS></EDITORGROUPS>
               ... content channel data goes here
     </CONTENT>
   </DATA>
</RESPONSE>
```

Sample Response:

```
<RESPONSE>
<STATUS>
<SERVERTIME><![CDATA[1140035992748]]></SERVERTIME>
<TYPE><![CDATA[getContentByContentId]]></TYPE>
<RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
<DATA_VERSION><![CDATA[1]]></DATA_VERSION>
<VALUE><![CDATA[1]]></VALUE>
<ERRORS></ERRORS>
</STATUS>
<DATA> <CONTENT RECORDID="0020f259d60108fe459b11007fe8">
<SERVERTIME><![CDATA[1140035991660]]></SERVERTIME>
<MASTERIDENTIFIER><![CDATA[test 2]]></MASTERIDENTIFIER>
<TYPE><![CDATA[BIO]]></TYPE>
<TYPE_GUID><![CDATA[8010149aafb30fe21075891007e80]]></TYPE_GUID>
<REPOSITORY><![CDATA[GILTEST]]></REPOSITORY>
<REPOSITORY_GUID><![CDATA[192-168-1-11-54ea79-f17a0979fe--7ff3]]></
REPOSITORY_GUID>
<DOCUMENTID><![CDATA[A485]]></DOCUMENTID>
<LOCALECODE><![CDATA[en_US]]></LOCALECODE>
<LOCALEVALUE><![CDATA[1033]]></LOCALEVALUE>
<MAJOR_VERSION><![CDATA[1]]></MAJOR_VERSION>
<MINOR_VERSION><![CDATA[0]]></MINOR_VERSION>
<AUTHOR><![CDATA[Super Admin]]></AUTHOR>
<AUTHORID><![CDATA[1921681235663a2-ed25809901-7f5d]]></AUTHORID>
<OWNER><![CDATA[Super Admin]]></OWNER>
<OWNERID><![CDATA[1921681235663a2-ed25809901-7f5d]]></OWNERID>
<STARTTIMESTAMP><![CDATA[2006-01-24 16:18:00 EST]]></STARTTIMESTAMP>
<STARTTIMESTAMP_MILLIS><![CDATA[1138137480000]]></STARTTIMESTAMP_MILLIS>
<ENDTIMESTAMP><![CDATA[2006-01-28 16:18:00 EST]]></ENDTIMESTAMP>
<ENDTIMESTAMP_MILLIS><![CDATA[1138483080000]]></ENDTIMESTAMP_MILLIS>
<LASTMODIFIEDTIMESTAMP><![CDATA[2006-01-24 16:19:04 EST]]></
LASTMODIFIEDTIMESTAMP>
<LASTMODIFIEDTIMESTAMP_MILLIS><![CDATA[1138137544000]]></
LASTMODIFIEDTIMESTAMP_MILLIS>
<METRICS>
<DEFAULT>
<VALUE ISNUMBER="Y"><![CDATA[0]]></VALUE>
<LASTACCESSED><![CDATA[2006-01-24 16:19:01.0]]></LASTACCESSED>
</DEFAULT>
</METRICS>
<RESOURCEPATH><![CDATA[sites/GILTEST/content/live/0020f259d60108fe459b11007fe8/
1033/]]></RESOURCEPATH>
<PUBLISHEDTIMESTAMP><![CDATA[2006-01-24 16:19:02 EST]]></
PUBLISHEDTIMESTAMP>
<PUBLISHEDTIMESTAMP_MILLIS><![CDATA[1138137542000]]></
PUBLISHEDTIMESTAMP_MILLIS>
```

```
<VIEWS>
<VIEW>
<NAME><![CDATA[GILTEST]]></NAME>
<REFERENCE_KEY><![CDATA[GILTEST]]></REFERENCE_KEY>
<GUID><![CDATA[192-168-1-11-54ea79-f17a0979fe--7ff3]]></GUID>
<OBJECTID><![CDATA[018]]></OBJECTID>
<PARENTID></PARENTID>
<CHILDCOUNT><![CDATA[4]]></CHILDCOUNT>
</VIEW>
</VIEWS>
<CATEGORIES></CATEGORIES>
<SECURITY>
<USERGROUP>
<NAME><![CDATA[EXECUTIVE]]></NAME>
<REFERENCE_KEY><![CDATA[EXECUTIVE]]></REFERENCE_KEY>
<GUID><![CDATA[80101466f430fbf03a3ae5007ffc]]></GUID>
<OBJECTID><![CDATA[193]]></OBJECTID>
<PARENTID></PARENTID>
<CHILDCOUNT><![CDATA[0]]></CHILDCOUNT>
</USERGROUP>
</SECURITY>
<EDITORGROUPS></EDITORGROUPS>
... content channel data goes here
</CONTENT>
</DATA>
</RESPONSE>
```

deleteContent

Method:

public java.lang.String deleteContent(java.lang.String token, java.lang.String contentid)

Description:

Method that deletes a specified content record from the user's repository - all associated locales and content history for that content record will also be deleted. The user associated with the security token must have permissions and privileges to delete the specified content record or an exception is flagged.

Specified by:	deleteContent in interface com.inquira.imws.interfaces.ContentServices	
Parameters:	**token**	security token
	contentid	content record to delete
Returns:	XML Formatted string containing response structure (see Sample Response)	

Sample Response:

```
<RESPONSE>
   <STATUS>
      <SERVERTIME><![CDATA[1139414879065]]></SERVERTIME>
      <TYPE><![CDATA[deleteContent]]></TYPE>
      <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
      <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
      <VALUE><![CDATA[1]]></VALUE>
      <ERRORS/>
    </STATUS>
    <DATA>
    </DATA>
</RESPONSE>
```

getContentData

Method:

java.lang.String getContentData(java.lang.String token, java.lang.String inputXml)

getContentData in interface com.inquira.imws.interfaces.ContentServices

Description:

Retrieve a content record by recordid

This method retrieves content in a manner similar to the get.content.data tag. the supplied attributes perform the same function as their conterparts in the get.content.data tag. This service allows the retrieval of unpublish content, by setting the LATEST_VERSION node to true. currently user group, view and editor group security needs to be passed in by the calling application. This method also can be used to return records providing a list of content ids or document ids. Use one or more CONTENTID nodes to return the records using content ids. Use one or more DOCUMENTID nodes to return the records using document ids. Only one or the other can be used. If you pass a mix of CONTENTID and DOCUMENTID nodes, only the ones using CONTENTID will be returned. When using this option, all other parameters are ignored only LOCALE_CODE, LATEST_VERSION, and ORDERINGS can be used. All other will be ignored.

Parameters:	**token**	Security token
	inputXml	XML formatted string containing input similar to the one shown in the sample input.
Returns:	XML Formatted string containing the web service response (see Sample Response)	

Sample Input:

```
<CONTENT>
   <CHANNEL_REFERENCE_KEY>NEWS</CHANNEL_REFERENCE_KEY>
   <LOCALE_CODE>en_US</LOCALE_CODE>
   <CONTENTID></CONTENTID>
   <DOCUMENTID>en_US</DOCUMENTID>
   <IGNORE_DISPLAY_DATES>true</IGNORE_DISPLAY_DATES>
   <UPDATED_SINCE/>
   <HIERARCHICAL_CATEGORIES/>
   <MAX_RECORDS>0</MAX_RECORDS>
   <DISPLAY_START_DATE></DISPLAY_START_DATE>
   <DISPLAY_END_DATE></DISPLAY_END_DATE>
   <MODE>validitydate</MODE>
   <LATEST_VERSION>true</LATEST_VERSION>
```

```
    <VIEWS>
      <VIEW>
        <REFERENCE_KEY></REFERENCE_KEY>
      </VIEW>
    </VIEWS>
    <CATEGORIES>
      <CATEGORY>
        <REFERENCE_KEY></REFERENCE_KEY>
      </CATEGORY>
    </CATEGORIES>
    <CASELINKS>
      <CASELINK>
        <CASENUMBER></CASENUMBER>
      </CASELINK>
    </CASELINKS>
    <SECURITY>
      <USERGROUP>
        <REFERENCE_KEY></REFERENCE_KEY>
      </USERGROUP>
    </SECURITY>
    <EDITORGROUPS>
      <EDITORGROUP>
        <REFERENCE_KEY/>
      </EDITORGROUP>
    </EDITORGROUPS>
    <ORDERINGS DIRECTION="ascending">
      <COLUMN>INDEXMASTERIDENTIFIERS</COLUMN>
    </ORDERINGS>
  </CONTENT>
```

Sample Response:

```
  <RESPONSE>
    <STATUS>
      <SERVERTIME><![CDATA[1147987960910]]></SERVERTIME>
      <TYPE><![CDATA[getContentData]]></TYPE>
      <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
      <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
      <VALUE><![CDATA[1]]></VALUE>
      <ERRORS/>
    </STATUS>
```

```
        <DATA>
          <CONTENTS COUNT="2">
            <CONTENT>
              <MASTERIDENTIFIER><![CDATA[Article one]]></MASTERIDENTIFIER>
              <RECORDID><![CDATA[00100bb2bc3010b012f4e3c007f56]]></RECORDID>
              <DOCUMENTID><![CDATA[NE44]]></DOCUMENTID>
              <AUTHOR><![CDATA[Joe Smith]]></AUTHOR>
              <AUTHORID><![CDATA[1921681235663a2-ed25809901-7f5d]]></AUTHORID>
              <OWNER><![CDATA[Joe Smith]]></OWNER>
              <OWNERID><![CDATA[1921681235663a2-ed25809901-7f5d]]></OWNERID>
              <PUBLISHED><![CDATA[false]]></PUBLISHED>
              <MAJOR_VERSION><![CDATA[0]]></MAJOR_VERSION>
              <MINOR_VERSION><![CDATA[2]]></MINOR_VERSION>
              <STARTTIMESTAMP><![CDATA[2006-05-09 08:56:00 EDT]]></
STARTTIMESTAMP>
              <STARTTIMESTAMP_MILLIS><![CDATA[1147179360000]]></
STARTTIMESTAMP_MILLIS>
              <ENDTIMESTAMP/>
              <ENDTIMESTAMP_MILLIS/>
              <LASTMODIFIEDTIMESTAMP><![CDATA[2006-05-17 13:11:23 EDT]]></
LASTMODIFIEDTIMESTAMP>
              <LASTMODIFIEDTIMESTAMP_MILLIS><![CDATA[1147885883000]]></
LASTMODIFIEDTIMESTAMP_MILLIS>
            </CONTENT>
            <CONTENT>
              <MASTERIDENTIFIER><![CDATA[Article two]]></MASTERIDENTIFIER>
              <RECORDID><![CDATA[0025faf4ec010acd24f90d007ff4]]></RECORDID>
              <DOCUMENTID><![CDATA[NE38]]></DOCUMENTID>
              <AUTHOR><![CDATA[Rafael Rodriguez]]></AUTHOR>
              <AUTHORID><![CDATA[00270980a0a0010a948d93f50072ad]]></AUTHORID>
              <OWNER><![CDATA[Rafael Rodriguez]]></OWNER>
              <OWNERID><![CDATA[00270980a0a0010a948d93f50072ad]]></OWNERID>
              <PUBLISHED><![CDATA[false]]></PUBLISHED>
              <MAJOR_VERSION><![CDATA[0]]></MAJOR_VERSION>
              <MINOR_VERSION><![CDATA[1]]></MINOR_VERSION>
              <STARTTIMESTAMP><![CDATA[2006-04-24 13:19:28 EDT]]></
STARTTIMESTAMP>
              <STARTTIMESTAMP_MILLIS><![CDATA[1145899168000]]></
STARTTIMESTAMP_MILLIS>
              <ENDTIMESTAMP/>
              <ENDTIMESTAMP_MILLIS/>
              <LASTMODIFIEDTIMESTAMP><![CDATA[2006-04-24 14:29:28 EDT]]></
LASTMODIFIEDTIMESTAMP>
              <LASTMODIFIEDTIMESTAMP_MILLIS><![CDATA[1145903368000]]></
LASTMODIFIEDTIMESTAMP_MILLIS>
            </CONTENT>
          </CONTENTS>
        </DATA>
      </RESPONSE>
```

getContentRecord

Method:

public java.lang.String getContentRecord(java.lang.String token, java.lang.String inputXml)

getContentRecord in interface com.inquira.imws.interfaces.ContentServices

Description:

Method to return a content record given its contentid or document id and locale. This service will first look for a record id either as an attribute of the CONTENT node or a node called RECORDID. If no record id found then it will look for a DOCUMENTID node. If staging records are desired, set the LATEST_VERSION node to true (the default behaviour is to return only published records) By setting the LATEST_VERSION node to true the service will return the latest version of the desired content record in the specified locale. A valid security token and locale code is required.

Parameters:	**token**	security token
	inputXml	XML formatted string containing input similar to the one shown in the sample input.
Returns:	XML Formatted string containing the web service response (see Sample Response)	

Sample Input:

```
<CONTENT RECORDID="">
   <RECORDID></RECORDID>
   <DOCUMENTID>NE30</DOCUMENTID>
   <LOCALECODE>en_US</LOCALECODE>
   <LATEST_VERSION>false</LATEST_VERSION>
   <INCREASEVIEWCOUNT>false</INCREASEVIEWCOUNT>
</CONTENT>
```

Sample Response:

```
<RESPONSE>
  <STATUS>
    <SERVERTIME><![CDATA[1147989111940]]></SERVERTIME>
    <TYPE><![CDATA[getContentRecord]]></TYPE>
    <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
    <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
    <VALUE><![CDATA[1]]></VALUE>
    <ERRORS/>
  </STATUS>
  <DATA>
    <CONTENT RECORDID="0028ec44cb010a94c8eed5007fed">
      <SERVERTIME_MILLIS><![CDATA[1147989111729]]></SERVERTIME_MILLIS>
      <MASTERIDENTIFIER><![CDATA[This is  a new content record]]></
MASTERIDENTIFIER>
      <TYPE><![CDATA[NEWS]]></TYPE>
      <TYPE_GUID><![CDATA[00270980a0a0010a948d93f500702d]]></TYPE_GUID>
      <REPOSITORY><![CDATA[GILTEST]]></REPOSITORY>
      <REPOSITORY_GUID><![CDATA[00270980a0a0010a948d93f500731f]]></
REPOSITORY_GUID>
      <DOCUMENTID><![CDATA[NE21]]></DOCUMENTID>
      <LOCALECODE><![CDATA[en_US]]></LOCALECODE>
      <LOCALEVALUE><![CDATA[1033]]></LOCALEVALUE>
      <MAJOR_VERSION><![CDATA[5]]></MAJOR_VERSION>
      <MINOR_VERSION><![CDATA[4]]></MINOR_VERSION>
      <PUBLISHED_MAJOR_VERSION><![CDATA[5]]></PUBLISHED_MAJOR_VERSION>
      <PUBLISHED_MINOR_VERSION><![CDATA[0]]></PUBLISHED_MINOR_VERSION>
      <AUTHOR><![CDATA[Joe Smith]]></AUTHOR>
      <AUTHORID><![CDATA[1921681235663a2-ed25809901-7f5d]]></AUTHORID>
      <OWNER><![CDATA[Jimmy Hendrix]]></OWNER>
      <OWNERID><![CDATA[00270980a0a0010a948d93f50072ab]]></OWNERID>
      <PUBLISHED><![CDATA[false]]></PUBLISHED>
      <WORKFLOW_STEP_NAME/>
      <STARTTIMESTAMP><![CDATA[2006-04-13 15:47:00 EDT]]></STARTTIMESTAMP>
      <STARTTIMESTAMP_MILLIS><![CDATA[1144957620000]]></
STARTTIMESTAMP_MILLIS>
      <ENDTIMESTAMP/>
      <ENDTIMESTAMP_MILLIS/>
      <LASTMODIFIEDTIMESTAMP><![CDATA[2006-04-13 16:07:05 EDT]]></
LASTMODIFIEDTIMESTAMP>
      <LASTMODIFIEDTIMESTAMP_MILLIS><![CDATA[1144958825000]]></
LASTMODIFIEDTIMESTAMP_MILLIS>
```

```
        <METRICS>
            <DEFAULT>
                <VALUE ISNUMBER="Y"><![CDATA[0]]></VALUE>
                <LASTACCESSED><![CDATA[2006-04-13 15:48:05.0]]></LASTACCESSED>
            </DEFAULT>
        </METRICS>
             <RESOURCEPATH><![CDATA[sites/GILTEST/content/live/
0028ec44cb010a94c8eed5007fed/1033/]]></RESOURCEPATH>
        <PUBLISHEDTIMESTAMP/>
        <PUBLISHEDTIMESTAMP_MILLIS/>
          <VIEWS>
            <VIEW>
              <NAME><![CDATA[GILTEST]]></NAME>
              <REFERENCE_KEY><![CDATA[GILTEST]]></REFERENCE_KEY>
              <GUID><![CDATA[00270980a0a0010a948d93f500731f]]></GUID>
              <OBJECTID><![CDATA[002]]></OBJECTID>
              <PARENTID/>
              <CHILDCOUNT><![CDATA[3]]></CHILDCOUNT>
            </VIEW>
          </VIEWS>
          <CATEGORIES/>
          <SECURITY/>
          <EDITORGROUPS/>
          <NEWS>
             <TITLE><![CDATA[This is  a new content record]]></TITLE>
             <BODY><![CDATA[asdfasdfasdfasdf sdfgsdfg]]></BODY>
          </NEWS>
        </CONTENT>
      </DATA>
    </RESPONSE>
```

getContentTemplate

Method:

java.lang.String getContentTemplate(java.lang.String token, java.lang.String channelRefKey)

Description:

Returns a blank content record for the specified channel. The user associated with the security token must have permissions and privileges to delete access the specified content channel or an exception is flagged. Only the first child node is created for channel attributes that are defined as nodes. Additional nodes can be added as needed.

Specified by:	getContentTemplate in interface com.inquira.imws.interfaces.ContentServices	
Parameters:	**token**	security token
	channelRefKey	channel reference key to get template for
Returns:	XML formatted string containing response structure (see Sample Response)	

Sample Response:

```
<RESPONSE>
  <STATUS>
    <SERVERTIME><![CDATA[1140033014845]]></SERVERTIME>
    <TYPE><![CDATA[getContentTemplate]]></TYPE>
    <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
    <DATA_VERSION><![CDATA[1]]></DATA_VERSION
    <VALUE><![CDATA[1]]></VALUE
    <ERRORS></ERRORS>
  </STATUS>
    <DATA>
    <CONTENT>
      <TYPE><![CDATA[BIO]]></TYPE>

<TYPE_GUID><![CDATA[8010149aafb30fe21075891007e80]]></TYPE_GUID>
          <REPOSITORY><![CDATA[GILTEST]]></REPOSITORY>
          <REPOSITORY_GUID><![CDATA[192-168-1-11-54ea79-f17a0979fe--7ff3]]></
REPOSITORY_GUID>
         <STARTTIMESTAMP><![CDATA[2006-02-15 14:50:12 EST]]></STARTTIMESTAMP>
          <STARTTIMESTAMP_MILLIS><![CDATA[1140033012735]]></
STARTTIMESTAMP_MILLIS>
          <ENDTIMESTAMP><![CDATA[2006-02-19 14:50:12 EST]]></ENDTIMESTAMP>
          <ENDTIMESTAMP_MILLIS><![CDATA[1140378612736]]></
ENDTIMESTAMP_MILLIS>
          <VIEWS><VIEW><REFERENCE_KEY><![CDATA[]]></REFERENCE_KEY>
          <GUID><![CDATA[]]></GUID>
    </VIEW></VIEWS>
```

```
          <CATEGORIES>
            <CATEGORY>
              <REFERENCE_KEY><![CDATA[]]></REFERENCE_KEY>
              <GUID><![CDATA[]]></GUID>
           </CATEGORY>
          </CATEGORIES>
        <SECURITY>
          <USERGROUP>
            <REFERENCE_KEY><![CDATA[]]></REFERENCE_KEY>
            <GUID><![CDATA[]]></GUID>
          </USERGROUP>
        </SECURITY>
          <NEWS>
          ... specific channel attributes available here
          </NEWS>
        </CONTENT>
      </DATA>
</RESPONSE>
```

modifyContent

Method:

public java.lang.String modifyContent(java.lang.String token, java.lang.String inputXml, boolean publish)

modifyContent in interface com.inquira.imws.interfaces.ContentServices

Description:

Method to modify an existing content record. A valid security token is required.

Parameters:	**token**	Security token
	inputXml	content record XML (see Sample Input)
	publish	boolean flag indicating whether the record should be automatically published
Returns:	XML Formatted string containing the web service response (see Sample Response)	

Sample Input:

```
<CONTENT RECORDID="0020f259d60108fe459b11007fe8">
   <SERVERTIME><![CDATA[1140035991660]]></SERVERTIME>
   <MASTERIDENTIFIER><![CDATA[test 2]]></MASTERIDENTIFIER>
   <TYPE><![CDATA[BIO]]></TYPE>
   <TYPE_GUID><![CDATA[8010149aafb30fe21075891007e80]]></TYPE_GUID>
   <REPOSITORY><![CDATA[GILTEST]]></REPOSITORY>
   <REPOSITORY_GUID><![CDATA[192-168-1-11-54ea79-f17a0979fe--7ff3]]></
REPOSITORY_GUID>
   <DOCUMENTID><![CDATA[A485]]></DOCUMENTID>
   <LOCALECODE><![CDATA[en_US]]></LOCALECODE>
   <LOCALEVALUE><![CDATA[1033]]></LOCALEVALUE>
   <MAJOR_VERSION><![CDATA[1]]></MAJOR_VERSION>
   <MINOR_VERSION><![CDATA[0]]></MINOR_VERSION>
   <AUTHOR><![CDATA[Super Admin]]></AUTHOR>
   <AUTHORID><![CDATA[1921681235663a2-ed25809901-7f5d]]></AUTHORID>
   <OWNER><![CDATA[Super Admin]]></OWNER>
   <OWNERID><![CDATA[1921681235663a2-ed25809901-7f5d]]></OWNERID>
   <STARTTIMESTAMP><![CDATA[2006-01-24 16:18:00 EST]]></STARTTIMESTAMP>
   <STARTTIMESTAMP_MILLIS><![CDATA[1138137480000]]></STARTTIMESTAMP_MILLIS>
   <ENDTIMESTAMP><![CDATA[2006-01-28 16:18:00 EST]]></ENDTIMESTAMP>
   <ENDTIMESTAMP_MILLIS><![CDATA[1138483080000]]></ENDTIMESTAMP_MILLIS>
   <LASTMODIFIEDTIMESTAMP><![CDATA[2006-01-24 16:19:04 EST]]></
LASTMODIFIEDTIMESTAMP>
   <LASTMODIFIEDTIMESTAMP_MILLIS><![CDATA[1138137544000]]></
LASTMODIFIEDTIMESTAMP_MILLIS>
```

```
<METRICS>
   <DEFAULT>
<VALUE ISNUMBER="Y"><![CDATA[0]]></VALUE>
<LASTACCESSED><![CDATA[2006-01-24 16:19:01.0]]></LASTACCESSED>
     </DEFAULT>
   </METRICS>
   <RESOURCEPATH><![CDATA[sites/GILTEST/content/live/0020f259d60108fe459b11007fe8/
1033/]]></RESOURCEPATH>
   <PUBLISHEDTIMESTAMP><![CDATA[2006-01-24 16:19:02 EST]]></
PUBLISHEDTIMESTAMP>
   <PUBLISHEDTIMESTAMP_MILLIS><![CDATA[1138137542000]]></
PUBLISHEDTIMESTAMP_MILLIS>
   <CASELINKS REUSECOUNT="1" DOCVALUE="90">
   <CASELINK DATEASSIGNED="2006-02-17 16:13:58.0">
   <CASENUMBER><![CDATA[upstart]]></CASENUMBER>
   <CASEDESCRIPTION><![CDATA[NEW --- Now the description!]]></CASEDESCRIPTION>
   <CASEINCIDENT><![CDATA[90]]></CASEINCIDENT>
   </CASELINK>
   </CASELINKS>
      <VIEWS>
         <VIEW>
            <NAME><![CDATA[GILTEST]]></NAME>
            <REFERENCE_KEY><![CDATA[GILTEST]]></REFERENCE_KEY>
            <GUID><![CDATA[192-168-1-11-54ea79-f17a0979fe--7ff3]]></GUID>
            <OBJECTID><![CDATA[018]]></OBJECTID>
            <PARENTID></PARENTID>
            <CHILDCOUNT><![CDATA[4]]></CHILDCOUNT>
         </VIEW>
      </VIEWS>
      <CATEGORIES></CATEGORIES>

 <SECURITY>
         <USERGROUP>
            <NAME><![CDATA[EXECUTIVE]]></NAME>
            <REFERENCE_KEY><![CDATA[EXECUTIVE]]></REFERENCE_KEY>
            <GUID><![CDATA[80101466f430fbf03a3ae5007ffc]]></GUID>
            <OBJECTID><![CDATA[193]]></OBJECTID>
            <PARENTID></PARENTID>
            <CHILDCOUNT><![CDATA[0]]></CHILDCOUNT>
         </USERGROUP>
      </SECURITY>
      <EDITORGROUPS></EDITORGROUPS>
            ... content channel data goes here
   </CONTENT>
```

Sample Response:

```
<RESPONSE>
<STATUS>
   <SERVERTIME><![CDATA[1140035992748]]></SERVERTIME>
   <TYPE><![CDATA[getContentByContentId]]></TYPE>
   <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
   <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
   <VALUE><![CDATA[1]]></VALUE>
   <ERRORS></ERRORS>
</STATUS>
<DATA>
   <CONTENT RECORDID="0020f259d60108fe459b11007fe8">
        <SERVERTIME><![CDATA[1140035991660]]></SERVERTIME>
        <MASTERIDENTIFIER><![CDATA[test 2]]></MASTERIDENTIFIER>
        <TYPE><![CDATA[BIO]]></TYPE>
        <TYPE_GUID><![CDATA[8010149aafb30fe21075891007e80]]></TYPE_GUID>
        <REPOSITORY><![CDATA[GILTEST]]></REPOSITORY>
        <REPOSITORY_GUID><![CDATA[192-168-1-11-54ea79-f17a0979fe--7ff3]]></
REPOSITORY_GUID>
        <DOCUMENTID><![CDATA[A485]]></DOCUMENTID>
        <LOCALECODE><![CDATA[en_US]]></LOCALECODE>
        <LOCALEVALUE><![CDATA[1033]]></LOCALEVALUE>
        <MAJOR_VERSION><![CDATA[1]]></MAJOR_VERSION>
        <MINOR_VERSION><![CDATA[0]]></MINOR_VERSION>
        <AUTHOR><![CDATA[Super Admin]]></AUTHOR>
        <AUTHORID><![CDATA[1921681235663a2-ed25809901-7f5d]]></AUTHORID>
        <OWNER><![CDATA[Super Admin]]></OWNER>
        <OWNERID><![CDATA[1921681235663a2-ed25809901-7f5d]]></OWNERID>
        <STARTTIMESTAMP><![CDATA[2006-01-24 16:18:00 EST]]></STARTTIMESTAMP>

   <STARTTIMESTAMP_MILLIS><![CDATA[1138137480000]]></STARTTIMESTAMP_MILLIS>
        <ENDTIMESTAMP><![CDATA[2006-01-28 16:18:00 EST]]></ENDTIMESTAMP>
        <ENDTIMESTAMP_MILLIS><![CDATA[1138483080000]]></ENDTIMESTAMP_MILLIS>
        <LASTMODIFIEDTIMESTAMP><![CDATA[2006-01-24 16:19:04 EST]]></
LASTMODIFIEDTIMESTAMP>
        <LASTMODIFIEDTIMESTAMP_MILLIS><![CDATA[1138137544000]]></
LASTMODIFIEDTIMESTAMP_MILLIS>
```

```
        <METRICS>
            <DEFAULT>
                <VALUE ISNUMBER="Y"><![CDATA[0]]></VALUE>
                <LASTACCESSED><![CDATA[2006-01-24 16:19:01.0]]></LASTACCESSED>
            </DEFAULT>
        </METRICS>
        <RESOURCEPATH><![CDATA[sites/GILTEST/content/live/
0020f259d60108fe459b11007fe8/1033/]]></RESOURCEPATH>
        <PUBLISHEDTIMESTAMP><![CDATA[2006-01-24 16:19:02 EST]]></
PUBLISHEDTIMESTAMP>
        <PUBLISHEDTIMESTAMP_MILLIS><![CDATA[1138137542000]]></
PUBLISHEDTIMESTAMP_MILLIS>
        <CASELINKS REUSECOUNT="1" DOCVALUE="90">
          <CASELINK DATEASSIGNED="2006-02-17 16:13:58.0">
            <CASENUMBER><![CDATA[upstart]]></CASENUMBER>
            <CASEDESCRIPTION><![CDATA[NEW --- Now the description!]]></
CASEDESCRIPTION>
            <CASEINCIDENT><![CDATA[90]]></CASEINCIDENT>
          </CASELINK>
        </CASELINKS>
            <VIEWS>
                <VIEW>
                    <NAME><![CDATA[GILTEST]]></NAME>
                    <REFERENCE_KEY><![CDATA[GILTEST]]></REFERENCE_KEY>
                    <GUID><![CDATA[192-168-1-11-54ea79-f17a0979fe--7ff3]]></GUID>
                    <OBJECTID><![CDATA[018]]></OBJECTID>
                    <PARENTID></PARENTID>
                    <CHILDCOUNT><![CDATA[4]]></CHILDCOUNT>
                </VIEW>
            </VIEWS>
            <CATEGORIES></CATEGORIES>
            <SECURITY>
                <USERGROUP>
                    <NAME><![CDATA[EXECUTIVE]]></NAME>
                    <REFERENCE_KEY><![CDATA[EXECUTIVE]]></REFERENCE_KEY>
                    <GUID><![CDATA[80101466f430fbf03a3ae5007ffc]]></GUID>
                    <OBJECTID><![CDATA[193]]></OBJECTID>
                    <PARENTID></PARENTID>
                    <CHILDCOUNT><![CDATA[0]]></CHILDCOUNT>
                </USERGROUP>
            </SECURITY>
            <EDITORGROUPS></EDITORGROUPS>
                    ... content channel data goes here
        </CONTENT>
    </DATA>
</RESPONSE>
```

removeCaseLink

Method:

java.lang.String removeCaseLink(java.lang.String token, java.lang.String inXml)

removeCaseLink in interface com.inquira.imws.interfaces.ContentServices

Description:

Method to remove a case number from a content record.

Parameters:	**token**	Security token
	inputXml	content record XML (see Sample Input)
Returns:	XML Formatted string containing the web service response (see Sample Response)	

Sample Input:

```
<CASELINK>
      <CONTENTID>ABC-123-DEF-456</CONTENTID>
      <DOCUMENTID>SOL123</DOCUMENTID>
      <CASENUMER>100</CASENUMER>
</CASELINK>
```

Sample Response:

```
<RESPONSE>
 <STATUS>
    <SERVERTIME><![CDATA[1184367277237]]></SERVERTIME>
    <TYPE/>
    <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
    <DATA_VERSION/>
    <VALUE><![CDATA[1]]></VALUE>
    <ERRORS/>
 </STATUS>

 <DATA>
    <CASELINKS>
    <CASELINK>
        <CASENUMBER><![CDATA[1]]></CASENUMBER>
        <CASEDESCRIPTION><![CDATA[Case 1]]></CASEDESCRIPTION>
    </CASELINK>
    <CASELINK>
        <CASENUMBER><![CDATA[2]]></CASENUMBER>
        <CASEDESCRIPTION><![CDATA[Case 2]]></CASEDESCRIPTION>
    </CASELINK>
```

```
    <CASELINK>
        <CASENUMBER><![CDATA[3]]></CASENUMBER>
        <CASEDESCRIPTION><![CDATA[Case 3]]></CASEDESCRIPTION>
     </CASELINK>
     </CASELINKS>
</DATA>
</RESPONSE>
```

User Services Interface

com.inquira.imws.interfaces

Interface UserServices

public interface **UserServices**

This interface defines the set of public methods available for managing user information in an Information Manager repository.

Each web service requires that a security token be passed as a parameter, and any parameter that requires the security token must pass the entire response generated by the `SecurityService.authenticate()` method, including the encrypted data. See *Information Manager Web Services Authentication* on page 12 for more information.

Information Manager Web Services calls typically use an XML-formatted string as an input parameter. The structure of the XML input varies, depending on the Information Manager Web Services that is being called.

Each web service will return a standardized response that contains relevant data about the transaction as described in *Information Manager Web Services XML Response* on page 13.

Method	Description
getUser on page 54	This method obtains data for single user based on a specified GUID and locale code.
getUsersForCategory on page 55	This method returns all users who have been assigned the specified content categories associated with their user profile (skills).
getUsersForEmail on page 56	This method returns the details for a user matching a specified email address.
getUsersForLocale on page 57	This method returns all users for a specified locale code.
The getUsersForEmail Method on page 58	This method returns the details for a user matching a specified login name.

getUser

Method:

java.lang.String getUser(java.lang.String token, java.lang.String inXml)

Description:

This method is a simple call that will get one User based on a guid and a locale code.

Specified by:	com.inquira.imws.interfaces.UserServices	
Parameters:	**token**	security token
	inputXml	XML formatted input containing the required data. See Sample Response below.
Returns:	An XML-formatted string containing the response structure. See Sample Response below.	

Sample Input:

```
<USER>
    <GUID>002830e3167001094073 76f0007da3</GUID>
    <LOCALE_CODE>en_US</LOCALE_CODE>
</USER>
```

Sample Response:

```
<RESPONSE>
  <tbd>
</RESPONSE>
```

getUsersForCategory

Method:

java.lang.String getUsersForCategory(java.lang.String token, java.lang.String inXml)

Description:

This method will return all users who have been assigned a category that matches the CATEGORY's reference key and is in the current site.

Specified by:	com.inquira.imws.interfaces.UserServices	
Parameters:	**token**	security token
	inputXml	XML formatted input containing the required data. See Sample Input below.
Returns:	An XML-formatted string containing the response structure. See Sample Response below.	

Sample Input:

```
<USER>
   <CATEGORY>
      ARTISTS
   </CATEGORY>
</USER>
```

Sample Response:

```
<RESPONSE>
   <tbd>
</RESPONSE>
```

getUsersForEmail

Method:

java.lang.String getUsersForEmail(java.lang.String token, java.lang.String inXml)

Description:

This method returns the user with the matching email and is in the current site.

Specified by:	com.inquira.imws.interfaces.UserServices	
Parameters:	**token**	security token
	inputXml	XML formatted input containing the required data. See Sample Input below.
Returns:	An XML-formatted string containing the response structure. See Sample Response below.	

Sample Input:

```
<USER>
    <EMAIL>someone@somehost.com</EMAIL>
    <LOCALE_CODE>en_US</LOCALE_CODE>
</USER>
```

Sample Response:

```
<RESPONSE>
   <tbd>
</RESPONSE>
```

getUsersForLocale

Method:

public java.lang.String **getUsersForLocale**+

Description:

This method will return all users based on the locale code input and the current site

Specified by:	com.inquira.imws.interfaces.UserServices	
Parameters:	**token**	security token
	inputXml	XML formatted input containing the required data. See Sample Input below.
Returns:	An XML-formatted string containing the response structure. See Sample Response below.	

Sample Input:

```
<USER>
   <LOCALE_CODE>en_US</LOCALE_CODE>
</USER>
```

Sample Response:

```
<RESPONSE>
   <tbd>
</RESPONSE>
```

The getUsersForEmail Method

Method:

java.lang.String getUsersForLogin(java.lang.String token, java.lang.String inXml)

Description:

This method returns the user with the matching login and is in the current site.

Specified by:	com.inquira.imws.interfaces.UserServices	
Parameters:	**token**	security token
	inputXml	XML formatted input containing the required data. See Sample Input below.
Returns:	An XML-formatted string containing the response structure. See Sample Response below.	

Sample Input:

```
<USER>
    <LOGIN>username</LOGIN>
    <LOCALE_CODE>en_US</LOCALE_CODE>
</USER>
```

Sample Response:

```
<RESPONSE>
   <tbd>
</RESPONSE>
```

Channel Services Interface

com.inquira.imws.interfaces

Interface ChannelServices

public interface **ChannelServices**

This interface defines the methods available for managing Information Manager content channels.

Each web service requires that a security token is passed in as a parameter. The security token can be obtained from the IM Security Service using the authenticate() method.

Information Manager Web Services calls typically use an XML-formatted string as an input parameter. The structure of the XML input varies, depending on the Information Manager Web Services that is being called.

Each web service will return a standardized response that contains relevant data about the transaction as described in Information Manager Web Services XML Response.

Method	Description
getChannel on page 60	This method returns the attribute details for a specific content channel. If the locale is not specified the repository default locale is used to return the channel attributes.
getChannels on page 64	This method returns a list of all of the channels for the repository, including a list of channel attributes and their properties (i.e., required, display type, and so forth). If the locale is not specified the repository default locale is used to return the channel attributes.

getChannel

Method:

java.lang.String getChannel(java.lang.String token, java.lang.String referenceKey, java.lang.String localeCode)

Description:

This method returns a Content Channel for the passed in reference key.

Specified by:	com.inquira.imws.interfaces.UserServices	
Parameters:	**token**	security token.
	referenceKey	Content Channel reference key.
	localeCode	The locale code for the requested records, i.e. en_US. The locale code is used for the display names of the attributes. If no locale code is passed in, the locale used will be the repository's default locale. See Sample Response below.
Returns:	An XML-formatted string containing the response structure. See Sample Response below.	

Sample Input:

Sample Response:

```
<RESPONSE>
  <STATUS>
    <SERVERTIME><![CDATA[1145633573669]]></SERVERTIME>
    <TYPE><![CDATA[getChannels]]></TYPE>
    <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
    <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
    <VALUE><![CDATA[1]]></VALUE>
    <ERRORS/>
  </STATUS>
  <DATA>
    <CHANNEL>
      <DISPLAY_NAME><![CDATA[News]]></DISPLAY_NAME>
      <GUID><![CDATA[00270980a0a0010a948d93f500702d]]></GUID>
      <REF_KEY><![CDATA[News]]></REF_KEY>
      <CHANNEL_PROPERTIES>
        <CHANNEL_PROPERTY NAME="ALLOWS_CHECKOUT"><![CDATA[true]]></
CHANNEL_PROPERTY>
        <CHANNEL_PROPERTY NAME="ASSOCIATED_CONTENT"/>
        <CHANNEL_PROPERTY NAME="GEOSPATIAL"><![CDATA[N]]></
CHANNEL_PROPERTY>
        <CHANNEL_PROPERTY NAME="HAS_PRIORITY"><![CDATA[N]]></
```

```
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_RELATEDCONTENT"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_TALKBACK"><![CDATA[N]]></
CHANNEL_PROPERTY>
           <CHANNEL_PROPERTY NAME="DISCUSSION_IS_MODERATED"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="REMOVE_VERSIONS"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="IS_EVENT"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_FILES"><![CDATA[true]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="TRACK_USER_ACTIVITY"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_WORKFLOW"><![CDATA[true]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="WORKFLOW_NAME"><![CDATA[Content]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY
NAME="WORKFLOW_REF_KEY"><![CDATA[CONTENT]]></CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_RATINGS"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <PREVIEW_URLS>
               <PREVIEW_URL NAME="url1"><![CDATA[http://localhost:8080/resources/sites/
<ID>/page]]></PREVIEW_URL>
            </PREVIEW_URLS>
         </CHANNEL_PROPERTIES>
         <CHANNEL_SCHEMA>
            <ATTRIBUTE>
               <NAME><![CDATA[News]]></NAME>
               <GUID><![CDATA[00270980a0a0010a948d93f5007026]]></GUID>
               <REF_KEY><![CDATA[NEWS]]></REF_KEY>
               <XPATH><![CDATA[//NEWS]]></XPATH>
               <DESCRIPTION/>
               <INDEXABLE><![CDATA[false]]></INDEXABLE>
               <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
               <SEARCHABLE><![CDATA[false]]></SEARCHABLE>
               <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
               <MINOCCURS><![CDATA[0]]></MINOCCURS>
               <TYPE><![CDATA[NODE]]></TYPE>
               <IS_READONLY><![CDATA[false]]></IS_READONLY>
               <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
            </ATTRIBUTE>
```

```
    <ATTRIBUTE>
      <NAME><![CDATA[Title]]></NAME>
      <GUID><![CDATA[00270980a0a0010a948d93f5007025]]></GUID>
      <REF_KEY><![CDATA[TITLE]]></REF_KEY>
      <XPATH><![CDATA[//NEWS/TITLE]]></XPATH>
      <DESCRIPTION/>
      <INDEXABLE><![CDATA[true]]></INDEXABLE>
      <IS_MASTER_IDENTIFIER><![CDATA[true]]></IS_MASTER_IDENTIFIER>
      <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
      <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
      <MINOCCURS><![CDATA[0]]></MINOCCURS>
      <TYPE><![CDATA[TEXT_FIELD]]></TYPE>
      <IS_READONLY><![CDATA[false]]></IS_READONLY>
      <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
    </ATTRIBUTE>
    <ATTRIBUTE>
      <NAME><![CDATA[Body]]></NAME>
      <GUID><![CDATA[00270980a0a0010a948d93f5007024]]></GUID>
      <REF_KEY><![CDATA[BODY]]></REF_KEY>
      <XPATH><![CDATA[//NEWS/BODY]]></XPATH>
      <DESCRIPTION/>
      <INDEXABLE><![CDATA[true]]></INDEXABLE>
      <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
      <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
      <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
      <MINOCCURS><![CDATA[0]]></MINOCCURS>
      <TYPE><![CDATA[TEXT_AREA]]></TYPE>
      <IS_READONLY><![CDATA[false]]></IS_READONLY>
      <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
    </ATTRIBUTE>
    <ATTRIBUTE>
      <NAME><![CDATA[securedfile]]></NAME>
      <GUID><![CDATA[00270980a0a0010a948d93f5007023]]></GUID>
      <REF_KEY><![CDATA[SECUREDFILE]]></REF_KEY>
      <XPATH><![CDATA[//NEWS/SECUREDFILE]]></XPATH>
      <DESCRIPTION/>
      <INDEXABLE><![CDATA[true]]></INDEXABLE>
      <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
      <SEARCHABLE><![CDATA[false]]></SEARCHABLE>
      <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
      <MINOCCURS><![CDATA[0]]></MINOCCURS>
      <TYPE><![CDATA[FILE]]></TYPE>
      <IS_READONLY><![CDATA[false]]></IS_READONLY>
      <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
    </ATTRIBUTE>
  </CHANNEL_SCHEMA>
  <META_SCHEMA/>
  <STYLESHEETS/>
```

```
        <WORKFLOW_STEPS>
          <STEP>
            <GUID><![CDATA[00270980a0a0010a948d93f50072b2]]></GUID>
            <DISPLAY_NAME><![CDATA[Creator]]></DISPLAY_NAME>
            <SORT_ORDER><![CDATA[1]]></SORT_ORDER>
          </STEP>
          <STEP>
            <GUID><![CDATA[00270980a0a0010a948d93f50072b3]]></GUID>
            <DISPLAY_NAME><![CDATA[Review]]></DISPLAY_NAME>
            <SORT_ORDER><![CDATA[2]]></SORT_ORDER>
          </STEP>
          <STEP>
            <GUID><![CDATA[00270980a0a0010a948d93f50072b1]]></GUID>
            <DISPLAY_NAME><![CDATA[Publisher]]></DISPLAY_NAME>
            <SORT_ORDER><![CDATA[3]]></SORT_ORDER>
          </STEP>
          <STEP>
            <GUID><![CDATA[00270980a0a0010a948d93f50072b0]]></GUID>
            <DISPLAY_NAME><![CDATA[Final Review]]></DISPLAY_NAME>
            <SORT_ORDER><![CDATA[4]]></SORT_ORDER>
          </STEP>
        </WORKFLOW_STEPS>
        <CATEGORIES>
          <CATEGORY>
            <DISPLAY_NAME><![CDATA[News]]></DISPLAY_NAME>
            <GUID><![CDATA[00270980a0a0010a948d93f500730a]]></GUID>
            <REF_KEY><![CDATA[NEWS]]></REF_KEY>
            <OBJECT_ID><![CDATA[001]]></OBJECT_ID>
          </CATEGORY>
          <CATEGORY>
            <DISPLAY_NAME><![CDATA[Sports]]></DISPLAY_NAME>
            <GUID><![CDATA[00270980a0a0010a948d93f5007306]]></GUID>
            <REF_KEY><![CDATA[SPORTS]]></REF_KEY>
            <OBJECT_ID><![CDATA[002]]></OBJECT_ID>
          </CATEGORY>
        </CATEGORIES>
        <USERGROUPS/>
      </CHANNEL>
    </DATA>
  </RESPONSE>
```

getChannels

Method:

java.lang.String getChannels(java.lang.String token, java.lang.String localeCode)

Description:

This method returns a list of Content Channels for the current repository.

Specified by:	com.inquira.imws.interfaces.UserServices	
Parameters:	**token**	security token
	localeCode	The locale code for the requested records, i.e. en_US. The locale code is used for the display names of the attributes. If no locale code is passed in, the locale used will be the repository's default locale. See Sample Response below.
Returns:	An XML-formatted string containing the response structure. See Sample Response below.	

Sample Input:

Sample Response:

```
<RESPONSE>
  <STATUS>
    <SERVERTIME><![CDATA[1145632561222]]></SERVERTIME>
    <TYPE><![CDATA[getChannels]]></TYPE>
    <RESPONSE_VERSION><![CDATA[1]]></RESPONSE_VERSION>
    <DATA_VERSION><![CDATA[1]]></DATA_VERSION>
    <VALUE><![CDATA[1]]></VALUE>
    <ERRORS/>
  </STATUS>

 <DATA>
   <CHANNELS>
     <CHANNEL>
       <DISPLAY_NAME><![CDATA[Bio]]></DISPLAY_NAME>
       <GUID><![CDATA[00270980a0a0010a948d93f5007285]]></GUID>
       <REF_KEY><![CDATA[Bio]]></REF_KEY>
       <CHANNEL_PROPERTIES>
         <CHANNEL_PROPERTY NAME="ALLOWS_CHECKOUT"><![CDATA[false]]></
CHANNEL_PROPERTY>
         <CHANNEL_PROPERTY NAME="ASSOCIATED_CONTENT"/>
         <CHANNEL_PROPERTY NAME="GEOSPATIAL"><![CDATA[N]]></
CHANNEL_PROPERTY>
         <CHANNEL_PROPERTY NAME="HAS_PRIORITY"><![CDATA[N]]></
CHANNEL_PROPERTY>
```

```
            <CHANNEL_PROPERTY NAME="HAS_RELATEDCONTENT"><![CDATA[Y]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_TALKBACK"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY
NAME="DISCUSSION_IS_MODERATED"><![CDATA[N]]></CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="REMOVE_VERSIONS"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="IS_EVENT"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_FILES"><![CDATA[true]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="TRACK_USER_ACTIVITY"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_WORKFLOW"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_RATINGS"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <PREVIEW_URLS>
              <PREVIEW_URL NAME="newurl"><![CDATA[http://localhost:8080/resources/sites/
<ID>/page]]></PREVIEW_URL>
              <PREVIEW_URL NAME="new12"><![CDATA[http://localhost:8080/resources/sites/
<ID>/page2]]></PREVIEW_URL>
              </PREVIEW_URLS>
           </CHANNEL_PROPERTIES>
           <CHANNEL_SCHEMA>
              <ATTRIBUTE>
                 <NAME><![CDATA[Bio]]></NAME>
                 <GUID><![CDATA[00270980a0a0010a948d93f5007276]]></GUID>
                 <REF_KEY><![CDATA[BIO]]></REF_KEY>
                 <XPATH><![CDATA[//BIO]]></XPATH>
                 <DESCRIPTION/>
                 <INDEXABLE><![CDATA[false]]></INDEXABLE>
                 <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
                 <SEARCHABLE><![CDATA[false]]></SEARCHABLE>
                 <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
                 <MINOCCURS><![CDATA[0]]></MINOCCURS>
                 <TYPE><![CDATA[NODE]]></TYPE>
                 <IS_READONLY><![CDATA[false]]></IS_READONLY>
                 <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
              </ATTRIBUTE>
```

```
                <ATTRIBUTE>
                  <NAME><![CDATA[Name]]></NAME>
                  <GUID><![CDATA[00270980a0a0010a948d93f5007275]]></GUID>
                  <REF_KEY><![CDATA[NAME]]></REF_KEY>
                  <XPATH><![CDATA[//BIO/NAME]]></XPATH>
                  <DESCRIPTION/>
                  <INDEXABLE><![CDATA[true]]></INDEXABLE>
                  <IS_MASTER_IDENTIFIER><![CDATA[true]]></IS_MASTER_IDENTIFIER>
                  <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
                  <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
                  <MINOCCURS><![CDATA[0]]></MINOCCURS>
                  <TYPE><![CDATA[TEXT_FIELD]]></TYPE>
                  <IS_READONLY><![CDATA[false]]></IS_READONLY>
                 <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
                </ATTRIBUTE>
                <ATTRIBUTE>
                  <NAME><![CDATA[Bio]]></NAME>

<GUID><![CDATA[00270980a0a0010a948d93f5007274]]></GUID>
                  <REF_KEY><![CDATA[BIO]]></REF_KEY>
                  <XPATH><![CDATA[//BIO/BIO]]></XPATH>
                  <DESCRIPTION/>
                  <INDEXABLE><![CDATA[true]]></INDEXABLE>
                  <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
                  <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
                  <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
                  <MINOCCURS><![CDATA[0]]></MINOCCURS>
                  <TYPE><![CDATA[WYSIWYG_EDIT]]></TYPE>
                  <IS_READONLY><![CDATA[false]]></IS_READONLY>
                 <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
                </ATTRIBUTE>
                <ATTRIBUTE>
                  <NAME><![CDATA[File]]></NAME>
                  <GUID><![CDATA[00270980a0a0010a948d93f5007273]]></GUID>
                  <REF_KEY><![CDATA[FILE]]></REF_KEY>
                  <XPATH><![CDATA[//BIO/FILE]]></XPATH>
                  <DESCRIPTION/>
                  <INDEXABLE><![CDATA[true]]></INDEXABLE>
                  <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
                  <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
                  <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
                  <MINOCCURS><![CDATA[0]]></MINOCCURS>
                  <TYPE><![CDATA[FILE]]></TYPE>
                  <IS_READONLY><![CDATA[false]]></IS_READONLY>
                 <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
                </ATTRIBUTE>
```

```
      <ATTRIBUTE>
        <NAME><![CDATA[Picture]]></NAME>
        <GUID><![CDATA[00270980a0a0010a948d93f5007272]]></GUID>
        <REF_KEY><![CDATA[PICTURE]]></REF_KEY>
        <XPATH><![CDATA[//BIO/PICTURE]]></XPATH>
        <DESCRIPTION/>
        <INDEXABLE><![CDATA[true]]></INDEXABLE>
        <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
        <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
        <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
        <MINOCCURS><![CDATA[0]]></MINOCCURS>
        <TYPE><![CDATA[FILE]]></TYPE>
        <IS_READONLY><![CDATA[false]]></IS_READONLY>
        <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
      </ATTRIBUTE>
    </CHANNEL_SCHEMA>
    <META_SCHEMA>
      <ATTRIBUTE>
        <NAME><![CDATA[META]]></NAME>
        <GUID><![CDATA[00270980a0a0010a948d93f500727a]]></GUID>
        <REF_KEY><![CDATA[META]]></REF_KEY>
        <XPATH><![CDATA[//META]]></XPATH>
        <DESCRIPTION/>
        <INDEXABLE><![CDATA[false]]></INDEXABLE>
        <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
        <SEARCHABLE><![CDATA[false]]></SEARCHABLE>
        <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
        <MINOCCURS><![CDATA[0]]></MINOCCURS>
        <TYPE><![CDATA[NODE]]></TYPE>
        <IS_READONLY><![CDATA[false]]></IS_READONLY>
        <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
      </ATTRIBUTE>
      <ATTRIBUTE>
        <NAME><![CDATA[Footer]]></NAME>
        <GUID><![CDATA[00270980a0a0010a948d93f5007279]]></GUID>
        <REF_KEY><![CDATA[FOOTER]]></REF_KEY>
        <XPATH><![CDATA[//META/FOOTER]]></XPATH>
        <DESCRIPTION/>
        <INDEXABLE><![CDATA[true]]></INDEXABLE>

<IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
        <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
        <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
        <MINOCCURS><![CDATA[0]]></MINOCCURS>
        <TYPE><![CDATA[TEXT_FIELD]]></TYPE>
        <IS_READONLY><![CDATA[false]]></IS_READONLY>
        <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
      </ATTRIBUTE>
```

```
            <ATTRIBUTE>
              <NAME><![CDATA[Shared Image]]></NAME>
              <GUID><![CDATA[00270980a0a0010a948d93f5007278]]></GUID>
              <REF_KEY><![CDATA[SHARED_IMAGE]]></REF_KEY>
              <XPATH><![CDATA[//META/SHARED_IMAGE]]></XPATH>
              <DESCRIPTION/>
              <INDEXABLE><![CDATA[true]]></INDEXABLE>
              <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
              <SEARCHABLE><![CDATA[false]]></SEARCHABLE>
              <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
              <MINOCCURS><![CDATA[0]]></MINOCCURS>
              <TYPE><![CDATA[FILE]]></TYPE>
              <IS_READONLY><![CDATA[false]]></IS_READONLY>
              <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
            </ATTRIBUTE>
            <ATTRIBUTE>
              <NAME><![CDATA[CODE]]></NAME>
              <GUID><![CDATA[00270980a0a0010a948d93f5007277]]></GUID>
              <REF_KEY><![CDATA[CODE]]></REF_KEY>
              <XPATH><![CDATA[//META/CODE]]></XPATH>
              <DESCRIPTION/>
              <INDEXABLE><![CDATA[true]]></INDEXABLE>

<IS_MASTER_IDENTIFIER><![CDATA[true]]></IS_MASTER_IDENTIFIER>
              <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
              <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
              <MINOCCURS><![CDATA[0]]></MINOCCURS>
              <TYPE><![CDATA[TEXT_FIELD]]></TYPE>
              <IS_READONLY><![CDATA[false]]></IS_READONLY>
              <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
            </ATTRIBUTE>
          </META_SCHEMA>
          <STYLESHEETS/>
          <CATEGORIES>
            <CATEGORY>
              <DISPLAY_NAME><![CDATA[News]]></DISPLAY_NAME>
              <GUID><![CDATA[00270980a0a0010a948d93f500730a]]></GUID>
              <REF_KEY><![CDATA[NEWS]]></REF_KEY>
              <OBJECT_ID><![CDATA[001]]></OBJECT_ID>
            </CATEGORY>
            <CATEGORY>
              <DISPLAY_NAME><![CDATA[Sports]]></DISPLAY_NAME>
              <GUID><![CDATA[00270980a0a0010a948d93f5007306]]></GUID>
              <REF_KEY><![CDATA[SPORTS]]></REF_KEY>
              <OBJECT_ID><![CDATA[002]]></OBJECT_ID>
            </CATEGORY>
          </CATEGORIES>
          <USERGROUPS/>
        </CHANNEL>
```

```
        <CHANNEL>
          <DISPLAY_NAME><![CDATA[News]]></DISPLAY_NAME>
          <GUID><![CDATA[00270980a0a0010a948d93f500702d]]></GUID>
          <REF_KEY><![CDATA[News]]></REF_KEY>
          <CHANNEL_PROPERTIES>
            <CHANNEL_PROPERTY NAME="ALLOWS_CHECKOUT"><![CDATA[true]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="ASSOCIATED_CONTENT"/>
            <CHANNEL_PROPERTY NAME="GEOSPATIAL"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY
NAME="HAS_PRIORITY"><![CDATA[N]]></CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_RELATEDCONTENT"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_TALKBACK"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY
NAME="DISCUSSION_IS_MODERATED"><![CDATA[N]]></CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="REMOVE_VERSIONS"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="IS_EVENT"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_FILES"><![CDATA[true]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="TRACK_USER_ACTIVITY"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_WORKFLOW"><![CDATA[true]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="WORKFLOW_NAME"><![CDATA[Content]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY
NAME="WORKFLOW_REF_KEY"><![CDATA[CONTENT]]></CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_RATINGS"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <PREVIEW_URLS>
              <PREVIEW_URL NAME="url1"><![CDATA[http://localhost:8080/resources/sites/
<ID>/page]]></PREVIEW_URL>
            </PREVIEW_URLS>
          </CHANNEL_PROPERTIES>
          <CHANNEL_SCHEMA>
            <ATTRIBUTE>
              <NAME><![CDATA[News]]></NAME>
              <GUID><![CDATA[00270980a0a0010a948d93f5007026]]></GUID>
              <REF_KEY><![CDATA[NEWS]]></REF_KEY>
              <XPATH><![CDATA[//NEWS]]></XPATH>
              <DESCRIPTION/>
              <INDEXABLE><![CDATA[false]]></INDEXABLE>
```

```
<IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
                <SEARCHABLE><![CDATA[false]]></SEARCHABLE>
                <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
                <MINOCCURS><![CDATA[0]]></MINOCCURS>
                <TYPE><![CDATA[NODE]]></TYPE>
                <IS_READONLY><![CDATA[false]]></IS_READONLY>
                <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
              </ATTRIBUTE>
              <ATTRIBUTE>
                <NAME><![CDATA[Title]]></NAME>
                <GUID><![CDATA[00270980a0a0010a948d93f5007025]]></GUID>
                <REF_KEY><![CDATA[TITLE]]></REF_KEY>
                <XPATH><![CDATA[//NEWS/TITLE]]></XPATH>
                <DESCRIPTION/>
                <INDEXABLE><![CDATA[true]]></INDEXABLE>
                <IS_MASTER_IDENTIFIER><![CDATA[true]]></IS_MASTER_IDENTIFIER>
                <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
                <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
                <MINOCCURS><![CDATA[0]]></MINOCCURS>
                <TYPE><![CDATA[TEXT_FIELD]]></TYPE>
                <IS_READONLY><![CDATA[false]]></IS_READONLY>
                <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
              </ATTRIBUTE>
              <ATTRIBUTE>
                <NAME><![CDATA[Body]]></NAME>
                <GUID><![CDATA[00270980a0a0010a948d93f5007024]]></GUID>
                <REF_KEY><![CDATA[BODY]]></REF_KEY>
                <XPATH><![CDATA[//NEWS/BODY]]></XPATH>
                <DESCRIPTION/>
                <INDEXABLE><![CDATA[true]]></INDEXABLE>

<IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
                <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
                <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
                <MINOCCURS><![CDATA[0]]></MINOCCURS>
                <TYPE><![CDATA[TEXT_AREA]]></TYPE>
                <IS_READONLY><![CDATA[false]]></IS_READONLY>
                <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
              </ATTRIBUTE>
```

```
            <ATTRIBUTE>
              <NAME><![CDATA[securedfile]]></NAME>
              <GUID><![CDATA[00270980a0a0010a948d93f5007023]]></GUID>
              <REF_KEY><![CDATA[SECUREDFILE]]></REF_KEY>
              <XPATH><![CDATA[//NEWS/SECUREDFILE]]></XPATH>
              <DESCRIPTION/>
              <INDEXABLE><![CDATA[true]]></INDEXABLE>
              <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
              <SEARCHABLE><![CDATA[false]]></SEARCHABLE>
              <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
              <MINOCCURS><![CDATA[0]]></MINOCCURS>
              <TYPE><![CDATA[FILE]]></TYPE>
              <IS_READONLY><![CDATA[false]]></IS_READONLY>
              <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
            </ATTRIBUTE>
          </CHANNEL_SCHEMA>
          <META_SCHEMA/>
          <STYLESHEETS/>
          <WORKFLOW_STEPS>
            <STEP>
              <GUID><![CDATA[00270980a0a0010a948d93f50072b2]]></GUID>
              <DISPLAY_NAME><![CDATA[Creator]]></DISPLAY_NAME>
              <SORT_ORDER><![CDATA[1]]></SORT_ORDER>
            </STEP>
            <STEP>

<GUID><![CDATA[00270980a0a0010a948d93f50072b3]]></GUID>
              <DISPLAY_NAME><![CDATA[Review]]></DISPLAY_NAME>
              <SORT_ORDER><![CDATA[2]]></SORT_ORDER>
            </STEP>
            <STEP>
              <GUID><![CDATA[00270980a0a0010a948d93f50072b1]]></GUID>
              <DISPLAY_NAME><![CDATA[Publisher]]></DISPLAY_NAME>
              <SORT_ORDER><![CDATA[3]]></SORT_ORDER>
            </STEP>
            <STEP>
              <GUID><![CDATA[00270980a0a0010a948d93f50072b0]]></GUID>
              <DISPLAY_NAME><![CDATA[Final Review]]></DISPLAY_NAME>
              <SORT_ORDER><![CDATA[4]]></SORT_ORDER>
            </STEP>
          </WORKFLOW_STEPS>
          <CATEGORIES>
            <CATEGORY>
              <DISPLAY_NAME><![CDATA[News]]></DISPLAY_NAME>
              <GUID><![CDATA[00270980a0a0010a948d93f500730a]]></GUID>
              <REF_KEY><![CDATA[NEWS]]></REF_KEY>
              <OBJECT_ID><![CDATA[001]]></OBJECT_ID>
            </CATEGORY>
```

```
            <CATEGORY>
               <DISPLAY_NAME><![CDATA[Sports]]></DISPLAY_NAME>
               <GUID><![CDATA[00270980a0a0010a948d93f5007306]]></GUID>
               <REF_KEY><![CDATA[SPORTS]]></REF_KEY>
               <OBJECT_ID><![CDATA[002]]></OBJECT_ID>
            </CATEGORY>
         </CATEGORIES>
         <USERGROUPS/>
      </CHANNEL>

 <CHANNEL>
         <DISPLAY_NAME><![CDATA[Products]]></DISPLAY_NAME>
         <GUID><![CDATA[00270980a0a0010a948d93f5006e7c]]></GUID>
         <REF_KEY><![CDATA[Products]]></REF_KEY>
         <CHANNEL_PROPERTIES>
            <CHANNEL_PROPERTY NAME="ALLOWS_CHECKOUT"><![CDATA[true]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="ASSOCIATED_CONTENT"/>
            <CHANNEL_PROPERTY NAME="GEOSPATIAL"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_PRIORITY"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_RELATEDCONTENT"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_TALKBACK"><![CDATA[Y]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY
NAME="DISCUSSION_IS_MODERATED"><![CDATA[N]]></CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="REMOVE_VERSIONS"><![CDATA[N]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="IS_EVENT"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_FILES"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="TRACK_USER_ACTIVITY"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_WORKFLOW"><![CDATA[false]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="HAS_RATINGS"><![CDATA[true]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY NAME="RATING_NAME"><![CDATA[Content Rating]]></
CHANNEL_PROPERTY>
            <CHANNEL_PROPERTY
NAME="RATING_REF_KEY"><![CDATA[CONTENT_RATING]]></CHANNEL_PROPERTY>
            <PREVIEW_URLS/>
         </CHANNEL_PROPERTIES>
         <CHANNEL_SCHEMA>
            <ATTRIBUTE>
               <NAME><![CDATA[Products]]></NAME>
```

```
<GUID><![CDATA[00270980a0a0010a948d93f5006e75]]></GUID>
                <REF_KEY><![CDATA[PRODUCTS]]></REF_KEY>
                <XPATH><![CDATA[//PRODUCTS]]></XPATH>
                <DESCRIPTION/>
                <INDEXABLE><![CDATA[false]]></INDEXABLE>
                <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
                <SEARCHABLE><![CDATA[false]]></SEARCHABLE>
                <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
                <MINOCCURS><![CDATA[0]]></MINOCCURS>
                <TYPE><![CDATA[NODE]]></TYPE>
                <IS_READONLY><![CDATA[false]]></IS_READONLY>
                <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
              </ATTRIBUTE>
              <ATTRIBUTE>
                <NAME><![CDATA[Name]]></NAME>
                <GUID><![CDATA[00270980a0a0010a948d93f5006e74]]></GUID>
                <REF_KEY><![CDATA[NAME]]></REF_KEY>
                <XPATH><![CDATA[//PRODUCTS/NAME]]></XPATH>
                <DESCRIPTION/>
                <INDEXABLE><![CDATA[true]]></INDEXABLE>
                <IS_MASTER_IDENTIFIER><![CDATA[true]]></IS_MASTER_IDENTIFIER>
                <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
                <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
                <MINOCCURS><![CDATA[1]]></MINOCCURS>
                <TYPE><![CDATA[TEXT_FIELD]]></TYPE>
                <IS_READONLY><![CDATA[false]]></IS_READONLY>
                <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
              </ATTRIBUTE>
              <ATTRIBUTE>
                <NAME><![CDATA[Code]]></NAME>
                <GUID><![CDATA[00270980a0a0010a948d93f5006e73]]></GUID>
                <REF_KEY><![CDATA[CODE]]></REF_KEY>
                <XPATH><![CDATA[//PRODUCTS/CODE]]></XPATH>
                <DESCRIPTION/>
                <INDEXABLE><![CDATA[true]]></INDEXABLE>
                <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
                <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
                <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
                <MINOCCURS><![CDATA[0]]></MINOCCURS>
                <TYPE><![CDATA[TEXT_FIELD]]></TYPE>
                <IS_READONLY><![CDATA[false]]></IS_READONLY>
                <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
              </ATTRIBUTE>
```

```
            <ATTRIBUTE>
              <NAME><![CDATA[Description]]></NAME>
              <GUID><![CDATA[00270980a0a0010a948d93f5006e72]]></GUID>
              <REF_KEY><![CDATA[DESCRIPTION]]></REF_KEY>
              <XPATH><![CDATA[//PRODUCTS/DESCRIPTION]]></XPATH>
              <DESCRIPTION/>
              <INDEXABLE><![CDATA[true]]></INDEXABLE>
              <IS_MASTER_IDENTIFIER><![CDATA[false]]></IS_MASTER_IDENTIFIER>
              <SEARCHABLE><![CDATA[true]]></SEARCHABLE>
              <MAXOCCURS><![CDATA[1]]></MAXOCCURS>
              <MINOCCURS><![CDATA[0]]></MINOCCURS>
              <TYPE><![CDATA[TEXT_AREA]]></TYPE>
              <IS_READONLY><![CDATA[false]]></IS_READONLY>
              <SECUREDFILERESOURCE><![CDATA[false]]></SECUREDFILERESOURCE>
            </ATTRIBUTE>
          </CHANNEL_SCHEMA>
          <META_SCHEMA/>
          <STYLESHEETS/>
          <CATEGORIES>
            <CATEGORY>
              <DISPLAY_NAME><![CDATA[News]]></DISPLAY_NAME>
              <GUID><![CDATA[00270980a0a0010a948d93f500730a]]></GUID>
              <REF_KEY><![CDATA[NEWS]]></REF_KEY>
              <OBJECT_ID><![CDATA[001]]></OBJECT_ID>
            </CATEGORY>
            <CATEGORY>
              <DISPLAY_NAME><![CDATA[Sports]]></DISPLAY_NAME>
              <GUID><![CDATA[00270980a0a0010a948d93f5007306]]></GUID>
              <REF_KEY><![CDATA[SPORTS]]></REF_KEY>
              <OBJECT_ID><![CDATA[002]]></OBJECT_ID>
            </CATEGORY>
          </CATEGORIES>
          <USERGROUPS/>
        </CHANNEL>
      </CHANNELS>
    </DATA>
  </RESPONSE>
```

Chapter 2 The Information Manager Tag Library

This section provides reference information and examples for Information Manager JSP tags. The tags are listed in alphabetical order, and each section includes tag format information and a description. Usage examples are provided where available.

auto.login

Description:

This tag automatically logs in a user by providing a user login or user record ID.

Format:

```
<IM:auto.login
login="string"          Login for the user
recordid="string"       Record id for the user

/>
```

benchmark

Description:

This tag displays a formatted box on the page where the tag is added. Each tag on the page is instrumented to show one or more performance metrics, such as total execution time, produced by the ***timer tag*** described in *timer* on page 275. The metric data is stored at the page scope of each JSP page.

To display the statistics of embedded templates you must add the <benchmark> tag to each page. For applications such as Information Center, you must add the <benchmark> tag to the `c_xxx.jsp` pages.

Format:

```
<IM:benchmark
/IM:benchmark>
```

Example:

See the section on the timer tag for an example of how the two tags are used together.

cache

Description:

This tag caches page data for quick retrieval.

Format:

```
<IM:cache
```

`id*="string"`	ID for this cached piece of HTML.
`minutes="non-scriptable integer"`	Sets this cache entry to expire in a certain number of minutes.

```
></IM:cache>
```

cache.destroy

Description:

This tag kills the specified cache.

Format:

<IM:cache.destroy

`id*="string"`	ID for this cached piece of HTML.

/>

create.category.data

Description:

This tag creates a category.

Format:

<IM:create.category.data

`referencekey*="string"`	The reference key for the new category. Reference keys must be unique in a Repository.
`name*="string"`	Display name for the new category
`parentcategory="string"`	Reference key for the parent category. If this value is present, the new category will added as a child of parentcategory. The parent category must exist

/>

dataset.batch

Description:

This tag calculates and displays the number of batches for a data set iterator.

Format:

<IM:dataset.batch

`id="non-scriptable string"`	A desired prefix to use for all scripting variables created by this tag. There is no default id. Make sure all ids are unique
`displayontop="integer"`	If set to true, the display batch will be showed at the first index of the repetition
`displayonbottom="integer"`	If set to true, the display batch will be showed at the last index of the repetition

></IM:dataset.batch >

dataset.batch.page.iterator

Description:

This tag iterates over the pages returned by the `dataset.batch` tag.

Format:

<IM:dataset.batch.page.iterator

`id="non-scriptable string"`	A desired prefix to use for all scripting variables created by this tag. There is no default id. Make sure all ids are unique.

></IM:dataset.batch.page.iterator >

debug

Description:

This tag displays detailed debugging information on all application, session, request and page level variables.

Format:

<IM:debug

`view="string"`	Scope of the debug information to view (all, session, request, application, or page)

/>

Example Usage:

<IM:debug view="all"/>

Scripting Variables:

none

Body Tags:

none

delete.content.casenumber

Description:

Tag that deletes a case number for a content record

Format:

<IM:delete.content.casenumber

`documentid="non-scriptable string"`	Delete the case number for the supplied document ID
`recordid="non-scriptable string"`	Delete the case number for the supplied record ID
`casenumber*="non-scriptable string"`	The case number to be deleted for the content record

></IM:delete.content.casenumber >

email.send.page

Description:

This tag automatically sends the specified page or url to the desired email address.

Format:

<IM:email.send.page

`recipient*="string"`	the recipient's email address
`recipientname="string"`	the recipient's display name
`from*="string"`	the sender's email address
`fromname="string"`	the sender's display name
`subject*="string"`	the subject description
`comments="string"`	optional comments to be appended. The comment will be inserted in the desired page only when the <code>SC_COMMENT_REPLACE</code> is present.
`page="string"`	the page to be sent. This attribute should be in the SiteMap format, for example: HOME
`url="string"`	the url of the page to be sent. This attribute should be in the http format, for example: http:// www.company.com/IM/index?page=home

></IM:email.send.page >

form.channel.contribution

Description:

This tag creates an HTML form to allow an end user to contribute content directly into a specific content channel through the client site. The user will not be able to edit the record once submitted. If workflow is setup for the channel, the new record will go in to the initial step of the workflow and await approval prior to publishing. Content Administrators can find and locate the contributed record just as if it had been created using the Management Console.

Format:

<IM:form.channel.contribution

`success="string"`	Page user will be directed to if the content was successfully added
`error="string"`	Page user will be directed to if creating the content failed.
`channel*="string"`	The channel this contribution is for.
`views="string"`	The department(s) this contribution is for - delimited by '+'.
`categories="string"`	The categories assigned to this contribution - delimited by '+'.
`groups="string"`	The user groups assigned to this contribution - delimited by '+'.
`publish="true" or "false"`	Set this attribute to false if you do not wish the contribution to be published. The default behavior is to publish the contributions if the specified channel does not have workflow turned on.
`name="string"`	HTML form name
`onsubmit="string"`	onSubmit event code
`onreset="string"`	onreset event code
`useeditor="true" or "false"`	If this attribute is set to true, every schema attribute that is of type Rich Text Editor will display a wysiwyg component. The default value is true.
`allowfilebrowsing="true" or "false"`	If this attribute is set to true and the useeditor attribute is set to true, allowed users will be able to browse and upload files to the server. The default value is true.
`casenumber="string"`	assign a case number to this record

`casedescription="string"`	assign a case description to this record
`caseincident="string"`	assign a case incident to this record

></IM:form.channel.contribution >

Example Usage:

<IM:form.channel.contribution channel="news" departments="department1+department2" success="thankyou" error="errorpage">

Title:

<IM:input.channel.contribution attribute="/news/title"/>

Body:

<IM:input.channel.contribution attribute="/news/body"/>
</IM:channel.contribute.form>

Scripting Variables:

None

Body Tags:

input.channel.contribution on page 223

form.dataform

Description:

This tag creates an HTML form to allow an end user to enter data into a data form. You can use this tag either independently, or as a child of a get.channel.record tag (for example, when there is a rating associated with a piece of content).

Format:

```
<IM:form.dataform
```

`success="string"`	the url of the page that the user will be directed to when the data form is saved.
`error="string"`	the url of the page that the user will be directed to if the data form cannot be saved due to an error.
`dataform="string"`	the Reference Key of the data form.
`target="string"`	the target window for success and error pages.
`view="string"`	the Reference Key of the content channel view to which the response should be assigned.
`name="string"`	the HTML form name
`onsubmit="string"`	an onSubmit event code
`onreset="string"`	an onreset event code
`localecode="string"`	the locale in which to store the submitted form data. The default is the session locale.

```
></IM:form.dataform >
```

Example Usage for Displaying the Form:

```
<IM:form.dataform dataform="INQUIRIES" success="requestthank" error="requestthank">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:input.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question> <br>
<input type="submit" value="Submit">
</IM:form.dataform>
```

Example Usage for Displaying Results

```
<IM:get.dataform.results name="nameofform" individual="false" aggregate="false">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<IM:get.dataform.question/><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:get.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question>
</IM:get.dataform.results>
```

Scripting Variables:

None

TagClass:

com.inquira.client.tags.TakeSurveyTag

form.login

Description:

This tag creates an HTML form to capture login in information such as user id and password.

Format:

<IM:form.login

`success="string"`	Page user will be directed to if the login was successful
`error="string"`	Page user will be directed to if the login failed, the original data entered will be redisplayed if directed back to the login page, error messages are available using the error.xxx tags
`target="string"`	Target window for success and error pages.
`name="string"`	HTML form name
`onsubmit="string"`	onSubmit event code
`onreset="string"`	onreset event code

/>

Example Usage:

```
<IM:form.login success="home" error="logon">
<IM:input.user.id/>
<IM:input.password/>
</IM:form.login>
```

Scripting Variables:

None

Body Tags:

input.user.id on page 240

input.user.password on page 244

form.lostpassword

Description:

This tag creates an HTML form to capture the email address of a user to email them their login information.

Format:

<IM:form.lostpassword

`success="string"`	Page user will be directed to if the login was successful
`error="string"`	Page user will be directed to if the email address failed, the original data entered will be redisplayed if directed back to the lost password page, error messages are available using the error.xxx tags
`name="string"`	HTML form name
`onsubmit="string"`	onSubmit event code
`onreset="string"`	onreset event code

/>

Example Usage:

```
<IM:form.lostpassword success="home" error="logon">
<IM:error.email/><br/>
<IM:input.user.email value="" size="40" maxlength="40"/><br/>
<input type="submit" value="Submit" name="B1">
</IM:form.lostpassword >
```

Scripting Variables:

none

Body Tags:

input.user.email on page 236

form.message

Description:

This tag creates an HTML form to allow users to contribute to a threaded message board.

Format:

<IM:form.message

`success="string"`	Page user will be directed to if the message was successfully added
`error="string"`	Page user will be directed to if adding the message failed
`parentrecordid*="string"`	The record id of the parent record for this message. Examples would be if an article has talkback associated with it, this attribute would have to contain the record id of the article (the talkback will be a child record of the article). Likewise, for a discussion group, this attribute would either contain the record id of the discussion group (top level talkback), or the record id of another talkback (the talkback becomes a child of another talkback aka. a response)
`emailto="string"`	Email address to send this talkbackresponse to. It could be used for sending it to a mailing list.
`emailfrom="string"`	Email address from who this response is being sent. If you whish to let the users input their own from email address, just create a html input field with name = 'emailfrom', and leave this attribute blank. If no from address is supplied the system will use the ADMIN_EMAIL parameter as the from address for the email. in order to send responses as emails, there must be a emailto address specified.
`emailfooter="string"`	Text that will be appended on email responses only, right after the text the user has input. This is an optional attribute, and it can be used to providing a link back to the threaded discussion. You could specify the email footer here as a attribute of this tag or provide a input field with name = 'emailfooter' for this parameter.
`name="string"`	HTML form name

`onsubmit="string"`	onSubmit event code
`onreset="string"`	onreset event code

/>

Example Usage for Displaying All Messages:

```
<!-- get message data -->
<IM:get.message.data dataset="data" topic="hammers"/>
<!-- iterate through messages in discussion records -->
<IM:iterate.message.records dataset="data">
<!-- get handle to individual record -->
<IM:get.message.record>
<!-- space message to show hierarchy and display message title and text-->
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.title/> <br>
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.text/> <br>
<!-- display a reply link that passes the message record id to the reply page -->
<a href="index?page=messagereply&rec=<%=message.recordid%>">Reply</a> <br>
</IM:get.message.record>
</IM:iterate.message.records>
```

Example Usage for Replying to a Message:

```
<!-- declare page variable and load recordid parameter off url from previous page -->
<% String rec = request.getParameter("rec");%>
<!-- create HTML form for input fields -->
<IM:form.message parentrecordid="<%=rec%>" success="successpage" error="errorpage">
<!-- display input fields for title and text -->
Title<br>
<input.message.title><br>
Text<br>
<input.message.text><br>
<!-- display submit button -->
<input type="submit" value="Submit">
</IM:form.message>
```

Scripting Variables:

None

Body Tags:

form.message ?

input.message.title on page 228

input.message.text on page 226

form.recommendation.contribution

Description:

Creates an HTML form to allow an end user to contribute a content recommendation.

Format:

<IM:form.recommendation.contribution

success="string"	Page user will be directed to if the content was successfully added
error="string"	Page user will be directed to if creating the content failed .
channel="string"	The channel this contribution is for.
priority="string"	The priority this contribution will have.
categories="string"	The categories assigned to this contribution - delimited by '+'.
name="string"	HTML form name
onsubmit="string"	onSubmit event code
onreset="string"	onreset event code

></IM:form.recommendation.contribution >

form.search.attribute

Description:

This tag creates an HTML form to perform attribute search on a specified Content Channel.

Format:

<IM:form.search.attribute

`channel="string"`	Name (referencekey) for the Channel to be searched
`matchtype="string"`	determines whether to match all criteria or any criteria. Valid values are 'ALL' or 'ANY'. case insensitive
`success="string"`	
`error="string"`	
`sortattribute="string"`	Specifies attribute used to sort results
`sortdirection="string"`	Specify either ASC (ascending) or DESC (descending) sort order
`categories="string"`	delimited (+) list of Content Categories (using reference key) to include in the full text search. i.e. CAT1+CAT2.
`matchallcategories="true" or "false"`	Used in conjunction with 'categories' attribute. Set to true to 'and' the categories together in the search, and false to 'or' the categories.
`views="string"`	delimited (+) list of Views (using reference key) to include in the full text search. i.e. VIEW1+VIEW2.
`maxrecords="integer"`	Max amount of records returned by this search. Default is set to 200 records. A value of 0 will return all possible records
`name="string"`	HTML form name
`onsubmit="string"`	onSubmit event code
`onreset="string"`	onreset event code

></IM:form.search.attribute

Example Usage:

```
<IM:form.search.attribute channel="directory" matchtype="ALL" success="searchresults"
error="errorpage">
First Name:<br>
<IM:input.search.attribute attribute="directory\first_name"/>
<br>
Last Name:<br>
<IM:input.search.attribute attribute="directory\last_name"/>
<br>
<input type="submit"/>
</IM:form.search.attribute>
```

Scripting Variables:

none

Body Tags:

input.search.attribute on page 231

form.search.fulltext

Description:

This tag creates an HTML form to perform a full text search on one or more selected Content Channels.

Format:

```
<IM:form.search.fulltext
```

`channels="string"`	A delimited (+) list of Content Channels (using reference key) to include in the full text search (i.e., NEWS+EVENTS).
`success="string"`	Page user will be directed to if the search was successfully created.
`error="string"`	Page user will be directed to if search failed.
`categories="string"`	delimited (+) list of Content Categories (using reference key) to include in the full text search (i.e., CAT1+CAT2).
`matchallcategories="true" or "false"`	Used in conjunction with 'categories' attribute. Set to true to 'and' the categories together in the search, and false to 'or' the categories.

`matchallterms="true" or "false"`	If this attribute is set to true, the search will AND all the words searched. If it is set to false, this tag will OR the words searched. The default value is true.
`views="string"`	delimited (+) list of Views (using reference key) to include in the full text search (i.e., VIEW1+VIEW2).
`maxrecords="integer"`	Max amount of records returned by this search. Default is set to 200 records. A value of 0 will return all possible records
`name="string"`	HTML form name
`onsubmit="string"`	onSubmit event code
`onreset="string"`	onreset event code

></IM:form.search.fulltext >

Example Usage:

```
<IM:form.search.fulltext channels="news+events+press" success="searchresults"
error="errorpage">
Search String: <br>
<IM:input.search.fulltext/>
<br>
<input type="submit"/>
</IM:form.search.fulltext>
```

Scripting Variables:

none

Body Tags:

input.search.fulltext on page 232

form.select.locale

Description:

Create an HTML form that displays a drop down box of available locals for the current domain.

Format:

<IM:form.select.locale

`success="string"`	Page user will be directed to if the locale change was successful
`error="string"`	Page user will be directed to if the locale change failed, the original data entered will be redisplayed if directed back to the login page, error messages are available using the error.xxx tags
`name="string"`	HTML form name
`onsubmit="string"`	onSubmit event code
`onreset="string"`	onreset event code

></IM:form.select.locale >

Example Usage:

```
<IM:form.select.locale success="successpage" error="errorpage">
<input type="submit"/>
</IM:form.select.locale>
```

Scripting Variables:

none

Body Tags:

none

form.send.email

Description:

This tag allows for the sending of a client html page as email.

Format:

<IM:form.send.email

`subject*="string"`	subject line of the email
`page="string"`	html page to send as the body of the email
`url="string"`	html page to send as the body of the email
`success="string"`	Page user will be directed to if the edit was successful
`error="string"`	Page user will be directed to if the edit failed, the original data entered will be redisplayed if directed back to the edit page, error messages are available using the error.xxx tags
`name="string"`	HTML form name
`onsubmit="string"`	onSubmit event code
`onreset="string"`	onreset event code

></IM:form.send.email >

form.shoppingcart.action

Description:

Creates an HTML form used for adding or removing items from a shopping cart.

Format:

<IM:form.shoppingcart.action

cart*="string"	Reference key for shopping cart.
options*="string"	Name of the HTML element containing the options or attributes of the product.
success="string"	Page user will be directed to if the subscribe action is successful
error="string"	Page user will be directed if the subscribe action is unsuccessful
cartaction*="string"	Action to be performed. Valid values for this field are add and <B>remove
item*="string"	Record ID of item to be added or deleted from shopping cart.
transactionid="string"	Optional field for setting the transaction ID if known. This attribute will get the transaction directly from the database instead of the user's session, or cookie.
name="string"	HTML form name
onsubmit="string"	onSubmit event code
onreset="string"	onreset event code

></IM:form.shoppingcart.action >

form.shoppingcart.update

Description:

Creates an HTML form used for updating items from a shopping cart.

Format:

<IM:form.shoppingcart.update

`cart*="string"`	Reference key for shopping cart.
`transactionid="string"`	Optional field for setting the transaction ID if known. This attribute will get the transaction directly from the database instead of the user's session, or cookie.
`success="string"`	Page user will be directed to if the update action is successful
`error="string"`	Page user will be directed if the update action is unsuccessful
`name="string"`	HTML form name
`onsubmit="string"`	onSubmit event code
`onreset="string"`	onreset event code

></IM:form.shoppingcart.update >

form.subscription.subscribe

Description:

Creates an HTML form used for subscribing and un-subscribing from subscriptions.

Format:

<IM:form.subscription.subscribe

`success="string"`	Page user will be directed to if the subscribe action is successful
`error="string"`	Page user will be directed if the subscribe action is unsuccessful
`name="string"`	HTML form name
`onsubmit="string"`	onSubmit event code
`onreset="string"`	onreset event code

></IM:form.subscription.subscribe >

form.user.attributes

Description:

Creates an HTML form to allow a user to edit their profile information.

Format:

<IM:form.user.attributes

`success="string"`	Page user will be directed to if the edit was successful
`error="string"`	Page user will be directed to if the edit failed, the original data entered will be redisplayed if directed back to the edit page, error messages are available using the error.xxx tags
`createnew="true" or "false"`	Set to <code>true</code> if you would like to create a new user. Otherwise set to <code>false</code>. The default behavior is set to <code>false</code>
`adminuser="true" or "false"`	Set to <code>true</code> if you would like to create a new admin user. Set to <code>false</code> to create Web Users. The default behavior is set to create web users
`autologin="true" or "false"`	Set to <code>true</code> if you would like to login the edited user. Otherwise set to <code>false</code>. The default behavior is set to <code>false</code>
`userid="string"`	User ID (GUID) of the user to be edited
`roles="string"`	When creating new users use this attribute to specified one or more Roles to be assigned to the new user. Use existing role reference keys separated by plus signs. If no roles are specified, then the default user role will be assigned to the new user.

`parenteditorgroup="string"`	If attribute present, a new Editor Group will be created using this attribute as its Parent Branch (make sure branch exists). The Editor Group creation feature is only applied for new users. The new Editor Group will have as reference key the new user's login and its Display Name format will be: user's Last Name, user's First Name.
`name="string"`	HTML form name
`onsubmit="string"`	onSubmit event code
`onreset="string"`	onreset event code

></IM:form.user.attributes >

Example Usage:

```
<IM:form.user.attributes success="successpage" error="errorpage">
User ID: <IM:input.user.id.error/><br>
<IM:input.user.id value="" system size="20"/><br>
Password: <IM:input.user.password .error/><br>
<IM:input.user.password value="" size="20"/><br>
First Name: <IM:input.user.firstname .error/><br>
<IM:input.firstname value="" size="30"/><br>
Last Name: <IM:input.user.lastname .error/><br>
<IM:input.user.lastname value="" size="30"/><br>
Email Address: <IM:input.user.email.error/><br>
<IM:input.user.email value="" size="30"/><br>
Phone Number:<br>
<IM:input.user.attribute attribute="demo/phone" value="" size="10"/><br>
Zip Code<br>
<IM:input.user.attribute attribute="demo/zipcode" value="" size="5"/><br>
<br>
<IM:input.submit value="Save Changes"/>
</IM:form.user.attributes>
```

Scripting Variables:

none

Body Tags:

input.user.id on page 240

input.user.password on page 244

input.user.firstname on page 238

input.user.lastname on page 242

input.user.email on page 236
input.user.attribute on page 234v

form.wizardform

Description:

Creates an HTML form to take input from a Process Wizard question.

Format

<IM:form.wizardform

`success="string"`	Page user will be directed to if the wizard encountered no problems
`error="string"`	Page user will be directed to if the wizard encountered problems
`target="string"`	Target window for success and error pages
`name="string"`	HTML form name
`onsubmit="string"`	onSubmit event code
`onreset="string"`	onreset event code
`wizardid="string"`	The id of the Process Wizard that will be used
`wizardstepid="string"`	The internal step id of the Process Wizard
`wizardnextstep="string"`	Reserved for future use
`id="string"`	The id of the form

/>

Example Usage:

```
<html>
<head>

</head>
<body>
... <IM:form.wizardform wizardid="<%=wizid%>" wizardstepid="<%=wizstep%>" id="id2"
success="searchtest" error="http://www.cnn.com">
</body>
</html>
```

Scripting Variables:

none

Body Tags:

A `form.wizardform` will generate the following hidden fields :

```
<input name="action" value="SearchWizardAction" type="hidden">
<input name="success" value="answers" type="hidden">
<input name="error" value="answers&amp;amp;er=y" type="hidden">
<input name="wizardid" value="WizardOfTime" type="hidden">
<input name="wizardstepid" value="1A" type="hidden">
<input name="wizaction" value="next" type="hidden">
(wizaction is in version 8.0.1.1)
```

The “wizaction” hidden field indicates which submit button was pressed. There are three available options: “next”, “back”, and “cancel”. You must use a script to set this parameter when a button is clicked. The following excerpt uses javascript to set each of the three values as in `onclick="javascript:wizaction.value = 'cancel';">`

```
<button value="cancel" name="inqwiz" onclick="javascript:wizaction.value = 'cancel';">Cancel</
button>
        <% if (id2.showback) { %><button type="submit" value="Back" name="inqwiz"
onclick="javascript:wizaction.value = 'back';">Back</button><% } %>
<button type="submit" class="button-feature" name="inqnext"
onclick="javascript:wizaction.value = 'next';">Next</button>
```

You can give the buttons any name you wish but the hidden “wizaction” must be set to either “next”, “back”, or “cancel”.

get.audit.record

Description:

Tag that gets the current object of the audit iteration. Scripting Variables: xml = XML Document as a String; doc = DOM xml document; recordid = the id of the record(GUID); index = current repetition index for this object; attributes = a Java Object array containing all the visible attributes for this record. The actual objects in the array are Strings;

Format:

```
<IM:get.audit.record
```

`id="non-scriptable string"`	A desired prefix to use for all scripting variables created by this tag. There is no default id. If nesting user-record tags, make sure all ids are unique
`recordid="string"`	ID of record to retrieve. If you are trying to retrieve an audit record by record id, you must supply the intended entity of that record.
`entity="string"`	The intended entity of the audit record to retrieve. This is a mandatory attribute if the record ID attribute is used. Valid values are content, user, security.

```
></IM:get.audit.record >
```

get.audit.data

Description:

Retrieves Audit records for the enclosed parameters. The allowed audit entities are: content, user, security.

Format:

<IM:get.audit.data

`dataset*="string"`	A unique identifier for the dataset, this identifier is used by the iterator tag to access records that are retrieved.
`entity*="string"`	This is the audit entity. Valid values are content, user, security
`matchall="true" or "false"`	If this evaluates to true all the enclosed qualifiers will be grouped by ANDs. If set to false all the enclosed qualifiers will be grouped by ORs. The default value is true
`sortby="string"`	List of entity attributes delimited by a + sign to be sorted by.
`direction="string"`	The direction of the sort. Ascending or descending. The default if no value provided will be ascending.

></IM:get.audit.data >

get.category.attribute

Description:

This tag displays the specified attribute for a category.

Format:

<IM:get.category.attribute

`attribute*="string"`	the name of attribute to display. The attributes supported are:	
	`name`	returns the name of the category
	`referencekey`	returns the reference key of the category
	`recordid`	returns the recordid(guid) of the category
	`level`	returns the depth level of the category
	`parentreferencekey`	returns the reference key of the parent of this category
	`parentid`	returns the recordid(guid) of the parent of this category

></IM:get.category.attribute >

Required Parent Tag:

`get.category.record` on page

get.category.data

Description:

Retrieves a list of categories. If no channels or views are specified, this tag will return all parent categories for the active domain. If views and/or channels are specified, this tag will return parent categories that are assigned to those values passed in. The views and channel assignment are performed on the SiteConnect Management Console.

Format:

<IM:get.category.data

`dataset*="string"`	A unique identifier for the data set. This identifier is used by the iterate.records tag to access records that are retrieved.
`view="string"`	List of views delimited by a + sign that the ad-hoc query will search on. This means that will return category parent records that have assigned the passed in views.
`channel="string"`	List of channels separated b y a + sign that the category query will search on. This means that will return categories parent records that have assigned the passed in channels
`sortby="string"`	This parameter determines which category parameter will be used for sorting. Allowed values are: name and referencekey.
`direction="string"`	The direction of the sort. Ascending or descending. The default if no value provided will be ascending.
`recordid="string"`	ID of the parent category to retrieve. If you provide a value for record id, this tag will return the children for the category referenced by that value
`referencekey="string"`	Reference Key of the parent category record to retrieve. If you provide a value for referencekey, this tag will return the children for the category referenced by that value

/>

Scripting Variables:

{count} is the total number of category objects retrieved.

get.category.record

Description:

This tag returns a category record, either from an iteration or from a reference key passed as a parameter.

Format:

<IM:get.category.record

`recordid="string"`	ID of record to retrieve.
`referencekey="string"`	Reference Key of category record to retrieve.
`id="non-scriptable string"`	A desired prefix to use for all scripting variables created by this tag. There is no default id. Make sure all ids are unique

></IM:get.category.record >

get.channel.attribute

Description:

This tag retrieves a specified attribute from the current content record. This tag can display an attribute without a body section; however, to access the scripting variables, set the `processbody` to `true` and optionally assign an id to the tag.

Format:

```
<IM:get.channel.attribute
```

`attribute="string"`
 	
`(Required)`
	a case-sensitive xpath to the desired content attribute.

`type="string"`	the type of the attribute for formatting purposes. Supported types include: • number • date • text The default is `text`.
`mask="string"`	specifies a pattern that determines exact formatting of a date or number when used with the `type` attribute. For example, you can use a mask to format a number as currency (such as `$1.00` or `$1` instead of the value `1`).

```
></IM:get.channel.attribute >
```

Example Usage:

```
<!--Get channel data records using query-->
<IM:get.channel.data query="topnews" dataset="mynews">

<!-- Iterate channel record -->
<IM:iterate.channel.records dataset="mynews">
   <!--Get handle to current record-->
   <IM:get.channel.record id="news">
      <!-- Display title attribute for current record -->
      <IM:get.channel.attribute attribute="NEWS\TITLE">
      <IM:get.channel.attribute attribute="NEWS\IMAGE" processbody="true" id="image">
         <%if (image.exists) {%>
           <img src="<%=news.resourcepath%>/<%=image.value%>">
         <%}%>
      </IM:get.channel.attribute>
    </IM:get.channel.record>
</IM:iterate.channel.records>
```

Scripting Variables:

None

Required Parent Tags:

None

TagClass:

com.inquira.client.tags.ChannelAttributeTag

get.channel.attribute.exists

Description:

This tag checks the current record to determine whether a specified attribute has a value. If the value is present, it will display the contents of this tag. If the `negate` parameter is set to `true`, this tag will display its contents only if the specified content attribute does not have a value.

Format:

```
<IM:get.channel.attribute.exists
```

`negate="string"` **(Non-scriptable string)**	reverses the conditional behavior of this tag. If the `negate` parameter is set to true, this tag will display its contents only if the specified content attribute does not have a value. The default value is `false`.
`attribute="string"` `(Required)`	specifies the desired attribute to check.

```
></IM:get.channel.attribute.exists >
```

Scripting Variables:

None

Reqired Parent Tags:

get.channel.record

Tag Class:

com.inquira.client.tags.ConditionalAttributeExistsTag

get.channel.attribute.value

Description:

This tag retrieves specified attributes from the current content record as scripting variables.

Format:

```
<IM:get.channel.attribute.value
```

`attribute="string"`
 	
`(Required)`
	a case-sensitive xpath to the desired content attribute.

`id="string"`
 	
`(Non-scriptable string)`
	a desired prefix to use for all scripting variables created by this tag. There is no default id.
	
	NOTE: Ensure that all ids within nested channel-iterator tags are unique.

`type="string"`	the type of the attribute for formatting purposes. Supported types include: • number • date • text The default is `text`.
`mask="string"`	specifies a pattern that determines exact formatting of a date or number when used with the `type` attribute. For example, you can use a mask to format a number as currency (such as `$1.00` or `$1` instead of the value `1`).

```
></IM:get.channel.attribute.value >
```

Example Usage:

```
<!--Get channel data records using query-->
<IM:get.channel.data query="topnews" dataset="mynews">
<!-- Iterate channel record -->
<IM:iterate.channel.records dataset="mynews">
    <!--Get handle to current record-->
    <IM:get.channel.record id="news">
        <!-- Display title attribute for current record -->
        <IM:get.channel.attribute attribute="NEWS\TITLE">
       <IM:get.channel.attribute attribute="NEWS\IMAGE" processbody="true" id="image">
            <%if (image.exists) {%>
                <img src="<%=news.resourcepath%>/<%=image.value%>">
            <%}%>
        </IM:get.channel.attribute>
     </IM:get.channel.record>
</IM:iterate.channel.records>
```

Scripting Variables:

Name	Scope	Description
{id}.exist	tag	indicates whether data exists in the attribute
{id}.value	tag	the value of the current attribute

Required Parent Tags:

None.

TagClass:

com.inquira.client.tags.ChannelAttributeValueTag

Using the Mask Parameter to Format Dates

You can specify explicit display formatting for dates when using the get.channel.attribute tag by specifying:

- A value of date for the type parameter
- A display conversion pattern using the mask parameter

The default mask pattern for a date attribute type is:

```
%Y-%m-%d %H:%M:%S %Z
```

To specify date formatting using the mask parameter:

- Set the type parameter to `date`

The mask parameter supports the following pattern specifiers to format dates for display:

Conversion Specifiers:

%%	a '%' character
%a	abbreviated weekday name
%A	full weekday name
%b	abbreviated month name
%B	full month name
%c	shorthand for "%X %x", the locale format for date and time
%d	day of the month as a decimal number (01-31)
%e	same as %d but does not print the leading 0 for days 1 through 9
%F	milliseconds as a decimal number (000-999)
%H	hour based on a 24-hour clock as a decimal number (00-23)
%I	hour based on a 12-hour clock as a decimal number (01-12)
%j	day of the year as a decimal number (001-366)
%m	month as a decimal number (01-12)
%M	minute as a decimal number (00-59)
%p	AM/PM designation for the locale
%S	second as a decimal number (00-59)
%w	weekday as a decimal number (0-6), where Sunday is 0
%x	date using the date representation for the locale
%X	time using the time representation for the locale
%y	year without century (00-99)
%Y	year with century (such as 1990)
%Z	time zone name (such as Pacific Daylight Time)
%z	time zone name (such as Pacific Daylight Time)

Using the Mask Parameter to Format Numbers

You can specify explicit display formatting for numbers when using the get.channel.attribute tag by specifying:

- A value of number for the type parameter
- A display conversion pattern using the mask parameter

The default mask pattern for a date attribute type is:

```
%Y-%m-%d %H:%M:%S %Z
```

To specify date formatting using the mask parameter:

- Set the type parameter to `date`

The mask parameter supports the following pattern specifiers to format numbers for display:

You can specify a pattern for numeric values using the mask parameter with the get.channel.attribute tag in conjunction with the type tag. For numeric formatting, the type tag must be set to `number`. When the mask parameter is used to convert a numeric value to a string, pattern strings can include the following types of characters:

Numbers	You can specify numeric characters within pattern strings. Numbers in pattern strings display as specified unless an input character in the same relative position overwrites them. For example, if a positive pattern string `9,990.00` is applied to the attribute value is `53.88`, the resulting display is `9,953.88`.
Separators	You can specify the period character (`.`) as a decimal separator, and comma character (`,`) as a thousand separator within pattern strings.

Placeholders	You can specify the pound sign character (#) to represent numeric characters that will be displayed to the user.
	
	For example, if a positive pattern `$#,##0.00` is applied to the attribute value `76329`, the resulting display is `$76,329.00`.
	
	NOTE: Placeholder characters do not serve a programmatic function; they merely serve to make pattern strings more easily read by people.
	
	For example, the strings:
	
	`,0.00`
	`#,#0.00`
	
	and
	
	`#,##0.00`
	
	are functionally equivalent; the mask formatter will use the specified separator characters regardless of the presence or location of any place holders.

Spaces	You can specify the underscore character (_) to include a space in a pattern string. The space character inserts a space if no numeric input character occupies that position.

<table>
<tr><td>Currency</td><td>The dollar sign character ($) is the canonical currency mark in pattern strings.

All other characters specified in a pattern string are displayed as typed. The following examples show the input value 1019.55 is displayed by different positive patterns:<table><tr><th>Pattern</th><th>Display</th></tr><tr><td>"#,##0.00"</td><td>1,019.55</td></tr><tr><td>"$#,##0.00"</td><td>$1,019.55</td></tr><tr><td>"___,__0.00"</td><td>1,019.55</td></tr></table></td></tr>
</table>

get.channel.data

Description:

This tag retrieves valid records from a specified content channel as specified by selection criteria or by a named query. The returned data is assigned to a named variable using the dataset parameter.

You can dynamically define selection criteria at runtime to retrieve specific records based on a user action by specifying additional parameters.

You can also specify these parameters to override selection criteria specified in a named query. Dynamic criteria selection parameters include:

features	List of features delimited by '+' sign to be added to xml stream. Valid values are : categories, departments, security. Default is all features off.
dataset (Required)	A unique identifier for the dataset. This identifier is used by the iterate.records tag to access records that are retrieved.
query	Reference key name of the stored query defined in the Management Console.
channel (Required)	Retrieve all valid records from the specified channels separated by plus(+) signs.
views	Restrict the data set to records assigned to specific views by specifying one or more views separated by plus(+) signs.
usergroups	Restrict the data set to records assigned to specific user groups by specifying one or more user groups separated by plus(+) signs.

where	Sets the where clause to be used on this query.
categories	Restrict the data set to records assigned to specific categories by specifying one or more categories separated by plus(+) signs. NOTE: The matchallcategories parameter specifies whether content records must match all (and) or any (or) of the specified categories.
editorgroups	Restrict the data set to records assigned to specific editor groups by specifying one or more editor groups separated by plus(+) signs.
expression	Expression representing date range.
startdate/enddate	Set or modify the date range of the data set by specifying the start date and end date parameters. NOTE: These parameters are relative to the publish date of a content record. The remove date of a content record is used to restrict access to records after the current date surpasses the end date.
mode	validitydates - content must be valid at specified time, startdates - content must start by specified date.
expression	Expression representing date range.
departments	Deprecated. Replace by views.
maxrecords	Limit the total number of records retrieved to the specified value.
updatedsince	Retrieve content that has been updated after this date.
postalcode	If the passed in channel has GeoSpatial capabilities, fetches records with the passed in postal code and within the proximity radius.
proximity	If the passed in channel has GeoSpatial capabilities, sets the proximity radius.
sortby	List of sort parameters separated b y a + sign.
direction	The direction of the sort. Ascending or descending. The default if no value provided will be ascending.
hierarchicalcategories	Used in conjunction with 'category' attribute. Set to true to use hierarchical categories, false otherwise. It defaults to true.

ignoredisplaydates	Set to true to fetch all content regardless of Display Start and Display End Date restrictions. The default behaviour is not to ignore these dates.
mostpopular	If this attribute is set to true, to query will return the most popular records only. The sortby attribute will be disabled, however you can specify a direction of the sort.
impressions	If this attribute is set to true, to query will return the impression count for each record.
ownerid	If the ownerid (login name) is passed in, the records that are fetched are restricted to the specified owner.
casenumber	If the casenumber is passed in, the records that are fetched are restricted to the specified casenumber.
newcontent	Set to true to return records that the active user has not seen yet. This will only apply if there is a logged in user.
reverseHierarchy	Set to true to return records associated with the parents of the passed in categories. Defaults to false.
viewedby	Return records that the passed in userid has already seen. If you would like to sort by the date seen use VIEWEDDATE in the sort parameters.
stagingonly	If this attribute is set to true, to query will return staging documents regardless if they are published or not.
matchallcategories	Used in conjunction with 'category' attribute. Set to true to 'and' the categories together in query, and false to 'or' the categories.
usecache	Set to true to cache the result set. This option is only for published documents not for staging documents
cachetime	If usecache is set to true, this parameter specifies the time in minutes the cache results will live. The default is set to 1 minute
adaptivequery	Set to false to turn off adaptive query. By default is set to true. This parameter optimizes queries order by startdate, enddate, mostpupular, eventstartdate, eventenddate and createdate. If maxrecords is set to 0, adaptive query will be turned off since all records will be returned. To reset it use cache=refresh on the url.

Format:

`<IM:get.channel.data`	
`features="string"`	a list of features to add to the XML stream. Valid values are: • categories • departments • security The default value is no features.
`dataset="string"` `(Required)`	a unique identifier for the data set. This identifier is used by the iterate.channel.records and get.channel.template tags to access the records that are retrieved.
`query="string"`	a reference key that identifies a stored query defined in the Management Console. IMPORTANT: You cannot specify a query and a channel in the same tag.
`channel="string"`	a content channel in which the desired content records are located. IMPORTANT: You cannot specify a channel and a query in the same tag.
`startdate="string"`	a start date associated with the desired content records.
`enddate="string"`	an end date associated with the desired content records.
`mode="string"`	validitydates - content must be valid at specified time startdates - content must start by specified date
`expression="string"`	an expression that represents a date range.

`casenumber="string"`	a case number to restrict the returned content records. The tag will return only content associated with the specified case numbers.
`ownerid="string"`	a content owner ID to restrict the returned content records. The tag will return only content associated with the specified content owner.
`departments="string"`	Deprecated. Replace by the views parameter.
`views="string"`	a list of views in which to locate content records. Use plus(+) signs to delimit list items. NOTE: This parameter will be used only if the channel attribute is specified, and will not be used if the query attribute is specified.
`categories="string"`	a list of content categories associated with the desired content records. Use plus(+) signs to delimit list items.
`editorgroups="string"`	a list of editor groups associated with the desired content records. Use plus(+) signs to delimit list items.
`where="string"`	sets the where clause to be used by this query.
`sortby="string"`	List of XPaths delimited by a + sign for sorting the results. a list of parameters to determine the sort order of the results. Valid values are: • dateadded • title • casenumber • priority • status Use plus(+) signs to delimit list items.

`direction="string"`	the direction of sorting. Valid values are: • ascending • descending Ascending is the default.
`matchallcategories="true | false"`	specifies whether to: `true` — include only topics associated with all specified categories (and), as specified in the categories parameter. `false` — include topics associated with any specified categories (or), as specified in the categories parameter.
`hierarchicalcategories="true | false"`	specifies whether the categories parameter will use hierarchical categories (`true`) or (`false`). The default is `true`.
`ignoredisplaydates="true" or "false"`	specifies whether to return all content regardless of Display Start and Display End Date restrictions (`true`) or to honor the restrictions (`false`). The default is false.
`maxrecords="integer"`	the maximum number of records to retrieve.
`updatedsince="string"`	specifies to return only content that has been updated since this date.
`/>`	

Example of Calling a Named Query:

```
<IM:get.channel.data query="todaysnews"/>
```

Example of Using Dynamic Selection Criteria:

```
<IM:get.channel.data channel="news"/>
```

Example of Specifying Views within Dynamic Selection Criteria :

```
<IM:get.channel.data channel="news" views="HR+IS+SALES" />
```

Example of Specifying Dates to Get News for Only One Day:

```
<IM:get.channel.data channel="news" startdate="08/31/2005" enddate="08/31/2005" />
```

Scripting Variables:

Name	Scope	Description
{dataset}.count	page	The number of records retrieved.

Body Tags:

None

TagClass:

com.inquira.client.tags.ChannelDataTag

Referring to Stored Content Retrieval Queries

You can define and store named queries for use with the `get.channel.data` tag to easily retrieve content records. You can use the Management Console query builder to define and store any number of content queries as described in the *Information Manager Administration Guide*. You use stored content queries by specifying the stored query in the query parameter of the `get.channel.data` tag.

get.channel.record

Description:

This tag returns a specific channel record using a specified record ID. You can use this tag within the body of an `iterate.channel.records` tag using no additional parameters, since the iterator will automatically pass in the current record. The `id` parameter defines the namespace for scripting variables within the body of the tag.

Format:

```
<IM:get.channel.record
```

`features="string"`	a list of features to add to the XML stream. Valid values are: • categories • departments • security The default value is no features.
`recordid="string"`	the ID of the record to retrieve.
`id="string"` `(Non-scriptable string)`	a desired prefix to use for all scripting variables created by this tag. There is no default id. NOTE: Ensure that all ids within nested channel-iterator tags are unique.
`impressions="true" or "false"`	Attribute to turn or off the impressions for this record. The default value is to increment the impressions. Set to false not to increment impressions.
`documentid="string"`	a document ID of the record to retrieve. This value overrides the record ID value

`activitytype="string"`
 	
`(Non-scriptable string)`
	an activity identifier that is stored with the user activity logs. This value can help in identifying visited pages.

`localecode="string"`	the locale code associated with the current request. The locale code is in the format: language_location For example: en_US *where:* en indicates the language and US indicates the location. The default is the value of the repository default locale.

```
></IM:get.channel.record >
```

Example Stand Alone Usage:

```
<!-- Get passed in record id form URL-->
<% String rec = request.getParameter("rec");%>

<!--Get handle to current record-->
<IM:get.channel.record recordid="<%=rec%>">
<!-- Display title and body attribute for current record -->
<IM:get.channel.attribute attribute="NEWS\TITLE"><br>
<IM:get.channel.attribute attribute="NEWS\BODY">
</IM:get.channel.record>
```

Example Usage within an Iterator:

```
<!--Get channel data records using query-->
<IM:get.channel.data query="topnews" dataset="mynews">
<!-- Iterate channel record -->
<IM:iterate.channel.records dataset="mynews">
<!--Get handle to current record-->
<IM:get.channel.record id="news">
<!-- Display title attribute for current record -->
<IM:get.channel.attribute attribute="NEWS\TITLE">
<!-- Provide hyperlink to detail page for current record -->
<a href="index?page=detail&rec=<%=news.recordid%>>Read more</a><br>
</IM:get.channel.record>
</IM:iterate.channel.records>
```

Scripting Variables:

Name	Scope	Description
`{id}.index`	tag	The index number of the current record. Only valid when used in iterator.
`{id}.recordid`	tag	The ID of the current record.

Required Parent Tags:

None (can be used within a `parent iterate.channel` tag)

TagClass:

com.inquira.client.tags.ChannelItemTag

get.channel.recordid

Description:

Prints to the file the record ID attribute of the current channel record.

Format:

```
<IM:get.channel.recordid
></IM:get.channel.recordid >
```

get.channel.record.exists

Description:

This tag checks whether a specified content record exists or is still valid.

Format:

```
<IM:get.channel.record.exists
```

`documentid="string"`	specifies the document ID of the channel record to be checked.
`negate="string"` `(Non-scriptable string)`	reverses the conditional behavior of this tag. Set this parameter to `true` to display the tag contents (for example, an error message) only when the requested record does not exist or is not valid. The default value is `false`.
`recordid="string"` `(Required)`	specifies the record ID of the channel record to be checked.

```
></IM:get.channel.record.exists >
```

Scripting Variables:

None

Required Parent Tags:

None

TagClass:

com.inquira.client.tags ConditionalChannelItemExistsTag

get.channel.template

Description:

Formats a channel data set using a specified XSL stylesheet.

Format:

<IM:get.channel.template

`dataset="string"`	Name of the <channel-data> recordset that contains the data to be transformed, this name corresponds to the id attribute of the channel-data tag, and is case sensitive
`template*="string"`	The reference key (not name) of the pre-defined XSL template that will be used to transform the data retrieved using the channel-data tag

/>

Example Usage:

This will get content data out of the xmldata variable and transform it using the NewsStyleSheet:

```
<IM:channel-format dataset="xmldata" template="NewsStyleSheet"/>
```

get.config.param.value

Description:

Gets the Management Console configuration parameter for this repository.

<IM:get.config.param.value

`id="string"`	ID for this record.
`propertyName*="string"`	property name to retrieve value of. This is displayed in the config.properties file

></IM:get.config.param.value >

get.content.locales

Description:

This tag retrieves all locales for a specified content record.

Format:

<IM:get.content.locales

`contentid="string"`
 	
`(Required)`
	specifies the ID of the specified content record.

`dataset="string"`
 	
`(Required)`
	specifies the name of a local variable used to store the resulting array of localized record references.

/>

Scripting Variables:

Name	Scope	Description
{defined_name}	page	The variable name is determined by the string passed to the data set parameter.

Reqired Parent Tags:

None

Tag Class:

com.inquira.client.tags.LocalizedDataTag

get.content.data

Description:

This tag retrieves a list of content records using a JDBC Connection pool. This tag looks similar to the get.channel.data tag but all the underlying structure is totally different.

dataset `(Required)`	A unique identifier for the dataset. This identifier is used by the iterate.records tag to access records that are retrieved.
channel `(Required)`	Retrieve all valid records from the specified channels separated by plus(+) signs.
views	Restrict the data set to records assigned to specific views by specifying one or more views separated by plus(+) signs.
usergroups	Restrict the data set to records assigned to specific user groups by specifying one or more user groups separated by plus(+) signs.
categories	Restrict the data set to records assigned to specific categories by specifying one or more categories separated by plus(+) signs. NOTE: The matchallcategories parameter specifies whether content records must match all (and) or any (or) of the specified categories.
editorgroups	Restrict the data set to records assigned to specific editor groups by specifying one or more editor groups separated by plus(+) signs.
startdate/enddate	Set or modify the date range of the data set by specifying the start date and end date parameters. NOTE: These parameters are relative to the publish date of a content record. The remove date of a content record is used to restrict access to records after the current date surpasses the end date.

mode	validitydates - content must be valid at specified time, startdates - content must start by specified date.
maxrecords	Limit the total number of records retrieved to the specified value.
sortby	List of sort parameters separated b y a + sign.
direction	The direction of the sort. Ascending or descending. The default if no value provided will be ascending.
hierarchicalcategories	Used in conjunction with 'category' attribute. Set to true to use hierarchical categories, false otherwise. It defaults to true.
ignoredisplaydates	Set to true to fetch all content regardless of Display Start and Display End Date restrictions. The default behaviour is not to ignore these dates.
mostpopular	If this attribute is set to true, to query will return the most popular records only. The sortby attribute will be disabled, however you can specify a direction of the sort.
impressions	If this attribute is set to true, to query will return the impression count for each record.
ownerid	If the ownerid (login name) is passed in, the records that are fetched are restricted to the specified owner.
casenumber	If the casenumber is passed in, the records that are fetched are restricted to the specified casenumber.
newcontent	Set to true to return records that the active user has not seen yet. This will only apply if there is a logged in user.
reverseHierarchy	Set to true to return records associated with the parents of the passed in categories. Defaults to false.
viewedby	Return records that the passed in userid has already seen. If you would like to sort by the date seen use VIEWEDDATE in the sort parameters.
stagingonly	If this attribute is set to true, to query will return staging documents regardless if they are published or not.
matchallcategories	Used in conjunction with 'category' attribute. Set to true to 'and' the categories together in query, and false to 'or' the categories.
usecache	Set to true to cache the result set. This option is only for published documents not for staging documents
cachetime	If usecache is set to true, this parameter specifies the time in minutes the cache results will live. The default is set to 1 minute

adaptivequery	Set to false to turn off adaptive query. By default is set to true. This parameter optimizes queries order by startdate, enddate, mostpupular, eventstartdate, eventenddate and createdate. If maxrecords is set to 0, adaptive query will be turned off since all records will be returned. To reset it use cache=refresh on the url.

Format:

`<IM:get.content.data`	
`Dataset=”string” (required)`	A unique identifier for the data set. This identifier is used by the iterate.channel.records and get.channel.template tags to access the records that are retrieved.
`channel="string"`	A content channel in which the desired content records are located. You cannot specify a channel and a query in the same tag.
`startdate="string"`	A start date associated with the desired content records.
`enddate="string"`	An end date associated with the desired content records.
`mode="string"`	validitydates - content must be valid at specified time startdates - content must start by specified date
`expression="string"`	An expression that represents a date range.
`casenumber="string"`	A case number to restrict the returned content records. The tag will return only content associated with the specified case numbers.
`ownerid="string"`	A content owner ID to restrict the returned content records. The tag will return only content associated with the specified content owner.
`departments="string"`	Deprecated. Replace by the views parameter.
`views="string"`	A list of views in which to locate content records. Use plus(+) signs to delimit list items. This parameter will be used only if the channel attribute is specified, and will not be used if the query attribute is specified.
`usergroups="string"`	A list of user groups associated with the desired content records. Use plus(+) signs to delimit list items.

`categories="string"`	A list of content categories associated with the desired content records. Use plus(+) signs to delimit list items.
`editorgroups="string"`	A list of editor groups associated with the desired content records. Use plus(+) signs to delimit list items.
`where="string"`	Sets the where clause to be used by this query.
`sortby="string"`	List of XPaths delimited by a + sign to determine the sort order of the results. Valid values are: CONTENTID CONTENTTEXTID INDEXMASTERIDENTIFIERS LOCALEID OWNERID OWNERNAME PRIORITYID PUBLISHEDDATE RECORDID REQUIRESTRANSLATION DATEADDED DATEMODIFIED CREATEDATE DISPLAYENDDATE DISPLAYSTARTDATE EVENTENDDATE EVENTSTARTDATE DOCUMENTID VIEWEDDATE
`direction="string"`	The direction of sorting. Valid values are: • ascending • descending Ascending is the default.
`hierarchicalcategories= "true | false"`	Specifies whether the categories parameter will use hierarchical categories (`true`) or (`false`). The default is `true`.
`ignoredisplaydates= "true" or "false"`	Specifies whether to return all content regardless of Display Start and Display End Date restrictions (`true`) or to honor the restrictions (`false`). The default is false.
`maxrecords="integer"`	The maximum number of records to retrieve.

`mostpopular= "true | false"`
 	
`/>`
	If this attribute is set to true, to query will return the most popular records only. This attribute overrides the sortby attribute, but you can use the direction attribute to specify a direction of the sort.

Example of Calling a Named Query:

```
<IM:get.content.data query="todaysnews"/>
```

Example of Using Dynamic Selection Criteria:

```
<IM:get.content.data channel="news"/>
```

Example of Specifying Views within Dynamic Selection Criteria :

```
<IM:get.content.data channel="news" views="HR+IS+SALES" />
```

Example of Specifying Dates to Get News for Only One Day:

```
<IM:get.content.data channel="news" startdate="08/31/2005" enddate="08/31/2005" />
```

Scripting Variables:

Name	Scope	Description
{dataset}.count	page	The number of records retrieved.

Body Tags:

None

TagClass:

com.inquira.client.tags.ContentDataTag

get.content.record

Description:

This tag retrieves a content record from an iteration.

Format:

```
<IM:get.content.record
```

`id="string"`
 	
`(Non-scriptable string)`
	specifies a desired prefix to use for all scripting variables created by this tag. Each id must be unique. There is no default id.

```
></IM:get.content.record >
```

Scripting Variables:

Name	Scope	Description
{record}.recordid	page	the value of the CONTENTID attribute for the returned content record.
{record}.docid	page	the value of the DOCUMENTID attribute for the returned content record.
{record}.masteridentifier	page	the value of the INDEXMASTERIDENTIFIERS attribute for the returned content record.
{record}.displayStartDate	page	the value of the DISPLAYSTARTDATE attribute for the returned content record.
{record}.displayEndDate	page	the value of the DISPLAYENDDATE attribute for the returned content record.
{record}.eventStartDate	page	the value of the EVENTSTARTDATE attribute for the returned content record.
{record}.eventEndDate	page	the value of the EVENTENDDATE attribute for the returned content record.
{record}.viewcount	page	the value of the COUNT - if (mostpopular) attribute for the returned content record.
{record}.datemodified	page	the value of the DATEMODIFIED attribute for the returned content record.

Required Parent Tags:

iterate.dataset (generic iterator tag)

get.contenthistory.data

Description:

This tag retrieves a content history from an iteration.

Format:

```
<IM:get.contenthistory.data
```

`dataset="string"` `(Required)`	A unique identifier for the dataset. This identifier is used by the iterate.records tag to access records that are retrieved.
`documentid="string"`	A content documentID. Prepend with 'S:' for unpublished content. A documentID OR recordID is required.
`recordid="string"`	A content recordID. Prepend with 'S:' for unpublished content. A recordID OR documentID is required.
`localecode="string"`	Returns records relating to content given localecode.
`login="string"`	Returns records relating to a given user login.
`version="string"`	Version for which to return history.
`maxversions="string"`	Number of versions before the version for which to retrieve history.

```
></IM:get.contenthistory.data >
```

Example Usage:

```
<h3>History</h3>
<IM:get.contenthistory.data dataset="rsData" documentid="<%=strID%>"/>
<%
if (rsData.count > 0) {
%>
<table border="0" cellpadding="2" cellspacing="0">
<tbody>
<thead>
<td><b>User</b></td><td><b>Action</b></td><td><b>Version</b></td><td><b>When</b></
td>
</thead>
<IM:iterate.dataset dataset="rsData" id="itData">
<IM:get.contenthistory.record id="crData">
<tr>
<td>
```

```
<IM:get.user.record id="historyUser" login="<%=crData.login%>" generatexml="false">
<a href="index?page=user_profile&user=<%=historyUser.guid%>">
<%
if(historyUser.showName){
%>
<%=historyUser.firstname%> <%=historyUser.lastname%>
<%
}else{
if(historyUser.alias!=null){
%>
<%=historyUser.alias%>
<%
}else{
%>
<%=historyUser.firstname%> <%=historyUser.lastname%><%
}
}
%>
</a>
</IM:get.user.record>
</td>
<td><%= crData.action%></td>
<td><%=crData.version%></td>
<td><%=timeAgo(crData.date)%></td>
</tr>
<% if( crData.comment!=null ) { %>
<tr>
<td colspan="4"><%= crData.comment%></td>
</tr>
<%
}
%>
</IM:get.contenthistory.record>
</IM:iterate.dataset>
</tbody>
</table>
<%
}
%>
```

Scripting Variables:

None

Required Parent Tags:

iterate.dataset (generic iterator tag)

get.contenthistory.record

Description:

This tag retrieves a content history record from an iteration.

Format:

```
<IM:get.contenthistory.record
```

`id="string"`	A desired prefix to use for all scripting variables created by this tag.

```
></IM:get.contenthistory.record >
```

Example Usage:

```
<h3>History</h3>
<IM:get.contenthistory.data dataset="rsData" documentid="<%=strID%>"/>
<%
if (rsData.count > 0) {
%>
<table border="0" cellpadding="2" cellspacing="0">
<tbody>
<thead>
<td><b>User</b></td><td><b>Action</b></td><td><b>Version</b></td><td><b>When</b></
td>
</thead>
<IM:iterate.dataset dataset="rsData" id="itData">
<IM:get.contenthistory.record id="crData">
<tr>
<td>
<IM:get.user.record id="historyUser" login="<%=crData.login%>" generatexml="false">
<a href="index?page=user_profile&user=<%=historyUser.guid%>">
<%
if(historyUser.showName){
%>
<%=historyUser.firstname%> <%=historyUser.lastname%>
<%
}else{
if(historyUser.alias!=null){
%>

<%=historyUser.alias%>
<%
}else{
%>
```

```
<%=historyUser.firstname%> <%=historyUser.lastname%><%
}
}
%>
</a>
</IM:get.user.record>
</td>
<td><%= crData.action%></td>
<td><%=crData.version%></td>
<td><%=timeAgo(crData.date)%></td>
</tr>
<% if( crData.comment!=null ) { %>
<tr>
<td colspan="4"><%= crData.comment%></td>
</tr>
<%
}
%>
</IM:get.contenthistory.record>
</IM:iterate.dataset>
</tbody>
</table>
<%
}
%>
```

Scripting Variables:

Name	Scope	Description
{id}.action	tag	The action name for the ContentHistory record, such as Localized, Created, Edited, Approved, Rejected, Published, Modified Site Visibility, Modified Category, Unpublished, Reverted Localization Requested, Localization Cleared, Metadata changed, Workflow Comment Added.
{id}.login	tag	The user login that made the change.
{id}.version	tag	The version of the content.
{id}.comment	tag	Comment text.
{id}.date	tag	Date of history entry.

Required Parent Tags:

iterate.dataset (generic iterator tag)

get.dataform.answer

Description:

Retrieves the answer for the current data form answer record.

Format:

```
<IM:get.dataform.answer
/>
```

Example Usage for Displaying the Form:

```
<IM:form.dataform dataform="INQUIRIES" success="requestthank" error="requestthank">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:input.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question> <br>
<input type="submit" value="Submit">
</IM:form.dataform>
```

Example Usage for Displaying Results:

```
<IM:get.dataform.results name="nameofform" individual="false" aggregate="false">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:get.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question>
</IM:get.dataform.results>
```

Scripting Variables:

None

Body Tags:

None

get.dataform.answer.record

Description:

This tag provides a handle to the current data form answer record.

Format:

```
<IM:get.dataform.answer.record
></IM:get.dataform.answer.record >
```

Example of Displaying the Form

```
<IM:form.dataform dataform="INQUIRIES" success="requestthank" error="requestthank">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:input.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question> <br>
<input type="submit" value="Submit">
</IM:form.dataform>
```

Example of Displaying Results

```
<IM:get.dataform.results name="nameofform" individual="false" aggregate="false">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:get.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question>
</IM:get.dataform.results>
```

Scripting Variables:

Name	Scope	Description
{id}.index	page	The index number of the current record.
{id}.recordid	page	The ID of the current record.

Body Tags:

get.dataform.answer on page 141

TagClass:

com.inquira.client.tags.SurveyAnswerItemTag

get.dataform.name

Description:

This tag returns the localized name of a specified data form by reference key passed as a parameter.

Format:

<IM:get.dataform.name

`referencekey*="string"`	The reference key of the desired Data Form.

></IM:get.dataform.name >

get.dataform.results

Description:

This tag retrieves the results associated with a specified data form. You can specify parameters to return individual and aggregate results.

Format:

```
<IM:get.dataform.results
```

`dataform="string"`	the name of the survey.
`individual="true | false"`	specifies whether to return individual results (`true`).

`aggregate="true" or "false"`	specifies whether to return aggregate results (`true`).

```
></IM:get.dataform.results >
```

Example Usage for Displaying the Form:

```
<IM:form.dataform dataform="INQUIRIES" success="requestthank" error="requestthank">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:input.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question> <br>
<input type="submit" value="Submit">
</IM:form.dataform>
```

Example Usage for Displaying Results:

```
<IM:get.dataform.results dataform="nameofform" individual="false" aggregate="false">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:get.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question>
</IM:get.dataform.results>
```

Scripting Variables:

None.

TagClass:

com.inquira.client.tags.SurveyResultsTag

get.dataform.question

Description:

Returns the question for the current data form question record.

Format:

```
<IM:get.dataform.question
/>
```

Example Usage for Displaying the Form

```
<IM:form.dataform dataform="INQUIRIES" success="requestthank" error="requestthank">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:input.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question> <br>
<input type="submit" value="Submit">
</IM:form.dataform>
```

Example Usage for Displaying Results

```
<IM:get.dataform.results name="nameofform" individual="false" aggregate="false">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:get.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question>
</IM:get.dataform.results>
```

Body Tags:

None

TagClass:

com.inquira.client.tags.SurveyQuestionTag

get.dataform.question.record

Description:

This tag provides a handle to the current data form question record.

Format:

```
<IM:get.dataform.question.record
></IM:get.dataform.question.record >
```

Example Usage for Displaying the Form

```
<IM:form.dataform dataform="INQUIRIES" success="requestthank" error="requestthank">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:input.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question> <br>
<input type="submit" value="Submit">
</IM:form.dataform>
```

Example Usage for Displaying Results

```
<IM:get.dataform.results name="nameofform" individual="false" aggregate="false">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:get.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question>
</IM:get.dataform.results>
```

Scripting Variables:

Name	Scope	Description
{id}.index	page	The index number of the current record.
{id}.recordid	page	The ID of the current record.

Body Tags:

get.dataform.question on page 146

iterate.dataform.answer on page 254

get.dataform.answer.record on page 142

get.dataform.answer on page 141

TagClass:

com.inquira.client.tags.SurveyQuestionItemTag

get.dataset.batch.page

Description:

This tag returns the current page from the batch page iterator.

```
<IM:get.dataset.batch.page
id="non-scriptable string"
></IM:get.dataset.batch.page >
```

A desired prefix to use for all scripting variables created by this tag. There is no default id. All ids must be unique.

get.domain.attribute

Description:

Get a system or custom domain attribute. Only one of the attribute parameters can be specified at one time.

Format:

<IM:get.domain.attribute

`attribute="string"`	xpath of extended site attribute to return. Note that extended site properties are defined via schema in the admin tool.
`systemattribute="string"`	Basic site property value to return. Valid values are: **domain** domain name of site **refkey** locale-neutral reference key for site
`view="string"`	Reference key for the repository view to return property for

/>

Example Usage:

```
<html>
<head>
<meta content="<IM:get.domain.attribute attribute="demo/keywords" />" name="KEYWORDS">
<title><IM:get.domain.attributes systemattribute="domain"/></title>
</head>
<body>
...
</body>
</html>
```

Scripting Variables:

none

Body Tags:

none

get.inquirasearch.data

Description:

This tag submits a request to the Inquira search engine by means of a SOAP call.

Format:

<IM:get.inquirasearch.data

type="string" (Required)	The type of SOAP request to make. Valid values are: empty – this is the first call to the search in order to get all of the facet filters search – perform a search based on the `searchstring` parameter narrow – turn a facet on or off by passing in the `facetid` in the `facettoggle` parameter forward, backward – for paging. Go to the next or previous set of answers. more – pass in the facetid of a facet that has its `isHasMore()` property set to true. There will be more sub facets returned for a particular facet. open – open an external document with the highlighting, In order for this request to work, you will need to pass in the `answerid`, `highlightinfo`, `iqaction`, and url of the answer you want to open. All 4 fields are members of the search answer variable (InquiraResult). wizard – available but not used by the TagLibrary. Use the `form.wizardfields` tag instead.

	link – used to open a click-thru link in order to log the analytics. Any time a search result link is opened or a portlet link is opened, a link request should be made and then redirect to the link by getting the `id.clickthrough` value. feedback – record a feedback from the ussr. Either the request can send the `feedbackcomments` with a user's textual comments, or a `feedbackrating` with a numeric value rating the question or both. similar – when a search result indicates that it has a similar result, you can create a link that will set the type to "similar" by checking if rsAns.similar_count > 0, and passing in the `answerid` and the `relatedIds`. The page will then contain the similar answers. same – to return to the last search results that were displayed, pass in a type of `same`. InitialContact – this is an equivalent of the empty request for a Submit A Case Online search request. This needs to be called to initialize the search before calling `AnswerContact` when the user searches before submitting a case. AnswerContact – this search request is used to perform a search in order to satisfy the submit a case request for escalation. RespondContact – this will inform the search instance of whether or not the user was satisfied. If the escalate field is populated with "true", then a case escalation will be logged in the search instance for analytics to record. If the escalate is set to false, then the user was satisfied with one of the search results.
`searchstring="string"`	The question string to search for
`facettoggle="string"`	The facet's id to toggle on or off for filtering results

`dataset="string"` (Required)	A unique identifier for the dataset, this identifier is used by the iterate.dataset to step through all of the returned answers and facets. To retrieve the facets or answers simply append a `.facets` or `.answers` to the value passed in.
pageobj=`"string"`	Future-not used now.
classlevel0=`"string"`	Have the formatted answers that are returned by the answer.getExcerpt method use a specified css class instead of the default "sippetLevel0"
classlevel1=`"string"`	Have the formatted answers that are returned by the answer.getExcerpt method use a specified css class instead of the default "sippetLevel1"
classlevel2=`"string"`	Have the formatted answers that are returned by the answer.getExcerpt method use a specified css class instead of the default "sippetLevel2"
classlevel3=`"string"`	Have the formatted answers that are returned by the answer.getExcerpt method use a specified css class instead of the default "sippetLevel3"
restrict=`"string"`	Tell the inquira search to only look for InfoManager documents, Discussions or Non-Discussions. The three valid values are IM, IM_DISCUSSION, and IM_CHANNEL respectively. If this parameter is not specified, no restrictions will be applied.
parsedHTML=`"string"`	An "open" request will return the html into this variable. The html will contain highlighted text.
iqaction=`"string"`	Value returned with an answer that is used for a subsequent 'open' document request.
highlightinfo=`"string"`	Information needed to create highlighting for an "open" request". Part of a returned answer.
answerid=`"string"`	A unique value for an answer. Used when calling the request to "open" an external document. Returned for each answer
relatedids=`"string"`	A unique value for an answers related ids. Used when calling the request to find "similar" answers.
url=`"string"`	The url for the target document.
id=`"string"` (Required)	The id of the target document.

feedbackcomments="string"	Comments regarding how well the search has performed. Use feedbackcomments or feedbackratings
feedbackrating="string"	A value from 1 to 5 indicating the rating of the search results, where 5 is the best rating. Use feedbackrating or feedbackcomments
segment="string"	A value passed into the SOAP request
escalate="string"	Used for a RespondContact search request type. Indicates whether or not to escalate based on user satisfaction
params="string"	A + delimited way of passing in extra params along with the SOAP request
views="string"	A + delimited way of passing in views to restrict by
validate="true | false"	Vallidate the search results against the database. Defaults to false
ccaanswersolutionslist= "string"	used by an AddSolution request
ccaextsollist="string"	used by an AddSolution request
ccasrkey="string"	used by an AddSolution request
ccatypes="string"	used by an AddSolution request
baseurl="string"	For analytics. To determine the origination of the request
querysrc="string"	For analytics. To determine the page of the request
uniqueid="string"	For analytics. To determine the page of the request
facetcollectionid="string"	Allows completely different navigation taxonomies
domaingroup="string"	Allows completely different navigation taxonomies
questiontype="string"	Assosiated with the questiontype(subject) and language to determine a domaingroup by search engine
wizardlabel="string"	For analytics. To determine the page of the request
stepuuid="string"	For analytics. To determine the page of the request
wizardid="string"	For analytics. To determine the page of the request

wizardstepid="string"	For analytics. To determine the page of the request
localecode="string"	Desired locale code for this search. If not valid or an exception occurs, the session's locale will be used.
resultlanguage="string"	Used as part of the constraint for the result filtering.

/>

Example Usage:

```
<html>
<body>
<IM:get.inquirasearch.data type="<%=type%>" dataset="rsData" id="srch"
segment="<%=userSegment%>"  searchstring="<%=question%>"
   facettoggle="<%=fac%>" pageobj="pag" restrict="<%= restrict %>" url="<%=url%>"
highlightinfo="<%=highlightinfo%>"
   iqaction="<%=iqaction%>" answerid="<%=answerid%>" relatedids="<%= related %>"
feedbackrating="<%=feedbackrating%>" feedbackcomments="<%=feedback%>" />
</body>
</html>
```

Scripting Variables:

Variable	Description
parsedHTML	An "open" request will return the HTML into this variable. The HTML will contain highlighted text. This is also used when passing in the XML for PDF highlighting.
iqxml	The raw IQXML containing the response.
pageobj	The pageobj parameter allows you get back specific information about the result set returned, such as if there are more results to navigate to, and it is called by whatever is set as id. For example: id="myid" … myid.pageobj.totalResults would return the total number of results.
facetcount	The number of facets returned.
wizardcount	The number of process wizards returned.
portletcount	The number of portlets returned.
question	The question that was asked.
clickthrough	The URL to jump to after a link request.
allowhighlight	The repository option to allow highlighting.
allowsimilar	The repository level option to allow similar results to be shown.
showingsimilar	Returns True if similar results are being shown instead of all results.
pagewarp	If not null, contains a page warp location to redirect to: `response.sendRedirect(id.pagewarp)`
foundFeatureContent	Boolean variable to indicate whether or not there is a FEATURE_CONTENT portlet that needs to be shown below the search results. You can control the title of the search results to "Answers" if this is false, or "Best Answers" if it is true.

Body Tags:

none

TagClass:

com.inquira.client.tags.ChannelDataTag

get.inquirasearch.answer

Description

Tag that receives the current InquiraResult object from inside the `iterate.dataset` tag.

Format

<IM:get.inquirasearch.answer

`id = "string"` (Required)	The unique ID representing the InquiraResult

/>

Example Usage:

```
<html>
<head>...
</head>
<body>
... <get.inquirasearch.answer id=”ns”>

</body>
</html>
```

Scripting Variables:

Variable	Description
<%= ans.display %>	the text to show for the link (the heading)
<%= ans.excerpt %>	a convenience method to return an HTML formatted string of all the snippets. It uses the supplied CSS class levels or defaults to "snippetClassx" where x is a value from 0-3.
<%= ans.clickThrough() %>	the click-thru link returned by Inquira Search

The next 3 are for creating a relevancy “progress bar” table.

Variable	Description
<%= ans.scoretext %>	determine the "score" html bar's percentage text
<%= ans.scorewidth %>	determine the "score" html bar's width
<%= ans.nonscorewidth %>	determine the "score" html bar's width - The right side of the two-celled table
<%= ans.score %>	Float – how closely this answer fits the question
<%= ans.uimode %>	currently, should be “answer”
<%= ans.highlightinfo %>	used by the Inquira engine to highlight relevant text inside an external document – not much use for the tags
<%= ans.answerid %>	
<%= ans.iqaction %>	
<%= ans.url %>	a URL to link to the actual document
<%= ans.cmsstatus %>	information on the Information Manager document specifying whether the content record is published or not
<%= ans.cmsguid %>	The GUID for an Information Manager record
<%= ans.isExternalContent %>	Boolean – an Information Manager content document in an external repository
<%= ans.isInternalContent %>	Boolean – an Information Manager content document in the current repository
<%= ans.isExternalDocument %>	Boolean – not an Information Manager document
<%= ans.iscontentdeleted %>	Boolean – An Information Manager document that has been indexed, but since been deleted. True if the content GUID cannot be found in the database
<%= ans.docType %>	String – the document type of this answer
<%= ans.snippets %>	NSMutableArray - each part of the text is contained in snippets that have various levels that show how closely the search term is qualified. Each snippet is concatenated into the excerpt for convenience and into span tags with the appropriate class levels.
<%= ans.similar_count %>	Integer – the number of similar results for the current answer

<%= ans.relatedIds %>	String – the IDs of related answers. This is used for a “similar” request
<%= ans.escurl %>	String – the html escaped url that is used to pass in a “link” request. Each snippet is of type InquiraSnippet and simply contains getLvl() and getText() methods
<%= ans.isWizard %>	Boolean – True if the answer is a type Process Wizard. If the answer is a wizard, you can modify the <a> to append the wizardid and the wizardstepid in order to pass the arguments into a `form.wizardfields` tag.
<%= ans.wizardLabel %>	String – the process wizard’s display label
<%= ans.wizardDesc %>	String – the description of the Process Wizard
<%= ans.wizardFirstStep %>	identifies which step ID the Process Wizard should start at
<%= ans.wizardId %>	String – identifies which Process Wizard to use
<%= ans.wizardDefaultStep %>	String – not currently used
<%= ans.wizard %>	Wizard – contains the actual raw Wizard object

The wizard fields above may be used like this:

```
if (rsAns.isWizard){ %>
        <a href="http://10.0.20.76:8080/TagLibrary/
index?page=<%=myPage%>&type=wizard&answerid=<%=rsAns.answerid%>&iqaction=<%=rs
Ans.iqaction%>&wizardid=<%=rsAns.wizardId%>&wizardstepid=<%=rsAns.wizardFirstStep%>
&wizardnextstep= "><%=rsAns.wizardLabel%> <br>
   </a> <%=rsAns.wizardDesc%><br>
```

Body Tags:

none

get.inquirasearch.answerfacet

Description

Tag that handles facets for answers. This tag receives the current InquiraResult object from inside the `iterate.dataset` tag.

Format

```
<IM:get.inquirasearch.answerfacet
id = "string"
(Required)
/>
```

The unique ID representing the AnswerFacet.

Example Usage:

```
<IM:iterate.dataset dataset="rsAns.facets"><IM:get.inquirasearch.answerfacet id="ansFac">
<% if ((ansFac != null) && (!ansFac.hidden) && (ansFac != null) && (ansFac.item(1) !=null)) {
boolean sli = !ansFac.item(1).selected;
String cla = "";
if (sli){
cla = "qna-answer-facet-link";
} else {
cla = "qna-answer-facet-selected";
}
%>
<div class="qna-answer-facet-block">
<span class="qna-answer-facet-title"><%= ansFac.item(0).description(0) %>:</span>
<% if (sli) { %><a class="<%= cla %>"
href="index?page=<%=strPage%>&type=narrow&fac=<%= ansFac.item(1).id%>" ><% }
%><%= ansFac.item(1).description(0) %><%if (sli) { %></a><% } %>
</div>
<% }%>
</IM:get.inquirasearch.answerfacet> </IM:iterate.dataset>
```

Scripting Variables:

Variable	Description
<%= ansFac.hidden %>	Boolean that is True if this answer facet is hidden.
<%= ansFac.items.count() %>	Number of items.
<%= ansFac.item(0) %>	The first item.
<%= ansFac.item(0).shape %>	The shape of the item
<%= ansFac.item(0).coords %>	The coordinates of the item
<%= ansFac.item(0).id %>	The item ID.

An Item can contain any number of Fields, Descriptions or Datas:

<%= ansFac.item(0).fields.count() %>	Number of fields
<%= ansFac.item(0).descriptions.count() %>	Number of descriptions
<%= ansFac.item(0).datas.count() %>	Number of datas
<%= ansFac.item(0).field(1) %>	Getting second field
<%= ansFac.item(0).description(1) %>	Getting second description
<%= ansFac.item(0).data(1) %>	Getting second data

Descriptions and Datas are returned as strings:

<%= ansFac.item(0).description(1) %>	String description
<%= ansFac.item(0).data(1) %>	String data

A Field can have the following properties:

<%= ansFac.item(0).field(0).content %>	String content
<%= ansFac.item(0).field(0).id %>	String id
<%= ansFac.item(0).field(0).type %>	String type
<%=a nsFac.item(0).field(0).description %>	String description
<%= ansFac.item(0).field(0).orig %>	String orig
<%= ansFac.item(0).field(0).isSharedType %>	Boolean isSharedType

A Field can contain any number of Maps or Options:

<%= ansFac.item(0).field(0).maps.count() %>	Get the map count
<%= asFac.item(0).field(0).options.count() %>	Get the option count
<%= ansFac.item(0).field(0).map(0) %>	Get first map
<%= ansFac.item(0).field(0).option(0) %>	Get first option

An Option can have the following properties:

<%= ansFac.item(0).field(0).option(0).content %>	String content
<%= ansFac.item(0).field(0).option(0).value %>	String value
<%= ansFac.item(0).field(0).option(0).id %>	String id
<%= ansFac.item(0).field(0).option(0).selected %>	String selected

A Map can have the following properties:

<%= ansFac.item(0).field(0).map(0).field %>	String field
<%= ansFac.item(0).field(0).map(0).variable %>	String variable

Body Tags:

none

get.inquirasearch.facet

Description

Tag that receives the current ResultFacet object from inside the iterate.dataset tag

Format

<IM:get.inquirasearch.facet

`id = "string"` (Required)	The unique id representing the ResultFacet

/>

Example Usage:

```
<html>
<head>

</head>
<body>
... <IM:get.inquirasearch.facet id="fac">
</body>
</html>
```

Scripting Variables:

Variable	Description
<%= fac.active %>	True if this facet is being used as a filter
<%= fac.count %>	the number of sub-facets in existence (not necessarily the number of facets returned)
<%= fac.display %>	the textual description to display
<%= fac.referencekey %>	a unique ID for this facet returned by InquiraSearch. An example might be CMS-CATEGORY-GILTEST-SPORTS.Football
<%= fac.childInEffect %>	a sub-facet of type ResultFacet, if any, that is being used as a filter
<%= fac.parent %>	a sub-facet's reference to its parent facet
<%= fac.subFacets %>	A list of all sub-facets under the next level on the current facet's tree
<%= fac.hasMore %>	Indicates whether or not there are more sub-facets that could be retrieved. If, for example, there are 1000 authors, only 10 may be returned in the first request
<%= fac.level %>	indicates the level of this facet in the hierarchy, starting at level 1

Body Tags:

none

get.inquirasearch.tablecell

Description

Tag that receives the current InquiraResult object from inside the `iterate.dataset` tag.

Format

```
<IM:get.inquirasearch.tablecell
```

`id="string"`
 	
`(Required)`
	A unique identifier representing the TableCell. This identifier is used by the iterate.records tag to access TableCells that are retrieved.

```
/>
```

Example Usage:

```
<% if (itemx.table != null){ %>
</tr><tr><td class="im-first"></td><td class="im-last">
<TABLE class="qna-result-table" border="1" cellpadding="0" cellspacing="0" >
<IM:iterate.dataset dataset="itemx.rows">
<IM:get.inquirasearch.tablerow id="tabrow">
<TR>
<IM:iterate.dataset dataset="tabrow.cells">
<IM:get.inquirasearch.tablecell id="tabcell">
<% if (tabrow.rownum == 1) { %>
<TD class="qna-result-table-header"><%= tabcell %> </TD>
<% } else { %>
<TD class="qna-result-table-text"><%= tabcell %> </TD>
<% } %>
</IM:get.inquirasearch.tablecell>
</IM:iterate.dataset>
</TR>
</IM:get.inquirasearch.tablerow>
</IM:iterate.dataset>
</TABLE>
```

Body Tags:

none

get.inquirasearch.tablerow

Description

Tag that receives the current InquiraResult object from inside the `iterate.dataset` tag.

Format

```
<IM:get.inquirasearch.tablerow
```

`id="string"`
 	
`(Required)`
	A unique identifier representing the TableRow. This identifier is used by the iterate.records tag to access TableRows that are retrieved.

```
/>
```

Example Usage:

```
<% if (itemx.table != null){ %>
</tr><tr><td class="im-first"></td><td class="im-last">
<TABLE class="qna-result-table" border="1" cellpadding="0" cellspacing="0" >
<IM:iterate.dataset dataset="itemx.rows">
<IM:get.inquirasearch.tablerow id="tabrow">
<TR>
<IM:iterate.dataset dataset="tabrow.cells">
<IM:get.inquirasearch.tablecell id="tabcell">
<% if (tabrow.rownum == 1) { %>
<TD class="qna-result-table-header"><%= tabcell %> </TD>
<% } else { %>
<TD class="qna-result-table-text"><%= tabcell %> </TD>
<% } %>
</IM:get.inquirasearch.tablecell>
</IM:iterate.dataset>
</TR>
</IM:get.inquirasearch.tablerow>
</IM:iterate.dataset>
</TABLE>
```

Body Tags:

none

get.inquirasearch.portlet

Description:

Tag that receives the current Portlet object from inside the iterate.dataset tag

Format:

```
<IM:get.inquirasearch.portlet
id="string"
(Required)

/>
```

the unique ID representing the Portlet

Example Usage:

```
<html>
<head>

</head>
<body>
... <Table border=0 align=right width=20%><tr><td>

<IM:iterate.search.portlets dataset="rsData.portlets"
order="PROMOTE+DEFINE+RELATED_TOPIC+FEEDBACK">
  <IM:get.inquirasearch.portlet id="porter">
  <% if (porter.type.equals("feedback")) { %>
        <form name="feedbackform" action="index" method=get>
        <input type=hidden name=page value="<%=myPage%>">
        <input type=hidden name=type value=feedback>
  <% } %>
  <table border=1 width=100%> <tr><th class="im-lightblue"><%= porter.name %></th> </tr>
           <IM:iterate.dataset dataset="porter.items" id="pitems">
             <IM:get.inquirasearch.portlet.item id="item">
<TR><td>Answer ID = <%= item.answerid %> <% if (porter.type.equals("feedback")) { %>
                  <%= item.excerpt %><BR>
                 <% if (pitems.index == 4) { %>
                       <input type=submit name=type value=submit>
                       </form>
                  <% } %>
        <% } else { %>
        <% if (item.ansType.equals("dictionary")) { %>
<a
href="index?page=<%=myPage%>&type=search&showdef=true&title=<%=item.getLinkText()%
>&def=<%=item.excerpt%>&answerid=<%=item.answerid%>&iqaction=<%=item.iqaction%>"><
%= item.getLinkText() %></a> <br><%= item.excerpt %>
<BR><BR>

                <% } else {%>
```

```
                <a href="<%=item.titleUrl%>"><%= item.getLinkText() %></a> <br><%=
item.excerpt %><BR><BR>
                <% } %>
        <% } %>
              </td></tr>
               </IM:get.inquirasearch.portlet.item>
             </IM:iterate.dataset>

  </IM:get.inquirasearch.portlet>
  </IM:iterate.search.portlets>

</td><tr></Table>

</body>
</html>
```

Scripting Variables:

Name

type

Body Tags:

none

get.locale.attribute

Description:

This tag displays the selected attribute for a locale. This tag must be included inside a get.locale.record tag.

Format:

```
<IM:get.locale.attribute
```

attribute="string" (Required)	the name of the attribute to display. The following are supported attribute values:
encoding	returns the character encoding of the locale
dateformat	returns the date format string of the locale used for storing the data
recordid	returns the record ID (guid) of the locale
dateformatdisplay	returns the date format display string for displaying the locale dates
timeformat	returns the time format string of the locale used for storing the data
timeformatdisplay	returns the time format string of the locale used for storing the data
description	the localized description of the locale description
code	the locale code for the locale

```
></IM:get.locale.attribute >
```

Scripting Variables:

None

Required Parent Tags:

```
get.locale.record
```

TagClass:

com.inquira.client.tags.LocalePropertyTag

get.locale.record

Description:

Retrieves locale information for a locale.

```
<IM:get.locale.record
```

`recordid="string"`	Record ID of content record to find localized versions of
`id="non-scriptable string"`	Name of a local variable used to store the resulting array of localized record references

```
></IM:get.locale.record >
```

get.inquirasearch.portlet.item

Description:

Tag that receives the current Portlet's InquiraResult object from inside the iterate.dataset tag

Format:

```
<IM:get.inquirasearch.portlet.item
```

`id="string"` (Required)	The unique id representing the Portlet's InquiraResult

```
/>
```

Example Usage:

```
<html>
<head>

</head>
<body>
... ... <Table border=0 align=right width=20%><tr><td>

<IM:iterate.search.portlets dataset="rsData.portlets"
order="PROMOTE+DEFINE+RELATED_TOPIC+FEEDBACK">
  <IM:get.inquirasearch.portlet id="porter">
  <% if (porter.type.equals("feedback")) { %>
        <form name="feedbackform" action="index" method=get>
        <input type=hidden name=page value="<%=myPage%>">
```

```
        <input type=hidden name=type value=feedback>
 <% } %>
 <table border=1 width=100%> <tr><th class="im-lightblue"><%= porter.name %></th> </tr>
           <IM:iterate.dataset dataset="porter.items" id="pitems">
             <IM:get.inquirasearch.portlet.item id="item">
<TR><td>Answer ID = <%= item.answerid %> <% if (porter.type.equals("feedback")) { %>
                   <%= item.excerpt %><BR>
                  <% if (pitems.index == 4) { %>
                        <input type=submit name=type value=submit>
                        </form>
                   <% } %>
        <% } else { %>
        <% if (item.ansType.equals("dictionary")) { %>
<a
href="index?page=<%=myPage%>&type=search&showdef=true&title=<%=item.getLinkText()%
>&def=<%=item.excerpt%>&answerid=<%=item.answerid%>&iqaction=<%=item.iqaction%>"><
%= item.getLinkText() %></a> <br><%= item.excerpt %>
<BR><BR>

                <% } else {%>
                <a href="<%=item.titleUrl%>"><%= item.getLinkText() %></a> <br><%=
item.excerpt %><BR><BR>
                <% } %>
      <% } %>
            </td></tr>
             </IM:get.inquirasearch.portlet.item>
           </IM:iterate.dataset>

 </IM:get.inquirasearch.portlet>
 </IM:iterate.search.portlets>

</td><tr></Table>

</body>
</html>Scripting Variables:
```

The scripting variables are the same as for the data tag *get.inquirasearch.answer* on page 156.

Body Tags:

none

get.localized.name

Description:

This tag returns the localized name of a specified object. Objects are specified by reference key and type.

Format:

```
<IM:get.localized.name
```

`id="string"` `(Non-scriptable string)`	a desired prefix to use for all scripting variables created by this tag. The default value is localizedname. NOTE: Ensure that all ids within nested channel-attribute tags are unique.
`referencekey="string"` `(Required)`	the reference key of the desired object.
`type="string"` `(Required)`	the type of the desired object. Valid options are: • CHANNEL • VIEW • ROLE • CATEGORY • EDITORGROUP • USERGROUP • DATALIST • NEWSLETTER • SHOPPINGCART • DATAFORM

```
></IM:get.localized.name >
```

Scripting Variables:

None

Required Parent Tags:

None

TagClass:

com.inquira.client.tags.LocalizedObjectNameTag

get.localized.name.value

Description:

Returns the localized name of a object passing in a reference key and a type as a scripting variable.

Format:

<IM:get.localized.name.value

`id="non-scriptable string"`	A desired prefix to use for all scripting variables created by this tag. The default value is 'localizedname'. If nesting channel-attribute tags, make sure all ids are unique
`referencekey*="string"`	The reference key of the desired object.
`type*="string"`	The type of the desired object. Valid options are: • CHANNEL • VIEW • ROLE • CATEGORY • EDITORGROUP • USERGROUP • DATALIST • NEWSLETTER • SHOPPINGCART • DATAFORM

></IM:get.localized.name.value >

get.localized.text

Description:

Displays localized string for the specified resource key from the resource text file. Used to localize static site text and images. The resource text file is located on the jsp server in the WEB-INF\classes directory. Create a file for each local supported by the site, appending the local code at the end of the filename.

Format:

```
<IM:get.localized.text
```

`key*="string"`	Localized resource key to get the string for as defined in LocaleStrings properties file under the resource directory

```
/>
```

ApplicationResources.properties	Default language resource file (if no local is specified by the browser).
ApplicationResources_en.properties	English file defined by "_en"
ApplicationResources_es.properties	Spanish file defined by "_es"

Example Usage:

```
<IM:get.localized.text key="welcome.text"/>
<img src="<IM:get.localized.text key="welcome.image"/>">
```

Scripting Variables:

none

Body Tags:

none

get.management.console.url

Description:

Prints the management console url stored in the configuration parameters for this repository.

```
<IM:get.management.console.url
></IM:get.management.console.url >
```

get.management.console.url.value

Description:

Gets the management console url stored in the configuration parameters for this repository.

```
<IM:get.management.console.url.value
id="string"
```

ID for this record.

```
></IM:get.management.console.url.value >
```

get.message.author

Description:

This tag gets message body text from a message board.

Format:

```
<IM:get.message.author
/>
```

Example Usage for Displaying All Messages:

```
<!-- get message data -->
<IM:get.message.data dataset="data" topic="hammers"/>
<!-- iterate through messages in discussion records -->
<IM:iterate.message.records dataset="data">
<!-- get handle to individual record -->
<IM:get.message.record>
<!-- space message to show hierarchy and display message title and text-->
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.title/> <br>
```

```
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.text/> <br>
<!-- display a reply link that passes the message record id to the reply page -->
<a href="index?page=messagereply&rec=<%=message.recordid%>">Reply</a> <br>
</IM:get.message.record>
</IM:iterate.message.records>
```

Example Usage for Replying to a Message:

```
<!-- declare page variable and load recordid parameter off url from previous page -->
<% String rec = request.getParameter("rec");%>
<!-- create HTML form for input fields -->
<IM:form.message parentrecordid="<%=rec%>" success="successpage" error="errorpage">
<!-- display input fields for title and text -->
Title<br>
<input.message.title><br>
Text<br>
<input.message.text><br>
<!-- display submit button -->
<input type="submit" value="Submit">
</IM:form.message>
```

Scripting Variables:

None

Body Tags:

None

get.message.data

Description:

Retrieve published messages for a specific topic and save the result set in a user defined variable. If no topic is defined all topics will be returned.

Format:

```
<IM:get.message.data
```

`dataset*="string"`	A unique identifier for the data set, this identifier is used by the channel-iterator and channel-format tags to access records that are retrieved.
`topic="string"`	Reference key name of the stored query defined in the Conviveon Server Administration tool

```
/>
```

Example Usage for Displaying All Messages:

```
<!-- get message data -->
<IM:get.message.data dataset="data" topic="hammers"/>
<!-- iterate through messages in discussion records -->
<IM:iterate.message.records dataset="data">
<!-- get handle to individual record -->
<IM:get.message.record>
<!-- space message to show hierarchy and display message title and text-->
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.title/> <br>
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.text/> <br>
<!-- display a reply link that passes the message record id to the reply page -->
<a href="index?page=messagereply&rec=<%=message.recordid%>">Reply</a> <br>
</IM:get.message.record>
</IM:iterate.message.records>
```

Example Usage for Replying to a Message:

```
<!-- declare page variable and load recordid parameter off url from previous page -->
<% String rec = request.getParameter("rec");%>
<!-- create HTML form for input fields -->
<IM:form.message parentrecordid="<%=rec%>" success="successpage" error="errorpage">
```

```
<!-- display input fields for title and text -->
Title<br>
<input.message.title><br>
Text<br>
<input.message.text><br>
<!-- display submit button -->
<input type="submit" value="Submit">
</IM:form.message>
```

Scripting Variables:

{id}.count	Total records retrieved

Body Tags:

iterate.message.records

get.metadata.record on page 183

get.message.title on page 180

get.message.text on page 179

form.message on page 87

input.message.title on page 228

input.message.text on page 226

get.message.record

Description:

This tag gets a handle to the current message record.

Format:

```
<IM:get.message.record
```

recordid=	the Content Text record id of record to display. Useful for a detailed display page where a user would select a link to a record and be directed to a page that displays the record.
id=	Prefix for all scripting variables within the scope of this tag.

```
></IM:get.message.record >
```

Example Usage for Displaying All Messages:

```
<!-- get message data -->
<IM:get.message.data dataset="data" topic="hammers"/>
<!-- iterate through messages in discussion records -->
<IM:iterate.message.records dataset="data">
<!-- get handle to individual record -->
<IM:get.message.record>
<!-- space message to show hierarchy and display message title and text-->
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.title/> <br>
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.text/> <br>
<!-- display a reply link that passes the message record id to the reply page -->
<a href="index?page=messagereply&rec=<%=message.recordid%>">Reply</a> <br>
</IM:get.message.record>
</IM:iterate.message.records>
```

Example Usage for Replying to a Message:

```
<!-- declare page variable and load recordid parameter off url from previous page -->
<% String rec = request.getParameter("rec");%>
<!-- create HTML form for input fields -->
<IM:form.message parentid="<%=rec%>" success="successpage" error="errorpage">
<!-- display input fields for title and text -->
Title<br>
<input.message.title><br>
Text<br>
<input.message.text><br>
<!-- display submit button -->
<input type="submit" value="Submit">
</IM:form.message>
```

Scripting Variables:

{id}.index	Current record index number
{id}.recordid	ID of current record
{id}.depth	Message hierarchy depth

Body Tags:

get.message.text on page 179

form.message on page 87

input.message.title on page 228

input.message.text on page 226

get.message.text

Description:

This tag returns message body text from a message board.

Format:

```
<IM:get.message.text
/>
```

Example Usage for Displaying All Messages:

```
<!-- get message data -->
<IM:get.message.data dataset="data" topic="hammers"/>
<!-- iterate through messages in discussion records -->
<IM:iterate.message.records dataset="data">
<!-- get handle to individual record -->
<IM:get.message.record>
<!-- space message to show hierarchy and display message title and text-->
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.title/> <br>
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.text/> <br>
<!-- display a reply link that passes the message record id to the reply page -->
<a href="index?page=messagereply&rec=<%=message.recordid%>">Reply</a> <br>
</IM:get.message.record>
</IM:iterate.message.records>
```

Example Usage for Replying to a Message:

```
<!-- declare page variable and load recordid parameter off url from previous page -->
<% String rec = request.getParameter("rec");%>
<!-- create HTML form for input fields -->
<IM:form.message parentrecordid="<%=rec%>" success="successpage" error="errorpage">
<!-- display input fields for title and text -->
Title<br>
<input.message.title><br>
Text<br>
<input.message.text><br>
<!-- display submit button -->
<input type="submit" value="Submit">
</IM:form.message>
```

Scripting Variables:

None

Body Tags:

None

get.message.title

Description:

This tag returns a message title from a record in a message board.

Format:

```
<IM:get.message.title
/>
```

Example Usage for Displaying All Messages:

```
<!-- get message data -->
<IM:get.message.data dataset="data" topic="hammers"/>
<!-- iterate through messages in discussion records -->
<IM:iterate.message.records dataset="data">
<!-- get handle to individual record -->
<IM:get.message.record>
<!-- space message to show hierarchy and display message title and text-->
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.title/> <br>
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.text/> <br>
<!-- display a reply link that passes the message record id to the reply page -->
<a href="index?page=messagereply&rec=<%=message.recordid%>">Reply</a> <br>
</IM:get.message.record>
</IM:iterate.message.records>
```

Example Usage for Replying to a Message:

```
<!-- declare page variable and load recordid parameter off url from previous page -->
<% String rec = request.getParameter("rec");%>
<!-- create HTML form for input fields -->
<IM:form.message parentrecordid="<%=rec%>" success="successpage" error="errorpage">
<!-- display input fields for title and text -->
Title<br>
<input.message.title><br>
Text<br>
<input.message.text><br>
<!-- display submit button -->
<input type="submit" value="Submit">
</IM:form.message>
```

Scripting Variables:

None

Body Tags:

None

get.metadata.attribute

Description:

This tag returns the specified attribute of a metadata object.

Format:

<IM:get.metadata.attribute

`attribute*="string"`	The attribute you want to return.

></IM:get.metadata.attribute >

get.metadata.attribute.value

Description:

This tag returns the specified attribute of a metadata object.

Format:

<IM:get.metadata.attribute.value

`attribute*="string"`	The attribute you want to return.
`id*="string"`	ID for this record.

></IM:get.metadata.attribute.value >

get.metadata.channels

Description:

Gets a list of channels for the specified Domain. If the registered value evaluates to "true" it returns the list of registered channels for the passed in view.

Format:

<IM:get.metadata.channels

`dataset*="string"`	A unique identifier for the data set, this identifier is used by the iterate.channel.records and get.channel.template tags to access records that are retrieved.
`registered*="true | false"`	Set to true if you would like to get registered channels.
`view="string"`	The view reference key for registered content channels.

></IM:get.metadata.channels >

get.metadata.dataform

Description:

Gets a list of data forms assigned to passed in view. If no view is passed in it will get the data forms assigned to the Domain.

Format:

<IM:get.metadata.dataform

`dataset*="string"`	A unique identifier for the data set, this identifier is used by the `iterate.channel.records` and `get.channel.template` tags to access records that are retrieved.
`view="string"`	The view reference key for registered content channels.
`parentviews="true" or "false"`	If set to true, it will fetch all Data Forms assigned to the passed in View and also all data forms assigned to all its parents in the tree.

></IM:get.metadata.dataform >

get.metadata.record

Description:

This tag returns the current object in the iteration.

Format:

<IM:get.metadata.record

`id*="string"`	ID for this record.

></IM:get.metadata.record >

get.meta.resourcepath.content

Description:

This tag prints the meta resource path for the current content record.

Format:

<IM:get.meta.resourcepath.content
></IM:get.meta.resourcepath.content >

get.privileges

Description:

Gets the privilege for the current content record.

Format:

<IM:get.privileges

`negate="non-scriptable string"`	Reverses the conditional of this tag. If it evaluates to "true" it will show the contents of this tag. The default behavior of this attribute is to be false.

></IM:get.privileges >

get.privileges.add

Description:

Gets the ADD content privilege for the logged in user and for the given channel and view. If no View is specified, it will assume the Domain level .

Format:

<IM:get.privileges.add

`negate="non-scriptable string"`	Reverses the conditional of this tag. If it evaluates to "true", if the user have privileges, it will show the contents of this tag. The default behavior of this attribute is to be false.
`channel*="string"`	The channel reference key.
`view="string"`	The view reference key.
`editorgroups="string"`	List of editorgroups delimited by a + sign that the user needs to have in order to add a record. This tag will allow a user to add a record if he/she has at least one of the passed in editor groups. In order to make this feature active, the passed in channel needs to have Editor Groups management turned on.

></IM:get.privileges.add >

get.privileges.value

Description:

Gets the user privilege for the current record. If the user has privilege displays its content, otherwise it doesn't display the contents.

Format:

<IM:get.privileges.value

`negate="non-scriptable string"`	Reverses the conditional of this tag. If it evaluates to "true", if the user have privileges, it will show the contents of this tag. The default behavior of this attribute is to be false.
`privilege*="string"`	ADD - EDIT - DELETE. The desired privilege to check on..

></IM:get.privileges.value >

get.recommendation.attribute

Description:

Tag that displays the selected attribute for a content recommendation. This tag must be included inside a get.recommendation.record tag.

Format:

<IM:get.recommendation.attribute

attribute*="string"	Name of attribute that should be displayed.	
	Allowed Attribute values:	
	recordid	recordid of the recommendation
	casenumber	case number
	completedby	completed by user
	requestedby	requested user
	title	recommendation title
	text	recommendation text
	comments	comments if recommendation has been processed
	contentid	content id if recommendation has been processed and a content record has been created
	docid	document id if recommendation has been processed and a content record has been created
	channel	content channel assigned to recommendation
	status	current status of this content recommendation
	priority	current priority of this content recommendation
	dateadded	creation date timestamp (java.util.Date)
	datemodified	last modified timestamp (java.util.Date)
mask="string"	Mask to use for date fields.	

></IM:get.recommendation.attribute >

get.recommendation.data

Description:

Retrieves a list of content recommendations based on passed in parameters

Format:

<IM:get.recommendation.data

`dataset*="string"`	A unique identifier for the data set. This identifier is used by the iterate.records tag to access records that are retrieved.
`channels="string"`	List of channels separated b y a + sign that the content recommendation query will search on.
`completedby="string"`	If present, this tag will return only recommendations that where completed by this user. A valid value must be the user's login name
`requestedby="string"`	If present, this tag will return only recommendations that where requested by this user. A valid value must be the user's login name.
`status="string"`	List of statuses separated b y a + sign. valid values are: new, duplicate, not_enough_info, unsuitable, other, created
`priority="string"`	List of priorities separated b y a + sign. valid values are: low, medium, and high
`sortby="string"`	List of sort parameters separated b y a + sign. Valid values are: dateadded, title, casenumber, priority, and status
`direction="string"`	The direction of the sort. Ascending or descending. The default if no value provided will be ascending.
`casenumber="string"`	returns content recommendations that have this casenumber

/>

get.recommendation.record

Description:

Tag that gets a recommendation record either from an iteration or by a recommendation id.

```
<IM:get.recommendation.record
```

recordid="string"	ID of record to retrieve.
id="non-scriptable string"	A desired prefix to use for all scripting variables created by this tag. There is no default id. Make sure all ids are unique

```
></IM:get.recommendation.record >
```

get.record.masteridentifier

Description:

This tag returns the values of the master identifier fields for a specified content record, based on a guid or documentid passed as a parameter.

Format:

```
<IM:get.record.masteridentifier
```

`id="string"` `(Non-scriptable string)`	specifies a desired prefix to use for all scripting variables created by this tag. The default value is `masteridentifier`. NOTE: Ensure that all nested tags have unique identifiers.
`recordid="string"` `(Required)`	specifies the content id for the text to be returned. NOTE: To retrieve records from staging environments, prefix the record ID with the characters `S:`.
`documentid="string"` `(Required)`	specifies the document ID of the content record.
`localecode="string"`	specifies an optional locale code for the text associated with the requested fields.

```
></IM:get.channel.attribute.exists >
```

Scripting Variables:

None

Reqired Parent Tags:

None

get.repository.attribute

Description:

Get a system or custom domain attribute. Only one of the attribute parameters can be specified at one time.

Format:

<IM:get.repository.attribute

`id="string"`	ID for this record.	
`attribute="string"`	xpath of extended site attribute to return. Note that extended site properties are defined via schema in the admin tool.	
`systemattribute="string"`	Basic site property value to return. Valid values are:	
	domain	domain name of site
	refkey	locale-neutral reference key for site
`view="string"`	Reference key for the repository view to return property for	
`processbody="true" or "false"`	set to 'true' if you create a body for this tag	

></IM:get.repository.attribute >

Example Usage:

```
<html>
<head>
<meta content="<IM:get.repository.attribute attribute="demo/keywords" />"
name="KEYWORDS">
<title><IM:get.repository.attributes systemattribute="domain"/></title>
</head>
<body>
...
</body>
</html>
```

Scripting Variables:

None

Required Parent Tags:

None

get.repository.attribute.value

Description:

This tag returns a system or custom domain attribute. You can specify only one of the attribute parameters in a single statement.

Format:

<IM:get.repository.attribute.value

`id="string"`	ID for this record.	
`attribute="string"`	xpath of extended site attribute to return. Note that extended site properties are defined via schema in the admin tool.	
`systemattribute="string"`	Basic site property value to return. Valid values are:	
	domain	domain name of site
	refkey	locale-neutral reference key for site
`view="string"`	Reference key for the repository view to return property for	

></IM:get.repository.attribute.value >

Example Usage:

```
<html>
<head>
<meta content="<IM:get.repository.attribute attribute="demo/keywords" />"
name="KEYWORDS">
<title><IM:get.repository.attributes systemattribute="domain"/></title>
</head>
<body>
...
</body>
</html>
```

Scripting Variables:

None

Body Tags:

None

get.resourcepath.content

Description:

This tag prints the resource path for the current content record.

Format:

```
<IM:get.resourcepath.content
></IM:get.resourcepath.content >
```

get.resourcepath.user

Description:

This tag prints the resource path for the logged in user.

Format:

```
<IM:get.resourcepath.user
></IM:get.resourcepath.user >
```

get.resourcepath.view

Description:

This tag prints the resource path for the specified view.

Format:

```
<IM:get.resourcepath.view
```

`view*="string"` The reference key of the desired view.

```
></IM:get.resourcepath.view >
```

get.role.attribute

Description:

Tag that displays the selected attribute for a role. This tag must be included inside a get.role.record tag.

Format:

<IM:get.role.attribute

`attribute*="string"`	Name of attribute that should be displayed. Allowed Attribute values: name — returns the name of the role referencekey — returns the reference key of the role recordid — returns the recordid(guid) of the role

></IM:get.role.attribute >

get.role.data

Description:

Retrieves a list of roles. If webroles is set to true it returns a list of all webroles for this repository, other wise set a user id or login to return a list of roles assigned to the desired user. If webroles is set to false and no user attribute is specified, this tag will attempt to get the security roles for the logged in user.

Format:

<IM:get.role.data

`dataset*="string"`	A unique identifier for the data set, this identifier is used by the iterate.data set tag to access records that are retrieved.
`webroles="true" or "false"`	Set to true to returns all the web roles for this repository. The default behavior is set to false.
`userid="string"`	Record id of the user.
`login="string"`	Login of the user.

/>

get.role.record

Description:

Tag that gets a role record from an iteration or from a passed in role identifier.

Format:

<IM:get.role.record

`recordid="string"`	ID of record to retrieve.
`role="string"`	Reference Key of Role to retrieve.
`id="non-scriptable string"`	A desired prefix to use for all scripting variables created by this tag. There is no default id. If nesting role-record tags, make sure all ids are unique

></IM:get.role.record >

get.schema.data

Description:

Retrieves a Schema Record with all its schema attributes. If a parent node is passed in it return the children of that node. If no node is passed in it assumes the root schema attribute.

<IM:get.schema.data

`dataset*="string"`	A unique identifier for the data set, this identifier is used by the iterator tag to access records that are retrieved.
`type*="string"`	This is the schema type. Allowed values are: • CHANNEL (Regular channel schema) • META (channel meta schema) • USER • REPOSITORY
`node="string"`	The parent node xpath. If not present this tag will assume that the parent node is the root.
`referencekey="string"`	This value is necessary only if the schema is of type CHANNEL or META and it should be the reference key of the channel owner of the schema.

></IM:get.schema.data >

get.schema.item

Description:

Tag that retrieves the current repetition item of type schema attribute

<IM:get.schema.item

`id="non-scriptable string"`	A desired prefix to use for all scripting variables created by this tag. The default value is 'attributerecord'. If nesting tags, make sure all ids are unique.

></IM:get.schema.item >

get.schema.item.attribute

Description:

Returns the schema item attribute value for the passed in key

<IM:get.schema.item.attribute

`id="string"`	ID for this record.
`attribute="string"`	Attribute that you are interested in. Allowed values are: XPATH, NAME, REFERENCEKEY, TYPE, TEXTHEIGHT, TEXTWIDTH
`processbody="true" or "false"`	set to 'true' if you create a body for this tag

></IM:get.schema.item.attribute >

get.schema.item.attribute.value

Description:

Returns the schema item attribute value for the passed in key

<IM:get.schema.item.attribute.value

`id="string"`	ID for this record.
`attribute="string"`	Attribute that you are interested in. Allowed values are: XPATH, NAME, REFERENCEKEY, TYPE, TEXTHEIGHT, TEXTWIDTH

></IM:get.schema.item.attribute.value >

get.search.attribute

Description:

Displays properties inside the returned IQXML. Either property or propertypath can be used, but not both. Note that if no user is logged in, this tag will come back with no data.

Format

<IM:get.search.attribute

`attribute="string"`	Case sensitive xpath of iqxml to return
`id="string"`	A desired prefix to use for all scripting variables created by this tag. The default value is 'searchattribute'. If nesting channel-attribute tags, make sure all IDs are unique
`processbody="true | false"`	Set to 'true' if you create a body for this tag

/>

Example Usage:

```
<html>
<head>

</head>
<body>
... <IM:get.search.attribute id='att' attribute='/message/params/param[@name="charset"]'>
</body>
</html>
```

Scripting Variables:

none

Body Tags:

none

get.search.data

Description:

Tag that searches the repository for the passed in attribute. This tag performs a either a full text search or an attribute level search.

Format:

<IM:get.search.data

channels="string"	If used as a full text search this attribute is a delimited (+) list of Content Channels (using reference key) to include in the full text search (i.e., NEWS+EVENTS). If used as an attribute level search this attribute should only contain one channel reference key (i.e., NEWS).
categories="string"	delimited (+) list of Content Categories (using reference key) to include in the full text search (i.e., CAT1+CAT2).
matchallcategories="true" or "false"	Used in conjunction with 'categories' attribute. Set to true to 'and' the categories together in the search, and false to 'or' the categories.
matchallterms="true" or "false"	FULL TEXT SEARCH: If this attribute is set to true, the search will AND all the words searched. If it is set to false, this tag will OR the words searched. The default value is true. ATTRIBUTE LEVEL SEARCH: If this attribute is set to true, it will AND all the search attributes. If it is set to false, this tag will OR the search attributes. The default value is true.
isattributelevel="true" or "false"	Set to true if you would like to perform an attribute level search. Set to false to perform a full text search. The default value is false
views="string"	delimited (+) list of Views (using reference key) to include in the full text search (i.e., VIEW1+VIEW2).
maxrecords="integer"	Max amount of records returned by this search. Default is set to 200 records. A value of 0 will return all possible records

></IM:get.search.data >

get.securefilecookie

The set.securefilecookie is called after all the tokens for a page have been generated and will take the list of encrypted values and store them in at least 2 cookies on the client's browser.

An example of using the new tags inside of iterate.channel.records and get.channel.record is as follows

```
<cas:iterate.channel.records dataset="records" id="test">
   <cas:get.channel.record id="rec">
<cas:get.channel.attribute attribute="BRANDNEW/SECFILE" processbody="true" id="file">

        <%if(file.exists){%>
          <a href="<%=rec.secureresourcepath%><%=file.value%><cas:get.securefiletoken/
>"><%=file.value%></a>
</cas:get.channel.attribute>
   </cas:get.channel.record>
</cas:iterate.channel.records>
```

A secure file will need "`<cas:get.securefiletoken/>`" appended to the URL. The Servlet filter will need this token to match against the cookies for its authentication. It will generate the string, "`?token=xxxxxxx`" where `xxxxxxx` is the encryption.

get.securefiletoken

Description:

Tag that generates a necessary token value to append to a secure file link. Works with the IMServletFilter feature.

The get.securefiletoken will generate a unique, encrypted token value and store the encrypted values in a request attribute for cookie writing.

Format:

```
<IM:get.securefiletoken
></IM:get.securefiletoken >
```

get.session.locale.attribute

Description:

Tag that displays the specified Locale attribute. If no attribute is specified it displays the Locale Code attribute(i.e., en_us)

Format:

<IM:get.session.locale.attribute

`attribute="string"`	Locale attribute key. Allowed values are: code value description dateformat dateformatdisplay timeformat timeformatdisplay encoding

></IM:get.session.locale.attribute >

Attribute Result Examples:

Code = en_us
Value = 1033
description = English
dateformat = %m/%d/%Y
dateformatdisplay = mm/dd/yyyy
timeformat = %I:%M %p
timeformatdisplay = hh:mm
encoding = UTF-8

get.session.locale.attribute.value

Description:

Tag that creates scripting variables for the active locale. These are the scripting variables created: code, value, description, dateformat, dateformatdisplay, timeformat, timeformatdisplay, and encoding.

Format:

```
<IM:get.session.locale.attribute.value
id="non-scriptable string"
></IM:get.session.locale.attribute.value >
```

A desired prefix to use for all scripting variables created by this tag. Make sure all ids are unique

Variables Examples:

```
Code = en_us
Value = 1033
description = English
dateformat = %m/%d/%Y
dateformatdisplay = mm/dd/yyyy
timeformat = %I:%M %p
timeformatdisplay = hh:mm
encoding = UTF-8
```

get.static.resourcepath

Description:

Prints to the specified static resource path for the active domain.

Format:

```
<IM:get.static.resourcepath
></IM:get.static.resourcepath >
```

get.static.resourcepath.value

Description:

Gets the specified static resource path for the active domain.

Format:

<IM:get.static.resourcepath.value

`id="string"` ID for this record.

></IM:get.static.resourcepath.value >

get.subscription.name

Description:

Tag that displays the name of the current subscription. This tag is intended to be a child of get.subscription.record tag.

Format:

<IM:get.subscription.name
></IM:get.subscription.name >

get.subscription.record

Description:

Tag that gets the current subscription in the iteration.

Format:

<IM:get.subscription.record
></IM:get.subscription.record >

get.user.attribute

Description:

Provides access to both system user attributes and custom defined user attributes. To retrieve system attributes use the “systemattribute” parameter and pass in the key for the system attribute to display. To retrieve custom attributes, use the “attribute” parameter and pass in the xpath to the attribute to retrieve. The format for the xpath is REPOSITRY_REF_KEY/ATTRIBUTE_REF_KEY.

Format:

<IM:get.user.attribute

`attribute="string"`	Case sensitive xpath of extended user property to return. Note that extended user properties are defined via schema in the admin tool.
`systemattribute="string"`	Basic user property value to return. Valid values case insensitive and are:

firstname	users first name
lastname	users last name
email	users email address
login	users login ID
localecode	users preferred locale code (i.e. 1033)
localedesc	users preferred locale description (i.e. en_US)
defaultview	users default view

></IM:get.user.attribute >

Example Usage:

```
FIRST Name: <IM:get.user.attribute attribute="FIRSTNAME"/> <br>
LAST Name: <IM:get.user.attribute systemattribute="LASTNAME"/><br>
EMAIL: <IM:get.user.attribute attribute="EMAIL"/><br>
GUID: <IM:get.user.attribute attribute="GUID"/><br>
Prefered Locale: <IM:get.user.attribute attribute="LOCALE/CODE"/><br>
EDITOR GROUP NAME: <IM:get.user.attribute attribute="EDITORGROUP/NAME"/><br>
EDITOR GROUP REFKEY: <IM:get.user.attribute attribute="EDITORGROUP/
REFERENCEKEY"/><br>
DEFAULT VIEW REFKEY: <IM:get.user.attribute attribute="DEFAULTVIEW/
REFERENCEKEY"/><br>
PHONE: <IM:get.user.attribute attribute="REPOSITORY_REF_KEY/phone"/><br>
```

get.user.attribute.value

Description:

Provides the same level of functionality as the *get.user.attribute* tag. However it enables the value to be passed back through a scripting variable. In some cases this is needed to use the value programmatically.

Format:

<IM:get.user.attribute.value

`attribute="string"`	Case sensitive xpath of extended user property to return. Note that extended user properties are defined via schema in the admin tool.
`systemattribute="string"`	Basic user property value to return. Valid values case insensitive and are: firstname — users first name lastname — users last name email — users email address login — users login ID localecode — users preferred locale code (i.e. 1033) localedesc — users preferred locale description (i.e. en_US)
`id="non-scriptable string"`	A desired prefix to use for all scripting variables created by this tag. The default value is 'attribute'. If nesting channel-attribute tags, make sure all ids are unique

></IM:get.user.attribute.value >

Example Usage:

```
<IM:get.user.attribute.value attribute="REPOSITORY_REF_KEY/phone" id="userPhone">
Phone = <%=userPhone.value%>
</IM:get.user.attribute.value>
```

get.user.data

Description:

Retrieves a list of users using the DefaultViews and EditorGroups as query parameters.

Format:

<IM:get.user.data

dataset*="string"	A unique identifier for the data set, this identifier is used by the iterate.user.records tag to access records that are retrieved.
defaultviews="string"	List of views delimited by a + sign that the ad-hoc query will search on. This means that it will return user records that have assigned as their default views the passed in views.
where="string"	Sql type where clause for fetching users. Allowed values are firstName, lastName, login, and email. These are case sensitive key names and the fetch is also case sensitive. EXAMPLES: <IM:get.user.data dataset="userdataset" sortby="firstname" direction="ascending" where="login like 'g*' and lastName like 'R*'" /> <IM:get.user.data dataset="userdataset" sortby="firstname" direction="ascending" where="firstName = 'John' and lastName like 'R*'" />
editorgroups="string"	List of editorgroups delimited by a + sign that the query will search on.
sortby="string"	This parameter determines which user record parameter will be used for sorting. Allowed values are: firstname, lastname, email, and login. The default behavior is lastname
direction="string"	The direction of the sort. Ascending or descending. The default if no value provided will be ascending.
roles="string"	List of roles delimited by a + sign that the query will search on.

`categories="string"`	List of categories delimited by a + sign that the query will search on.
`localecode="string"`	Locale code for the current request. Locale code is of format en_US where en=language, US=location. If not specified, it defaults to default locale of repository.

/>

Example Response:

```
<USER>
   <LOGIN><![CDATA[eclapton]]></LOGIN>
   <FIRSTNAME><![CDATA[Eric]]></FIRSTNAME>
   <LASTNAME><![CDATA[Clapton]]></LASTNAME>
   <EMAIL><![CDATA[gil@conviveon.com]]></EMAIL>
   <ACTIVE><![CDATA[Y]]></ACTIVE>
   <DEFAULTVIEW>
      <REFERENCEKEY><![CDATA[PROMOTERS]]></REFERENCEKEY>
      <NAME><![CDATA[Promoters]]></NAME>
   </DEFAULTVIEW>
   <LOCALE>
      <CODE><![CDATA[en_US]]></CODE>
      <DESCRIPTION><![CDATA[English]]></DESCRIPTION>
   </LOCALE>
   <GUID><![CDATA[00281939151101094a846fad007db3]]></GUID>
   <USERGROUP>
      <NAME><![CDATA[All]]></NAME>
      <REFERENCEKEY><![CDATA[ALL]]></REFERENCEKEY>
   </USERGROUP>
   <USERGROUP>
      <NAME><![CDATA[MANAGEMENT]]></NAME>
      <REFERENCEKEY><![CDATA[MANAGEMENT]]></REFERENCEKEY>
   </USERGROUP>
   <CATEGORY>
      <NAME><![CDATA[My Web Service]]></NAME>
      <REFERENCEKEY><![CDATA[WEBSERVICE]]></REFERENCEKEY>
   </CATEGORY>
   <CATEGORY>
      <NAME><![CDATA[989898]]></NAME>
      <REFERENCEKEY><![CDATA[989898]]></REFERENCEKEY>
   </CATEGORY>
```

```
 <CATEGORY>
       <NAME><![CDATA[GOLF]]></NAME>
      <REFERENCEKEY><![CDATA[GOLF]]></REFERENCEKEY>
   </CATEGORY>
   <CATEGORY>
      <NAME><![CDATA[Instruments]]></NAME>
      <REFERENCEKEY><![CDATA[INSTRUMENTS]]></REFERENCEKEY>
   </CATEGORY>
   <ROLE>
      <NAME><![CDATA[WEBROLE1]]></NAME>
      <REFERENCEKEY><![CDATA[WEBROLE1]]></REFERENCEKEY>
   </ROLE>
   <ROLE>
      <NAME><![CDATA[View Work for Promoters]]></NAME>
      <REFERENCEKEY><![CDATA[VIEW_WORK_FOR_PROMOTERS]]></
REFERENCEKEY>
   </ROLE>
   <ROLE>
      <NAME><![CDATA[View Work For Artist]]></NAME>
      <REFERENCEKEY><![CDATA[VIEW_WORK_FOR_ARTIST]]></REFERENCEKEY>
   </ROLE>
   <ROLE>
      <NAME><![CDATA[WEBROLE2]]></NAME>
      <REFERENCEKEY><![CDATA[WEBROLE2]]></REFERENCEKEY>
   </ROLE>
   <ROLE>
      <NAME><![CDATA[TESTWEB]]></NAME>
      <REFERENCEKEY><![CDATA[TESTWEB]]></REFERENCEKEY>
   </ROLE>
   <METRICS>
      <DEFAULT>
         <USERLEVEL>
            <TITLE>Expert</TITLE>
            <NUMBER>5</NUMBER>
         </USERLEVEL>
         <DOCUMENTACCESSCOUNT>0</DOCUMENTACCESSCOUNT>
         <DOCUMENTSAUTHORED>3</DOCUMENTSAUTHORED>
         <DOCUMENTSOWNED>0</DOCUMENTSOWNED>
         <DOCUMENTREUSECOUNT>0</DOCUMENTREUSECOUNT>
         <DOCUMENTVALUE>0</DOCUMENTVALUE>
      </DEFAULT>
   </METRICS>
   <NOTIFYASSIGNED>Y</NOTIFYASSIGNED>
   <NOTIFYPERFORM>Y</NOTIFYPERFORM>
<EXTENDED>
<GILTEST>
<ADDRESS><![CDATA[XXX SAVING FROM CLIENT XXX]]></ADDRESS>
<ZIP/>
<PHONE/>
<CITY/>
<STATE/>
```

```
<HTMLDOC/>
<PICTURE/>
        <INSTRUMENTS>
          <NAME/>
        </INSTRUMENTS>
        <CURRENT/>
        <UID><![CDATA[UID115]]></UID>
      </GILTEST>

   </EXTENDED>
</USER>
```

get.user.has.role

Description:

Checks if the specified user has the specified role. You can specify the user by userid, login. If you don't specify a user attribute, and this tag is inside a get.user.record, it will be applied to the user object of the get.user.record. If it is not inside a get.user.record, it will be applied to the logged in user. You can specify the role by roleid, role reference key, or if this tag is inside a get.role.record, it will be applied to the role object of the get.role.record.

Format:

```
<IM:get.user.has.role
```

`roleid="non-scriptable string"`	The role id of the security role to be checked.
`role="non-scriptable string"`	The role referencekey of the security role to be checked.
`userid="non-scriptable string"`	The user id of the user to be checked.
`login="non-scriptable string"`	The login of the user to be checked.
`negate="non-scriptable string"`	Reverses the conditional of this tag. If it evaluates to "true" it will show the contents of this tag if the value is not present. The default behavior of this attribute is to be false.

```
></IM:get.user.has.role >
```

get.user.record

Description:

Tag that gets a user record either from iterate.user.records iteration, from a passed record ID, or a passed in login name.

Format:

<IM:get.user.record

`recordid="string"`	ID of record to retrieve.
`login="string"`	Login name of user record to retrieve.
`id="non-scriptable string"`	A desired prefix to use for all scripting variables created by this tag. There is no default id. If nesting user-record tags, make sure all ids are unique

></IM:get.user.record >

get.views

Description:

Get a list of all compartments for the current domain.

Format:

<IM:get.views

`dataset*="string"`	Name of the data set that contains the department list information
`startview="string"`	Reference key of the site to start on

/>

Example Usage:

<IM:get.departments dataset="departments"/>

Scripting Variables:

none

Body Tags:

none

get.view.template

Description:

Returns the template for the passed in view, if no view is passed in it returns the template for the domain.

Format:

<IM:get.view.template

`view*="string"`	The reference key of the desired view.

></IM:get.view.template >

get.view.template.value

Description:

Returns the template for the passed in view, if no view is passed in it returns the template for the domain.

Format:

<IM:get.view.template.value

`view*="string"`	The reference key of the desired view.
`id*="string"`	ID for this record.

></IM:get.view.template.value >

get.xpath.attribute

Description:

Tag that returns the value stored in a XML Document for the passed in xpath. The xml document is retrieved from the parent tag if and only if the parent tag supports this functionality. Scripting Variables: value = the actual value found; exists = boolean to determine if the attribute exists or not.

Format:

<IM:get.xpath.attribute

`attribute*="string"`	Case sensitive xpath to attribute in parent tag to return.
`id="non-scriptable string"`	A desired prefix to use for all scripting variables created by this tag. The default value is 'attribute'. If nesting channel-attribute tags, make sure all ids are unique
`processbody="true" or "false"`	set to 'true' if you create a body for this tag

></IM:get.xpath.attribute >

get.xpath.attribute.value

Description:

Tag that returns the value stored in a XML Document for the passed in xpath. The xml document is retrieved from the parent tag if and only if the parent tag supports this functionality. Scripting Variables: value = the actual value found; exists = boolean to determine if the attribute exists or not.

Format:

<IM:get.xpath.attribute.value

`attribute*="string"`	Case sensitive xpath to attribute in parent tag to return.
`id="non-scriptable string"`	A desired prefix to use for all scripting variables created by this tag. The default value is 'attribute'. If nesting channel-attribute tags, make sure all IDs are unique

></IM:get.xpath.attribute.value >

get.wizardfield.record

Description:

Tag that receives the current WizardFieldObject object from inside the iterate.wizardform.fields tag.

Format:

```
<IM:get.wizardfield.record
id="string"
(Required)
/>
```

The unique id representing the WizardFieldObject

Example Usage:

```
<html>
<head>

</head>
<body>
... <IM:iterate.wizardform.fields>
   <br>
        <IM:get.wizardfield.record id="wizf">
   <% if (wizf.type.equals("select")) { %>
        <IM:input.wizardfield.record css="dropdown"/>
   <% } else if (wizf.type.equals("checkbox")) { %>
        <IM:input.wizardfield.record css="chekbx"/>
   <% } else if (wizf.type.equals("text")) { %>
        <IM:input.wizardfield.record css="letext"/>
   <% } else if (wizf.type.equals("radio")) { %>
        <IM:input.wizardfield.record css="amfm"/>
   <% } else { %>
             <br><b><%=wizf.text%></b> <br>
   <% } %>
        </IM:get.wizardfield.record>

   </IM:iterate.wizardform.fields>

</body>
</html>
```

Scripting Variables:

WizardFieldObject:

```
public String type = null;
public boolean selected = false;
public String groupid = null;
public String itemid = null;
public String value = null;
public String text = null;
public String groupdesc = null;
public String itemdesc = null;
public int order = 0;
public boolean last = false;
```

Body Tags:

none

get.wizard.previous.response

Description:

Tag that receives the current PreviousResponse object from inside the iterate.wizard.previous.responses tag.

Format:

```
<IM:get.wizard.previous.response
```

`id="string"` (Required)	The unique id representing the PreviousResponse object

```
/>
```

Example Usage:

```
<html>
<head>

</head>
<body>
... <IM:iterate.wizard.previous.responses>
<IM:get.wizard.previous.response id="wpr">

<font color=blue size="3"><%= wpr.question %></font><br>
<font color=blue size="2"><%= wpr.answer %></font> <br><br>
</IM:get.wizard.previous.response>
</IM:iterate.wizard.previous.responses>

</body>
</html>
```

Scripting Variables:

question
answer
stepid

Body Tags:

none

get.wizardfield.record

Description:

Tag that receives the current WizardFieldObject object from inside the iterate.wizardform.fields tag.

Format:

<IM:get.wizardfield.record

`id="string"` (Required)	The unique id representing the WizardFieldObject

/>

Example Usage:

```
<html>
<head>

</head>
<body>
... <IM:iterate.wizardform.fields>
   <br>
        <IM:get.wizardfield.record id="wizf">
   <% if (wizf.type.equals("select")) { %>
        <IM:input.wizardfield.record css="dropdown"/>
   <% } else if (wizf.type.equals("checkbox")) { %>
        <IM:input.wizardfield.record css="chekbx"/>
   <% } else if (wizf.type.equals("text")) { %>
        <IM:input.wizardfield.record css="letext"/>
   <% } else if (wizf.type.equals("radio")) { %>
        <IM:input.wizardfield.record css="amfm"/>
   <% } else { %>
             <br><b><%=wizf.text%></b> <br>
   <% } %>
        </IM:get.wizardfield.record>

   </IM:iterate.wizardform.fields>

</body>
</html>
```

Scripting Variables:

WizardFieldObject:

```
public String type = null;
public boolean selected = false;
public String groupid = null;
public String itemid = null;
public String value = null;
public String text = null;
public String groupdesc = null;
public String itemdesc = null;
public int order = 0;
public boolean last = false;
```

Body Tags:

none

index.next

Description:

Create HTML hyperlink to the next batch of records from an iterate.channel.records tag.

Format:

```
<IM:index.next
ifnull="string"

></IM:index.next >
```

Example Usage:

```
<table width="100%" cellpadding="0" cellspacing="0">
<IM:iterate.channel.records dataset="news" maxpageitems="10" maxindexpages="20">
<IM:get.channel.record id="newsrec">
<tr>
<td align="left">
<a href="index?page=detail&rec=<%=newsrec.recordid%>"><IM:get.channel.attribute
attribute="news/title"/></a>
</td>
</tr>
</IM:get.channel.record>
<IM:iterate.index>
<tr>
<td>
<br>
Result Pages:
<IM:index.prev>
 <a href="<%= pageUrl %>">[<< Prev]</a>
</IM:index.prev>
<IM:index.pages>
<% if (pageNumber.intValue() < 10) { %> <% } %>
<% if (pageNumber == iteratorpagenumber) { %>
<b><%= pageNumber %></b>
```

```
<% } else { %>
<a href="<%= pageUrl %>"><%= pageNumber %></a>
<%}%>
</IM:index.pages>
<IM:index.next>
 <a href="<%= pageUrl %>">[Next >>]</a>
</IM:index.next>
<br>
</td>
</tr>
</IM:iterate.index>
</IM:iterate.channel.records>
</table>
```

Scripting Variables:

none

Body Tags:

none

index.pages

Description:

Create HTML hyperlink to each batch page of records from an `iterate.channel.records` tag.

Format:

```
<IM:index.pages
></IM:index.pages >
```

Example Usage:

```
<table width="100%" cellpadding="0" cellspacing="0">
<IM:iterate.channel.records dataset="news" maxpageitems="10" maxindexpages="20">
<IM:get.channel.record id="newsrec">
<tr>
<td align="left">
<a href="index?page=detail&rec=<%=newsrec.recordid%>"><IM:get.channel.attribute
attribute="news/title"/></a>
</td>
</tr>
</IM:get.channel.record>
<IM:iterate.index>
<tr>
<td>
<br>
Result Pages:
<IM:index.prev>
 <a href="<%= pageUrl %>">[<< Prev]</a>
</IM:index.prev>
<IM:index.pages>
<% if (pageNumber.intValue() < 10) { %> <% } %>
<% if (pageNumber == iteratorpagenumber) { %>
<b><%= pageNumber %></b>
<% } else { %>
<a href="<%= pageUrl %>"><%= pageNumber %></a>
<%}%>
</IM:index.pages>
<IM:index.next>
 <a href="<%= pageUrl %>">[Next >>]</a>
</IM:index.next>
<br>
</td>
</tr>
</IM:iterate.index>
</IM:iterate.channel.records>
</table>
```

Scripting Variables:

none

Body Tags:

none

index.prev

Description:

Create HTML hyperlink to page with previous batch of records from an iterate.channel.records tag.

Format:

```
<IM:index.prev
ifnull="string"

></IM:index.prev >
```

Example Usage:

```
<table width="100%" cellpadding="0" cellspacing="0">
<IM:iterate.channel.records dataset="news" maxpageitems="10" maxindexpages="20">
<IM:get.channel.record id="newsrec">
<tr>
<td align="left">
<a href="index?page=detail&rec=<%=newsrec.recordid%>"><IM:get.channel.attribute
attribute="news/title"/></a>
</td>
</tr>
</IM:get.channel.record>
<IM:iterate.index>
<tr>
<td>
<br>
```

```
Result Pages:
<IM:index.prev>
 <a href="<%= pageUrl %>">[<< Prev]</a>
</IM:index.prev>
<IM:index.pages>
<% if (pageNumber.intValue() < 10) { %> <% } %>
<% if (pageNumber == iteratorpagenumber) { %>
<b><%= pageNumber %></b>
<% } else { %>
<a href="<%= pageUrl %>"><%= pageNumber %></a>
<%}%>
</IM:index.pages>

<IM:index.next>
 <a href="<%= pageUrl %>">[Next >>]</a>
</IM:index.next>
<br>
</td>
</tr>
</IM:iterate.index>
</IM:iterate.channel.records>
</table>
```

Scripting Variables:

none

Body Tags:

none

input.channel.contribution

Description:

Creates an HTML input element for an attribute of an existing content channel.

Format:

<IM:input.channel.contribution

attribute*="string"	Path to custom user attribute. To contribute display start and end dates, or event start and end dates, you must be specified one of the following keywords and a date mask DISPLAYSTARTDATE, DISPLAYENDATE, EVENTSTARTDATE, EVENTENDDATE.
mask="string"	Date mask for Content Dates.
size="string"	Size of the input field.
maxlength="string"	Maximum length of the input field.
value="string"	Initial value of the input field. if nothing is entered for the value, the value will default to the information in the database.
css="string"	CSS Class to be used for the input field
cols="string"	If this input is a text area, this attribute defines how many columns it will have
rows="string"	If this input is a text area, this attribute defines how many rows it will have

/>

Example Usage:

```
<IM:form.channel.contribution channel="news" departments="department1+department2"
success="thankyou" error="errorpage">
Title:
<IM:input.channel.contribution attribute="/news/title"/>
Body
<IM:input.channel.contribution attribute="/news/body"/>
</IM:channel.contribute.form>
```

Scripting Variables:

None

Body Tags:

input.channel.contribution on page 223

input.dataform.answer

Description:

This tag creates an HTML input element that matches an answer type with pre-defined answers.

Format:

```
<IM:input.dataform.answer
```

`css="string"`	CSS Class to be used for the input field

```
></IM:input.dataform.answer >
```

Example Usage for Displaying the Form

```
<IM:form.dataform dataform="INQUIRIES" success="requestthank" error="requestthank">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:input.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question> <br>
<input type="submit" value="Submit">
</IM:form.dataform>
```

Example Usage for Displaying Results

```
<IM:get.dataform.results name="nameofform" individual="false" aggregate="false">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:get.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question>
```

```
</IM:get.dataform.results>
```

Scripting Variables:

None

TagClass:

com.inquira.client.tags.SurveyAnswerInputTag

input.message.author

Description:

Creates an HTML INPUT element for a message author.

Format:

```
<IM:input.message.author
```

`size="string"`	Size of the input field.
`maxlength="string"`	Maximum length of the input field.
`value*="string"`	Initial value of the input field.

```
/>
```

Example Usage for Displaying All Messages:

```
<!-- get message data -->
<IM:get.message.data dataset="data" topic="hammers"/>
<!-- iterate through messages in discussion records -->
<IM:iterate.message.records dataset="data">
<!-- get handle to individual record -->
<IM:get.message.record>
<!-- space message to show hierarchy and display message title and text-->
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.title/> <br>
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.text/> <br>
<!-- display a reply link that passes the message record id to the reply page -->
<a href="index?page=messagereply&rec=<%=message.recordid%>">Reply</a> <br>
</IM:get.message.record>
</IM:iterate.message.records>
```

Example Usage for Replying to a Message:

```
<!-- declare page variable and load recordid parameter off url from previous page -->
<% String rec = request.getParameter("rec");%>
```

```
<!-- create HTML form for input fields -->
<IM:form.message parentid="<%=rec%>" success="successpage" error="errorpage">
<!-- display input fields for title and text -->
Title<br>
<input.message.title><br>
Text<br>
<input.message.text><br>
<!-- display submit button -->
<input type="submit" value="Submit">
</IM:form.message>
```

Scripting Variables:

none

Body Tags:

none

input.message.text

Description:

Creates an HTML INPUT element for a message body text.

Format:

```
<IM:input.message.text
```

`size="string"`	Size of the input field.
`maxlength="string"`	Maximum length of the input field.
`value*="string"`	Initial value of the input field.

```
/>
```

Example Usage for Displaying All Messages:

```
<!-- get message data -->
<IM:get.message.data dataset="data" topic="hammers"/>
<!-- iterate through messages in discussion records -->
<IM:iterate.message.records dataset="data">
<!-- get handle to individual record -->
<IM:get.message.record>
<!-- space message to show hierarchy and display message title and text-->
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.title/> <br>
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.text/> <br>
```

```
<!-- display a reply link that passes the message record id to the reply page -->
<a href="index?page=messagereply&rec=<%=message.recordid%>">Reply</a> <br>
</IM:get.message.record>
</IM:iterate.message.records>
```

Example Usage for Replying to a Message:

```
<!-- declare page variable and load recordid parameter off url from previous page -->
<% String rec = request.getParameter("rec");%>
<!-- create HTML form for input fields -->
<IM:form.message parentid="<%=rec%>" success="successpage" error="errorpage">
<!-- display input fields for title and text -->
Title<br>
<input.message.title><br>
Text<br>
<input.message.text><br>
```

```
<!-- display submit button -->
<input type="submit" value="Submit">
</IM:form.message>
```

Scripting Variables:

none

Body Tags:

none

input.message.title

Description:

Creates an HTML INPUT element for a message title.

Format:

```
<IM:input.message.title
```

`size="string"`	Size of the input field.
`maxlength="string"`	Maximum length of the input field.
`value*="string"`	Initial value of the input field.

```
/>
```

Example Usage for Displaying All Messages:

```
<!-- get message data -->
<IM:get.message.data dataset="data" topic="hammers"/>
<!-- iterate through messages in discussion records -->
<IM:iterate.message.records dataset="data">
<!-- get handle to individual record -->
<IM:get.message.record>
<!-- space message to show hierarchy and display message title and text-->
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.title/> <br>
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.text/> <br>
<!-- display a reply link that passes the message record id to the reply page -->
<a href="index?page=messagereply&rec=<%=message.recordid%>">Reply</a> <br>
</IM:get.message.record>
</IM:iterate.message.records>
```

Example Usage for Replying to a Message:

```
<!-- declare page variable and load recordid parameter off url from previous page -->
<% String rec = request.getParameter("rec");%>
<!-- create HTML form for input fields -->
<IM:form.message parentid="<%=rec%>" success="successpage" error="errorpage">
<!-- display input fields for title and text -->
Title<br>
<input.message.title><br>
Text<br>
<input.message.text><br>
<!-- display submit button -->
<input type="submit" value="Submit">
</IM:form.message>
```

Scripting Variables:

none

Body Tags:

none

input.recommendation.contribution

<IM:input.recommendation.contribution

attribute*="string"	Attribute to contribute. Valid values are: title -- Recommendation Title text -- Recommendation text case number -- Case Num channel -- Content Channel for recommendation categories -- Categories associated with recommendation priority -- priority for recommendation
size="string"	Size of the input field.
maxlength="string"	Maximum length of the input field.
value="string"	Initial value of the input field. if nothing is entered for the value, the value will supplied by the information in the database.
css="string"	CSS Class to be used for the input field
cols="string"	If this input is a text area, this attribute defines how many columns it will have
rows="string"	If this input is a text area, this attribute defines how many rows it will have

/>

Description:
Creates an HTML input element for a content recommendation attribute

input.search.attribute

Description:

Creates an HTML INPUT submit element for performing attribute searches on a content channel.

Format:

```
<IM:input.search.attribute
attribute*="string"

/>
```

Example Usage for Capturing Search Text:

```
<IM:form.search.attribute channel="directory" matchtype="ALL" success="searchresults"
error="errorpage">
First Name:<br>
<IM:input.search.attribute attribute="directory\first_name"/>
<br>
Last Name:<br>
<IM:input.search.attribute attribute="directory\last_name"/>
<br>
<input type="submit"/>
</IM:form.search.attribute>
```

Example Usage Displaying Search Results:

```
<IM:iterate.channel.records dataset="searchresults" maxpageitems="2">
<IM:get.channel.record id="currentrecord">
<b>Channel: <IM:get.channel.attribute attribute="TYPE" /></b><br>
<a href="index?page=detail&guid=<%=currentrecord.recordid%>"><IM:get.channel.attribute
attribute="TITLE" /></a><br>
</IM:get.channel.record>
</IM:iterate.channel.records>
```

Scripting Variables:

none

Body Tags:

none

input.search.fulltext

Description:

Creates an HTML INPUT element for capturing the full text search text.

Format:

```
<IM:input.search.fulltext
size="string"
maxlength="string"
value="string"

/>
```

Example Usage for Capturing Search Text:

```
<IM:form.search.fulltext channels="news+events+press" success="searchresults"
error="errorpage">
Search String: <br>
<IM:input.search.fulltext/>
<br>
<input type="submit"/>
</IM:form.search.fulltext>
```

Example Usage for Displaying Search Results:

```
<IM:iterate.channel.records dataset="searchresults" maxpageitems="2">
<IM:get.channel.record id="currentrecord">
<b>Channel: <IM:get.channel.attribute attribute="TYPE" /></b><br>
<a href="index?page=detail&guid=<%=currentrecord.recordid%>"><IM:get.channel.attribute
attribute="TITLE" /></a><br>
</IM:get.channel.record>
</IM:iterate.channel.records>
```

Scripting Variables:

none

Body Tags:

none

input.subscription

Description:

Tag that display a checkbox for the current subscription in the repetition.

Format:

<IM:input.subscription

`css="string"`	CSS Class to be used for the input field
`showtext="true" or "false"`	If evaluates to true, the text of the current subscription will be displayed next to the checkbox. Otherwise only the checkbox will be displayed. In this case please use the get.subscription.name tag to get the subscription name. If this argument is omitted, the default value is set to true.
`checked="true" or "false"`	If evaluates to true, all the checkboxes in the repetition will be pre-checked.

></IM:input.subscription >

input.subscription.email

Description:

Creates an HTML input element for an email address to be search for. Typically used for searching email address in the subscribe/unsubscribe process.

Format:

<IM:input.subscription.email

`size="string"`	Size of the input field.
`maxlength="string"`	Maximum length of the input field.
`css="string"`	CSS Class to be used for the input field

/>

input.user.attribute

Description:

Creates an HTML input element for a custom user attribute.

Format:

<IM:input.user.attribute

attribute*="string"	Path to custom user attribute.
size="string"	Size of the input field.
maxlength="string"	Maximum length of the input field.
value*="string"	Initial value of the input field. if nothing is entered for the value, the value will default to the information in the database.
css="string"	CSS Class to be used for the input field
cols="string"	If this input is a text area, this attribute defines how many columns it will have
rows="string"	If this input is a text area, this attribute defines how many rows it will have

/>

Example Usage:

```
<IM:form.user.attributes success="successpage" error="errorpage">
User ID: <IM:input.user.id.error/><br>
<IM:input.user.id value="" system size="20"/><br>
Password: <IM:input.user.password .error/><br>
<IM:input.user.password value="" size="20"/><br>
First Name: <IM:input.user.firstname .error/><br>
<IM:input.firstname value="" size="30"/><br>
Last Name: <IM:input.user.lastname .error/><br>
<IM:input.user.lastname value="" size="30"/><br>
Email Address: <IM:input.user.email.error/><br>
<IM:input.user.email value="" size="30"/><br>
Phone Number:<br>
<IM:input.user.attribute attribute="demo/phone" value="" size="10"/><br>
Zip Code<br>

<IM:input.user.attribute attribute="demo/zipcode" value="" size="5"/><br>
<br>
<IM:input.submit value="Save Changes"/>
</IM:form.user.attributes>
```

Scripting Variables:

none

Body Tags:

none

input.user.email

Description:

Creates an HTML input element for a users email address.

Format:

```
<IM:input.user.email
```

`size="string"`	Size of the input field.
`maxlength="string"`	Maximum length of the input field.
`value*="string"`	Initial value of the input field. if nothing is entered for the value, the value will default to the information in the database.
`css="string"`	CSS Class to be used for the input field

```
/>
```

Example Usage:

```
<IM:form.user.attributes success="successpage" error="errorpage">
User ID: <IM:input.user.id.error/><br>
<IM:input.user.id value="" system size="20"/><br>
Password: <IM:input.user.password .error/><br>
<IM:input.user.password value="" size="20"/><br>
First Name: <IM:input.user.firstname .error/><br>
<IM:input.firstname value="" size="30"/><br>
Last Name: <IM:input.user.lastname .error/><br>
<IM:input.user.lastname value="" size="30"/><br>
Email Address: <IM:input.user.email.error/><br>
<IM:input.user.email value="" size="30"/><br>
Phone Number:<br>
<IM:input.user.attribute attribute="demo/phone" value="" size="10"/><br>
Zip Code<br>
<IM:input.user.attribute attribute="demo/zipcode" value="" size="5"/><br>
<br>
<IM:input.submit value="Save Changes"/>
</IM:form.user.attributes>
```

Scripting Variables:

none

Body Tags:

none

input.user.email.error

Description:

Displays email errors message from Action requests when user enters incorrect email address.

Format:

```
<IM:input.user.email.error
/>
```

Example Usage:

```
<IM:form.user.attributes success="successpage" error="errorpage">
User ID: <IM:input.user.id.error/><br>
<IM:input.user.id value="" system size="20"/><br>
Password: <IM:input.user.password .error/><br>
<IM:input.user.password value="" size="20"/><br>
First Name: <IM:input.user.firstname .error/><br>
<IM:input.firstname value="" size="30"/><br>
Last Name: <IM:input.user.lastname .error/><br>
<IM:input.user.lastname value="" size="30"/><br>
Email Address: <IM:input.user.email.error/><br>
<IM:input.user.email value="" size="30"/><br>
Phone Number:<br>
<IM:input.user.attribute attribute="demo/phone" value="" size="10"/><br>
Zip Code<br>
<IM:input.user.attribute attribute="demo/zipcode" value="" size="5"/><br>
<br>
<IM:input.submit value="Save Changes"/>
</IM:form.user.attributes>
```

Scripting Variables:

none

Body Tags:

none

input.user.firstname

Description:

Creates an HTML input element for a users fist name.

Format:

```
<IM:input.user.firstname
```

`size="string"`	Size of the input field.
`maxlength="string"`	Maximum length of the input field.
`value*="string"`	Initial value of the input field. if nothing is entered for the value, the value will default to the information in the database.
`css="string"`	CSS Class to be used for the input field

```
/>
```

Example Usage:

```
<IM:form.user.attributes success="successpage" error="errorpage">
User ID: <IM:input.user.id.error/><br>
<IM:input.user.id value="" system size="20"/><br>
Password: <IM:input.user.password .error/><br>
<IM:input.user.password value="" size="20"/><br>
First Name: <IM:input.user.firstname .error/><br>
<IM:input.firstname value="" size="30"/><br>
Last Name: <IM:input.user.lastname .error/><br>
<IM:input.user.lastname value="" size="30"/><br>
Email Address: <IM:input.user.email.error/><br>
<IM:input.user.email value="" size="30"/><br>
Phone Number:<br>
<IM:input.user.attribute attribute="demo/phone" value="" size="10"/><br>
Zip Code<br>
<IM:input.user.attribute attribute="demo/zipcode" value="" size="5"/><br>
<br>
<IM:input.submit value="Save Changes"/>
</IM:form.user.attributes>
```

Scripting Variables:

none

Body Tags:

none

input.user.firstname.error

Description:

Displays errors message from action requests when no fist name is entered.

Format:

```
<IM:input.user.firstname.error
/>
```

Example Usage:

```
<IM:form.user.attributes success="successpage" error="errorpage">
User ID: <IM:input.user.id.error/><br>
<IM:input.user.id value="" system size="20"/><br>
Password: <IM:input.user.password .error/><br>
<IM:input.user.password value="" size="20"/><br>
First Name: <IM:input.user.firstname .error/><br>
<IM:input.firstname value="" size="30"/><br>
Last Name: <IM:input.user.lastname .error/><br>
<IM:input.user.lastname value="" size="30"/><br>
Email Address: <IM:input.user.email.error/><br>
<IM:input.user.email value="" size="30"/><br>
Phone Number:<br>
<IM:input.user.attribute attribute="demo/phone" value="" size="10"/><br>
Zip Code<br>
<IM:input.user.attribute attribute="demo/zipcode" value="" size="5"/><br>
<br>
<IM:input.submit value="Save Changes"/>
</IM:form.user.attributes>
```

Scripting Variables:

none

Body Tags:

none

input.user.id

Description:

Creates an HTML input element for a user id.

Format:

<IM:input.user.id

`size="string"`	Size of the input field.
`maxlength="string"`	Maximum length of the input field.
`value*="string"`	Initial value of the input field. if nothing is entered for the value, the value will default to the information in the database.
`css="string"`	CSS Class to be used for the input field

/>

Example Usage:

```
<IM:form.user.attributes success="successpage" error="errorpage">
User ID: <IM:input.user.id.error/><br>
<IM:input.user.id value="" system size="20"/><br>
Password: <IM:input.user.password .error/><br>
<IM:input.user.password value="" size="20"/><br>
First Name: <IM:input.user.firstname .error/><br>
<IM:input.firstname value="" size="30"/><br>
Last Name: <IM:input.user.lastname .error/><br>
<IM:input.user.lastname value="" size="30"/><br>
Email Address: <IM:input.user.email.error/><br>
<IM:input.user.email value="" size="30"/><br>
Phone Number:<br>
<IM:input.user.attribute attribute="demo/phone" value="" size="10"/><br>
Zip Code<br>
<IM:input.user.attribute attribute="demo/zipcode" value="" size="5"/><br>
<br>
<IM:input.submit value="Save Changes"/>
</IM:form.user.attributes>
```

Scripting Variables:

none

Body Tags:

none

input.user.id.error

Description:

Displays errors message from action requests when no user ID is entered.

Format:

```
<IM:input.user.id.error
/>
```

Example Usage:

```
<IM:form.user.attributes success="successpage" error="errorpage">
User ID: <IM:input.user.id.error/><br>
<IM:input.user.id value="" system size="20"/><br>
Password: <IM:input.user.password .error/><br>
<IM:input.user.password value="" size="20"/><br>
First Name: <IM:input.user.firstname .error/><br>
<IM:input.firstname value="" size="30"/><br>
Last Name: <IM:input.user.lastname .error/><br>
<IM:input.user.lastname value="" size="30"/><br>
Email Address: <IM:input.user.email.error/><br>
<IM:input.user.email value="" size="30"/><br>
Phone Number:<br>
<IM:input.user.attribute attribute="demo/phone" value="" size="10"/><br>
Zip Code<br>
<IM:input.user.attribute attribute="demo/zipcode" value="" size="5"/><br>
<br>
<IM:input.submit value="Save Changes"/>
</IM:form.user.attributes>
```

Scripting Variables:

none

Body Tags:

none

input.user.lastname

Description:

Creates an HTML input element for a user last name.

Format:

<IM:input.user.lastname

`size="string"`	Size of the input field.
`maxlength="string"`	Maximum length of the input field.
`value*="string"`	Initial value of the input field. if nothing is entered for the value, the value will default to the information in the database.
`css="string"`	CSS Class to be used for the input field

/>

Example Usage:

```
<IM:form.user.attributes success="successpage" error="errorpage">
User ID: <IM:input.user.id.error/><br>
<IM:input.user.id value="" system size="20"/><br>
Password: <IM:input.user.password .error/><br>
<IM:input.user.password value="" size="20"/><br>
First Name: <IM:input.user.firstname .error/><br>
<IM:input.firstname value="" size="30"/><br>
Last Name: <IM:input.user.lastname .error/><br>
<IM:input.user.lastname value="" size="30"/><br>
Email Address: <IM:input.user.email.error/><br>
<IM:input.user.email value="" size="30"/><br>
Phone Number:<br>
<IM:input.user.attribute attribute="demo/phone" value="" size="10"/><br>
Zip Code<br>
<IM:input.user.attribute attribute="demo/zipcode" value="" size="5"/><br>
<br>
<IM:input.submit value="Save Changes"/>
</IM:form.user.attributes>
```

Scripting Variables:

none

Body Tags:

none

input.user.lastname.error

Description:

Displays errors message from action requests when no last name is entered.

Format:

```
<IM:input.user.lastname.error
/>
```

Example Usage:

```
<IM:form.user.attributes success="successpage" error="errorpage">
User ID: <IM:input.user.id.error/><br>
<IM:input.user.id value="" system size="20"/><br>
Password: <IM:input.user.password .error/><br>
<IM:input.user.password value="" size="20"/><br>
First Name: <IM:input.user.firstname .error/><br>
<IM:input.firstname value="" size="30"/><br>
Last Name: <IM:input.user.lastname .error/><br>
<IM:input.user.lastname value="" size="30"/><br>
Email Address: <IM:input.user.email.error/><br>
<IM:input.user.email value="" size="30"/><br>
Phone Number:<br>
<IM:input.user.attribute attribute="demo/phone" value="" size="10"/><br>
Zip Code<br>
<IM:input.user.attribute attribute="demo/zipcode" value="" size="5"/><br>
<br>
<IM:input.submit value="Save Changes"/>
</IM:form.user.attributes>
```

Scripting Variables:

none

Body Tags:

none

input.user.password

Description:

Creates an HTML input element for a user password.

Format:

```
<IM:input.user.password
```

`size="string"`	Size of the input field.
`maxlength="string"`	Maximum length of the input field.
`value*="string"`	Initial value of the input field. a default password will NEVER be supplied by the database
`css="string"`	CSS Class to be used for the input field

```
/>
```

Example Usage:

```
<IM:form.user.attributes success="successpage" error="errorpage">
User ID: <IM:input.user.id.error/><br>
<IM:input.user.id value="" system size="20"/><br>
Password: <IM:input.user.password .error/><br>
<IM:input.user.password value="" size="20"/><br>
First Name: <IM:input.user.firstname .error/><br>
<IM:input.firstname value="" size="30"/><br>
Last Name: <IM:input.user.lastname .error/><br>
<IM:input.user.lastname value="" size="30"/><br>
Email Address: <IM:input.user.email.error/><br>
<IM:input.user.email value="" size="30"/><br>
Phone Number:<br>
<IM:input.user.attribute attribute="demo/phone" value="" size="10"/><br>
Zip Code<br>
<IM:input.user.attribute attribute="demo/zipcode" value="" size="5"/><br>
<br>
<IM:input.submit value="Save Changes"/>
</IM:form.user.attributes>
```

Scripting Variables:

none

Body Tags:

none

input.user.password.error

Description:

Displays errors message from action requests when no password is entered.

Format:

```
<IM:input.user.password.error
/>
```

Example Usage:

```
<IM:form.user.attributes success="successpage" error="errorpage">
User ID: <IM:input.user.id.error/><br>
<IM:input.user.id value="" system size="20"/><br>
Password: <IM:input.user.password .error/><br>
<IM:input.user.password value="" size="20"/><br>
First Name: <IM:input.user.firstname .error/><br>
<IM:input.firstname value="" size="30"/><br>
Last Name: <IM:input.user.lastname .error/><br>
<IM:input.user.lastname value="" size="30"/><br>
Email Address: <IM:input.user.email.error/><br>
<IM:input.user.email value="" size="30"/><br>
Phone Number:<br>
<IM:input.user.attribute attribute="demo/phone" value="" size="10"/><br>
Zip Code<br>
<IM:input.user.attribute attribute="demo/zipcode" value="" size="5"/><br>
<br>
<IM:input.submit value="Save Changes"/>
</IM:form.user.attributes>
```

Scripting Variables:

none

Body Tags:

none

input.wizardfield.record

Description:

Tag that provides the input html representing the current WizardFieldObject

Format:

<IM:input.wizardfield.record

`css="string"` The unique id representing the WizardFieldObject

/>

Example Usage:

```
<html>
<head>

</head>
<body>
... <IM:input.wizardfield.record css="dropdown"/> ....
</body>
</html>
```

Scripting Variables:

none

Body Tags:

none

is.admin

Description:

Checks if the logged in user (if any) is an admin user. If it is, it displays the contents inside this tag, and if the user is not an admin user, it doesn't display the contents.

Format:

<IM:is.admin

`negate="non-scriptable string"`	Reverses the conditional of this tag. If it evaluates to "true" it will show the contents of this tag if the user is not an admin user. The default behavior of this attribute is to be false.

></IM:is.admin >

Example Usage:

```
<IM:is.admin>
HTML for admin users. In example: a link for the admin application
<a href="http://www.someadminurl.com?<IM:logincredentials/>">Go To Admin</a>
</IM:is.admin>
<IM:is.admin negate="true">
HTML for non admin users
</IM:is.admin>
```

is.inquirasearch.enabled

Description:

Checks if InQuira Search has been configured and is enabled.

Format:

<IM:is.inquirasearch.enabled

`negate="string"`	Reverses the conditional of this tag. If it evaluates to "true", it will show the contents of this tag if Search is enabled. The default behavior of this attribute is false.

></IM:is.inquirasearch.enabled >

Scripting Variables:

none

Body Tags:

none

is.loggedin

Description:

Checks if there is a logged in user in the session. If there is one, it displays the contents inside this tag, and if there is no user logged in it doesn't display the contents.

Format:

<IM:is.loggedin

`negate="non-scriptable string"`	Reverses the conditional of this tag. If it evaluates to "true" it will show the contents of this tag if the user is not logged in. The default behavior of this attribute is to be false.

></IM:is.loggedin >

Example Usage:

<IM:is.loggedin>

HTML for logged in users
</IM:is.loggedin>
<IM:is.loggedin negate="true">

iterate.channel.child.records

Description:

Iterates over content records associated with the current records.

Format:

<IM:iterate.channel.child.records

`url="string"`	The url of the page we are currently viewing. DO NOT use, for internal application use only
`maxpageitems="integer"`	Used by the paging mechanism, this defines the maximum items to display (iterate over) per page.
`maxindexpages="integer"`	Used by the paging mechanism, this defines the maximum number of pages to display.
`isoffset="string"`	The offset of the page we are currently viewing. DO NOT use, for internal application use only.

></IM:iterate.channel.child.records >

Example Usage:

```
<!-- Get passed in record id form URL-->
<% String rec = request.getParameter("rec");%>

<!--Get handle to current record-->
<IM:get.channel.record recordid="<%=rec%>">
<!-- Display current record title and body attribute for current record -->
<IM:get.channel.attribute attribute="NEWS\TITLE"><br>
<IM:get.channel.attribute attribute="NEWS\BODY"><br>
Related Content:<br>
<!-- Iterate channel child record -->
<IM:iterate.channel.child.records id="relatedrecords">
<!--Get handle to current record-->
<IM:get.channel.record id="relatedrecord">
<!-- Display title attribute for current record -->
<IM:get.channel.attribute attribute="NEWS\TITLE">
<!-- Provide hyperlink to same detail page for the related record -->

<a href="index?page=detail&rec=<%=relatedrecord.recordid%>>Read more</a><br>
</IM:get.channel.record>
```

</IM:iterate.channel.child.records>
</IM:get.channel.record>

Scripting Variables:

{id}.faqsmaxitems	used internally for page batching
{id}.faqspagenumber	used internally for page batching
{id}.faqsoffset	used internally for page batching

Body Tags:

get.channel.record on page 123

get.channel.attribute on page 108

iterate.channel.parent.records

Description:

Iterates over content records that associated the current record.

Format:

<IM:iterate.channel.parent.records

`url="string"`	The url of the page we are currently viewing. DO NOT use, for internal use only
`maxpageitems="integer"`	Used by the paging mechanism, this defines the maximum items to display (iterate over) per page.
`maxindexpages="integer"`	Used by the paging mechanism, this defines the maximum number of pages to display.
`isoffset="string"`	The offset of the page we are currently viewing. DO NOT use, for internal use only.

></IM:iterate.channel.parent.records >

Example Usage:

```
<!-- Get passed in record id form URL-->
<% String rec = request.getParameter("rec");%>

<!--Get handle to current record-->
<IM:get.channel.record recordid="<%=rec%>">
<!-- Display current record title and body attribute for current record -->
<IM:get.channel.attribute attribute="NEWS\TITLE"><br>
<IM:get.channel.attribute attribute="NEWS\BODY"><br>
Parent Content Records:<br>
<!-- Iterate channel child record -->
<IM:iterate.channel.parent.records id="parentrecords">
<!--Get handle to current record-->
<IM:get.channel.record id="parentrecord">
<!-- Display title attribute for current record -->
<IM:get.channel.attribute attribute="NEWS\TITLE">
<!-- Provide hyperlink to same detail page for the related record -->

<a href="index?page=detail&rec=<%=parentrecord.recordid%>>Read more</a><br>
</IM:get.channel.record>
</IM:iterate.channel.parent.records>
</IM:get.channel.record>
```

Scripting Variables:

{id}.faqsmaxitems	used internally for page batching
{id}.faqspagenumber	used internally for page batching
{id}.faqsoffset	used internally for page batching

Body Tags:

get.channel.record on page 123

get.channel.attribute on page 108

iterate.channel.records

Description:

Iterate through all content records in a dataset.

Format:

```
<IM:iterate.channel.records
```

`dataset="string"`	the name of the dataset to iterate through.
`id="string"` `(Non-scriptable string)`	a desired prefix to use for all scripting variables created by this tag. The default value is `iterator`. NOTE: Ensure that all ids within nested channel-iterator tags are unique.
`node="string"`	a path to the desired node(s).
`url="string"`	the URL of the currently active page in the web application. IMPORTANT: Do Not Use; this parameter is reserved for internal use.
`maxpageitems="integer"`	the maximum number of items per page to display (iterate through). This parameter is used by the paging mechanism.
`maxindexpages="integer"`	the maximum number of pages to display. This parameter is used by the paging mechanism.
`isoffset="string"`	The offset of the currently active page in the web application. IMPORTANT: Do Not Use; this parameter is reserved for internal use.

```
></IM:iterate.channel.records >
```

Example Usage:

```
<!--Get channel data records using query-->
<IM:get.channel.data query="topnews" dataset="mynews">
<!-- Iterate channel record -->
<IM:iterate.channel.records dataset="mynews">
<!--Get handle to current record-->
<IM:get.channel.record id="news">
<!-- Display title attribute for current record -->
<IM:get.channel.attribute attribute="NEWS\TITLE">
<!-- Provide hyperlink to detail page for current record -->
<a href="index?page=detail&rec=<%=news.recordid%>>Read more</a><br>
</IM:get.channel.record>
</IM:iterate.channel.records>
```

Scripting Variables:

None

Required Parent Tags:

requires a parent tag that creates a dataset

TagClass:

com.inquira.client.tags.ChannelIteratorTag

iterate.dataform.answer

Description:

This tag iterates through the answer records associated with a data form question.

Format:

```
<IM:iterate.dataform.answer
></IM:iterate.dataform.answer >
```

Example Usage for Displaying the Form:

```
<IM:form.dataform dataform="INQUIRIES" success="requestthank" error="requestthank">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:input.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question> <br>
<input type="submit" value="Submit">
</IM:form.dataform>
```

Example Usage for Displaying Results

```
<IM:get.dataform.results name="nameofform" individual="false" aggregate="false">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:get.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question>
</IM:get.dataform.results>
```

Scripting Variables:

Name	Scope	Description
{id}.faqsmaxitems	page	Used internally for page batching.

{id}.faqspagenumber	page	Used internally for page batching.
{id}.faqsoffset	page	Used internally for page batching.

Body Tags:

get.dataform.answer.record on page 142

get.dataform.answer on page 141

input.dataform.answer on page 224

TagClass:

com.inquira.client.tags.SurveyAnswerIteratorTag

iterate.dataform.question

Description:

This tag iterates over the question records in a data form. You can use this tag to dynamically build a form.

Format:

```
<IM:iterate.dataform.question
></IM:iterate.dataform.question >
```

Attributes:

None

Example Usage for Displaying the Form

```
<IM:form.dataform dataform="INQUIRIES" success="requestthank" error="requestthank">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:input.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question> <br>
<input type="submit" value="Submit">
</IM:form.dataform>
```

Example Usage for Displaying Results:

```
<IM:get.dataform.results name="nameofform" individual="false" aggregate="false">
<IM:iterate.dataform.question>
<IM:get.dataform.question.record>
<b><IM:get.dataform.question/></b><br>
<IM:iterate.dataform.answer>
<IM:get.dataform.answer.record>
<IM:get.dataform.answer/><br>
</IM:get.dataform.answer.record>
</IM:iterate.dataform.answer>
</IM:get.dataform.question.record>
</IM:iterate.dataform.question>
</IM:get.dataform.results>
```

Scripting Variables:

None

Required Parent Tags:

get.dataform.question.record on page 147

get.dataform.question on page 146

iterate.dataform.answer on page 254

get.dataform.answer.record on page 142

get.dataform.answer on page 141

input.dataform.answer on page 224

TagClass:

com.inquira.client.tags.SurveyQuestionIteratorTag

iterate.dataset

Description:

This is a generic tag that iterates over a list of records returned in the specified data set.

Format:

```
<IM:iterate.dataset
```

`dataset="string"` `(Required)`	specifies the data set containing the record.
`id="string"` `(Non-scriptable string)`	specifies a prefix to use for all scripting variables created by this tag. Each ID must be unique. There is no default ID.

```
></IM:iterate.dataset >
```

Scripting Variables:

Name	Scope	Description
index	page	Returns the current index of the iteration.
count	page	Returns the total number of objects provided by the data set.

Required Parent Tags:

requires get.content.data or another tag to generate the data set

iterate.index

Description:

Format:

```
<IM:iterate.index
></IM:iterate.index >
```

iterate.message.records

Description:

Iterate over message records passed in with the data set.

Format:

```
<IM:iterate.message.records
```

`url="string"`	The url of the page we are currently viewing. DO NOT use. For internal application use only
`maxpageitems="integer"`	Used by the paging mechanism, this defines the maximum items to display (iterate over) per page.
`maxindexpages="integer"`	Used by the paging mechanism, this defines the maximum number of pages to display.
`isOffset="string"`	The offset of the page we are currently viewing. DO NOT use. For internal application use only.
`talkbackdepth="integer"`	the depth of talkback to retrieve
`contentid=`	guid of talkback record to iterate over

```
></IM:iterate.message.records >
```

Displaying All Messages:

```
<!-- get message data -->
<IM:get.message.data dataset="data" topic="hammers"/>
<!-- iterate through messages in discussion records -->
<IM:iterate.message.records dataset="data">
<!-- get handle to individual record -->
<IM:get.message.record>
<!-- space message to show hierarchy and display message title and text-->
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.title/> <br>
<img src="images\spacer.gif" width="<% =message.depth %>" aligh="left"/
><IM:get.message.text/> <br>

<!-- display a reply link that passes the message record id to the reply page -->
<a href="index?page=messagereply&rec=<%=message.recordid%>">Reply</a> <br>
```

```
</IM:get.message.record>
</IM:iterate.message.records>
```

Example Usage for Replying to a Message:

```
<!-- declare page variable and load recordid parameter off url from previous page -->
<% String rec = request.getParameter("rec");%>
<!-- create HTML form for input fields -->
<IM:form.message parentid="<%=rec%>" success="successpage" error="errorpage">
<!-- display input fields for title and text -->
Title<br>
<input.message.title><br>
Text<br>
<input.message.text><br>
<!-- display submit button -->
<input type="submit" value="Submit">
</IM:form.message>
```

Body Tags:

get.message.record on page 177

get.message.title on page 180

get.message.text on page 179

form.message on page 87

input.message.title on page 228

input.message.text on page 226

iterate.metadata.records

Description:

This tag iterates over a dataset of metadata records.

Format:

<IM:iterate.metadata.records

`dataset*="string"`	this is a unique identifier for the data set used by the `iterate.channel.records` and `get.channel.template` tags to access records that are retrieved.

></IM:iterate.metadata.records >

iterate.node

Description:

Tag that iterates over a list of nodes defined in the xpath of a user.

Format:

<IM:iterate.node

`node*="string"`	Xpath of node to be iterated
`id="non-scriptable string"`	A desired prefix to use for all scripting variables created by this tag. There is no default id.

></IM:iterate.node >

iterate.search.portlets

Description:

Iterate over the list of the portlets.

Format:

<IM:iterate.search.portlets

`order="string"`	The unique id representing `PreviousResponses` up to the wizard step id passed in
`dataset="string"` (Required)	A unique identifier for the data set which is used by `iterate.dataset` to step through all of the returned Portlets to retrieve the value passed in.

/>

Example Usage:

```
<html>content="<iterate.search.portlets
<head>

</head>
<body>
... ... <Table border=0 align=right width=20%><tr><td>

<IM:iterate.search.portlets dataset="rsData.portlets"
order="PROMOTE+DEFINE+RELATED_TOPIC+FEEDBACK">
  <IM:get.inquirasearch.portlet id="porter">
  <% if (porter.type.equals("feedback")) { %>
        <form name="feedbackform" action="index" method=get>
        <input type=hidden name=page value="<%=myPage%>">
        <input type=hidden name=type value=feedback>
  <% } %>
  <table border=1 width=100%> <tr><th class="im-lightblue"><%= porter.name %></th> </tr>
          <IM:iterate.dataset dataset="porter.items" id="pitems">
            <IM:get.inquirasearch.portlet.item id="item">
<TR><td>Answer ID = <%= item.answerid %> <% if (porter.type.equals("feedback")) { %>
                  <%= item.excerpt %><BR>
                 <% if (pitems.index == 4) { %>
                       <input type=submit name=type value=submit>
                       </form>
                  <% } %>
        <% } else { %>
        <% if (item.ansType.equals("dictionary")) { %>
```

```
<a
href="index?page=<%=myPage%>&type=search&showdef=true&title=<%=item.getLinkText()%
>&def=<%=item.excerpt%>&answerid=<%=item.answerid%>&iqaction=<%=item.iqaction%>"><
%= item.getLinkText() %></a> <br><%= item.excerpt %>
<BR><BR>

                <% } else {%>
                <a href="<%=item.titleUrl%>"><%= item.getLinkText() %></a> <br><%=
item.excerpt %><BR><BR>
                <% } %>
        <% } %>
              </td></tr>
               </IM:get.inquirasearch.portlet.item>
             </IM:iterate.dataset>

  </IM:get.inquirasearch.portlet>
  </IM:iterate.search.portlets>

</td><tr></Table>

</body>
</html>
```

Scripting Variables:

none

Body Tags:

none

iterate.subscription.records

Description:

Tag that is used for iterating over the list of available News Letter.

Format:

```
<IM:iterate.subscription.records
```

`view="string"`	The reference key of the desired view.
`dataform="string"`	The name of the data form from which get the available subscriptions.

```
></IM:iterate.subscription.records >
```

iterate.wizardform.fields

Description:

Iterate over the ProcessWizard form's WizardFieldObjects

Format:

```
<IM:iterate.wizardform.fields
/>
```

Example Usage:

```
<html>
<head>

</head>
<body>
... <IM:iterate.wizardform.fields>
   <br>
        <IM:get.wizardfield.record id="wizf">
   <% if (wizf.type.equals("select")) { %>
        <IM:input.wizardfield.record css="dropdown"/>
   <% } else if (wizf.type.equals("checkbox")) { %>
        <IM:input.wizardfield.record css="chekbx"/>
   <% } else if (wizf.type.equals("text")) { %>
        <IM:input.wizardfield.record css="letext"/>
   <% } else if (wizf.type.equals("radio")) { %>
        <IM:input.wizardfield.record css="amfm"/>
   <% } else { %>
             <br><b><%=wizf.text%></b> <br>
   <% } %>
        </IM:get.wizardfield.record>

   </IM:iterate.wizardform.fields>

</body>
</html>
```

Scripting Variables:

none

Body Tags:

none

iterate.wizard.previous.responses

Description:

Iterate over the list of the Process Wizard's previous responses

Format:

```
<IM:iterate.wizard.previous.responses
```

`stepid=string` (Required)	The unique ID representing `PreviousResponses` up to the wizard step id passed in

```
/>
```

Example Usage:

```
<html>
<head>

</head>
<body>
... <IM:iterate.wizard.previous.responses>
<IM:get.wizard.previous.response id="wpr">

<font color=blue size="3"><%= wpr.question %></font><br>
<font color=blue size="2"><%= wpr.answer %></font> <br><br>
</IM:get.wizard.previous.response>
</IM:iterate.wizard.previous.responses>

</body>
</html>
```

Scripting Variables:

none

Body Tags:

none

logout

Description:

This tag invalidates the current session. To use just place this tag in the logout page.

Format:

```
<IM:logout
></IM:logout >
```

manage.content

Description:

Returns a link to the admin tool used to manage content. It should specify if the action is EDIT, DELETE, ADD, or MANAGE . The result is a link with encrypted user credentials used for logging in the admin application. If the content ID and success and error attributes are passed, the user will be brought directly to that content record for editing.

Format:

<IM:manage.content

`contentid="string"`	If a content ID is provided with the login credentials, the user is directed directly to the associated content record for editing.
`category="string"`	If a category reference key is passed in, it will be preselected in the content edit screen and it will hide the category section.
`success="string"`	fully qualified URL to return the user to if the update was successful
`error="string"`	fully qualified URL to return the user to if an error occurs
`channel="string"`	Reference Key for ContentChannel. This parameter is required if adding content
`view="string"`	This is a view reference key. If this property is set and the action is add, then the content will be assigned to this view if the logged in user has enough permissions
`action*="string"`	EDIT - DELETE - ADD - MANAGE . Desired action to be performed. This is a required parameter that defines what action should be performed.

/>

Usage Example:

```
<a href="http://www.applicationurl.com?<IM:user.admin.link/>">Go To Admin</a>
```

node.count

Description:

Counts the number of nodes returned by an xpath.

Format:

<IM:node.count

`attribute*="string"`	Xpath to the nodes
`id="non-scriptable string"`	id of scripting variable

/>

pdf

Description:

This tag converts everything that is enclosed to a PDF document.

Format:

<IM:pdf

`filename="string"`	Name for the generated PDF file
`fontsize="string"`	Base font size for the generated PDF Document
`font="string"`	Sets the font of the document. Allowed values: Arial, Courier, Helvetica, Monospace, Sans-Serif, Serif, Symbol, Times

></IM:pdf >

save.content.attribute

Description:

This tag specifies the attribute and value to be modified on a content record using the `save.content.data` tag.

Format:

<IM:save.content.attribute

`attribute*="string"`	Attribute to be modified. To edit display start and end dates, or event start and end dates, you must specify one of the following key words and a date mask DISPLAYSTARTDATE, DISPLAYENDATE, EVENTSTARTDATE, EVENTENDDATE.
`value*="string"`	Value to be saved
`mask="string"`	Date mask for Content Dates.

/>

Required Parent Tag:

save.content.data

save.content.data

Description:

This tag automatically saves content data into a content record. This tag is used in conjunction with the save.content.attribute tag.

Format:

<IM:save.content.data

`contentid*="string"`	Content id of the content to be modified.
`publish="true" or "false"`	Flag that determines if the modified content record should be published or not.

></IM:save.content.data >

save.user.attribute

Description:

This tag will save a value to the passed in attribute path. If this tag is embedded within a get.user.record tag, it will save the property to the user specified in that tag, otherwise it will save the property to the logged in user.

Format:

<IM:save.user.attribute

`attribute*="string"`	The path to save the value
`value*="string"`	The value to be saved.

></IM:save.user.attribute >

set.audit.qualifier

Description:

Tag that is used in conjunction with the get.audit.data tag to created qualifiers to fetch the audit data.

Format:

<IM:set.audit.qualifier

`attribute*="string"`	The attribute to create the qualifier (i.e., userlogin, userid, contentid, ipaddress, etc.).
`operator*="string"`	The operator to be used. Allowed values--> Relational Operators: ("=", "<>", "<", ">", "<=", ">="), String Operators: "like", "caseInsensitiveLike"
`value*="string"`	The value to be set. You can also append wildcards (* or ?) if the operator is like or caseInsensitiveLike. If the value is of type dateadded or datemodified, you must supply the value in one of the following formats: "03/30/2005 16:30:05" OR "03/30/2004".

/>

set.content.casenumber

Description:

This tag assigns a case number to a content record.

<IM:set.content.casenumber

documentid="non-scriptable string"	Set the case number for the supplied document ID
recordid="non-scriptable string"	Set the case number for the supplied record ID
casenumber*="non-scriptable string"	The value to set the case number for the content record
casedescription*="non-scriptable string"	The value to set the case description for the content record
caseincident*="non-scriptable string"	The value to set the case incident for the content record

></IM:set.content.casenumber >

set.search.term

Description:

Tag that is used in conjunction with the get.search.data tag. This tag sets the text to be searched.

Format:

<IM:set.search.term

`attribute="string"`	If performing an attribute level search, set the desired attribute here.
`value="string"`	The actual text to search.

/>

set.securefilecookie

Description:

Tag that generates the required client cookies. Use this tag after all calls to the get.securefiletoken tag on a page.

```
<IM:set.securefilecookie
></IM:set.securefilecookie >
```

sitemap

Description:

This tag creates a sitemap entry used to specify security and login requirements for a page.

The Sitemap provides:

- A mechanism to protect the underlying structure of your site
- A concise and user-friendly URL for bookmarking

This tag is required only for JSP pages that are not components of other pages. For example, if Home.jsp includes several other files, only Home.jsp would require this tag. The included files would not need a sitemap tag.

Format:

```
<IM:sitemap
```

`pagename="string"` `(Required, Non-scriptable string)`	the name of the page that will be used in URLs by the Index servlet, for example: http://www.company.com/index?page=**pagename**
`requireslogon="true | false"` `(Non-scriptable string)`	specifies whether the user must be logged on to view this page (`true`). The default is `false`.
`ssl="true | false"` `(Non-scriptable string)`	specifies whether the page will use (https) a secure socket layer (SSL) (`true`).
`logonpage="string"` `(Non-scriptable string)`	the name of the page to which users will be redirected if they are not yet logged on. This name should be the name of another site map page entry.

`default="true | false"` `(Non-scriptable string)`	specifies whether the current page is the default page that will show if there is an error finding the specified page.
`namespace="string"`	specifies the namespace that this page will respond to. If set, only the specific namespace (case sensitive) will contain the page.

```
/>
```

To display the sitemap, append `&sitemap` to your URL:

```
http://www.company.com/index?page=home&sitemap
```

Scripting Variables:

None

Required Parent Tags:

None

TagClass:

com.inquira.client.tags.SitemapTag

template.definition

Description:

This tag creates a template definition.

Format:

```
<IM:template.definition
```

`template="string"` `(Required)`	Sitemap name for the template to use

```
></IM:template.definition >
```

Scripting Variables:

None

Required Parent Tags:

None

TagClass:

com.inquira.client.tags.TemplateInsertTag

template.get

Description:

This tag gets a template definition entry for content.

Format:

```
<IM:template.get
```

`name="string"` `(Required)`	Name of template entry

```
></IM:template.get >
```

Scripting Variables:

None

Required Parent Tags:

None

TagClass:

com.inquira.client.tags.TemplateGetTag

template.put

Description:

This tag adds a template definition entry.

Format:

```
<IM:template.put
```

`name="string"` `(Required)`	the sitemap name that the template will use.
`direct="string"`	specifies whether to include the code as JSP or as part of the template.
`content="string"` `(Required)`	the JSP file include or HTML to use for the template.

```
></IM:template.put >
```

Scripting Variables:

None

Required Parent Tags:

None

TagClass:

com.inquira.client.tags.TemplatePutTag

timer

Description:

This tag is used to time blocks of JSP code and is used in conjunction with the ***benchmark tag*** see *benchmark* on page 76 to display the resulting statistics.

Format:

<IM:timer

`id*="string"`	Optional ID for the timing block on the benchmark results page.

/IM:timer>

Example:

```
...
<IM:timer id=”test”
<IM:iterate.dataset dataset="newsData" id="itNewsData"> <IM:get.content.record id="crData">
<IM:get.channel.record id="cr" recordid="<%=crData.recordid%>">

<tr valign="top"><IM:get.channel.attribute.value
attribute="<%=Configuration.getStringForKey("channelForNews")+"/
"+Configuration.getStringForKey("attributeForImage")%>" id="image">

<%if(image.exists){%><td width="1%" valign="top"><span class="news_image"><a
href="index?page=content&id=<%= crData.docid %>"><img
src="<%=cr.resourcepath%><%=image.value%>" width="75" alt="" border="0"></a></span></
td>

<td valign="top"><h3><a href="index?page=content&id=<%= crData.docid %>"><%=
crData.masteridentifier %></a></h3>

<span class="small-info"><%=timeAgo(crData.displayStartDate)%><br /></span>

<IM:get.channel.attribute attribute="<%=Configuration.getStringForKey("channelForNews")+"/
"+Configuration.getStringForKey("defaultSummaryAttribute")%>"/></td><%} else {%>


<td valign="top" colspan="2"><h3><a href="index?page=content&id=<%= crData.docid
%>"><%=timeAgo(crData.displayStartDate)%> - <%= crData.masteridentifier %></a></h3>

<IM:get.channel.attribute attribute="<%=Configuration.getStringForKey("channelForNews")+"/
"+Configuration.getStringForKey("defaultSummaryAttribute")%>"/></td><%}%>

</IM:get.channel.attribute.value>

</tr>
```

```
</IM:get.channel.record></IM:get.content.record></IM:iterate.dataset>
<IM:iterate.dataset dataset="newsData" id="itNewsData"> <IM:get.content.record id="crData">
<IM:get.channel.record id="cr" recordid="<%=crData.recordid%>">

<tr valign="top"><IM:get.channel.attribute.value
attribute="<%=Configuration.getStringForKey("channelForNews")+"/
"+Configuration.getStringForKey("attributeForImage")%>" id="image">

<%if(image.exists){%><td width="1%" valign="top"><span class="news_image"><a
href="index?page=content&id=<%= crData.docid %>"><img
src="<%=cr.resourcepath%><%=image.value%>" width="75" alt="" border="0"></a></span></
td>

<td valign="top"><h3><a href="index?page=content&id=<%= crData.docid %>"><%=
crData.masteridentifier %></a></h3>

<span class="small-info"><%=timeAgo(crData.displayStartDate)%><br /></span>

<IM:get.channel.attribute attribute="<%=Configuration.getStringForKey("channelForNews")+"/
"+Configuration.getStringForKey("defaultSummaryAttribute")%>"/></td><%} else {%>

<td valign="top" colspan="2"><h3><a href="index?page=content&id=<%= crData.docid
%>"><%=timeAgo(crData.displayStartDate)%> - <%= crData.masteridentifier %></a></h3>

<IM:get.channel.attribute attribute="<%=Configuration.getStringForKey("channelForNews")+"/
"+Configuration.getStringForKey("defaultSummaryAttribute")%>"/></td><%}%>

</IM:get.channel.attribute.value>

</tr>

</IM:get.channel.record></IM:get.content.record></IM:iterate.dataset>
</IM:timer>
<IM:benchmark/>
...
```

update.content.metric

Description:

This tag automatically updates a specified metric for a content record in increments of 1. The initial metric is set to 0, and the initial event is counted as 1. You can specify a custom metric to increment by specifying its reference key; the default metric is the number of views.

Format:

<IM:update.content.metric

`recordid*="string"`	Record id of the content to be updated.
`metric="string"`	Reference key of the custom metric to be used.

></IM:update.content.metric >

user.admin.link

Description:

Returns the credentials for the logged in user encrypted. These credentials are used for login in the admin application. If the contentid and success and error attributes are passed, the user will be brought directly to that content record for editing.

Format:

<IM:user.admin.link

`contentid="string"`	If a content ID is provided with the login credentials, the user is directed directly to the associated content record for editing.
`view="string"`	This is a view reference key. If this property is set and the action is add, then the content will be assigned to this view if the logged in user has enough permissions
`success="string"`	fully qualified URL to return the user to if the update was successful
`error="string"`	fully qualified URL to return the user to if an error occurs

/>

Usage Example:

```
<a href="http://www.applicationurl.com?<IM:user.admin.link/>">Go To Admin</a>
```

user.input.remove.file

Description:

Creates an HTML input element for removing a file from the server for the specified name stored in the attribute.

Format:

```
<IM:user.input.remove.file
```

`attribute*="string"`	Path to custom user attribute.
`value="string"`	Initial value of the input field. if nothing is entered for the value, the value will default to the information in the database.
`css="string"`	CSS Class to be used for the input field

```
/>
```

Chapter 3 Creating and Deploying Custom Java Server Pages (JSPs)

You configure the Information Manager to display content on your site by creating and deploying custom Java Server pages (JSPs) using the Information Manager TagLibrary.

You deploy the JSPs by placing them in the directory:

<Information Manager_HOME>/server/webapps/

where:

`<Information Manager_HOME>` specifies the Information Manager installation directory.

You can use any text editor to create JSPs. This section provides examples of the following JSPs:

- The page template
- The template definition
- The listing page
- The detail page

NOTE: If you use Dreamweaver (Macromedia/Adobe), you can load the Information Manager Tag Library Descriptor located in the `/WEB-INF/tlds/Information Manager.tld` file where your web application is installed.

The Page Template

Page templates define the overall appearance of the page. Page templates can contain style sheets, static images, constant navigation, copyright information, and any other data that is consistent over many pages.

This template defines:

- The basics of an HTML page
- One named region, `contents`

Most templates would require many named regions, such as title, subnavigation, and footer.

The two unique and important lines of code in the following example are the first, which makes the Information Manager Tag Library available to this page:

```
<%@ taglib uri="/IMtaglib" prefix="IM" %>
```

and the seventh, which defines the named region with the `template.get` tag:

```
<IM:template.get name="contents" />
<%@ taglib uri="/IMtaglib" prefix="IM" %>
<html>
<head>
   <title>Demo</title>
</head>
<body>
   ....
   <IM:template.get name="contents" />
   ....
</body>
</html>
```

The Template Definition

Template definitions contain the definitions for the page elements that make up the general site layout.

```
<%@ taglib uri="/CAStaglib" prefix="CAS" %>
<% String id = request.getParameter("id"); %>
<IM:sitemap pagename="news"/>
<IM:template.definition template="t_template.jsp">
   <% if (id != null) { %>
      <IM:template.put name="contents" content="detail.jsp"/>
   <% } else { %>
      <IM:template.put name="contents" content="list.jsp"/>
   <% } %>
</IM:template.definition>
```

The `sitemap` tag;

```
<IM:sitemap pagename="news"/>
```

defines the name of the page, which is how it will be accessed via navigation; so that the URL of this page would end `index?page=news`.

The template.definition statement;

```
<IM:template.definition template="t_template.jsp">
```

selects the template, in this case a file in the root directory named `t_template.jsp`.

The following adds the content into the named region (content, as described in *The Page Template* on page 280):

```
<% if (id != null) { %>
   <IM:template.put name="contents" content="detail.jsp"/>
<% } else { %>
   <IM:template.put name="contents" content="list.jsp"/>
<% } %>
```

The template definition specifies that the news content will contain a detail and a list. A value that is made present in the querystring for accessing the detail determines which jsp will be used.

If the value is present;

```
<% if (id != null) { %>
```

then the rendered detail code is displayed in the named region contents;

```
<IM:template.put name="contents" content="detail.jsp"/>
```

If the value is not present, the rendered list code is used:

```
<IM:template.put name="contents" content="list.jsp"/>
```

The List Template Definition

The template definition lists basic information about all content records in reverse chronological order.

```
<%@ taglib uri="/CAStaglib" prefix="CAS" %>
<% String strPage = request.getParameter("page"); %>
....
<IM:get.channel.data channel="NEWS" dataset="connNews" />
<% if(connNews.count!=0){ %>
<IM:iterate.channel.records dataset="connNews">
   <IM:get.channel.record id="crNews">
      <p><strong><a href="index?page=<%= strPage %>&id=<IM:get.channel.recordid /
      "><IM:get.channel.attribute attribute="NEWS/TE" /></a></strong> -
<IM:get.channel.attribute attribute="NEWS/SUMMARY" /></p>
   </IM:get.channel.record>
</IM:iterate.channel.records>
<% } else { %>
   There is no news to report.
<% } %>
....
```

There are three steps to accessing the data:

- The `get.channel.data` tag;

  ```
  <IM:get.channel.data channel="NEWS" dataset="connNews" />
  ```

  specifies:

  - the channel to use
  - a name for the returned records

- The `iterate.channel.records` tag;

  ```
  <IM:iterate.channel.records dataset="connNews">
  ```

  loops through all of the records returned

- The `get.channel.record` tag;

  ```
  <IM:get.channel.record id="crNews">
  ```

  provides access to the current record in the iterator.

At this point, an individual content record is exposed and it is possible to extract the relevant content.

The news listing requires a link to the detail, the article title, and its abstract. The `get.channel.recordid` tag returns the GUID (globally unique identifier) of the current record, which is needed to select the detail;

```
<p><strong><a href="index?page=<%= strPage %>&id=<IM:get.channel.recordid />">
```

The `get.channel.attribute` tag returns any attributes in the schema of the channel being used, such as the title, `NEWS/TE`, and the abstract, `NEWS/SUMMARY`.

```
<IM:get.channel.attribute attribute="NEWS/TE" /></a></strong> - <IM:get.channel.attribute
attribute="NEWS/SUMMARY" /></p>
```

The example page also contains statements to check that records were returned and display an appropriate message if no records were returned.

The Detail Template Definition

The detail template definition also uses the `get.channel.data` tag; however, the `get.channel.data` tag and channel iterator are not required, since the GUID of the specific record is known.

Since this channel has an embedded Pictures node, a channel iterator is used to cycle through all of the pictures in the node.

Having a reference to the record allows access to any attribute within the schema of the channel being used.

After storing the JSP pages in the application server deployment directory, you can access the demo pages at:

`http://localhost:8080/demo/index?page=news`

```
<%@ taglib uri="/CAStaglib" prefix="CAS" %>
<% String id = request.getParameter("id");%>
<IM:get.channel.record recordid="<%=id%>" id="cr">
<table border=1 width=100%>
   <tr bgcolor="#999999">
      <td>Attribute</td>
      <td>Value</td>
   </tr>
   <tr>
      <td>NEWS/TE</td>
      <td><IM:get.channel.attribute attribute="NEWS/TE"/></td>
   </tr>
   <tr>
      <td>NEWS/SUMMARY</td>
      <td><IM:get.channel.attribute attribute="NEWS/SUMMARY"/></td>
   </tr>
   <tr>
       <td>NEWS/BODY</td>
      <td><IM:get.channel.attribute attribute="NEWS/BODY"/></td>
   </tr>
```

```
    <tr>
      <td colspan="2">NEWS/PICS</td>
      <IM:iterate.channel.records node="NEWS/PICS">
      <IM:get.channel.record id="crPics">
    <tr>
      <td>NEWS/PICS/IMAGE</td>
      <td>
        <img src="<%=crPics.resourcepath%><IM:get.channel.attribute attribute="IMAGE"/>"/>
      </td>
    </tr>
    <tr>
      <td>NEWS/PICS/CAPTION</td>
      <td>
        <IM:get.channel.attribute attribute="CAPTION"/>
      </td>
    </tr>
</IM:get.channel.record>
</IM:iterate.channel.records>
</table>
</IM:get.channel.record>
```