

Endeca® Discovery Framework

Power User's Guide



Contents

Preface.....	9
About this guide.....	9
Who should use this guide.....	9
Conventions used in this guide.....	9
Contacting Endeca Customer Support.....	10
 Chapter 1: Introduction to Discovery Framework.....	 11
About the Discovery Framework.....	11
About Discovery Framework power users.....	11
Discovery Framework user interface.....	11
Prerequisites to Discovery Framework application development.....	13
Obtaining more information about the Endeca MDEX Engine.....	13
 Chapter 2: Liferay Portal Basics.....	 15
About the Liferay Portal.....	15
Liferay Portal features used in the Discovery Framework.....	15
Dock menu functions.....	15
Control Panel functions.....	16
About plugins.....	18
About Liferay components.....	18
Accessing Liferay Portal documentation.....	19
 Chapter 3: Configuring Data Sources.....	 21
About data sources.....	21
Configuring data sources.....	21
Sample MDEX Engine data sources.....	21
Data source syntax.....	22
Specifying a default data source.....	23
Adding data sources to the Discovery Framework.....	23
Changing an Endeca component's data source.....	24
Configuring aggregated records in a data source.....	24
Connecting to a secured MDEX Engine.....	25
Data source role-based security.....	26
Data source relationships.....	26
Provided QueryFunction classes.....	27
Creating a QueryFunction class.....	30
Implementing the QueryFunction class.....	30
Using your custom QueryFunction.....	31
Adding the new .jar file to your Eclipse build path.....	31
Obtaining data source results.....	31
 Chapter 4: Building Discovery Framework Applications.....	 33
Applying a theme to a page.....	33
Adding a page.....	34
Renaming a page.....	35
Deleting a page.....	35
Applying a layout template to the page.....	36
Adding Endeca standard components.....	37
Renaming components.....	39
Editing components.....	39
Changing the company logo.....	40
 Chapter 5: Discovery Framework Logging.....	 41
Modifying logging in your Discovery Framework components.....	41
Adjusting the logging verbosity level in the Control Panel.....	41

Modifying portal-log4j-ext.xml.....	43
Setting up logging.....	43
log4j.properties.....	44
Learning about log4j.....	44
Chapter 6: Administrative Components.....	45
Accessing the administrative components.....	45
Attribute Settings.....	45
Data Sources.....	46
Configuring Data Sources.....	46
Data Source Bindings.....	48
Configuring Data Source Bindings.....	48
Framework Settings.....	49
Configuring Framework Settings.....	50
Performance Metrics.....	52
Using Performance Metrics.....	53
Sample Endeca Portlet.....	54
Chapter 7: Managing Attributes.....	57
About Endeca attributes.....	57
Accessing the Attribute Settings component.....	58
Attribute Settings dialog.....	58
Creating new attribute sets.....	59
Removing attributes from an attribute set.....	60
Adding attributes to attribute sets.....	61
Deleting attribute sets.....	61
About attribute display names.....	62
Changing the global display name of an attribute.....	62
Chapter 8: Layout Components.....	65
Tabbed Component Container.....	65
Using the Tabbed Component Container.....	65
Configuring the Tabbed Component Container.....	65
Chapter 9: Using Endeca Analytics.....	69
About Endeca Analytics.....	69
Analytics requirements for power users.....	69
Components that support Analytics.....	69
Chapter 10: Personalization Components.....	71
Bookmarks.....	71
Using Bookmarks.....	71
Configuring the Bookmarks component.....	73
Setting up the email server for Bookmarks support.....	74
Bookmarks saved data for component types.....	75
Chapter 11: Results Components.....	77
Results Table.....	77
Using Results Table.....	78
Configuring the Results Table component.....	79
Grid Preferences Editor.....	79
Configuring the Results Table for Analytics results.....	83
Record Details.....	84
Configuring the Record Details component.....	85
Configuring the Action menu.....	86
Configuring the Attribute List.....	86
Configuring the header field attribute.....	89
Results Table (Beta enhancements).....	90
Initial configuration of the component.....	91
Advanced configuration.....	92

Chapter 12: Data Visualization Components.....	97
Advanced Visualization.....	97
Using Advanced Visualization.....	97
Configuring the Advanced Visualization component.....	99
Chart.....	99
Using Chart.....	100
Configuring the Chart component.....	102
Chart styles.....	104
Cross Tab.....	109
Using Cross Tab.....	110
Configuring the Cross Tab component.....	110
Adjusting the Cross Tab table.....	111
Compare.....	114
Using Compare.....	114
Configuring the Compare component.....	115
Metrics Bar.....	119
Using Metrics Bar.....	120
Configuring the Metrics Bar component.....	120
Tag Cloud.....	121
Using Tag Cloud.....	121
Configuring the Tag Cloud component.....	123
Chapter 13: Filtering Components.....	125
Guided Navigation.....	125
Using Guided Navigation.....	126
Setting the Guided Navigation configuration options.....	128
Configuring the Attribute list.....	130
Breadcrumbs.....	131
Using Breadcrumbs.....	131
Configuring the Breadcrumbs component.....	132
Search Box.....	134
Using Search Box.....	134
Configuring Search Configurations.....	135
Configuring Search Box type-ahead suggestions.....	139
Range Filter.....	141
Using Range Filter.....	141
Configuring range filters.....	142
Find Similar.....	144
Using Find Similar.....	145
Configuring the Find Similar component.....	147
Chapter 14: Incorporating Liferay Components.....	151
Liferay component support.....	151
The Component Container.....	151
The Languages component.....	152
About Liferay Web Content Management components.....	152
Links component.....	152
Web Content component.....	153
Web Content Display component.....	153
Web Content List component.....	153
Web Proxy component.....	154
Chapter 15: Implementing Page Transitions.....	155
About page transitions.....	155
Page transition syntax.....	155
Creating page transitions using component IDs.....	156
Chapter 16: Using Deep Linking.....	159
About deep linking.....	159
Deep linking URL format.....	159

Deep linking syntax.....	160
Using NavByValue filters.....	161
Deep linking examples.....	161
Resetting data source query state.....	161
How security is handled with deep linking.....	161



Copyright and disclaimer

Product specifications are subject to change without notice and do not represent a commitment on the part of Endeca Technologies, Inc. The software described in this document is furnished under a license agreement. The software may not be reverse engineered, decompiled, or otherwise manipulated for purposes of obtaining the source code. The software may be used or copied only in accordance with the terms of the license agreement. It is against the law to copy the software on any medium except as specifically allowed in the license agreement.

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose without the express written permission of Endeca Technologies, Inc.

Copyright © 2003-2011 Endeca Technologies, Inc. All rights reserved. Printed in USA.

Portions of this document and the software are subject to third-party rights, including:

Corda PopChart® and Corda Builder™ Copyright © 1996-2005 Corda Technologies, Inc.

Outside In® Search Export Copyright © 2008 Oracle. All rights reserved.

Rosette® Globalization Platform Copyright © 2003-2005 Basis Technology Corp. All rights reserved.

Teragram Language Identification Software Copyright © 1997-2005 Teragram Corporation. All rights reserved.

Trademarks

Endeca, the Endeca logo, Guided Navigation, MDEX Engine, Find/Analyze/Understand, Guided Summarization, Every Day Discovery, Find Analyze and Understand Information in Ways Never Before Possible, Endeca Latitude, Endeca InFront, Profind, Endeca Navigation Engine, Don't Stop at Search, and other Endeca product names referenced herein are registered trademarks or trademarks of Endeca Technologies, Inc. in the United States and other jurisdictions. All other product names, company names, marks, logos, and symbols are trademarks of their respective owners.

The software may be covered by one or more of the following patents: US Patent 7.035.864, US Patent 7.062.483, US Patent 7.325.201, US Patent 7.428.528, US Patent 7.567.957, US Patent 7.617.184, US Patent 7.856.454, Australian Standard Patent 2001268095, Republic of Korea Patent 0797232, Chinese Patent for Invention CN10461159C, Hong Kong Patent HK1072114, European Patent EP1459206, European Patent EP1502205B1, and other patents pending.

Endeca Discovery Framework Power User's Guide • March 2011

Version 1.5

Preface

Endeca® Latitude applications guide people to better decisions by combining the ease of search with the analytic power of business intelligence. Users get self-service access to the data they need without needing to specify in advance the queries or views they need. At the same time, the user experience is data driven, continuously revealing the salient relationships in the underlying data for them to explore.

The heart of Endeca's technology is the MDEX Engine.™ The MDEX Engine is a hybrid between an analytical database and a search engine that makes possible a new kind of Agile BI. It provides guided exploration, search, and analysis on any kind of information: structured or unstructured, inside the firm or from external sources.

Endeca Latitude includes data integration and content enrichment tools to load both structured and unstructured data. It also includes the Discovery Framework, a set of tools to configure user experience features including search, analytics, and visualizations. This enables IT to partner with the business to gather requirements and rapidly iterate a solution.

About this guide

This guide contains information for the power user tasked with building end-user applications in the Endeca Discovery Framework.

The Endeca Discovery Framework enables rapid configuration of search applications—often in just minutes—that offer the highly interactive Guided Navigation® user experience across a full range of structured and unstructured enterprise data. It includes a Component SDK, which is a packaged development environment for portlets, themes, layout templates, and other portal element. Endeca has modified Liferay's version of its Plugins SDK to include the Endeca enhancements, such as the `EndecaPortlet` core class.

Who should use this guide

This guide is intended for power users of the Endeca Discovery Framework on Windows or Linux.

Conventions used in this guide

This guide uses the following typographical conventions:

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

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

Contacting Endeca Customer Support

The Endeca Support Center provides registered users with important information regarding Endeca software, implementation questions, product and solution help, training and professional services consultation as well as overall news and updates from Endeca.

You can contact Endeca Standard Customer Support through the Support section of the Endeca Developer Network (EDeN) at <http://eden.endeca.com>.



Chapter 1

Introduction to Discovery Framework

This section provides an overview for getting started building applications with the Discovery Framework.

About the Discovery Framework

The Endeca Discovery Framework is a portal application, built on the Liferay Portal, that you populate with portlet components.

The Discovery Framework enables rapid configuration of search applications that offer the highly interactive Guided Navigation® user experience across a full range of structured and unstructured enterprise data. Lightweight and easy to deploy, the Discovery Framework is ideal for the development of enterprise-quality search applications. Due to the Discovery Framework's component-based nature, applications built with it are simple to control, adapt, and extend. Granular layout and configuration control let users manage and personalize their own experiences.

About Discovery Framework power users

This guide is intended for Discovery Framework power users.

Power users configure Discovery Framework content for *end users*, who are Discovery Framework content consumers.

Several job roles might fit into the power user category. One example would be a business analyst who configures the Discovery Framework for end users and determines what components and data they can access. This includes executives seeking a dashboard view as well as others who need to drill through interactive visualizations and reports.

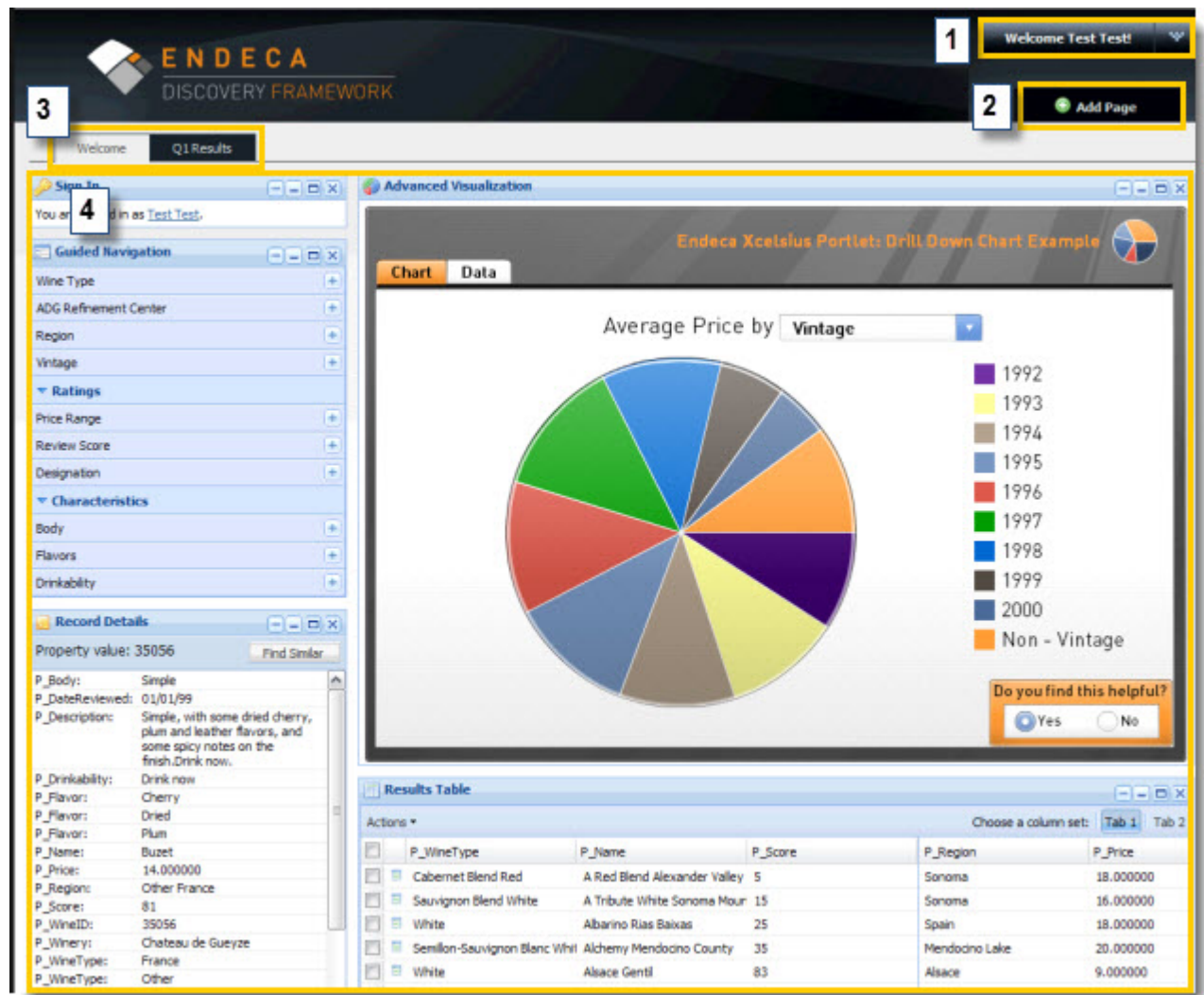
Typically, the Discovery Framework is configured so that end users cannot access the edit controls found on the Preferences page of each component.

Discovery Framework user interface

This topic provides a brief overview of a Discovery Framework user interface.

The look-and-feel of Discovery Framework applications varies according to the customization performed by the application's developers. A generic example of an application is used here to show the user interface.

When you first access the Discovery Framework application, you will see an interface similar to this one:



The numbered areas are the following controls and work areas:

1. The Dock gives you access to such function areas as the Control Panel (for administrative functions), Add Component (for adding Endeca and Liferay components), Layout Template, and Manage Pages (for the administration of the pages).
2. Add Page lets you add pages to the application's interface.
3. Tabbed pages are added by the power user or application developer and can be considered as home pages for the various components of the application. Placing the components on several tabbed pages provides a more useful navigation experience for the end user than having all the components in a single page.
4. The portal workspace is where users access the components. For example, the Guided Navigation component (just under the number 4) provides end users with contextual navigation across the application's data set. The records returned by the end user's navigation are shown in the Results Table component. This portal workspace example also shows a pie chart created by the application's Chart component.

More information on these features is provided later in this guide.

Prerequisites to Discovery Framework application development

Before you begin developing your application in the Discovery Framework, make sure the following items are in place.

The following tasks are typically the responsibility of the power user:

- Understanding your use cases and business case, so you can assemble Discovery Framework components to match these cases.
- Building application pages.
- Creating charts.
- Configuring controls for the various components.

The following tasks are typically the responsibility of the system administrator:

- Deploying and maintaining the server that will host the Discovery Framework application.
- Connecting the Discovery Framework to enterprise systems.
- Setting up Discovery Framework logging.
- Establishing and maintaining the Discovery Framework connection to its underlying database.
- Managing ongoing clustering and load balancing.
- Deploying extension points provided by the developer.

The following optional activities are typically the responsibility of a developer:

- Modifying existing components, or creating new ones, to better meet your organization's unique business requirements.
- Developing Discovery Framework extension points, for example to enable security.

The task of creating new themes is typically the responsibility of a designer.

Obtaining more information about the Endeca MDEX Engine

This guide assumes that you already have experience working with the Endeca MDEX Engine.

This includes having a solid understanding of Endeca Analytics as well as of record roll-ups, aggregation, and other forms of data manipulation.

If you need more information about the MDEX Engine, consult the documentation for the version you are using. The documentation is available in the Knowledge Base of the Endeca Developer's Network (EDeN).



Chapter 2

Liferay Portal Basics

Because the Discovery Framework is built upon the Liferay Portal, you need to know Liferay basics before you begin developing your Discovery Framework application.

About the Liferay Portal

The Discovery Framework is build upon the Liferay Portal Enterprise Edition.

Liferay Portal is an open-source JSR-286 portal technology. The Discovery Framework extends basic Liferay functionality to provide enhanced user management, security, and cross-component interaction, as well as performance-optimized communication with the Endeca MDEX Engine.

This version of the Discovery Framework is built upon Liferay Portal 5.2 Enterprise Edition Service Pack 5.

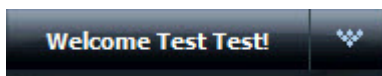
Liferay Portal features used in the Discovery Framework

This section describes Liferay Portal features that have been incorporated into the Discovery Framework.

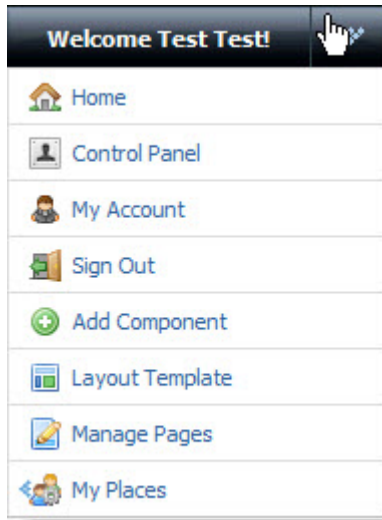
Dock menu functions

The Liferay Portal Dock houses the main menu for the portal application.

The Dock is located in the upper right corner of the application window and provides navigation links to various functions. The Dock is typically labeled "Welcome, <user name>!"



Hovering over the Dock displays the application menu:



If you are not signed in, only the **Home** and **Sign In** links are available.

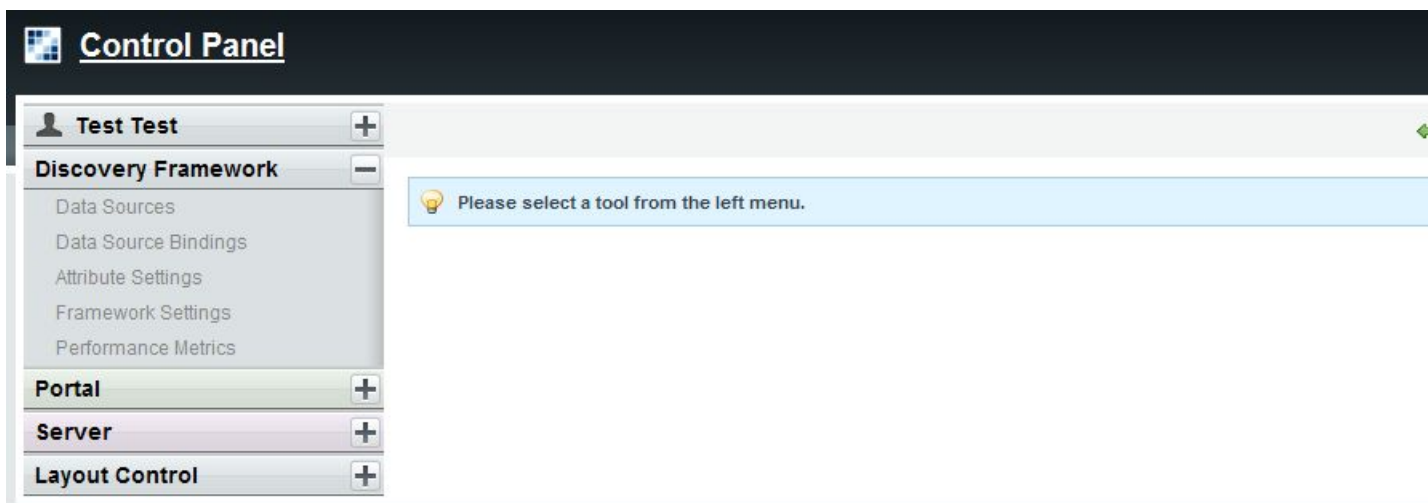
The application menu links provide the following functions:

Dock Link	Purpose
Home	Takes you to the home page.
Control Panel	Accesses the Liferay and Endeca administrative functions.
My Account	Lets you edit your user information, including changing your login password.
Sign Out	Logs you out.
Add Component	Lets you add Endeca and Liferay components to the application.
Layout Template	Displays the Layout Template window, which lets you apply a different layout to a page.
Manage Pages	Accesses the Manage Pages menu, where you can add and delete pages, change their order, and perform other page administration functions.
My Places	Displays the community and organization pages.

Control Panel functions

The Liferay Portal Control Panel contains a large number of tools for system administrators as well as for power users.

You access the **Control Panel** from the **Dock**. The left menu has a list of tool sections, with each section containing a number of user and administrator tool components. The following sample shows all the sections collapsed, except for the **Discovery Framework** section.



The default function headings are briefly described below.

Power users generally use the Control Panel to access Discovery Framework administrative components, such as the **Data Sources**, **Attribute Settings**, and **Framework Settings** components.

User Name section

The User Name section (Test Test in the sample) is the logged-in user's personal space, which can be managed with these functions:

- **My Account** lets you edit your user information, including changing your login password.
- **My Pages** lets you edit your public and private pages.

Discovery Framework section

Power users generally use the Control Panel to access the administrative components in the **Discovery Framework** section:

- The **Data Sources** component views and reloads data sources.
- The **Data Source Bindings** component associates data sources with components.
- The **Attribute Settings** component creates and modifies attribute sets.
- The **Framework Settings** component provides access to state, security, and other settings.
- The **Performance Metrics** component displays information about component and MDEX Engine query performance.

For details on these components, see the "Administrative Components" chapter.

Portal section

The Portal section is intended for portal administrators to manage the user community. The administrative components in this section are:

- The **Users** component adds and modifies user accounts.
- The **Organizations** component creates and edits Organizations, which are hierarchical collections of users.
- The **Communities** component creates and edits Communities, which are collections of users who have a common interest.
- The **User Groups** component creates and edits User Groups, which are simple, arbitrary collections of users.

- The **Roles** component manages the three types of roles, which are used to define permissions across their scope. The types of roles are a Portal role (which grants access across the portal), Community role (which grants access only within a single Community), and Organization role (which grants access only within a single Organization). The portal administrator can use these roles to control permissions in the application, such as who gets to see which pages.
- The **Password Policies** component sets the password rules, such as the password strength and the password expiration.
- The **Settings** component configures most of the global portal settings, including authentication, e-mail configuration, and the default landing page.
- The **Monitoring** component monitors all the live sessions in the portal (if monitoring is enabled).
- The **Plugins Configuration** component configures which portal roles have access to the plugins.

Server section

The Server section provides the following administration functions:

- The **Server Administration** component allows you to perform administrative tasks for the overall portal server, including resource management, setting log levels, data migration between databases, e-mail server configuration, and shutting down the Liferay Portal server.
- The **Portal Instances** component lets you add and configure multiple portal instances on a single server.
- The **Plugins Installation** component lets you view the installed plugins and also add new ones.

Layout Control section

- The **Web Content** component lets you add Web content, structures, templates, and RSS feeds.
- The **Links** component lets you add folder links.

About plugins

Plugins are `.war` files that allow you to add functionality to the Discovery Framework. In the Liferay Portal, portlets, themes, and layout templates are all considered plugins.

Because plugins are separate from the core application, they can be developed and deployed without disrupting application performance.

Creating plugins with the Component SDK

The Discovery Framework provides component developers with a Component SDK that they can use to modify or create portlet components. These tasks are not usually done by power users. For more information about the Component SDK, see the *Discovery Framework Extension Guide*.

About Liferay components

In addition to Endeca standard components, the Discovery Framework included some Liferay components by default.

Among these components are the **Tabbed Component Container** and the **Content Management** set of components.

Accessing Liferay Portal documentation

Because the Discovery Framework is built upon the Liferay Portal, you can access Liferay's documentation for more information about how to perform various tasks.

Specifically, the Liferay Portal Administrator's Guide provides extensive information about installing, configuring, and maintaining a portal. To access a free PDF download of this guide, go to <http://www.liferay.com> and navigate to the **Docs & Resources** section.

In addition to its formal administrator documentation, Liferay offers developer assistance in the form of blogs, wikis, and forums. To access these resources, go to <http://www.liferay.com> and navigate to the **Community** section.



Chapter 3

Configuring Data Sources

Data sources and query states are means of representing connections between the Discovery Framework and the MDEX Engines, as well as between queries and the MDEX Engines. This section discusses how to create and configure data sources, and outlines the default interaction model between related data sources.

About data sources

Every instance of a component that needs to query the MDEX Engine is backed by a particular data source. That data source, which represents a pointer to a specific MDEX Engine, is used to maintain application state for each user's session.

The different components in your Discovery Framework application can connect to different data sources.

Configuring data sources

Data sources and query states are how the Discovery Framework represents connections to MDEX Engines and queries to those MDEX Engines. This section discusses how to create and configure data sources, and outlines the default interaction model between related data sources.

Sample MDEX Engine data sources

The Discovery Framework installation package includes sample data sources. You can use the samples as a model for your own data sources.

The following sample data sources are located in the `endeca-portal\data\endeca-data-sources` directory:

- `base-filters-data-source.json.sample`
- `child-data-source.json.sample`
- `secured-data-source.json.sample`
- `simple-data-source.json.sample`
- `ssl-enabled-data-source.json.sample`

This directory also contains a `default.json` file, containing host and port information only. The default data source is discussed later in this section.

Data source syntax

Data source files are written in the JSON language.

Every data source must include `server` and `port` keys. All other settings are optional. If `name` and `id` are not included, they are both given the name of the file defining the data source by default. For example, a file named `parts.json` results in a data source with the `name` and `id` "parts." The `id` cannot contain spaces. Therefore, if you do not specify an explicit `id`, make sure that your file name does not contain any spaces.

The `name` setting is normally the only user-visible identification of a data source. The `id` setting is only used internally, and the `description` setting is only used for logging and debugging.



Note: SSL configuration file paths are relative to the directory containing the JSON data source files. This is typically, but not always, the `endeca-portal\data\endeca-data-sources` directory.

Data source example

The following example shows a JSON data source file based on wine merchant data.

```
{
  "server": "server01.lab.acme.com",
  "port": "15000",
  "apiVersion": "ENE_QUERY",
  "sslConfig": {
    "caFile": "truststore.ks",
    "caPassword": "endeca",
    "certFile": "keystore.ks",
    "certPassword": "endeca"
  },
  "id": "ds-id",
  "name": "Descriptive DataSource name",
  "description": "Detailed information about this DataSource",
  "baseFunctions": [
    {
      "class": "com.endeca.portal.data.functions.RecordFilter",
      "recordFilter": "Regions:Midwest"
    },
    {
      "class": "com.endeca.portal.data.functions.RangeFilter",
      "property": "P_Price",
      "rangeOperator": "GTEQ",
      "value1": "25"
    },
    {
      "class": "com.endeca.portal.data.functions.RefinementFilter",
      "dimValId": "123",
      "dimensionId": "121"
    }
  ],
  "securityEnabled": "true",
  "inheritSecurity": "true",
  "securityFilters": {
    "frenchFilter": {
```

```

        "class": "com.endeca.portal.data.functions.RecordFilter",
        "recordFilter": "OR(Region:Bordeaux,Region:Burgundy)"
    },
    "rolePermissions": {
        "French Wine": ["frenchFilter"]
    }
}

```

Specifying a default data source

Among the data sources in your Discovery Framework application, you must always include a default data source. This data source is automatically assigned to all data-source-backed components when they are initially added to a page.

When you initially start the Discovery Framework, the `id` of this data source is **default**. While this setting remains unchanged, you must include a data source with the `id` **default**. Upon installation, a `default.json` data source file (with an implicit `id` of **default**) is included in the `endeca-portal\data\endeca-data-sources` directory.

Alternatively, you can change the default data source name to that of another data source file. To do so:

1. In the **Liferay Control Panel**, go to the **Discovery Framework Settings** page.
2. In the **Discovery Framework Settings** page, change the `df.defaultDataSource` setting to a different value.
3. Restart the **Discovery Framework**.
4. Make sure you include a data source with that new value as its `id`.

Adding data sources to the Discovery Framework

It is possible to add new data sources to the Discovery Framework. Your Endeca components can then access the data sources you have added.

To add a data source to the Discovery Framework:

1. Create a new JSON file in `endeca-portal\data\endeca-data-sources`.
For definition examples, see the sample data sources located in the same directory.
2. After creating the new file on disk, do one of the following:
 - Restart the Discovery Framework.
 - In the Control Panel, select **Data Sources**, and then click **Update data sources**.




Note: If your data source does not appear after completing step 2, it probably means that your data source definition file contains invalid JSON syntax. You can confirm this by looking for a message about invalid syntax in the Discovery Framework log `df.log`. Check the log, edit your syntax, and try the steps above again.

Changing an Endeca component's data source

If more than one data source has been configured for the application, the data source for an individual portlet instance can be changed.

To change the data source for an Endeca component that can be bound to a data source:

1. In the header of the component whose data source you want to change, select the  icon, and then select **Preferences**.
2. Select the new data source in the drop down menu, and click **Update data source**.
You should see that the component has been successfully bound to a new data source.
3. Click **Return to Full Page**.



Note: This procedure only changes the data source for that single instance of the component.

Configuring aggregated records in a data source

Aggregated records allow you to group records by dimension or property values.

By configuring aggregated records, you enable the MDEX Engine to handle a group of multiple records as though it were a single record, based on the value of the rollup key. A rollup key can be any property or dimension that has its rollup attribute enabled.

You can use parent-child data source relationships to create multiple data sources with different aggregations. The parent data source can use base records, while multiple child data sources each aggregate by a different rollup key.

For details on how individual components handles record aggregation, see the *Discovery Framework Component Catalog*.



Note: Aggregated records in the Discovery Framework are subject to the same constraints as they are in the MDEX Engine: you cannot control the order of base records within an aggregate, and you cannot control which record is used as the representative record (beyond controlling overall sort order).

Aggregated record configuration

To configure a data source for aggregated records, you include the `RecordAggregator QueryFunction` class in the data source's JSON configuration file. The `RecordAggregator` contains two configuration properties: `rollupKey` and `aggCount`.

The `rollupKey` property is used to specify the dimension or property by which records in a navigation object's record list should be aggregated.

The `aggCount` property is used to limit the number of base records returned with the aggregated record. There are three possible values for `aggCount`:

- `ZERO_EREC_PER_AGGR`: none of the records that compose the aggregated record will be returned. (In most cases, `ZERO_EREC_PER_AGGR` is not useful. Because the MDEX Engine returns zero records, your components have almost no data to work with.)
- `ONE_EREC_PER_AGGR`: one of the records that composes the aggregated record will be returned.
- `ALL_EREC_PER_AGGR`: all of the records that compose the aggregated record will be returned. (Due to performance implications, use of `ALL_EREC_PER_AGGR` is discouraged.)

If no `aggCount` value is set, the default is `ONE_EREC_PER_AGGR`.

Aggregated record example

In the following example data source configuration file, the **P_Winery** property is the rollup key, and one of the records that composes the aggregated record will be returned.

```
{
  "server": "server01.lab.acme.com",
  "port": "15000",
  "baseFunctions": [
    {
      "class": "com.endeca.portal.data.functions.RecordAggregator",
      "rollupKey": "P_Winery",
      "aggCount": "ONE_EREC_PER_AGGR"
    }
  ]
}
```

Connecting to a secured MDEX Engine

This topic provides a high-level description of how you can set up the Discovery Framework to connect to a secured (HTTPS) MDEX Engine.

Several of the steps below refer to Chapter 3 (entitled "Using Endeca SSL Certificate Utilities") of the *Endeca Platform Services Security Guide*. The *Endeca Platform Services Security Guide* is available as part of the Platform Services documentation set on EDeN. Before attempting these steps, make sure you have a copy of that guide at hand.



Note: The steps below assume you are using the Discovery Framework Tomcat bundle.

To connect to a secured MDEX Engine:

1. Stop the Discovery Framework.
2. Generate the SSL certificate files for the Dgraph using the `enecerts` utility. This utility is available in both the Platform Services and MDEX Engine installation. For instructions, refer to the section "Generating SSL certificates" in the *Endeca Platform Services Security Guide*.
3. Generate the Java KeyStore (JKS) files using the `endeca-key-importer.jar` utility from the Endeca Platform Services installation. For instructions, refer to the topic "Converting PEM-format keys to JKS format" in chapter 3 of the *Endeca Platform Services Security Guide*.
4. Place the JKS keys into the directory containing the JSON data source files. This is typically, but not always, the `endeca-portal\data\endeca-data-sources` directory.
5. Specify the `caFile`, `certFile`, `caPassword`, and `certPassword` in the appropriate JSON data source file.

The following example is extracted from a larger data source file:

```
"sslConfig": {
  "caFile": "truststore.ks",
  "caPassword": "endeca",
  "certFile": "keystore.ks",
  "certPassword": "endeca"
}
```

6. Restart the Discovery Framework.

Data source role-based security

The default `MDEXSecurityManager` implementation supports the configuration of filters associated with Liferay roles that have been assigned to a user's login.

The supported filters are as follows:

- **securityEnabled:** enabling/disabling security filters on all queries issued to this data source.
- **securityFilters:** definition of all security filters to be used by this data source (any extension of `QueryFilter`).



Note: Record filters are the only supported type of `securityFilter`.

- **rolePermissions:** role mappings to the security filters that have been defined for the data source.
- **inheritSecurity:** enabling/disabling of security filter inheritance, based on data source relationships defined via the `parentDataSource` property.
- **parentDataSource:** when `inheritSecurity` is true, this property is used to find all ancestors of each data source being secured. For each data source, the list of security filters to be applied is a combination of all security filters configured for every ancestor data source, plus any found with the data source configuration itself.



Note: For more information on security, see the chapter "Security extensions to the Discovery Framework" in the *Discovery Framework Extension Guide*.

Data source relationships

This topic describes the relationship between parent and child data sources.



Important: The way data sources interact with each other can vary based on the Data Source State Manager your portal is using. The information below only applies to the default implementation of the State Manager provided with the released version of the Discovery Framework.

Discovery Framework data sources can be configured as parents and children. These relationships can be set in a data source's JSON file through the `parentDataSource:parent-ds-id` property.

In a parent-child behavior, child data sources act like immutable filter sets. If you are using a child data source, the `baseFunctions` setting in the data source file should be specified with some set of filters. When a component connected to a child data source attempts to get the current query state, the query state is the query state of the data source's parent with all of the static filters on the child data source (specified in configuration as `baseFunctions`) appended to it.

When a component connected to a child data source attempts to change that data source's query state (by setting a refinement from Guided Navigation, for example), the change takes place on the parent data source. Such an operation keeps applying to a data source's parent until it finds a data source without a parent. When any data source's query state is changed, the query states of all of its children are effectively changed as well, and any components connected to them will refresh.



Note: Role-based security filters can be inherited by descendant data sources when using the Security Manager, via the `parentDataSource` property.

Provided QueryFunction classes

This topic defines the QueryFunction classes included in the Discovery Framework.

Filters

Function Class	ENEQuery Equivalent	Configuration Properties	Notes
EQLFilter	setNavRecordStructureExpr()	eqlFilter: String	Applying an EQL filter always overwrites a previous EQLFilter
NegativeRefinementFilter	setNavRecordFilter()	dimValName: String dimValId: String dimensionName: String dimensionId: String ancestors: String[]	
RangeFilter	setNavRangeFilters()	property: String rangeOperator: (LT LTEQ GT GTEQ BTWN GCLT GCGT GCBTWN) value1: numeric value2: numeric (optional) value3: numeric (optional)	
RecordAggregator	setNavRollupKey()	rollupKey: String aggCount: ({ {ONE_EREC_PER_AGGR ALL_EREC_PER_AGGR ZERO_EREC_PER_AGGR } })	
RecordFilter	setNavRecordFilter()	recordFilter: String	
RecordSpecListFilter	setERecs()	recSpecs: String[]	Applying a Record SpecList Filter always overwrites a previous

Function Class	ENEQuery Equivalent	Configuration Properties	Notes
			Record Spec ListFilter
RefinementFilter	setNavDescriptors()	dimValId: long dimensionId: long multiSelect: (AND OR NONE) (optional) navigable: (true false) (optional)	
SearchFilter	setNavERecSearches()	searchInterface: String terms: String matchMode: (ALL PARTIAL ANY ALLANY ALLPARTIAL PARTIALMAX BOOLEAN)	

Configuration functions

Function Class	ENEQuery Equivalent	Configuration Properties	Notes
AnalyticsQueryConfig	setAnalyticsQuery()	analyticsQuery: String	
AttributeValueSearch- Config	setDimSearchTerms() setDimSearch- NavDescriptors() setDimSearch- NavRangeFilters() setDimSearch- NavRecordFilters() setDimSearch- NumDimValues() setDimSearch- NavRecordStructure Expr()	searchTerm: String maxValuesToReturn: int (optional) attribute: String (optional) searchWithIn: List<String> (optional - list of attributes) matchMode: (ALL PARTIAL ANY ALLANY ALLPARTIAL PARTIALMAX BOOLEAN) (optional)	searchWithIn is a list of attributes against which to search; attribute property is automatically included in this list.

Function Class	ENEQuery Equivalent	Configuration Properties	Notes
		relevanceRankingStrategy: String (optional)	
ExposeRefinement	setNavExposedRefinements()	dimValId: String dimensionId: String ownerId: String (optional) dimExposed: boolean (optional) exposeAll: boolean (optional) maxRefinements: int (optional; ignored for ENEQueries) groupKey: String (optional; ignored for ENEQueries) groupExposed: boolean (optional; ignored for ENEQueries)	At least one of dimValId or dimensionId is required. ownerId can be the ID of a NavConfig instance. dimExposed indicates whether the dimension is exposed. exposeAll indicates whether the dimension is fully exposed (that is, the "More..." link is selected).
NavConfig	setNavAllRefinements	exposeAllRefinements: boolean	
ResultsConfig	setNavNumERecs() setNavERecsOffset() setSelection	recordsPerPage: long offset: long (optional) columns: String[] (optional) numBulkRecords: int (optional)	
ResultsSummaryConfig	N/A		Sets the number of returned ERecs and AggrERecs to 0
SortConfig	setNavActiveSortKeys()	property: String ascending: boolean	

Creating a QueryFunction class

This topic describes the steps required to create a new `QueryFunction` class.

The steps below create a `QueryFilter` class, but the steps are analogous for creating a `QueryConfig` class.



Note: In order to create `QueryFunction` classes, you must install the Component SDK, which is a separate download.

To create a new `QueryFilter` class:

1. In a terminal, change your directory to `endeca-extensions` within the Component SDK's root directory (normally called `components`).
2. Run one of the following commands:
 - On Windows: `.\create-queryfilter.bat your-query-filter-name`
 - On Linux: `./create-queryfilter.sh your-query-filter-name`

This command creates a `<your-query-filter-name>-filter` directory under `endeca-extensions`. This directory is an Eclipse project that can be imported directly into Eclipse if that is your IDE.

The `endeca-extensions` directory also contains an empty sample implementation of either a `QueryFilter` or a `QueryConfig`, depending on which batch script you ran. This has no effect on `QueryState` in its original form. The skeleton implementation creates source files that do the following:

- Extends either `QueryFilter` or `QueryConfig`.
- Creates stubs for the abstract methods you need to implement: `applyToENEQuery`, `applyToDiscoveryServiceQuery`, and `toString`.
- Creates default implementations for `getSetters` and `getGetters`. These use static `setters` and `getters` member properties that use reflection to extract the appropriate methods from the class.
- Creates a no-argument, protected, empty constructor. (The protected access modifier is optional, but recommended.)
- Creates a private member variable for logging.

Implementing the QueryFunction class

This topic describes the steps needed to implement your new `QueryFunction` class.

To implement your new `QueryFunction`, you must:

- Add private filter or configuration properties.
- Create getters and setters for any filter properties you add.
- For any property that is not a `String`, create a setter property that takes a `String` and does conversion.
- Define a no-argument constructor (protected access modifier optional, but recommended).
- Implement the abstract methods `getSetters`, `getGetters`, `applyToENEQuery`, `applyToDiscoveryServiceQuery`, and `toString`. You can use the `getSetters` and `getGetters` methods from the sample `QueryFunction`.



Note: Because `.toString()` is used in `.equals()`, you should make sure that two `QueryFunction` objects that are the same return the same value. Specifically, `.toJSON().toString()` does not guarantee ordering of JSON properties, so two

`QueryFunction` objects with the same member values may not return the same value if `.toString()` was implemented using `.toJSON().toString`.

Using your custom QueryFunction

In order to use your new `QueryFunction`, you must deploy it to the Discovery Framework.

If you are using the default portal bundle, the `your-query-filter-name-filter|config` directory that you created contains an ant build file. The ant `deploy` task places a `.jar` file containing the custom `QueryFunction` into the `endeca-portal/tomcat-<version>/lib/ext` directory.

To deploy your custom `QueryFunction` (if you are not using the default portal bundle):

1. Put the new `QueryFunction.jar` into the container's global classpath.
2. Restart the Discovery Framework so that the portal picks up the new class file.

Once you have deployed your custom `QueryFunction`, you can use it in any component.

Adding the new .jar file to your Eclipse build path

If you are using Eclipse as your IDE, you need to add the new `.jar` file to your build path of your custom component.

To add the new `.jar` file to your Eclipse build path:

1. Right-click on the project, and select **Build Path > Configure Build Path**.
2. Click the **Libraries** tab.
3. Click **Add Variable**, select **DF_GLOBAL_LIB** (which you should have added while setting up the SDK), and then click **Extend**.
4. Open the `ext/` directory and select the `.jar` file containing your custom `QueryFunction`.
5. Click **OK**.

After adding the `.jar` file to the build path, you can import the class, and use your custom `QueryFunction` or `QueryConfig` to modify your `QueryState`.

Obtaining data source results

The `ENEQueryResults` class from the Presentation API is used to represent results of queries.

Components are always encouraged to add the relevant `QueryConfig` to specify what types of results they need. Calls to `DataSource.execute()`, without any arguments, will continue to work on ENE Presentation API data sources, but are deprecated.:

```
QueryState query = getDataSource(request).getQueryState();
query.addFunction(new NavConfig());
QueryResults results = getDataSource(request).execute(query);
```

You can then get the underlying API results and do whatever manipulation is required by your component.

```
ENEQueryResults eneResults = results.getENEQueryResults();
```

You can also make other local modifications to your query state before executing by adding filters or configurations to your query:

```
QueryState query = getDataSource(request).getQueryState();
query.addFunction(new ResultsConfig());
query.addFunction(new RecordFilter("Region:Midwest"));
QueryResults results = getDataSource(request).execute(query);
```

If you need to make modifications to your query that can't be represented on a `QueryState`, you can use `ENEQuery` instances directly:

```
DataSource ds = getDataSource(request);
ENEQuery eneQuery = ds.createENEQuery();
//modify query...
ENEQueryResults eneResults = ds.execute(eneQuery);
```

When you need to update a data source's state so that all associated components are updated, you cannot use `ENEQuery` instances—you must use `QueryState` instances.

```
DataSource ds = getDataSource(request);
QueryState query = ds.getQueryState();
query.addOperation(new RecordFilter("Region:Midwest"));
ds.setQueryState(query);
```




Chapter 4

Building Discovery Framework Applications

As a Discovery Framework power user, you build Web-based applications for end users who share specific job roles. This section provides a high-level overview of that process.

Applying a theme to a page

Themes control the look and feel of your Discovery Framework application.

Themes are hot deployable plugins that you can use to customize the appearance of your application. By using your own theme, your Discovery Framework application can adhere to the look-and-feel standards which may be used across all of the Web sites and Web applications in your company's infrastructure.

By default, Discovery Framework ships with the Endeca theme and the Liferay Classic theme. By going to the Liferay Web site, you can download themes from Liferay's Community Plugin Library, which are contributed by the Liferay community of users:

<http://www.liferay.com/downloads/liferay-portal/community-plugins>

Your application developer can also create customized themes for your application. For information on theme development, see the *Liferay Developer's Guide*, which is available from the Liferay Web site.

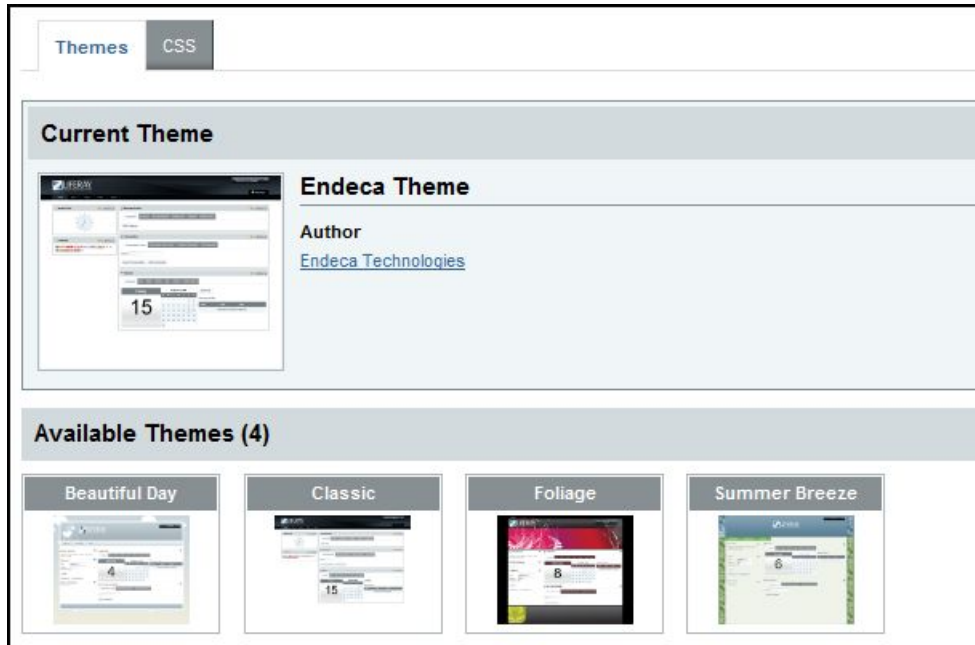
To apply an existing theme to a page:

1. From the Dock, choose **Manage Pages**.
2. In the Manage Pages dialog, click on a page that is listed in the left column.



3. Click the **Look and Feel** tab.

- The **Themes** tab shows the theme that is currently being used by the page and lists any available themes. To apply another theme, click on that theme.



If the new theme was successfully applied, you will see this message:



Note that each page can have a different theme.

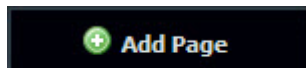
Adding a page

When you first launch the Discovery Framework, the interface consists of a single page. However, you can add additional pages.

Adding pages to your application maximizes application logic while minimizing visual clutter. Spreading components among several pages also helps improve the performance of the application.

To add a page to your Discovery Framework application:

- Click **Add Page**.



- An empty page label is created. Type a name for your new page.



- Click **Save**.

A new named tab is added to the application.

The name that you gave the page (at step 2) is actually its display name. When the page is created, it has a friendly URL name that you can see from the **Manage Page** dialog for that page, as in this example for the "Data Results" page:

The screenshot shows a 'Manage Page' dialog box. It has several fields: 'Name' with the value 'Data Results', 'HTML Title' (empty), 'Type' with a dropdown menu showing 'Portlet', 'Hidden' with an unchecked checkbox, and 'Friendly URL' with the value 'http://localhost:8080/web/guest /data-results'. The 'Friendly URL' field is highlighted with a yellow background.

The display name is shown in the page tab. But because its display name is not the same as its friendly URL name, you can easily change its display name, as explained in the following topic.

Once you have created additional pages, you can create page transitions to target the output of individual components to the page you specify. You can also apply a new theme to the page.

Renaming a page

After you create a page, you can change the display name that appears on its tab.

When you rename a page, you are changing the display name of the page. Because of this, you do not have to change any existing page transitions.

To rename a page:

1. Click the page tab, so that you go to the page.
2. Click on the page tab.
3. Type in the new name.



4. Click **Save**.

You can also rename a page from the **Manage Page** dialog for that page.

Deleting a page

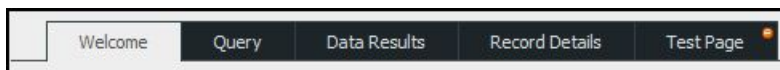
You can easily delete pages from the application.

When deleting a page, you cannot be accessing the page when you try to delete it. That is, you must delete a page from another page. Note that if the page to be deleted has components, the components will also be deleted along with the page.

To delete a page:

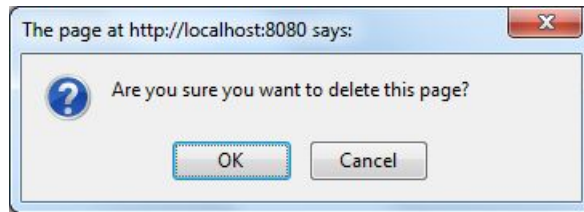
1. From any page (other than the one you want to delete), place the cursor on the page tab of the page to be deleted.

A red dot (with a white bar inside) is displayed in the right upper corner of the page tab, as in the Test Page below.



2. Place the cursor on the red dot and click once.

A delete confirm prompt is displayed:



3. Click **OK** in the prompt.

You can also delete a page from the **Manage Page** dialog. As with the procedure above, you cannot delete a page from **Manage Page** if you are currently accessing the page to be deleted.

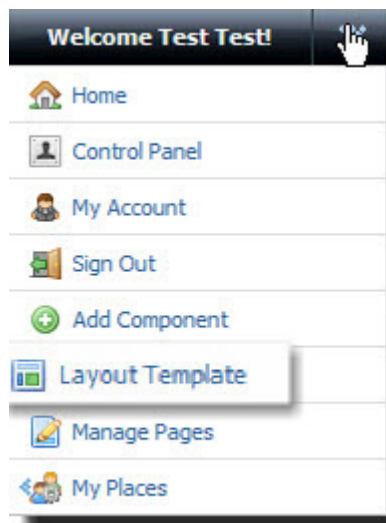
Applying a layout template to the page

Liferay provides you with a number of default layout templates that allow you to establish the layout of the pages in your Discovery Framework application.

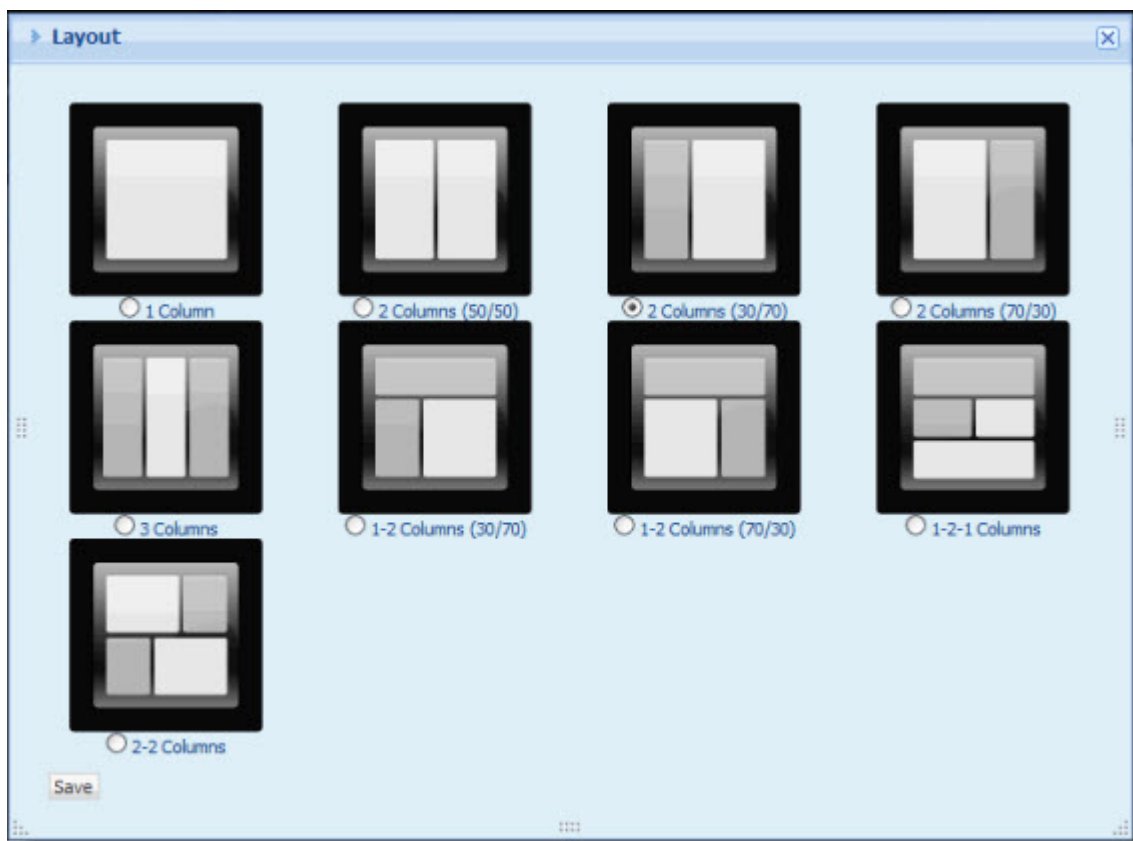
Layout templates specify the way components are organized on each individual page. For example, you can select a layout with two horizontal columns of different widths on one page, and one with a banner and two equally-wide columns on another.

To change the layout of a page:

1. Click the page tab, so that you go to the page.
2. From the **Dock** menu, go to **Layout Template**.



3. In the **Layout** window, select a layout option.



4. Click **Save**.
The new layout is applied to the page, with any existing components organized accordingly.



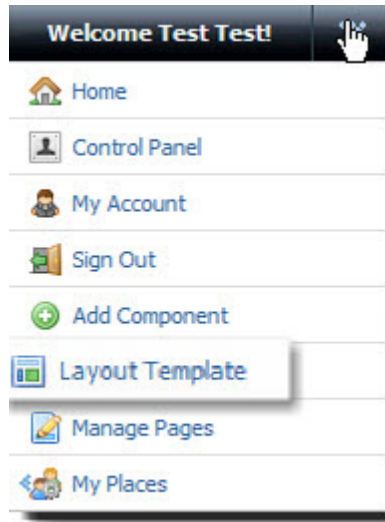
Note: If the default layout templates supplied by Liferay do not meet your needs, your developer can create custom layout template plugins for you, and your system administrator can make them available to your application.

Adding Endeca standard components

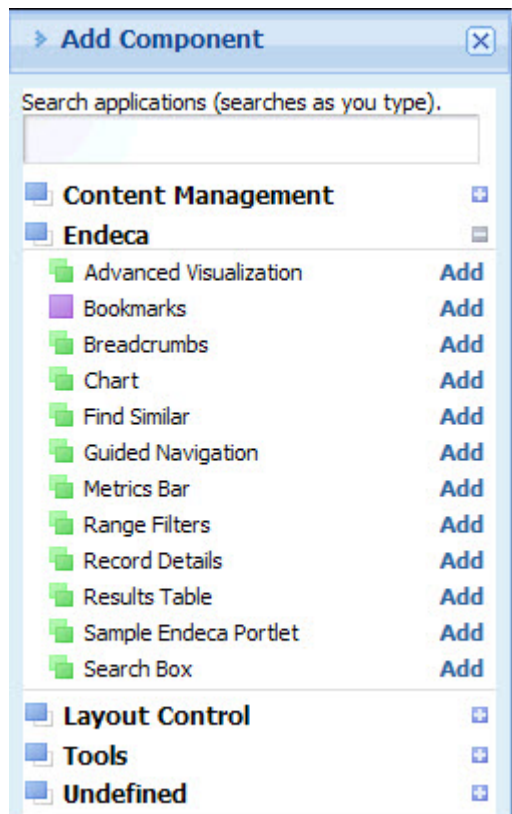
The Discovery Framework contains several Endeca standard components. These components make it possible for you to add Endeca functionality to your application.

To add an Endeca component to your Discovery Framework application:

1. Point the cursor at the **Dock** in the upper-right corner of the page.



2. In the drop-down menu, select **Add Component**.
The **Add Component** dialog box opens.
3. In the **Add Component** dialog box, expand the **Endeca** category.



A list of the available Endeca components appears.

4. To add a component to the main page layout, click **Add**.

Renaming components

After you add a component to your application, you can rename it.

To rename a component:

1. Click the title bar of the component.

The default title becomes editable.



2. Overwrite the existing title with your new one.
3. Click the cursor outside of the title bar.
The new title is applied to the component.

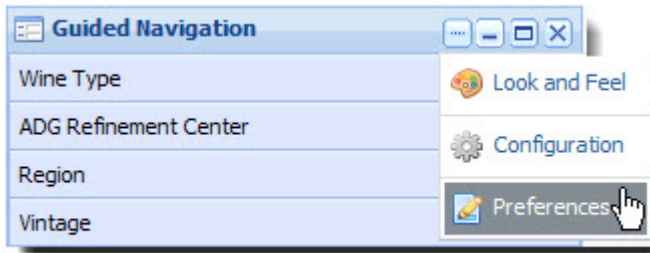
Editing components

This topic provides generic information about Discovery Framework component editing. For detailed information about configuring specific components, see the component documentation.

By editing Discovery Framework components, you further customize your end users' experience. For example, you can specify what data source backs a given component, what set of attributes end users have access to, or what style of chart they will see.

To edit an Endeca standard component:

1. In the component's title bar, click, and then click **Preferences**.



The component's editor appears.

2. Edit the settings for the component.

If the component has a **Save Preferences** or an **Apply** button, make sure to click it after you have made your changes.

3. Click **Return to Full Page**.



Changing the company logo

You can replace the Endeca Discovery Framework portal-wide logo with your own company logo.



Note: Creating a new logo is a design task not normally performed by the power user.

To change the company logo:

1. In the **Control Panel**, navigate to **Portal > Settings > Display Settings**.
2. In the **Logo** section of the **Settings** page, click **Change**.
3. Browse to the logo .png file you want to use, open it, and click **Save**.
4. On the **Settings** page, click **Save**.
The new logo is applied to the application's pages.



Chapter 5

Discovery Framework Logging

This topic describes how to set up the Apache `log4j` logging utility for your Discovery Framework application.

Modifying logging in your Discovery Framework components

There are two ways to modify logging for your Discovery Framework components.

From a developer's perspective, you will most frequently need to adjust a logging verbosity level for a given class or class hierarchy. The easiest way to do this is through the Liferay Control Panel. If you need to modify logging in a more complex manner, or want to change default settings, you can modify the `portal-log4j-ext.xml`.

Both of these methods are described in this section.



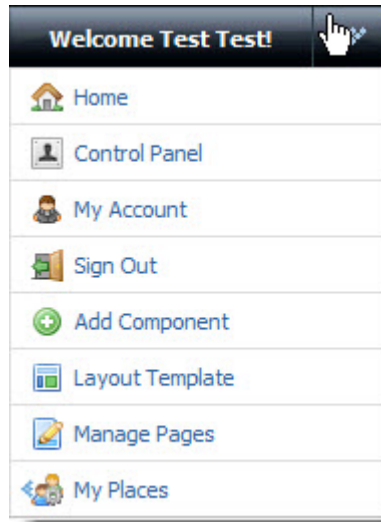
Note: Using either of these methods causes Liferay to adjust the log verbosity for both `log4j` and the Java Utility Logging Implementation (JULI). Code using either of these loggers should respect this configuration.

Adjusting the logging verbosity level in the Control Panel

The easiest way to dynamically adjust logging verbosity levels for any class hierarchy is from the Liferay Control Panel.

To adjust logging verbosity in the Liferay Control Panel:

1. In the Discovery Framework, point the cursor at the Dock in the upper-right corner of the page. The Dock is labeled "Welcome *<user name>!*"



2. From the drop-down menu, choose **Control Panel**.
3. From the **Control Panel** tool menu, choose **Server Administration**.
4. In the **Server Administration** pane, choose the **Log Levels** tab.

Server [Back to Guest](#)

Server Administration

Liferay Portal Enterprise Edition 5.2 EE SP3 (Augustine / Build 5207 / January 7, 2010)
Uptime: 00:44:03

Resources **Log Levels** Properties Data Migration File Uploads Mail OpenOffice Shutdown

Update Categories Add Category

Showing 1 - 20 of 224 results. Items per Page 20 Page 1 of 12 First Previous Next Last

Category	Level
com.ecyrd.jspwiki	ERROR
com.endeca	INFO
com.endeca.portal.instrumentation	INFO
com.germinus.easyconf	ERROR
com.liferay	ERROR

5. Scroll to find the class hierarchy you want to modify, and then adjust the logging level in the drop-down list. The available options are:
 - OFF
 - FATAL
 - ERROR
 - WARN

- INFO
- DEBUG
- ALL



Note: When you modify a class hierarchy, all classes that fall under that class hierarchy are also changed.

6. When you have finished adjusting log levels, click **Save**.



Note: By default, Endeca sets log levels for `com.endeca` and `com.endeca.portal.instrumentation`. You can adjust these levels. In addition, you can set the verbosity for a specific class or package by using the **Add Category** tab.

Modifying portal-log4j-ext.xml

Liferay's primary log configuration is managed in the `portal-log4j.xml` file (which is packed inside the portal application's `WEB-INF/lib/portal-impl.jar`).

As with many other configuration files, Liferay provides administrators with a second configuration file, `portal-log4j-ext.xml` (located in the portal application's `/WEB-INF/classes/META-INF/` directory), which can be used to override settings in the main `portal-log4j.xml` file.

Both of these files are in standard `log4j` XML configuration format. Both files allow creating and modifying appenders, binding appenders to loggers, and adjusting the default log verbosity of different classes and packages. By default, the Endeca override file `portal-log4j-ext.xml` specifies a log verbosity of `INFO` for the `com.endeca` and `com.endeca.portal.instrumentation` packages. Endeca does not override any of the default log verbosity settings specified for non-Endeca components packaged in `portal-log4j.xml`.

Setting up logging

The Discovery Framework uses the Apache `log4j` logging utility.

Liferay's primary log configuration is managed in `portal-log4j.xml` file (which is packed inside portal application's `WEB-INF/lib/portal-impl.jar`). As with many other configuration files, Liferay provides administrators with a second configuration file, `portal-log4j-ext.xml`, which can be used to override settings in the main `portal-log4j.xml` file.

Both of these files are in standard `log4j` XML configuration format. Both files allow creating and modifying appenders, binding appenders to loggers, and adjusting the default log verbosity of different classes/packages. By default, the Endeca override file specifies a log verbosity of `INFO` for the `com.endeca` and `com.endeca.portal.instrumentation` packages. Endeca does not override any of the default log verbosity settings specified for non-Endeca components packaged in `portal-log4j.xml`.

The Endeca log configuration specifies three appenders. The main root logger prints all messages to two locations: the console, which is typically redirected to the application server's output log (`catalina.out` in Tomcat and `SystemOut.log` in WAS), and a file called `df.log`. That file is specified relative to the working directory. If not adjusted, the `df.log` file can typically be found in one of the following locations:

1. If Tomcat was started by running the `startup.bat` or `startup.sh` script, the log is found wherever the script was run. For example, if you navigate to `tomcat-<version>/bin` and execute the startup script, your logs appear in `tomcat-<version>/bin/df.log`.
2. If Tomcat was registered and started as a Windows service, the log files may be located in `C:\Windows\System32\df.log` or `C:\Windows\SysWOW64\df.log`.
3. If Tomcat is a server inside of Eclipse, the log files may be located in the root of the Eclipse directory (such as `C:\eclipse\df.log`).
4. If running WAS 6.1 or 7, the log files may be located relative to the profile's working directory (such as `/localdisk/WAS/AppServer/profiles/AppSrv01/df.log`).

In addition to the console and `df.log` appenders, Endeca also provide a second file appender for capturing metrics logging. This appender creates a file called `df-metrics.log`, which is generated in the same location as `df.log`. All log entries produced by classes in `com.endeca.portal.instrumentation` are routed to `df-metrics.log`—they are not printed to the console or to `df.log`. By keeping performance information separate, administrators can easily distinguish server logs from performance logs, and can easily run analysis scripts on the performance logs.



Note: For further details on log4j logging in Liferay, see the *Liferay Portal Administrator's Guide*.

log4j.properties

This topic describes the different versions of the `log4j.properties` file.

The version of the `log4j.properties` file that is located in `common/endorsed/log4j.properties.jar` is used to configure logging for the Tomcat bundle. The file ensures that there is some preliminary log4j configuration, because log4j is initialized before the Discovery Framework in the Tomcat bundle. This `log4j.properties` file provides minimal configuration, which ensures that initial messages are logged to the console in the same format as the default configuration in `portal-log4j-ext.xml`. The settings in the `log4j.properties` file only affect a small number of messages printed as the server is starting. Once the Discovery Framework starts and loads its XML configuration file, it overrides the settings in the `log4j.properties` file. Therefore, it should not be necessary for administrators to modify this properties file.

In addition, all deployed portlets, as well as the Discovery Framework application itself, have their own `log4j.properties` files, located in `WEB-INF/classes`. Because the Discovery Framework uses XML configuration files, these properties files have no effect.

Learning about log4j

This topic provides links to additional information about log4j.

The [Apache log4j site](#) provides general information about log4j, along with documentation.

For more information about log4j logging in Liferay, see the [Liferay documentation](#), including the *Liferay Portal Administrator's Guide*.



Chapter 6

Administrative Components

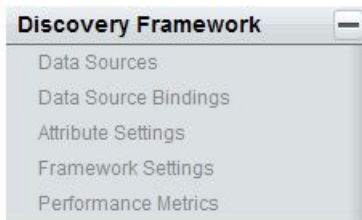
The Discovery Framework contains a number of administrative components. Many administrative components are chiefly used by system administrators, but some can be modified by power users.

Accessing the administrative components

You access Discovery Framework administrative components from the Liferay Control Panel.

This is in contrast to other Discovery Framework standard components, which you access from the **Add Component** menu.

The main menu of the Discovery Framework administrative components looks like this:



The five administrative components are described in this chapter.

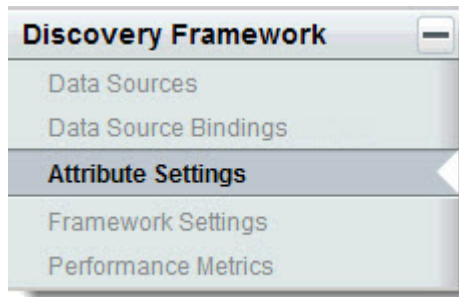
To access the Discovery Framework administrative components:

1. In the Discovery Framework, point the cursor at the **Dock** in the upper-right corner of the page. The **Dock** is labeled "Welcome <user name>!"
2. From the drop-down menu, choose **Control Panel**.
A number of component menus are displayed, including the **Discovery Framework** main menu shown above.
3. In the **Discovery Framework** main menu, click the administrative component you want to edit.
4. After you finish editing the component, click **Return to <user name>**.

Attribute Settings

The **Attribute Settings** component allows the power user to create, edit, and delete attribute sets and change display names for any attribute in a selected data source.

An attribute set is a collection of dimensions and properties associated with the named data source. They allow you to organize your properties and dimensions into named groups.

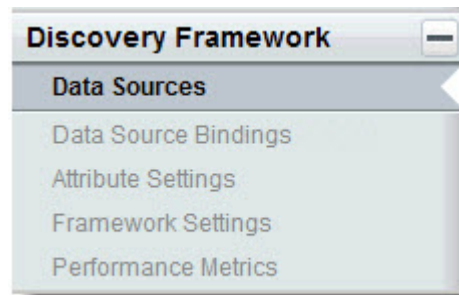


For details on using the **Attribute Settings** component, see Chapter 7 ("Managing Attributes") in this guide.

Data Sources

The **Data Sources** component allows you to view configured data sources and test the connection to them. In addition, you can reload updated configuration based on edits you have made on disk.

This component appears in the Liferay **Control Panel** and is not accessible from the **Add Component** menu.



Each Discovery Framework component that needs to query the MDEX Engine is tied to one particular data source. Changing any individual component's data source is done in that component's edit controls.

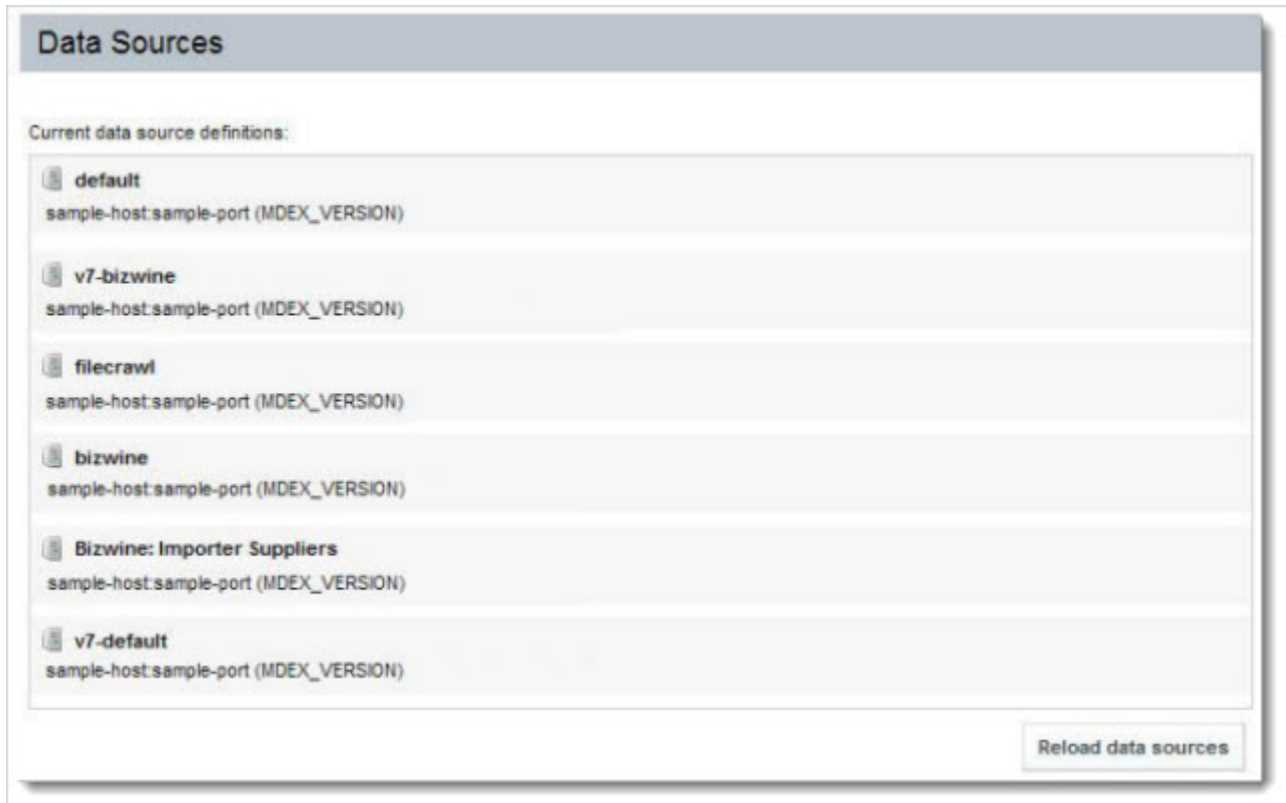


Note: For general information about Discovery Framework data sources, see Chapter 3 ("Configuring Data Sources") in this guide.

Configuring Data Sources

The **Data Source** component requires no configuration—you simply place it on the page.

When you open the **Data Sources** component, you can see all of the data sources that have been configured for your application, as shown in this example:



Each data source definition consists of:

- The name of the data source.
- The version of the MDEX Engine they are using (`ENE_QUERY` for MDEX Engine Version 6).

By clicking a data source's icon (highlighted below), you can test the data source's connection. The icon changes and mousing over it provides a confirmation message.



To reload updated configuration based on edits made on disk, you click the **Reload data sources** button. (For details on making these edits, see the *Discovery Framework Extension Guide*.)

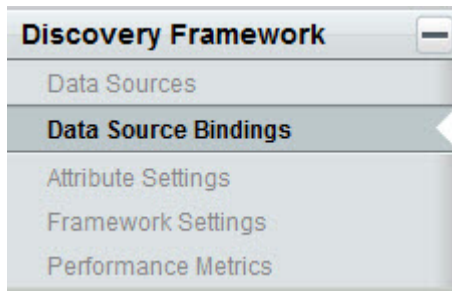




Note: Configuration of data sources themselves takes place offline.

Data Source Bindings

The **Data Source Bindings** component allows the power user to associate different configured data sources with selected components in a single operation, rather than on a per-component basis.



Configuring Data Source Bindings

Because the **Data Source Bindings** component does not display as an independent portlet, it is not configured in the same way as most Endeca standard components. Instead, you modify these settings in the **Control Panel**.

The **Data Source Bindings** component allows you to see all data-source-backed components on each page in your application and associate data sources with them, without having to configure them individually.

To associate components with a particular data source, check the components, select the data source from the drop down list, and click **Update data source**.

Data Source Bindings

Welcome

- ☐ Breadcrumbs : default
- ☒ Guided Navigation : default
- ☒ Record Details : default
- ☒ Find Similar Portlet : default

mdex7

- ☐ Breadcrumbs : v7 bizwine
- ☐ Guided Navigation : v7 bizwine
- ☐ Results Table : v7 bizwine

appcity

Endeca Record Details Portlet

- ☐ Record Details : default
- ☐ Find Similar Portlet : default
- ☐ Results Table : default

Find-Similar-Page

Select a data source

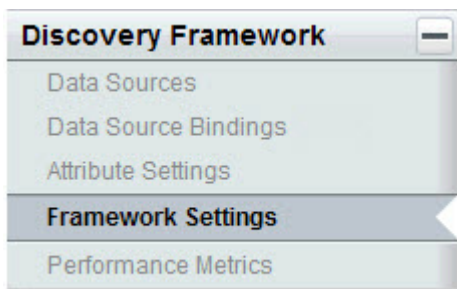
Framework Settings

The **Framework Settings** component provides access to state, security, and other settings.



Note: This component appears in the **Control Panel** and is not accessible from the **Add Component** menu.

Many settings related to Discovery Framework can be adjusted from the **Discovery Framework Settings** section of the **Control Panel**.



The default values of these settings are created automatically upon first use. You cannot add or delete settings from the **Control Panel**—you can only edit them. Settings only appear after the feature(s) that use them have been executed at least once. For example, if you have never used the **Chart** component, the **Corda Server URL** settings will not appear.

Configuring Framework Settings

Because the **Framework Settings** component does not display as an independent portlet, it is not configured in the same way as most Endeca standard components. Instead, you modify these settings in the **Control Panel**.



Note: If you do not see the **Discovery Framework Settings** in the **Control Panel**, it probably means you did not install the `endeca-framework-settings-portlet-<version>.war` file. Please review your installation settings.

The **Framework Settings** page looks like this:

Framework Settings

Warning! Incorrect values for these settings can cause serious problems with your Discovery Framework application. Please do not change these settings unless you are sure of what you are doing.

You must restart the Discovery Framework in order for changes to these settings to take effect.

df.cordalmageAuthEnabled:	<input type="text" value="false"/> <p><small>Image Authorization (BETA): Whether to secure Corda chart image URLs so they're only accessible to the user who requested them</small></p>
df.cordaServerExternalUrl:	<input type="text" value="http://appdev-x2k8.ne.endeca.com:8080/corda/server"/> <p><small>The externally-accessible URL of the Corda server.</small></p>
df.cordaServerInternalUrl:	<input type="text" value="http://localhost:8080/corda/server"/> <p><small>The internally-accessible URL of the Corda server.</small></p>
df.cordaServerRedirectorUrl:	<input type="text" value="http://appdev-x2k8.ne.endeca.com:8080/endeca-corda-chart-portlet/ctRedirect"/> <p><small>The URL of the Corda Redirector.</small></p>
df.dataSourceDirectory:	<input type="text" value="\${liferay.home}/data/endeca-data-sources"/> <p><small>The directory on disk from which to load the Data Source definition files. This must be an absolute path. You may start this value with the token "\${liferay.home}" to represent the Liferay portal root.</small></p>
df.deepLinkPortletName:	<input type="text" value="endecadeeplinkportlet_WAR_endecadeeplinkportlet"/> <p><small>The name of the deep link portlet.</small></p>
df.defaultDataSource:	<input type="text" value="default"/> <p><small>The id of the data source to be used by default for new portlets.</small></p>
df.defaultExporter:	<input type="text" value="com.endeca.export.CSVExport"/> <p><small>The default exporter class.</small></p>
df.exportPortletName:	<input type="text" value="endeca-results-export-portlet_WAR_endeca-results-export-portlet"/> <p><small>This is the default name of the export portlet that can be used with p_o_id.</small></p>
df.healthCheckTimeout:	<input type="text" value="5000"/> <p><small>The time, in milliseconds, to query timeout when checking data source availability</small></p>

The **Framework Settings** component contains the following settings:

Framework Setting	Meaning
df.cordalmageAuthEnabled:	Controls whether Corda chart image URLs are secured.
df.cordaServerExternalUrl and df.cordaServerInternalUrl:	The externally- and internally-accessible URLs of the Corda Server, which is used by the Chart component.
df.cordaServerRedirectorUrl:	The URL of the Corda Redirector, which can be used by the Chart component.
df.dataSourceDirectory:	The directory on disk from which to load data source definition files.
df.deepLinkPortletName:	The name of the deep link component.
df.defaultDataSource:	The name of the data source to use as the default.

Framework Setting	Meaning
df.defaultExporter:	The default exporter class.
df.exportPortletName:	The default name of the export portlet.
df.healthCheckTimeout:	The time (in milliseconds) for query timeout when checking data source availability on initialization.
df.maxExportAnalyticsRecords:	The maximum allowable number of Analytics records that can be exported.
df.maxExportBaseErrors:	The maximum allowable number of non-Analytics records that can be exported.
df.mdexCacheManager:	The fully-qualified class name to use for the MDEX Cache Manager. Changing this setting is currently experimental and unsupported, and should be used only for research purposes. This interface will change in upcoming releases.
df.mdexStateManager:	The fully-qualified class name to use for the MDEX State Manager.
df.mdexSecurityManager:	The fully-qualified class name to use for the MDEX Security Manager.
df.metadataCacheEnabled:	If set to <code>false</code> , retrieves this metadata directly from the data source whenever required (this will be slow). This setting will take effect for new sessions. Note that the use of this setting in a production environment is not supported.

To change a Framework setting:

1. On the **Framework Settings** page, change a setting by providing a new value in its configuration field.



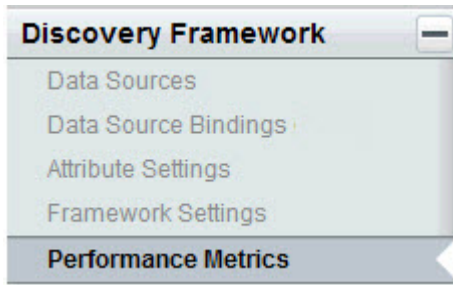
Note: Take care when modifying these settings, as incorrect values can cause problems with your Discovery Framework application.

2. Click **Update Settings**.
3. For the changes to be applied, restart Discovery Framework.

Performance Metrics

The **Performance Metrics** component displays information about component and MDEX Engine query performance.

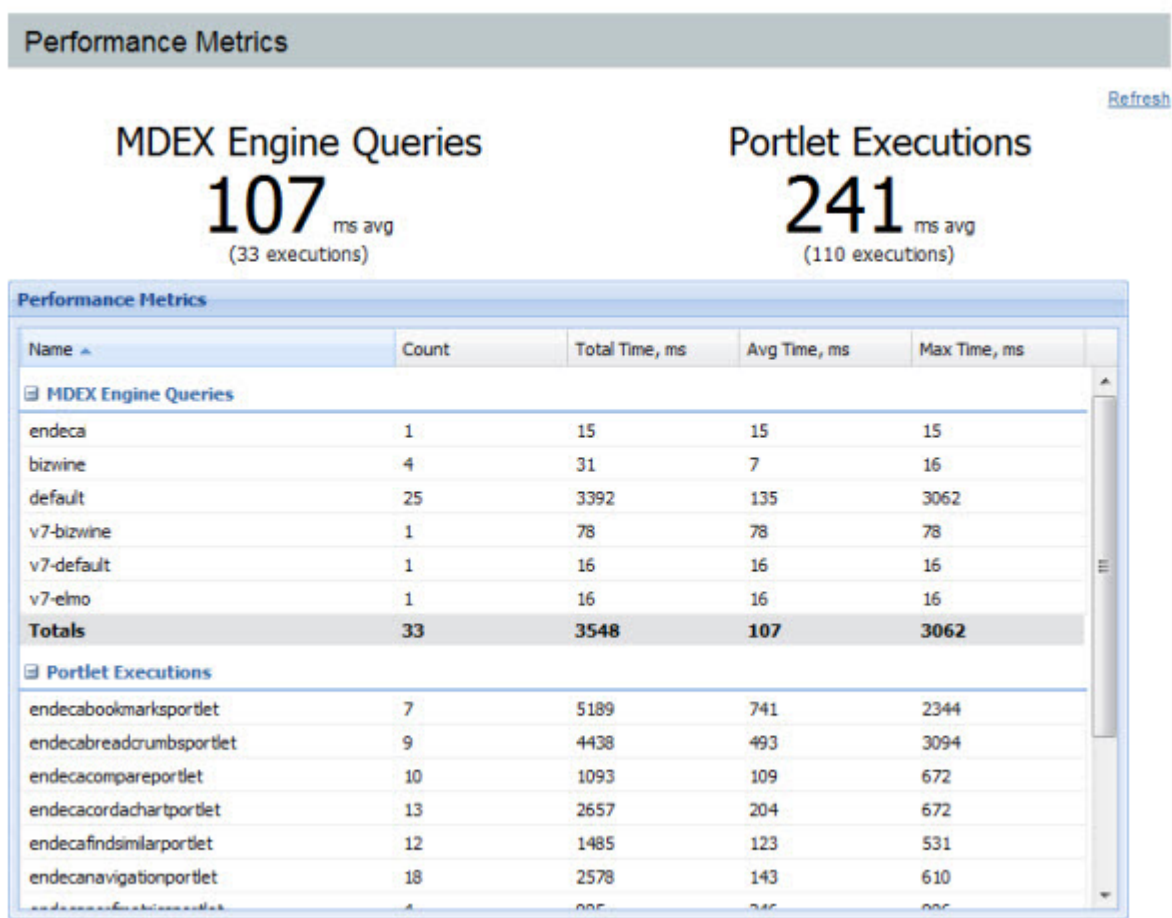
This component appears in the Liferay **Control Panel** and is not accessible from the **Add Component** menu.



The **Performance Metrics** component comes pre-configured and does not require a backing data source.

Using Performance Metrics

This topic describes how an end user can use the **Performance Metrics** component.

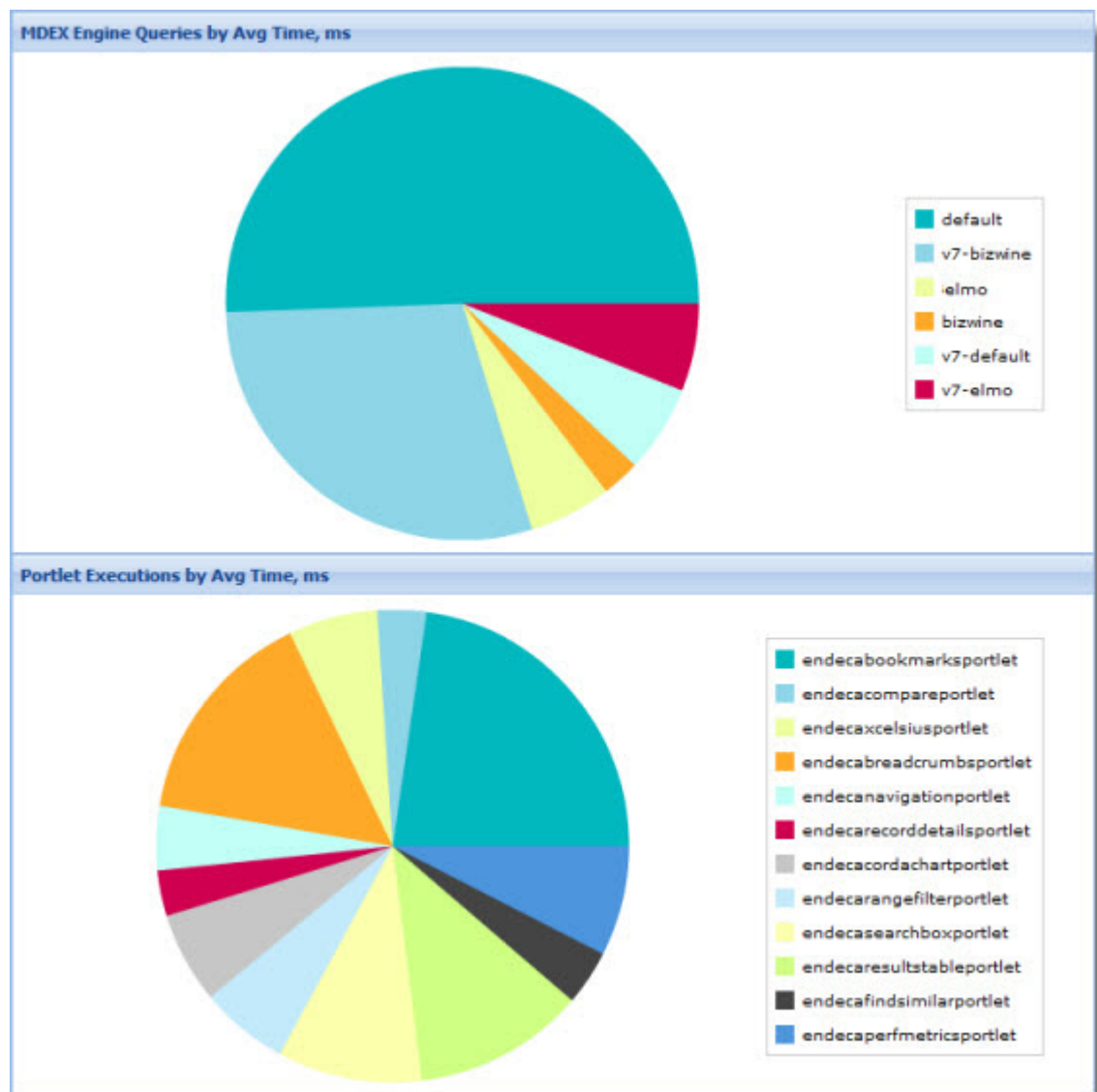


The **Performance Metrics** component tracks total MDEX Engine queries as well as queries per data source. It also tracks the overall performance of portlet components as well as the performance of each individual component.

The top portion of the component contains breakdowns of MDEX Engine queries by data source, as well as details on individual component executions. The **Performance Metrics** component tracks the

total number of queries, total execution time of queries, average execution time of queries, and maximum query execution time.

The bottom portion of the component summarizes the average time metrics in pie-chart form.



Note: This component provides a view into MDEX Engine query performance, which does not correlate directly to a page view, as there are often multiple MDEX Engine queries powering a single page.

Sample Endeca Portlet

The **Sample Endeca Portlet** component provides developers with a template from which they can build their own custom components.

This component is provided as a starting point for developers who want to create their own components.

☐ Sample Endeca Portlet Return to Full Page

Save

Data Source Configuration

default

Update data source

Grid Configuration

Max Result Count Threshold:

10000

Fields for Display (Drag-and-Drop to Reorder)

Fields

☐ spec

☒ P_WineType

☒ P_Name

☒ P_Score

☐ P_Region

☒ P_Price

☐ P_DateReviewed

☒ P_Year

☐ P_Winery

☐ P_WineID

☐ P_Description



Chapter 7

Managing Attributes

This chapter describes how the power user can create, edit, and delete attribute sets and change display names for any attribute in a selected data source.

About Endeca attributes

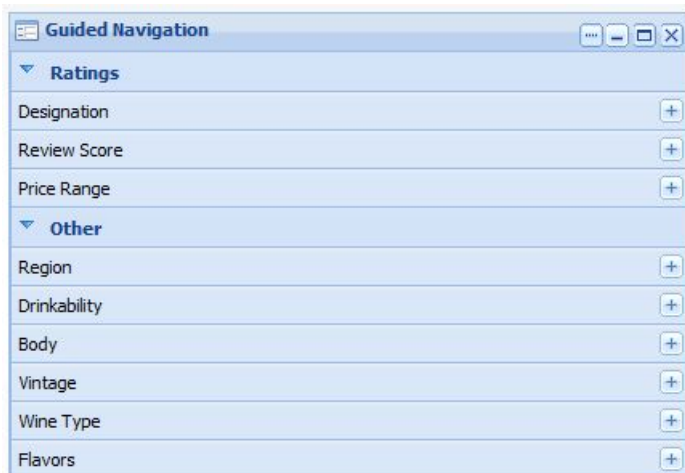
Endeca properties and dimensions are collectively called Endeca attributes.

Attributes can be grouped into attribute sets. An attribute set is a collection of dimensions and properties associated with the named data source. They allow you to organize your properties and dimensions into named groups.

Attribute sets are displayed in these components:

- The **Guided Navigation** component.
- The **Record Details** component.

For example, if you created an attribute set named Ratings, it would appear in the **Guided Navigation** component:

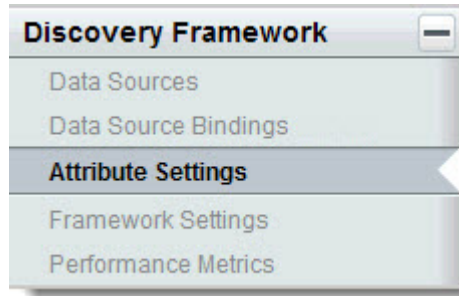


Accessing the Attribute Settings component

The **Attribute Settings** component allows the power user to create, edit, and delete attribute sets and change display names for any attribute in a selected data source.

To access the **Attribute Settings** component:

1. In the Discovery Framework, point the cursor at the **Dock** in the upper-right corner of the page.
2. From the drop-down menu, choose **Control Panel**.
3. From the **Control Panel** menu, select **Discovery Framework**.
4. From the **Discovery Framework** menu, select **Attribute Settings**.



The **Attribute Settings** dialog is displayed.

Attribute Settings dialog

You can manage your attributes and attribute sets with the **Attribute Settings** component's dialog.

When you access the **Attribute Settings** component, its dialog looks like this example:

Attribute Settings

Allows you to create, edit, and delete attribute sets and attribute display names.

Select a data source: default Load from MDEX

All Attributes		Attribute Sets
<input type="checkbox"/>	Attribute Name ▲ Display Name	Other
<input type="checkbox"/>	ADG Refinement Center ADG Refinement Center	P_WineType
<input type="checkbox"/>	Body Body	Endeca.NumWords
<input type="checkbox"/>	Designation Designation	P_Flavor
<input type="checkbox"/>	Drinkability Drinkability	P_WineID
<input type="checkbox"/>	Endeca.DataSize Endeca.DataSize	Endeca.NumAssigns
<input type="checkbox"/>	Endeca.NumAssigns Endeca.NumAssigns	Region
<input type="checkbox"/>	Endeca.NumWords Endeca.NumWords	P_Score
<input type="checkbox"/>	Flavors Flavors	Drinkability
<input type="checkbox"/>	P_Body P_Body	P_Region
<input type="checkbox"/>	P_DateReviewed P_DateReviewed	P_DateReviewed
<input type="checkbox"/>	P_Description P_Description	P_Year
<input type="checkbox"/>	P_Designation P_Designation	P_Winery
<input type="checkbox"/>	P_Drinkability P_Drinkability	P_Designation
<input type="checkbox"/>	P_Flavor P_Flavor	P_Body
<input type="checkbox"/>	P_Name P_Name	P_Drinkability
<input type="checkbox"/>	P_Price P_Price	Body
<input type="checkbox"/>	P_Region P_Region	P_Price
		Price Range

Add selected attributes to set Add
Create a new set ... Add

Creating new attribute sets

Use the **Attribute Settings** dialog to create new attribute sets.

To create a new attribute set:

1. Open the **Attribute Settings** component.
2. Select a data source from the drop-down list.

Select a data source:

default ▼

Select a data source

default

bizwine

Bizwine: Importer Suppliers

v7-default

3. In the **Create a new set** text box, enter the name of the attribute set and click **Add**.

 A screenshot of a user interface element consisting of a text input field with the placeholder text "Create a new set ..." and a button to its right labeled "Add" with a plus icon.

The empty attribute set is added to the **Attribute Sets** panel.

4. In the **All Attributes** panel, check the attributes that you want to add to the new set.
5. Select the new set from the **Add selected attributes to set** drop-down list and click **Add**.

 A screenshot of a user interface element showing a drop-down menu with the text "Add selected attributes to set" and a plus icon, followed by an "Add" button with a plus icon.

The added attributes are displayed in the set in the **Attribute Sets** panel.

After the attribute set is added, it will appear in the **Guided Navigation** and **Record Details** components.

Removing attributes from an attribute set

Use the **Attribute Settings** dialog to remove attributes from attribute sets.

Removing an attribute from a given attribute set does not remove it from other attribute sets or from the MDEX Engine.



Note: You cannot remove attributes from the **Other** default attribute set.

To remove an attribute from an attribute set:

1. Access the **Attribute Settings** dialog.
2. Select a data source from the drop-down list.

 A screenshot of a drop-down menu titled "Select a data source:". The menu is open, showing a list of options: "default" (highlighted), "bizwine", "Bizwine: Importer Suppliers", and "v7-default".

3. In the **Attribute Sets** panel, open the attribute set you want to manage.
4. To remove an attribute, click the delete icon next to the attribute name. Note that you will not be asked to confirm the operation.

 A screenshot of a panel titled "Attribute Sets". It contains a list of attribute sets: "Ratings", "Price Range", "Review Score", and "Designation". To the right of each attribute set name is a red "X" icon, which is the delete button. The "X" icon next to "Price Range" is highlighted with a yellow box.

5. If desired, repeat step 4 to remove more attributes from the set.

Adding attributes to attribute sets

Attributes can be added at any time to an existing attribute set.

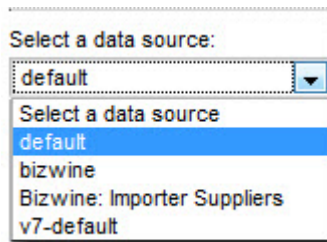
You can add any attribute that is listed in the **All Attributes** panel. An attribute can be added to multiple attribute sets.



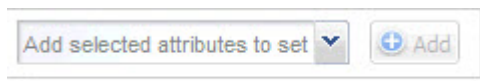
Note: You cannot add attributes to the **Other** default attribute set.

To add an attribute to an existing attribute set:

1. Access the **Attribute Settings** dialog.
2. Select a data source from the drop-down list.



3. In the **All Attributes** panel, check the attributes that you want to add to the set.
4. Select the set from the **Add selected attributes to set** drop-down list and click **Add**.



The added attributes will be displayed in the set in the **Attribute Sets** panel.

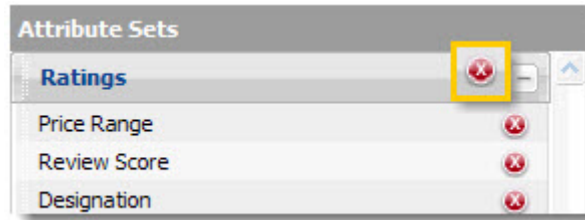
Deleting attribute sets

Use the **Attribute Settings** dialog to delete attribute sets.

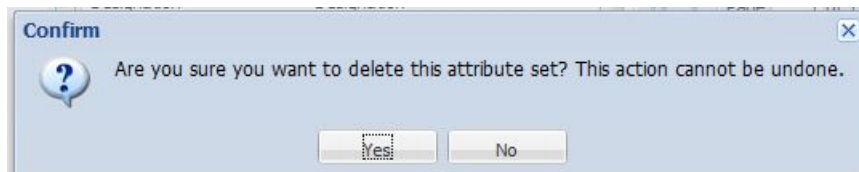
You can delete any attribute set except for the **Other** default attribute set. Deleting an attribute set does not delete its attributes from the MDEX Engine or from other attribute sets.

To delete an attribute set:

1. Access the **Attribute Settings** dialog.
2. In the **Attribute Sets** panel, click the delete icon on the attribute set you want to delete.



3. You are prompted to confirm the delete operation. Click **Yes** to delete the attribute set.



The attribute set is removed from the MDEX Engine and from the **Attribute Sets** panel.

About attribute display names

Display names provide user-friendly names for attributes on your data records.

Display names are names for the properties and dimensions of your data records that are expressed in an easy-to-understand way. Once defined, display names appear in various components, such as the **Guided Navigation**, **Results Table**, and **Record Details** components.

Defining display names is useful because they provide an easy way for the end users to navigate data records. For example, users can refine their search results, or understand and analyze results of analytics statements by selecting display names for different properties on the data records.

Keep in mind that changing an attribute's display name does not change the attribute's name as tagged on the data record. Tagged attribute names on records are in an NCName format, while display names can include non-NCName characters such as spaces and colons. For example, the tagged attribute name on the record will be "WineType" while the display name can be "Wine Type".

Changing the global display name of an attribute

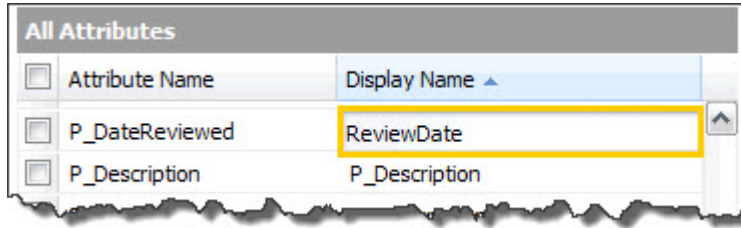
You change the display names for attributes in the **All Attributes** panel.

You can change display names for attributes at a time. The display name does not have to be unique.

Because the display name is used in the front-end application, Endeca recommends that you choose a reasonably short name that is easy to understand by your end users.

To change the global display name of an attribute:

1. Access the **Attribute Settings** dialog.
2. In the **All Attributes** panel, locate the attribute whose display name you want to change.
3. In the **Display Name** column for the attribute, double-click the cell containing the current display name. The cell changes to an editable field.
4. In the edit field, type in a new display name and press **Enter**.



The **All Attributes** panel is updated with the new display name. Any component that lists the display name is also immediately updated.



Chapter 8

Layout Components

Layout components provide additional control over Discovery Framework application design and display.

Tabbed Component Container

The **Tabbed Component Container** allows developers to create a tabbed interface within a region of a page and then store different components on various tabs.

The **Tabbed Component Container** is the only Endeca-developed layout control. This component is available under the **Layout Control** section of the **Add Component** menu, and not the **Endeca** section.

This component allows a user to create a tabbed interface within a region of a page, and then nest different components on the available tabs. For example, you might choose to put a **Results Table** component containing customer-based information on one tab, and another **Results Table** component containing product-based information on another tab.



Note: After placing the **Tabbed Component Container** on the page, make sure to refresh the page before attempting to add other components to the tabs.

Using the Tabbed Component Container

End users do not access the **Tabbed Component Container** directly, though a power user might include one or more instances in the end users' application.

Configuring the Tabbed Component Container

The power user can set the following preferences for the **Tabbed Component Container** component.

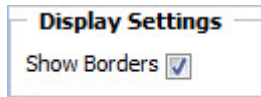


Note: Only the configuration tasks on the **Setup** tab are relevant to our purposes. You may disregard the other tabs.

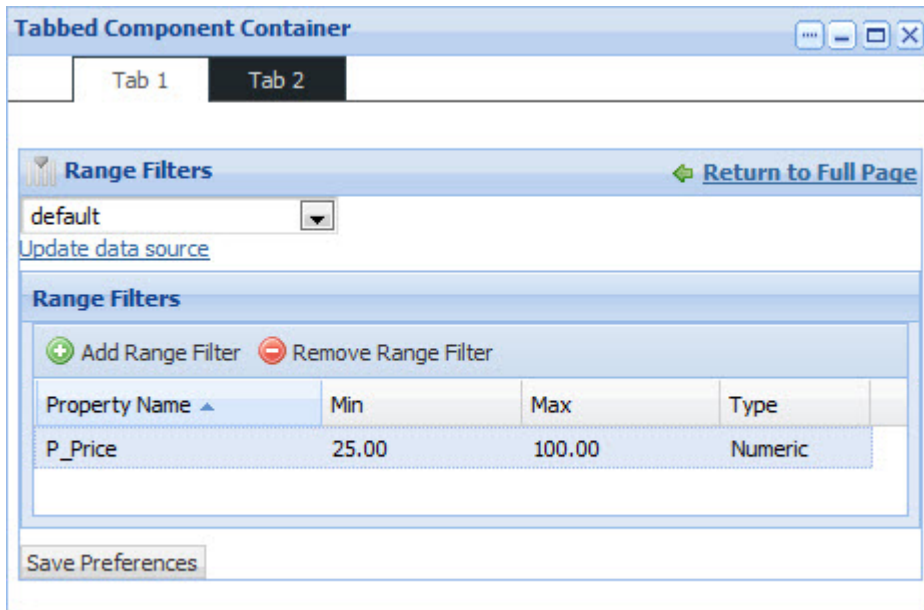
- You can edit the current tabbed component container, or select an archived version (if available).

- Tabs:** Add or delete tabs, as well as change their layouts, display names, and order on the page.

- Display Settings:** Check **Show Borders** to draw the component container around the components on each tab.



Drag the configured **Tabbed Component Container** onto the portal work area, then drag other components onto the tabs and configure them as usual. In the example below, the **Range Filter** component has been dragged onto Tab 1.





Chapter 9

Using Endeca Analytics

This section introduces Endeca Analytics, describes its interaction with the Discovery Framework, and points you to additional information about the Analytics language.

About Endeca Analytics

Endeca Analytics builds on the core capabilities of the Endeca MDEX Engine to enable applications that examine aggregate information such as trends, statistics, analytical visualizations, and comparisons.

Key extensions to the Endeca MDEX Engine associated with Analytics include the following:

- An Analytics language, which allows users to explore aggregate and statistical views of large databases using a Guided Navigation interface.
- Support for date and time data types, which allows applications to work with temporal data, performing time-based sorting, filtering, and analysis.

Before using Analytics, the Endeca MDEX Engine being used as a data source must be enabled for the Endeca Analytics feature. For more information, contact your Endeca representative.

Analytics requirements for power users

The Discovery Framework power user needs to be able to write Endeca Analytics statements. These statements are used to create Discovery Framework charts and other visualizations.

The Endeca Analytics language is documented in the *Endeca Analytics Guide*, which is available in the Knowledge Base of the Endeca Developer's Network (EDeN).

Components that support Analytics

The following Discovery Framework components support the use of Analytics.

- **Advanced Visualization** component
- **Chart**
- **Cross Tab** component
- **Metrics Bar** component
- **Results Table** component



Chapter 10

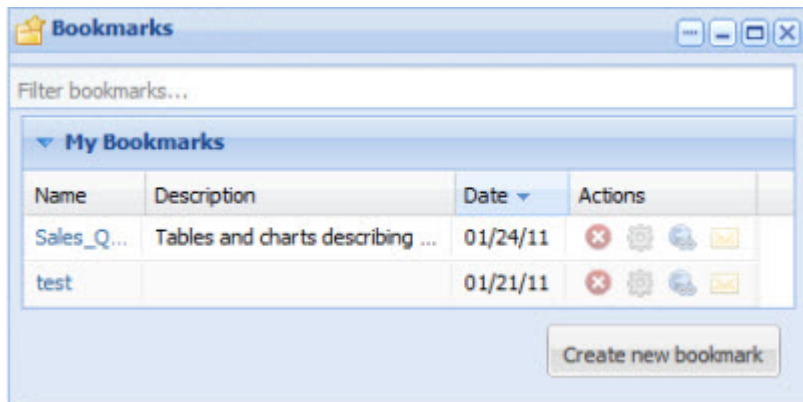
Personalization Components

Personalization components allow end users to customize their Discovery Framework application.

Bookmarks

Bookmarks is a personalization component that allows the end user to save a given navigation and component state in order to be able to share it or return to it at a later time.

Once a bookmark has been created, the user can click upon it to return to that state. Bookmarks are listed in the order they were created, along with an accompanying description and date.



Note: The **Bookmarks** component does not require a backing data source. It stores and retrieves data from the underlying Liferay database.

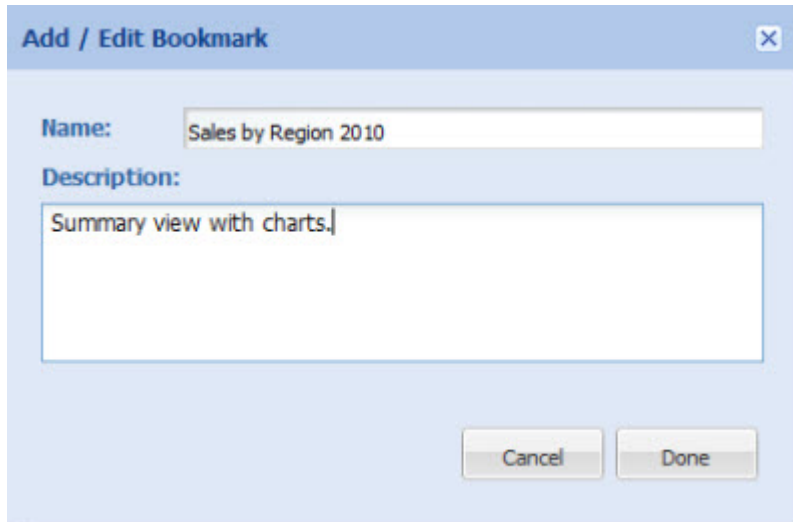


Note: The **Bookmarks** component supports the use of aggregated records.

Using Bookmarks

The **Bookmarks** component allows the end user to create, edit, delete, and share bookmarks.

To add a new bookmark, click **Create new bookmark**. The **Add / Edit Bookmark** dialog box, where you can fill in the **Name** and **Description**, displays. Click **Done** to add the bookmark.











Add / Edit Bookmark


Name: Sales by Region 2010


Description:
Summary view with charts.

Cancel Done

Once a bookmark has been created, four icons (Delete, Edit, Create link, Email) appear next to it:

Parts by ...	Jim's view	01/24/11				
Sales by ...	Summary view with charts.	01/24/11				

To delete a bookmark, click the **Delete** icon .

To edit a bookmark, click the **Edit** icon , which re-opens the **Add / Edit Bookmark** dialog box.


To capture and copy the URL of a bookmark, click the **Create link** icon , which opens the **Bookmark URL** dialog box:

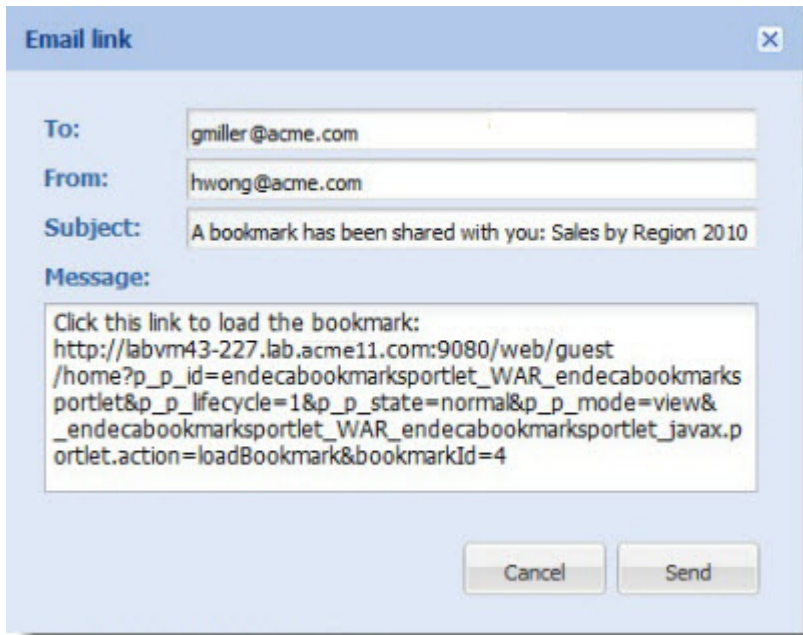


Bookmark URL

Copy and paste this URL: http://aturn43-227-361-253.us-east-1.elb.amazonaws.com:9080/web/guest/home?p_p_id=endecabookmarksportlet_WAR_endecabookmarksportlet&p_p_lifecycle=1&p_p_state=normal&p_p_mode=view&_endecabookmarksportlet_WAR_endecabookmarksportlet_javax.portlet.action=loadBookmark&bookmarkId=4 [select](#)

Done

To email the URL of a bookmark to another user, click the **Email** icon , which opens the **Email link** dialog box, pre-populated with everything except the **To:** field:



The **Email link** dialog box is shown with the following fields:

- To:** gmler@acme.com
- From:** hwong@acme.com
- Subject:** A bookmark has been shared with you: Sales by Region 2010
- Message:**

Click this link to load the bookmark:
http://labvm43-227.lab.acme11.com:9080/web/guest/home?p_p_id=endecabookmarksportlet_WAR_endecabookmarksportlet&p_p_lifecycle=1&p_p_state=normal&p_p_mode=view&_endecabookmarksportlet_WAR_endecabookmarksportlet_javax.portlet.action=loadBookmark&bookmarkId=4

Buttons: **Cancel** and **Send**

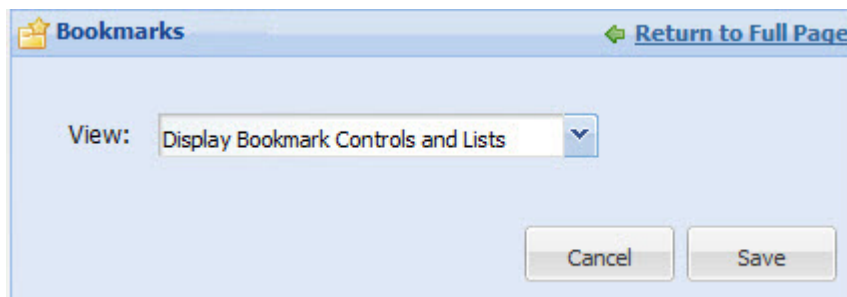


Note: In order to use the email feature, the Discovery Framework administrator must set up the email server, as described later in this section.

Configuring the Bookmarks component

The power user can set the following preferences for the **Bookmarks** component.

The **Bookmarks** configuration dialog allows the power user to select a display mode for the component. This controls the end user's view.

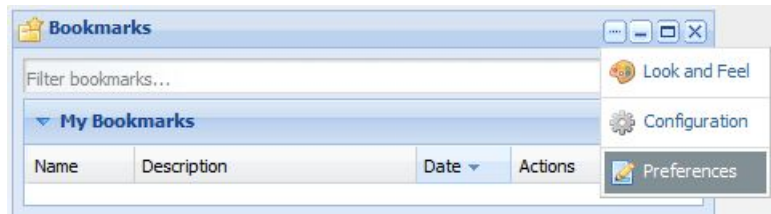


The **Bookmarks** configuration dialog is shown with the following elements:

- View:** Display Bookmark Controls and Lists (dropdown menu)
- Buttons: **Cancel** and **Save**
- Link: [Return to Full Page](#)

To configure the **Bookmarks** component:

1. After adding a **Bookmarks** component to a page, click its button and then click **Preferences**.



2. From the drop-down menu in the **Bookmarks** configuration dialog, select a display mode:
 - **Display Bookmark Controls only** allows the end user to create new bookmarks, but does not display the entire list of saved bookmarks.
 - **Display Bookmark Lists only** allows the end user to display the list of existing bookmarks.
 - **Display Bookmark Controls and Lists** allows the end user to both create new bookmarks and display the list of existing bookmarks. This is the default.
3. Click **Save** to save your changes.

Setting up the email server for Bookmarks support

Before end users can email bookmarks to other users from within the **Bookmarks** component, your Discovery Framework administrator must configure the mail server in the Liferay Control Panel.



Note: For additional information about this and other administrative tasks, see the [Liferay Portal Administrator's Guide](#).

To set up the mail server:

1. Point the cursor at the **Dock** in the upper-right corner of the page. The **Dock** is labeled "**Welcome <user name>!**"
2. From the drop-down menu, choose **Control Panel**.
3. In the **Server** section of the **Control Panel Portal** menu, click **Server Administration**.
4. In the **Server Administration** panel, click the **Mail** tab.
5. On the **Mail** panel, fill out the following outgoing email settings:
 - **Outgoing SMTP Server**
 - **Outgoing Port**
 - **User Name**
 - **Password**

Here is an example of a completed setup:

Outgoing SMTP Server	acme.com.s7a1.pstmp.com
Outgoing Port	25
Use a Secure Network Connection	<input type="checkbox"/>
User Name	user_user@acme.com
Password	12345

6. Click **Save**.

Bookmarks saved data for component types

This topic describes what parts of the component state are saved in the bookmarks created by the **Bookmarks** component.



Note: Administrative components are not included here because they are not saved in bookmarks.

Component	Persisted States	Comments
Advanced Visualization	Developer-driven	<p>The data persisted in a bookmark for this component is specific to the SAP Dashboard Design (Xcelsius) movie being used. It is therefore the responsibility of the movie developer to define which elements are to be persisted. This requires two movie components to be defined via the Xcelsius Data Manager:</p> <ol style="list-style-type: none"> 1. Definition of name-value pairs (established in the Flash Variables data connection) for each stateful design element in the UI. For example, the Drilldown Chart example movie includes an "Average Price by: " drop-down control that drives the Analytics grouping used to render the chart. This is backed by a Flash Variable called "selectedGroupBy" whose value is bound to that of the drop-down selection, so that when a bookmark is loaded, the correct grouping is applied based on which grouping was selected when the bookmark was created. That is, if a user selected "Vintage" from the drop-down and then created a bookmark, then Vintage should be the selected drop-down value when the movie is loaded from a bookmark. 2. An External Interface Connection (EIC) variable called "movieState" that is bound to a cell that contains an ampersand-delimited list of all variables defined in step one. For example, the Drilldown Chart example also includes a radio button control that can either be "yes" or "no" at any given point in time, whose corresponding Flash Variable is called "selectedRadioButton." In order to persist both the drop-down and radio button controls properly, the "movieState" EIC variable is bound to a cell that contains a string that looks like this: <pre>selectedGroupBy=Vintage&selectedRadionButton=yes</pre> <p>The Xcelsius developer is responsible for setting up both sets of connections properly and binding them to appropriate cells in the spreadsheet that is leveraged by the visualization.</p>
Bookmark	None	Nothing about the Bookmarks component itself is stateful.
Breadcrumbs	Expanded multi-select dimensions	When multi-select dimension values are present in the Breadcrumbs component, they may be grouped into a collapse/expand control (depending on the number of values and the threshold set in the component's preferences). The state of each multi-select dimension's collapse/expand control is stateful and will be stored in a bookmark. For example, if

Component	Persisted States	Comments
		you select Flavors of "Apple", "Berry" and "Cherry" in the sample wine data set, the collapse/expand control appears and is collapsed by default. If you expand it and then create a bookmark, the control will be expanded when that bookmark is loaded.
Chart	Metric, x-axis, and cross tab drop-down lists	When configuring a chart, the power user can define multiple metrics, x-axis, and cross tab groupings. When the chart is configured in this way, drop-down lists are provided for each. The user's selection for the Metric , Group By (X-Axis) , and Cross Tab drop-down lists are persisted with a bookmark.
Compare	Attributes: persists the order of attributes and the expand and collapse state of attribute sets, Records: persists the order of records and their highlighted state	
Find Similar	None	
Guided Navigation	Expanded dimensions and collapsed dimension groups	Any dimension that is expanded so that its refinements are exposed is persisted in a bookmark, so that it will still be expanded when the bookmark is loaded. Dimension groups are, by default, expanded. If you collapse a dimension group, this is persisted in a bookmark, so that it will be collapsed when the bookmark is loaded.
Metrics Bar	None	
Range Filter	None	
Record Details	None	
Results Table	Sort state and number of records per page	If you have sorted on a column and create a bookmark, the sort will be applied when the bookmark is loaded. If you have selected a certain number of records per page and create a bookmark, the results table will display that number of records per page when you load the bookmark.
Results Table (BETA Enhancements)	The selected column set state	
Search Box	None	
Tabbed Component Container	The tab in focus	If you create a bookmark when on a tab other than the first one, that tab will be reloaded properly when the bookmark is loaded.
Tag Cloud	None	



Chapter 11

Results Components

The **Results Table** and **Record Details** components provide a detailed view of records returned from queries.

Results Table

The **Results Table** component provides a simple interface for displaying results. The **Results Table** component can show results from Analytics and non-Analytics queries.

It provides a list of records in table form and enables users to view a record's details. Users can page through, sort, and scroll across large tables, switch between tabs, and drill down on selected results. Pinned columns help users keep track of the data they are viewing.

The screenshot shows a window titled "Results Table" with a table of wine records. The table has columns: P_WineType, P_Name, P_Score, P_Region, P_Price, and P_DateRev. The table is displayed with 10 rows of data. The interface includes a "Choose a column set:" dropdown with "Tab 1" and "Tab 2" options. At the bottom, there is a pagination bar showing "Page 1 of 5708" and "Records per page" with a refresh icon. The status bar at the bottom right indicates "Displaying records 1 - 10 of 57076".

P_WineType	P_Name	P_Score	P_Region	P_Price	P_DateRev
Zinfandel	Zinfandel Central	88	Other California	14.000000	10/15/96
White	Albarino Rias Bai	25	Spain	18.000000	06/15/93
White	Alsace Gentil	83	Alsace	9.000000	11/15/94
White	Alella Marques de	76	Spain	11.000000	04/15/94
White	Alsace Grand Cru	84	Alsace	16.000000	11/15/94
White	Chardonnay Nap.	82	Napa	11.000000	07/31/95
White	Chardonnay Nap.	81	Napa	18.000000	07/31/96
White	Sauvignon Blanc	79	New Zealand		05/15/94
White	Greece Laoutari	78	Greece	19.000000	04/30/97
White	Chardonnay Utiel	75	Spain	6.000000	04/30/95



Note: The **Results Table** component supports the use of aggregated records. For details about configuring Discovery Framework data sources for aggregated records, see Chapter 3 ("Configuring Data Sources") in this guide.

Dependencies

To see the details of a selected record, you need to have an instance of the **Record Details** component available in the application, either on the same page or on a page specified as the target.

Only Analytics statements with aggregation are supported in the **Results Table** component. If you are using an Analytics statement in the **Results Table** component, that statement must include a GROUP BY. If it does not, the table displays only a single row data on each page, and the records per page option does not have any effect.

About attribute names

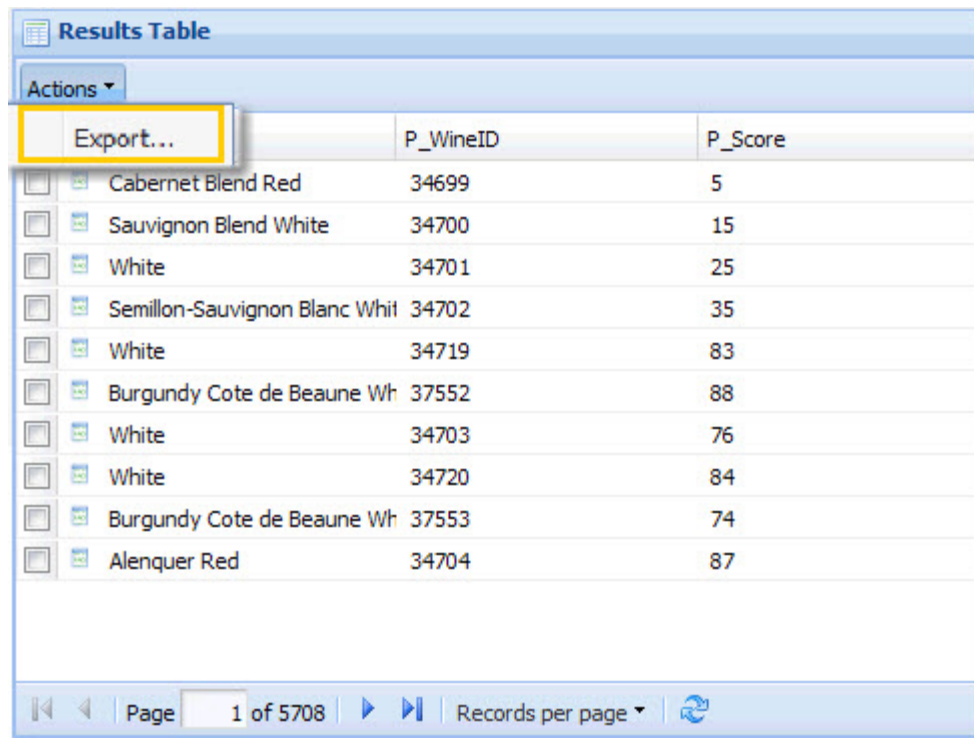
- Attributes (properties and dimensions) with parentheses in their names are not supported in the **Results Table** component.
- As a best practice, you should use NCName-compliant attribute names. If more user-friendly display names for attributes are required, you can provide them by using the display name feature of the **Attribute Settings** administrative component.

Using Results Table

This topic describes how an end user can use the **Results Table** component.

The **Results Table** component provides end users with a list of records in table form and enables them to view record details for both Analytics and non-Analytics queries. The **Results Table** component enables end users to page through records, sort them by clicking the column headers and drill-down to view a record's details in the **Record Details** component.

In addition, the end user can export the record list to Excel, by clicking **Actions > Export**:



The screenshot shows the 'Results Table' component with a table of wine records. The 'Actions' dropdown menu is open, and the 'Export...' option is highlighted. The table has columns for wine names, P_WineID, and P_Score. The footer shows pagination information: Page 1 of 5708 and a 'Records per page' dropdown.

	P_WineID	P_Score
<input type="checkbox"/> Cabernet Blend Red	34699	5
<input type="checkbox"/> Sauvignon Blend White	34700	15
<input type="checkbox"/> White	34701	25
<input type="checkbox"/> Semillon-Sauvignon Blanc Whit	34702	35
<input type="checkbox"/> White	34719	83
<input type="checkbox"/> Burgundy Cote de Beaune Wh	37552	88
<input type="checkbox"/> White	34703	76
<input type="checkbox"/> White	34720	84
<input type="checkbox"/> Burgundy Cote de Beaune Wh	37553	74
<input type="checkbox"/> Alenquer Red	34704	87

Configuring the Results Table component

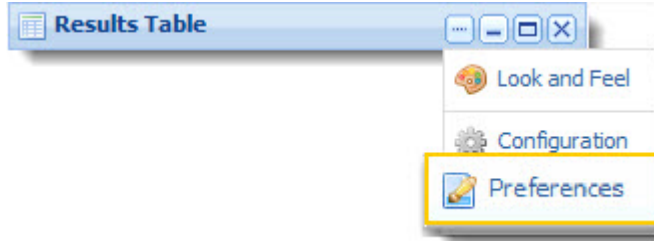
This topic describes the high-level steps involved in configuring the **Results Table** component.

There are two configuration sections for this component:

- The **Analytics Configuration** section is for the Analytics feature.
- The **Results Table Configuration** section sets the appearance and actions of the table.

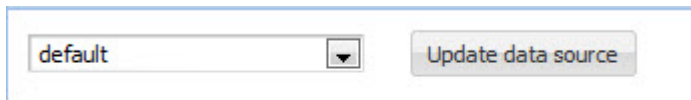
To configure the **Results Table** component:

1. After adding a **Results Table** component to a page, click its button and then click **Preferences**.



The **Results Table** edit view appears.

2. Select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.

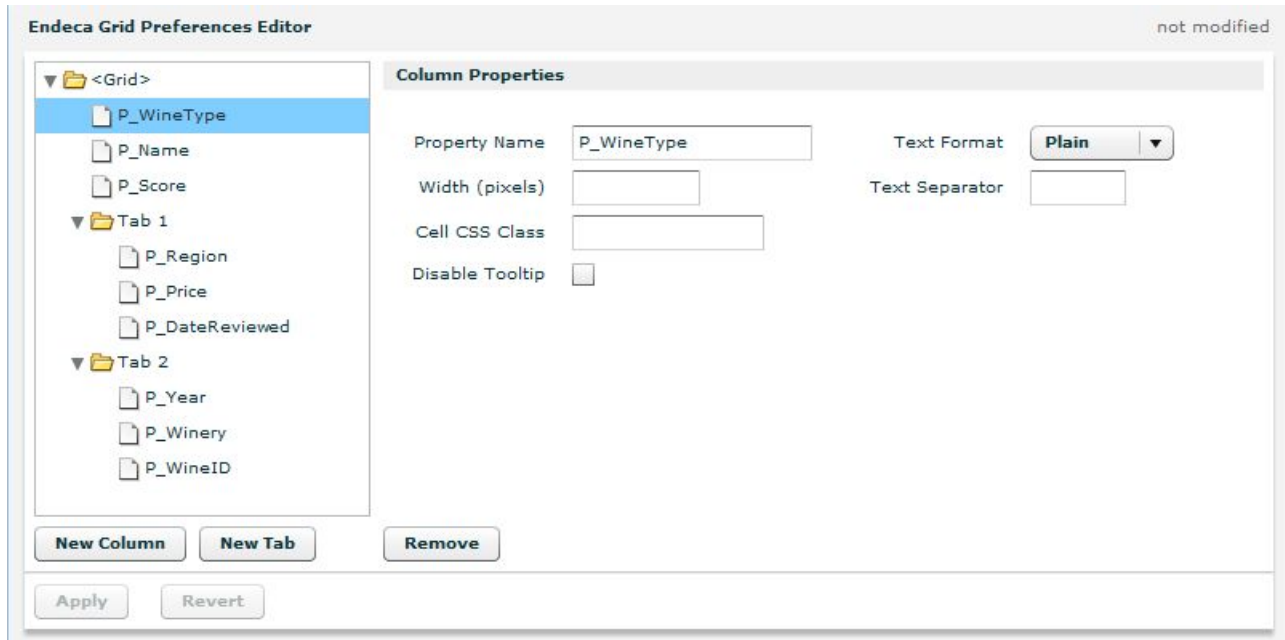


3. In the **Analytics Configuration** section, you can provide a table-based results display for Analytics statements. For details, see the "Configuring the Results Table for Analytics results" topic.
4. Use the **Endeca Grid Preferences Editor** to design the grid that end users will see.
See the next topic for details on this editor.
5. Click **Apply** to apply your changes.
6. Click **Return to Full Page** to return to the **Results Table** end-user view.

Grid Preferences Editor

This editor allows you to customize the columns and content of the Results Table.

The Grid Preferences Editor looks like this example:



The functions of the controls and editing dialogs are:

Control/Dialog	Purpose
Tab/Column pane	The pane (on the left side) from which you select the item you want to edit. The item can be a tab (such as Tab 1 or the Grid tab) or a column.
Properties dialog	The dialog (on the right side) where you edit the properties of the item. The three properties dialogs are the Grid Properties dialog, the Column Properties dialog (shown in the example), and the Tab Properties dialog. Each dialog has a different set of properties that are appropriate for the selected item.
New Column button	Adds a new column to one of the tabs.
New Tab button	Adds a new tab under the Grid tab.
Remove button	Deletes a column or an empty tab (that is, a tab that does not have columns).
Apply button	Applies your changes to the Results Table component.
Revert button	Undoes any changes that have not been applied and reverts the configuration to the last saved state.
Status indicator	A field at the upper-right corner of the editor that indicates the status of the configuration: <code>not modified</code> (the configuration is unchanged), <code>MODIFIED</code> (the configuration has been changed but the changes have not been applied), and <code>SAVED</code> (the configuration has been changed and the changes have been applied).

Adding or editing columns

You can add new columns to the tabs in the Results Table or modify the properties of existing columns.

To add a new column to a tab or modify an existing column:

1. Access the **Grid Preferences Editor**.
2. In the **Tab/Column** pane, do one of the following:
 - To add a new column, click the tab in which you want to add the column and then click the **New Column** button. Note that you can add columns to the **Grid** tab.
 - To edit an existing column, click the column name.

In either case, the **Column Properties** dialog is displayed.

The **Column Properties** dialog box is shown. It has the following fields and controls:

- Property Name:** A text input field.
- Width (pixels):** A text input field.
- Cell CSS Class:** A text input field.
- Disable Tooltip:** A checkbox.
- Text Format:** A dropdown menu currently showing 'Plain'.
- Text Separator:** A text input field.

3. Configure the column properties via the fields in the **Column Properties** dialog:
 - **Property Name** sets the property or dimension whose value will be displayed in this column. The display name of the property or dimension will be used as the name of the column tab.
 - **Text Format** determines how the value is displayed: **Plain** (the value displays as-is), **HTML**, **Integer**, or **Currency**.
 - **Width** sets the width of the column in pixels.
 - **Text Separator** sets the separator to use if the column has multiple values.
 - **Cell CSS Class** is currently not supported.
 - **Disable Tooltip** lets you show or hide the mouse-over tool tip for the column.
4. If desired, you can change the position of the column by dragging it up or down within the **Tab/Column** pane.
5. Click **Apply** to apply your changes to the component.

Adding or editing tabs

You can add a new tab to the **Results Table** or change the name of an existing tab.

To add a new tab or change the name of an existing tab:

1. Access the **Grid Preferences Editor**.
2. In the **Tab/Column** pane, do one of the following:
 - To add a new tab, select an existing tab and then click the **New Tab** button. The new tab will be added below the selected existing tab.
 - To edit an existing tab, click the tab name.

In either case, the **Tab Properties** dialog is displayed.

The **Tab Properties** dialog box is shown. On the left, a tree view displays the grid structure:

- <Grid>
 - P_WineType
 - P_Name
 - P_Score
 - New Tab** (selected)
 - Tab 1

The **Tab Properties** dialog box has a **Tab Text** field containing the text 'New Tab'.

3. In the **Tab Properties** dialog, use the **Tab Text** field to enter a new name or change an existing name.
4. If desired, you can change the position of the tab by dragging it up or down within the **Tab/Column** pane.
5. Click **Apply** to apply your changes to the component.

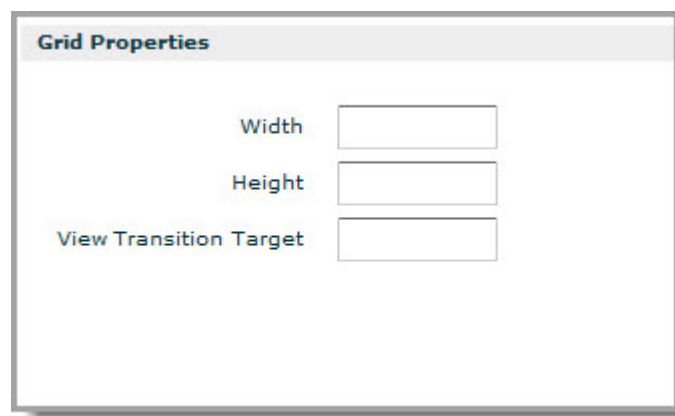
Editing the Grid properties

The **Grid Properties** dialog sets the dimensions of the grid and indicates where the results will display.

To edit Grid properties:

1. Access the **Grid Preferences Editor**.
2. In the **Tab/Column** pane, click the **Grid** tab.

The **Grid Properties** dialog is displayed.

The image shows a dialog box titled "Grid Properties". It contains three labeled input fields: "Width", "Height", and "View Transition Target". Each label is in blue text and is followed by a white rectangular input box with a thin gray border. The dialog box has a light gray header bar with the title "Grid Properties" in blue.

3. Configure the Grid properties via the fields in the **Grid Properties** dialog:
 - **Width** sets the width of the grid in pixels.
 - **Height** sets the height of the grid in pixels.
 - **View Transition Target** specifies a target page. If a target page is specified, clicking the record detail link changes to the specified page and causes any **Record Detail** component on the target page to display the selected record. The target can be specified as a relative path from `/web/guest` or an absolute path. If **View Transition Target** is blank, the record detail link triggers events on the same page, causing any **Record Detail** component on the same page to display the selected record.
4. Click **Apply** to apply your changes to the component.

Deleting tabs and columns

You can delete columns and empty tabs with the **Grid Preferences Editor**.

Keep in mind that you cannot delete a non-empty tab (that is, a tab that has least one column). If you select a non-empty tab, the **Grid Preferences Editor** disables (grays out) the **Remove** button.

To delete a tab or column:

1. Access the **Grid Preferences Editor**.
2. In the **Tab/Column** pane, select the tab or column you want to delete.
3. Click the **Remove** button.

The tab or column is removed without a confirmation prompt.

- Click **Apply** to apply your changes to the component.

Configuring the Results Table for Analytics results

If your application is using Endeca Analytics, you can customize the **Results Table** to display table-based results for Analytics statements.

Before you can use Analytics statements in your application, the MDEX Engine must be enabled for the Analytics option. In addition, you should have a sample Analytics statement that can be used as a basis for setting up the columns for the Results Table. For example, the results of this query:

```
return test as select avg("P_Price") as avgprice, avg("P_Score") as avgscore
group by "Vintage"
```

will create three columns: avgprice, avgscore, and Vintage.

To configure the **Results Table** for Analytics statements:

- Access the **Analytics Configuration** section.
- Check the **Use analytics statement** checkbox.
- Enter an Analytics statement in the query box:

- Click the **Test analytics query** button. If the Analytics statement is valid, the results are displayed under the Analytics query box:

avgscore	avgprice	Vintage
83.745283	21.408067	1992

- Click the **Save Analytics Configuration** button.
As the Caution message reminds you, saving the Analytics statement overwrites the columns configured in the **Grid Preferences Editor** panel. That is, the current columns are replaced with a new set of Analytics columns.
- Use the **Grid Preferences Editor** to configure the new columns and/or add more columns.
- Click **Apply** to apply your changes to the component.

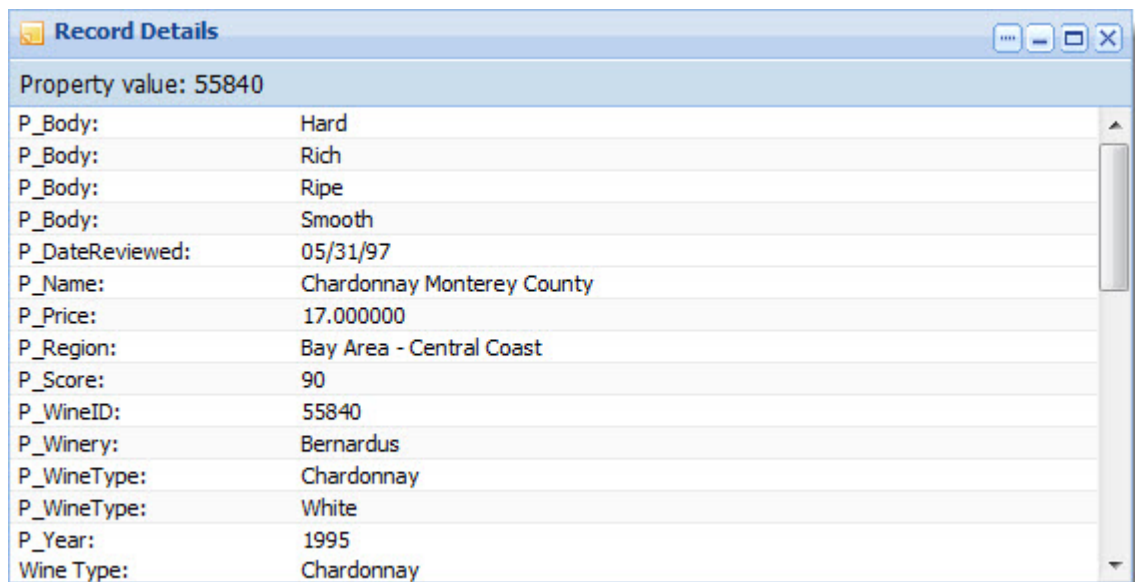
To revert to a non-Analytics **Results Table**, uncheck the **Use analytics statement** checkbox. Doing so will present you with a **Disable Analytics Statement** button, which you can click to revert to the standard **Results Table** grid.

Record Details

The **Record Details** component displays all of the properties and dimensions for a record in the **Results Table** component.

The **Record Details** component shows all properties first, followed by all dimensions, in the order they are returned from the Presentation API (that is, alphabetically). The user cannot change this order. However, the user can edit the render configuration part of the pipeline to control which properties and dimensions are displayed.

The example below shows the relationship between the **Results Table** and **Record Details** components. Selecting a record in the **Results Table** component and clicking the **View record detail** icon populates the **Record Details** component with details about the selected record.



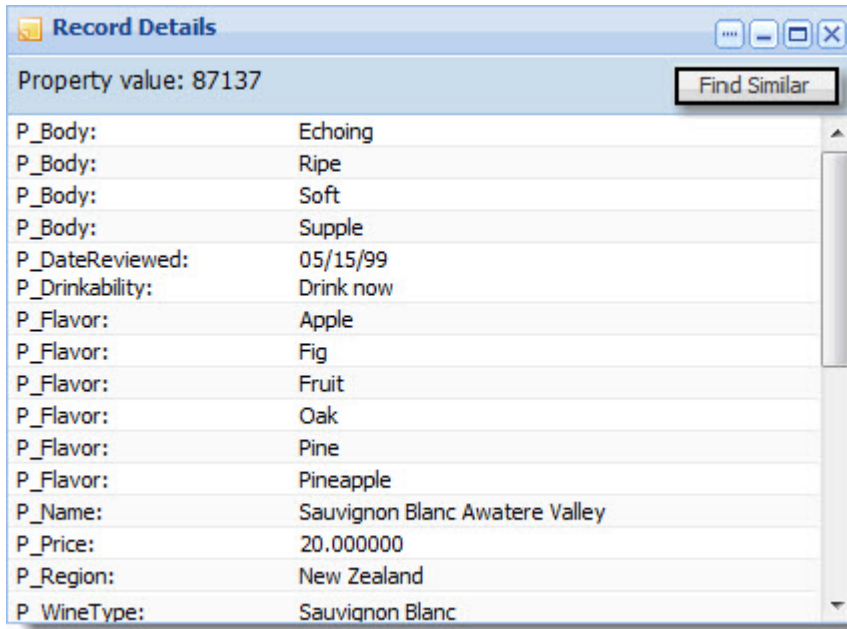
The screenshot shows a window titled "Record Details" with a standard Windows-style title bar. Below the title bar, there is a header bar with the text "Property value: 55840". The main content area is a table with two columns: the first column contains property names, and the second column contains their corresponding values. The table is scrollable, as indicated by a vertical scrollbar on the right side. The data in the table is as follows:

Property value: 55840	
P_Body:	Hard
P_Body:	Rich
P_Body:	Ripe
P_Body:	Smooth
P_DateReviewed:	05/31/97
P_Name:	Chardonnay Monterey County
P_Price:	17.000000
P_Region:	Bay Area - Central Coast
P_Score:	90
P_WineID:	55840
P_Winery:	Bernardus
P_WineType:	Chardonnay
P_WineType:	White
P_Year:	1995
Wine Type:	Chardonnay

Dependencies

The **Record Details** component must be used with the **Results Table** component, because only the **Results Table** provides the required detail links. However, as long as a page transition is specified, the two components do not have to reside on the same page.

The **Record Details** component also works in conjunction with the **Find Similar** component. When a record is displayed in **Record Details**, the user can click the **Find Similar** button (highlighted below) to search through records with similar attributes.



Note: The **Record Details** component supports the use of aggregated records. For details about configuring Discovery Framework data sources for aggregated records, see Chapter 3 ("Configuring Data Sources") in this guide.

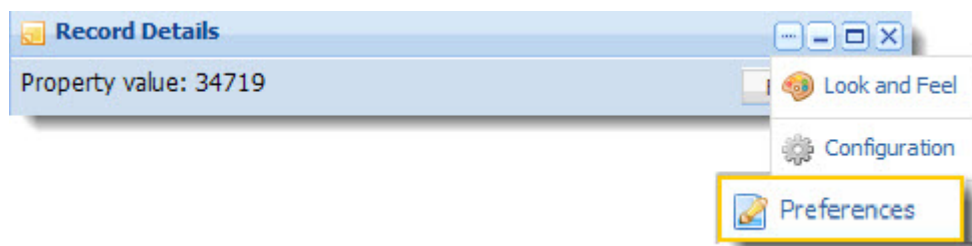
Configuring the Record Details component

This topic describes the high-level steps involved in configuring the **Record Details** component.

Subsequent topics describe certain individual steps in detail.

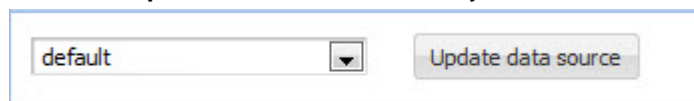
To configure the **Record Details** component:

1. After adding a **Record Details** component to a page, click its button and then click **Preferences**.



The **Record Details** edit view appears.

2. In the **Record Details** edit view, select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.



3. Configure the **Action** menu, as described in a following topic.

4. Configure the **Attribute List**, as described in a following topic.
5. Configure the **Header Field**, as described in a following topic.
6. Click **Save Preferences** to save your changes.

Configuring the Action menu

You can configure the actions that your end users can access when they click the **Actions** button.

The **Configure Action Menu** is where you configure the actions that an end user can perform with the record in the **Record Details** table:

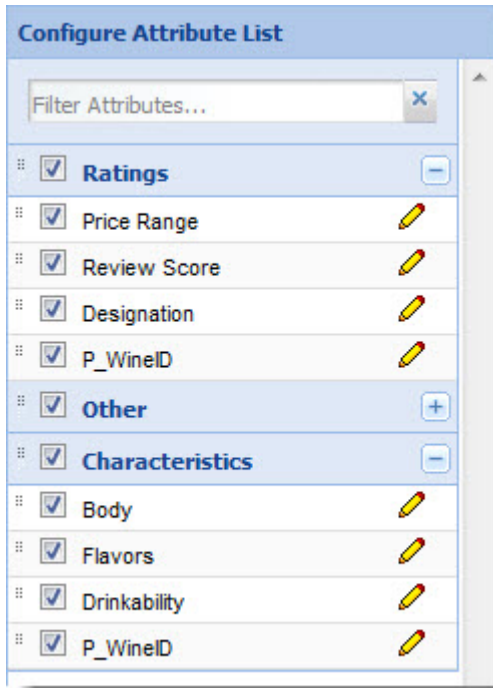
To configure the **Action** menu:


1. Access the **Record Details** edit view.
2. Open the **Configure Action Menu** control.
The menu shown above is displayed.
3. To make the **Action** menu available to users, mark the **Enable action menu** checkbox.
4. To change the name of the menu, enter a new name in the **Action menu name** field.
5. To enable an action for users, move it from the **Available actions** panel to the **Active actions** panel:
 - **Export** allows the user to export the record details to Excel. The single record is exported in CSV format that displays the record as a single Excel row.
 - **Print** displays the record details in another browser window. The end user prints as usual via the **File** menu or keyboard shortcut.
 - **Find Similar** launches the **Find Similar** component for the record in question. Optionally, you can specify a different page as the target for **Find Similar**.
6. Click **Save Preferences** to save your changes.

Configuring the Attribute List

The **Attribute List** defines the attributes that are displayed to the end user in the **Record Details** and **Compare** tables.

The **Configure Attribute List** dialog is where you configure the attributes shown with the record in the **Record Details** table and in the **Compare** table:



The  pencil icon (next to the attribute name) is used to access the **Edit Attribute Display** dialog for that attribute.

To configure the **Attribute List**:

1. Access the **Record Details** or **Compare** edit view.
The **Configure Attribute List** dialog is one of the displayed dialogs.
2. If you need to find a specific attribute, use the **Filter Attributes** search box to search for an attribute by name (with type-ahead).
3. In the Attribute List, check the attributes and attribute sets that you want displayed in the **Record Details** or **Compare** table. Unchecked attributes or attribute sets are not displayed in the table.
4. If you want to change the formatting of an attribute, click the pencil icon for the attribute to open the **Edit Attribute Display** dialog.
See the next topic for details on formatting options.
5. Click **Save Preferences** to save your changes.

Edit Attribute Display dialog

Use this dialog box to set the output formatting for an attribute.

The **Edit Attribute Display** dialog, shown below, lets you change the default format in which the values of some attributes are output in the **Record Details** and **Compare** tables. For example, you can specify that monetary values are prepended with dollar signs or appended percent signs to integers or decimal numerals.

Edit Attribute Display

Include attribute: ☒

Format: Integer

Formatter Options

Thousand Separator: Comma (,)

Add percent (%) suffix: ☐

Cancel OK

When changing the default formatting, you must make sure that the new formatting changes are appropriate to the data type of the attribute. For example, do not attempt to configure **Currency** formats to a string attribute. If the new formatting changes are invalid for the chosen attribute, the Discovery Framework software will revert the formatting to the defaults for the attribute's data type.

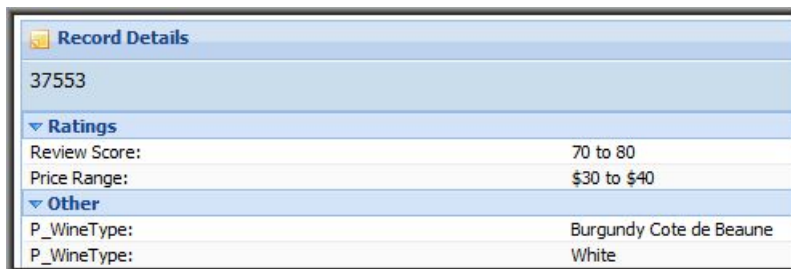
Format	Purpose	Formatter Options
Integer	Formats the output of integer values.	<ul style="list-style-type: none"> • Thousand Separator sets the thousands delimiter to a comma, period, space, or apostrophe. For example, for a value of "15000", specifying the Comma option would format the output as "15,000" while the Period option would produce a "15.000" output. • Add percent suffix appends a % character to the numeric output. For example, a rating score of "25" can be shown as "25%".
Currency	Formats the output of monetary values, such as floating point values.	<ul style="list-style-type: none"> • Thousand Separator sets the thousands delimiter to a comma, period, space, or apostrophe. • Decimal Separator sets a comma or a period as the mark between the integral and the fractional parts of a decimal numeral. • Decimal Places lets you specify the number of decimal places of a decimal numeral. • Currency Symbol either prepends or appends a dollar, pound, euro, or yen monetary symbol to the output.
Decimal	Formats the output of decimal values, such as floating point values.	<ul style="list-style-type: none"> • Thousand Separator sets the thousands delimiter to a comma, period, space, or apostrophe.

Format	Purpose	Formatter Options
		<ul style="list-style-type: none"> • Decimal Separator sets a comma or a period as the mark between the integral and the fractional parts of a decimal numeral. • Decimal Places lets you specify the number of decimal places of a decimal numeral. • Add percent suffix appends a % character to the numeric output.
String	Formats the output of string values.	<ul style="list-style-type: none"> • Default makes no change to the string value stored in the record, which means that the original casing is preserved. • Lower Case changes all the characters to lower case. • Upper Case changes all the characters to upper case. • Title Case changes the first character of each word to lower case.

Configuring the header field attribute

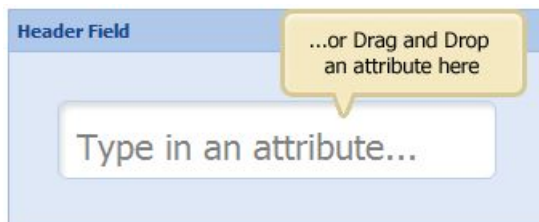
You can change the attribute to use as the display name in the end user's view of the record.

The header field attribute is the attribute whose value is used for the header of the **Record Details** table. In this cropped example, the default attribute for the header field is the P_WineID record specifier property, whose value for this record is 37553:



Record Details	
37553	
▼ Ratings	
Review Score:	70 to 80
Price Range:	\$30 to \$40
▼ Other	
P_WineType:	Burgundy Cote de Beaune
P_WineType:	White

You use the **Header Field** dialog to change the header field attribute:



To change the header field attribute:

1. Access the **Record Details** edit view.
2. Either type the name of an attribute in the **Header Field** text box or drag an attribute into it from the **Attribute List**.

If you changed the header field attribute to the P_Name property, the **Header Field** dialog should look like this:

Header Field

P_Name

- Click **Save Preferences** to save your changes.

The header field of the **Record Details** table should now look like this example:

Record Details

P_Name: Meursault Les Cras

Ratings

Review Score: 70 to 80

Price Range: \$30 to \$40

Other

P_WineType: Burgundy Cote de Beaune

P_WineType: White

Results Table (Beta enhancements)

This release of the Discovery Framework includes an enhanced version of the **Results Table** component. This new version is in Beta for this release.

With the enhanced version of the **Results Table** component, you can view and sort matching results by any dimension or metric, rename and format fields for user-friendly display, and expose high volumes of information through multiple tabs.

Results Table (BETA Enhancements)

Actions ▾ Choose a column set: Ratings Characteristics **Other**

<input type="checkbox"/> P_Name	P_WineType	Region	P_Score
<input type="checkbox"/> A Red Blend Alexander Valley	Cabernet Blend, Red	Sonoma	5
<input type="checkbox"/> A Tribute White Sonoma Mour	Sauvignon Blend, White	Sonoma	15
<input type="checkbox"/> Albarino Rias Baixas	White	Spain	25
<input type="checkbox"/> Alchemy Mendocino County	Semillon-Sauvignon Blanc, Wh	Mendocino Lake	35
<input type="checkbox"/> Sauvignon Blanc California Co	Sauvignon Blanc, White	Other California	50
<input type="checkbox"/> Riesling Auslese Rheingau Rux	Auslese, White	Germany	55
<input type="checkbox"/> Coteaux du Tricastin Rose Bel	Blush, Rhone	Rhone	60
<input type="checkbox"/> Cotes de Provence Rose	Blush, Other France	Provence	55
<input type="checkbox"/> White Grenache Cucamonga V	Blush, Grenache	South Coast	59
<input type="checkbox"/> Chassagne-Montrachet Clos S	Burgundy Cote de Beaune, W	Burgundy	60
<input type="checkbox"/> Chassagne-Montrachet La Gr	Burgundy Cote de Beaune, W	Burgundy	60
<input type="checkbox"/> Chassagne-Montrachet Les Pi	Burgundy Cote de Beaune, W	Burgundy	60

Page 1 of 2854 Records per page Displaying records 1 - 20 of 57076

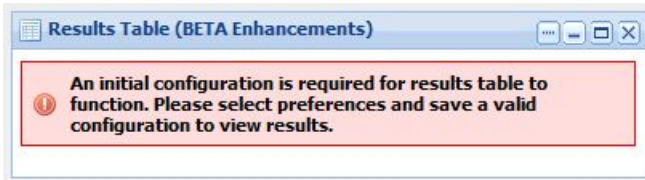
The enhanced **Results Table** component includes the following features:

- Support for attribute sets.
- Power users can configure a link from each record to an external system.
- Power users has enhanced configuration options for custom actions.
- Power users can configure a default sort order for results in the table.
- Power users can configure a default number of records per page.
- Power users can disable end user paging/sorting controls.
- Power users can configure a threshold for maximum records over which analytics queries will not be performed.
- End users can refine results by clicking dimension values in table cells.
- End users can view a totals/summary row for analytics results in the table.
- End users can print the **Results Table**.

Initial configuration of the component

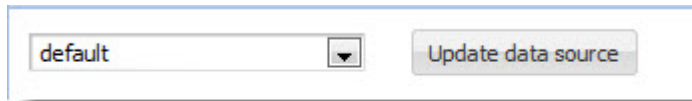
This topic describes how to quickly configure the **Results Table (BETA Enhancements)** component so that it can begin to display query results.

When you first add a **Results Table (BETA Enhancements)** component to a page, it displays a message asking that an initial configuration is required for the component to function:

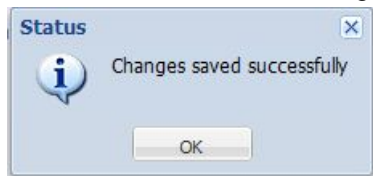


To initially configure the **Results Table (BETA Enhancements)** component:

1. After adding the component to a page, click its button and then click **Preferences**. The component edit view is displayed.
2. Select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.



3. In the **Attribute sets** list, check one or more attributes to display.
4. Click **Save Preferences** to apply your changes.
5. Click **OK** In the Status message.



6. Click **Return to Full Page** to return to the **Results Table (BETA Enhancements)** end-user view.

When you return to the end-user view, you should see the table populated with records.

After the initial configuration, you can later make more advanced the configuration changes to suit your application.

Advanced configuration

The power user can set the following preferences for the enhanced version of the **Results Table** component.

After initially configuring the component, you have more configuration options to customize the component for your end users. The fu

- **Analytics Configuration:** Provide table-based results display for Analytics statements.

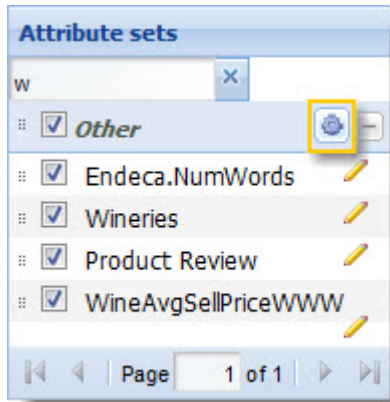


Note: If **Use analytics statement** is not selected, the **Analytics Configuration** options below do not appear in the **Preferences** display.

The screenshot shows a dialog box titled "Analytics Configuration:". It contains the following elements:

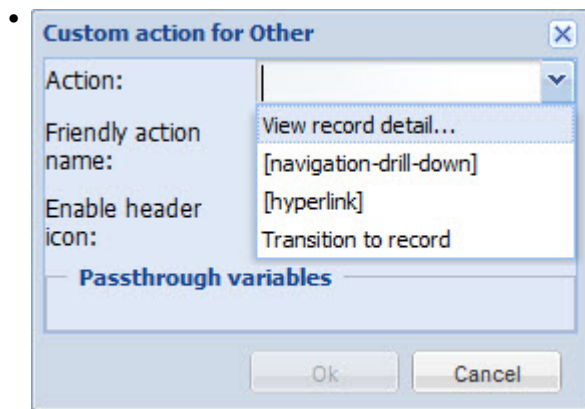
- A checked checkbox labeled "Use analytics statement".
- A text input field labeled "Analytics statement threshold (records):".
- A text input field labeled "Analytics statement threshold message:".
- A large text area labeled "Analytics statement:".
- At the bottom, there are two buttons: "Save Analytics Configuration" and "Test analytics query...".

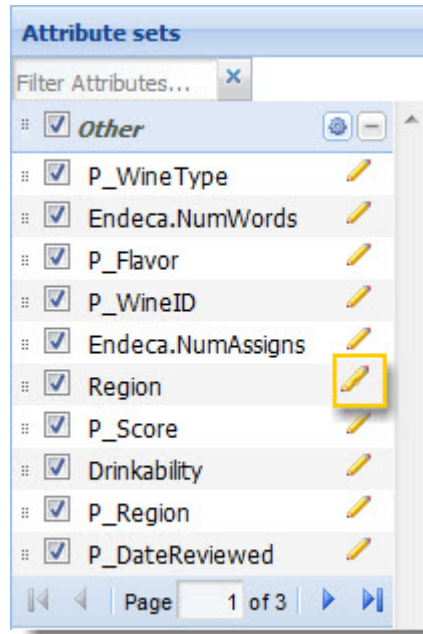
- **Locked Columns:** Pin attributes to ensure visibility when switching tabs and scrolling.
- **Attribute sets:** Select the attributes to be displayed to the end user. The power user can search for an attribute by name (with type-ahead), as well as check and uncheck attributes and attribute sets directly.



The **Actions** button (highlighted above) opens the **Custom action** dialog box, where the power user can choose from the following actions:

- View record detail, and indicate where the results will display. If **View Transition Target** is blank, the record detail link triggers events on the same page, causing any **Record Detail** component on the same page to display the selected record. If **View Transition Target** specifies a target page, clicking the record detail link changes to the specified page, and causes any **Record Detail** component on the target page to display the selected record. The target can be specified as a relative path from /web/guest or an absolute path.
- Navigation drill down
- Hyperlink, which allows the end user to click a link to access an external system.





The **Edit** button (highlighted above) opens the **Options** dialog box, where the power user can set column format (Integer, Currency, Decimal, or String) and associated formatting details. Additionally, there are options for showing or hiding the tool tip and setting the width of the column. The power user can also enable drill-down on the dimension value, if the attribute is a dimension, or bind a custom action link to the attribute value (for example, a record detail view could be hyperlinked to an external system).

Options

Body

Width (pixels):

Disable tooltip: ☒

Format: String

Formatter Options

☒ Default (no change)
☐ Lower Case
☐ Upper Case
☐ Title Case

☐ Enable drilldown
☐ Enable custom action

Cancel Save

- **Grid properties:** Set the following:
 - The dimensions of the grid.
 - The default sort order for each attribute.
 - The number of records displayed on each page in the **Results Table**, and the number of records on each page.
 - Whether end users can set records per page, pagination, or sorting in their own display.
 - Whether Analytics results are totaled in a summary row in the **Results Table**.

Grid properties

Height:

Width:

Default sort order:

- Price Range
- Vintage
- Wine Type

Records per page:

Default records per page: 10

☒ Enable end-user records per page:
☒ Enable end-user pagination:
☒ Enable end-user sorting:
☒ Enable analytics summary:

- **Action menu properties:** Enable and name an action menu for end users, and select and order their menu options.

The screenshot shows a dialog box titled "Action menu properties". It contains the following elements:

- Enable action menu:** A checkbox that is checked.
- Action menu name:** A text input field containing the word "Actions".
- Available actions:** A list box containing the text "exportAction".
- Selected actions:** An empty list box.
- Navigation buttons:** A vertical column of six buttons between the "Available actions" and "Selected actions" list boxes. From top to bottom, they are: an up arrow, a down arrow, a right arrow, a left arrow, an up arrow, and a down arrow.



Chapter 12

Data Visualization Components

Data visualization components provide a more detailed view of your data.

Advanced Visualization

The **Advanced Visualization** component provides SAP Crystal Dashboard Design (Xcelsius) visualizations within your application.

This integration provides Discovery Framework end users with dynamic insight into MDEX Engine powered data.

The **Advanced Visualization** component ships with three sample Flash movies in SWF format and the associated source files in XLF format.

Dependencies

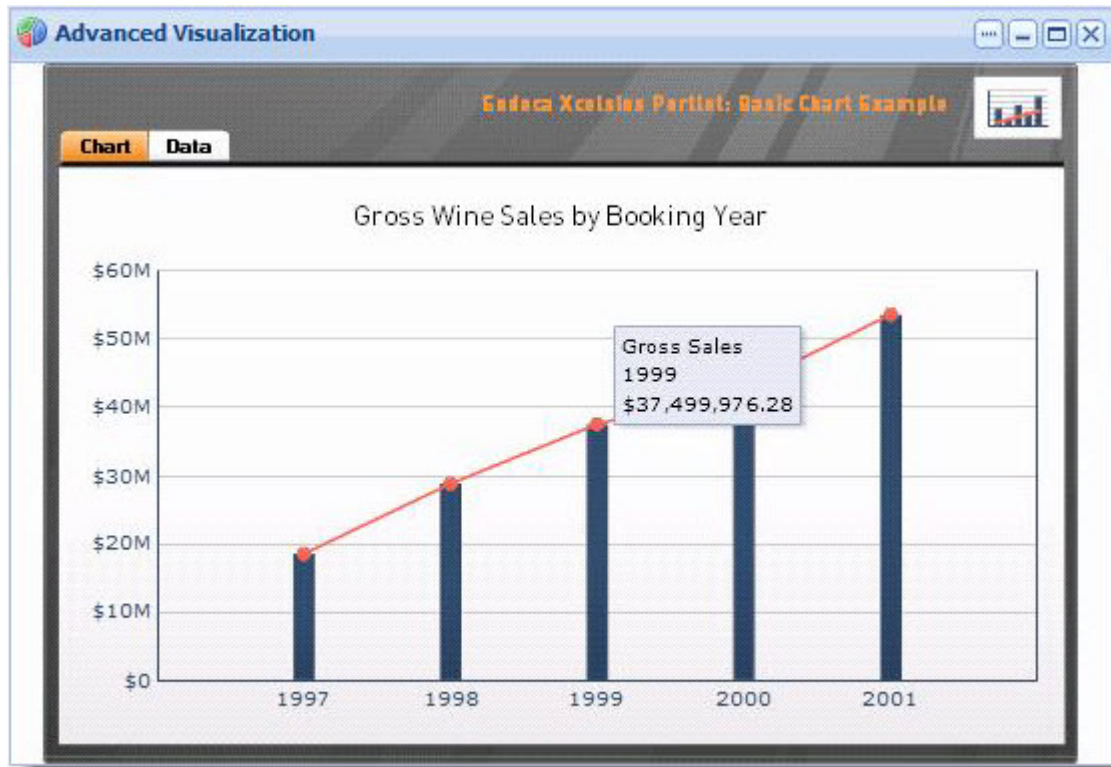
The **Advanced Visualization** component has the following dependencies:

- In order to use this component, you must purchase a license for SAP Crystal Dashboard Design 3.1. Once you have done so, you can download the software, along with a temporary license, from the Discovery Framework downloads page on [EDeN](#). To obtain a permanent license, contact Endeca Support.
- An experienced SAP Crystal Dashboard Design developer needs to create the Flash movies associated with your data. Once he or she has done so, it is easy to add the new movies to the Discovery Framework.
- This component requires the use of Endeca Analytics.

Using Advanced Visualization

This topic describes how an end user can use the **Advanced Visualization** component.

The **Advanced Visualization** component provides SAP Crystal Dashboard Design interactive data visualization of your Discovery Framework data. Many different chart styles are available to your SAP Crystal Dashboard Design developer.



You can mouse over elements of the chart to see the related data. In addition, you can click the **Data** tab to see the underlying data and its state.

The screenshot shows the same 'Advanced Visualization' window, but with the 'Data' tab selected. The 'Chart' tab is now inactive. The 'Raw Data' section displays a table with the following data:

Booking Year	Gross (\$)
1997	\$18,568,161.04
1998	\$28,812,393.85
1999	\$37,499,976.28
2000	\$43,000,000.00
2001	\$53,519,333.05

Below the table is a section titled 'Data Source State' which is currently empty.

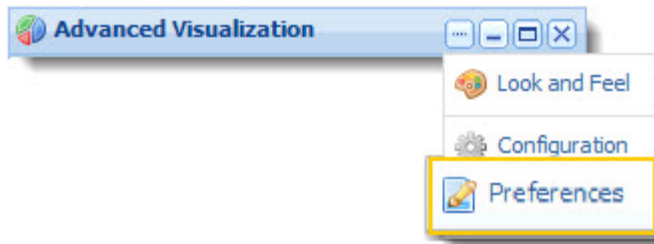
Configuring the Advanced Visualization component

This topic describes the high-level steps involved in configuring the **Advanced Visualization** component.

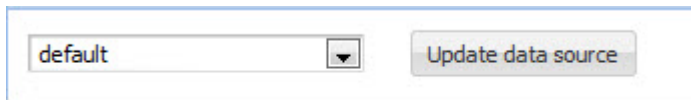
Subsequent topics describe certain individual steps in detail.

To configure the **Advanced Visualization** component:

1. After adding an **Advanced Visualization** component to a page, click its button and then click **Preferences**.



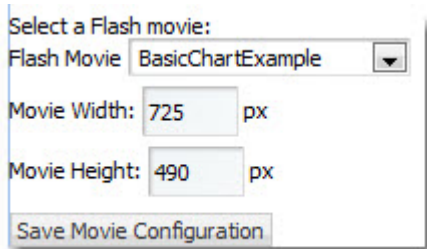
2. Select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.



3. To make sure that you have all the available Flash movies, click **Reload movies** to recognize any newly-added movies.

[Reload movies](#)

4. Use the drop-down list to choose from the available SAP Crystal Dashboard Design movies and set the component size.



Note that the SAP Crystal Dashboard Design movies must be supplied by an SAP Crystal Dashboard Design developer.

5. Click **Save Movie Configuration**.
6. Click **Return to Full Page** to return to the **Advanced Visualization** end-user view.

Chart

The **Chart** component lets you access Corda-based Analytics charting of your data.

The end user can mouse over the chart to reveal numeric data, and click on a chart element to filter results by the values displayed in the chart. In addition, the **Chart** component can be configured so that end users can switch the chart view by selecting drop-down choices. For example, an end user may switch the Metric or Group By that is summarized in the chart.

Dependencies

The **Chart** component has the following dependencies:

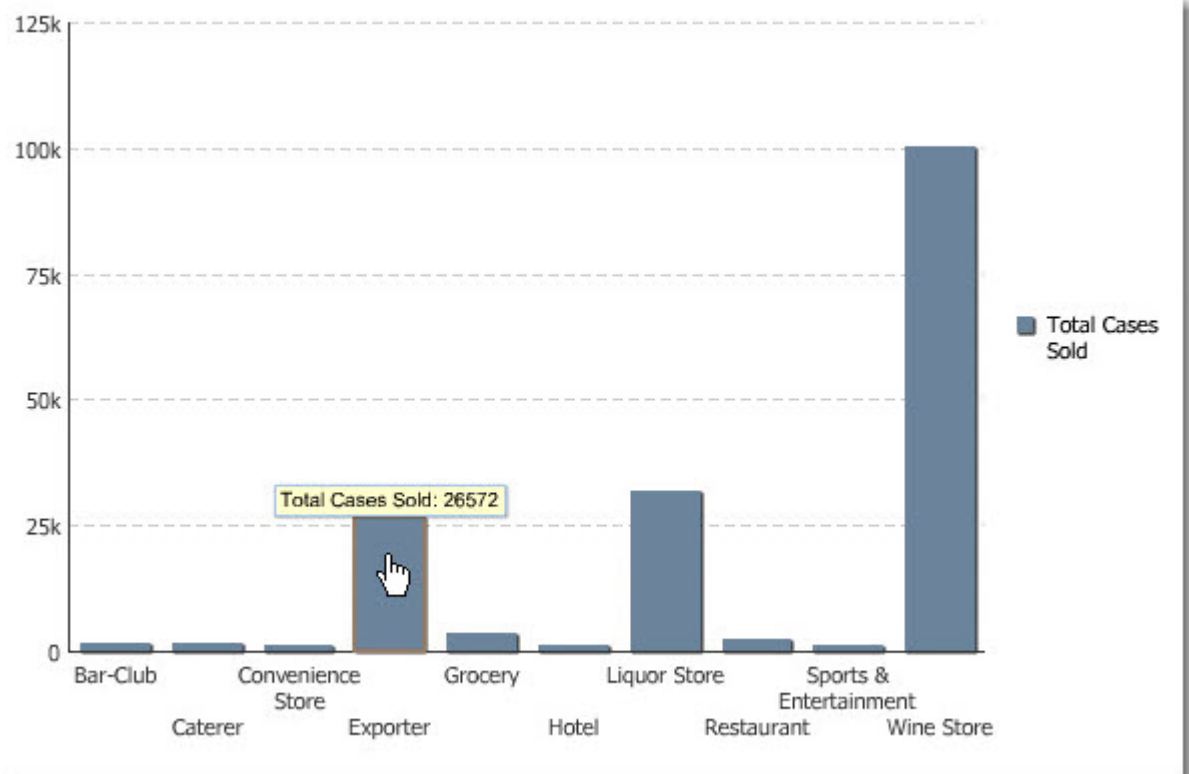
- In order to use this component, you must install Corda. For details, see the *Discovery Framework Installation Guide*.
- This component requires the use of Endeca Analytics.

Using Chart

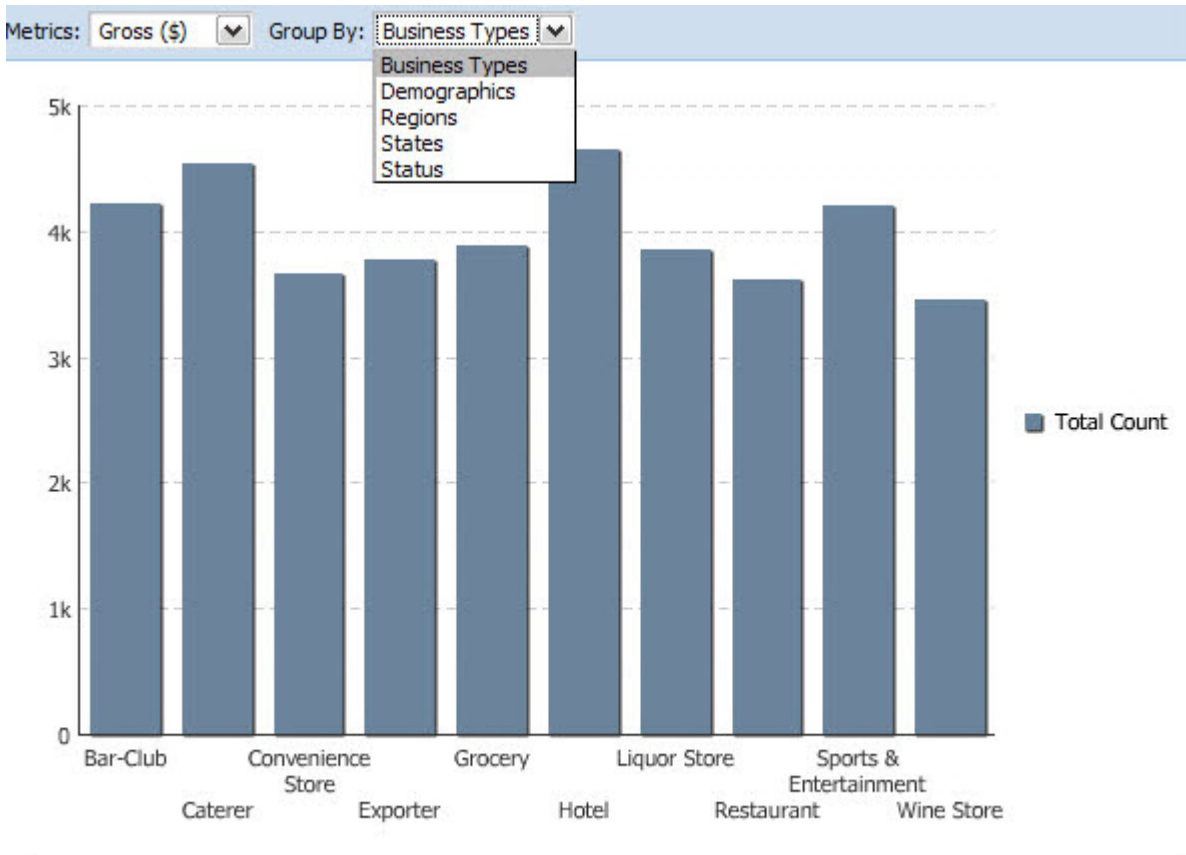
This topic describes how an end user can use the **Chart** component.

The power user who configures your Analytics statements can provide you with two kinds of charts: fixed and parametric. Examples of each are shown below.

The following chart is based on a fixed Analytics statement. You can mouse over chart elements to see the related data, or click to drill down. Drill down is supported for dimension GROUP BY statements but not for property GROUP BY statements.



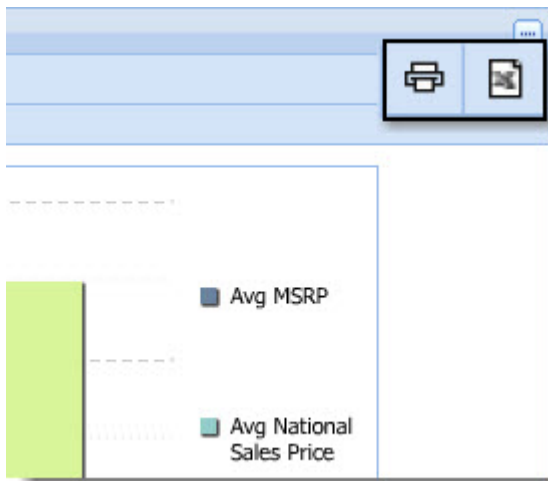
The following chart is based on a parametric Analytics statement. You can modify the chart display by selecting different values from the drop-down lists at the top of the chart.



In addition, the power user has the ability to configure multiple charts, based on different Analytics statements and chart styles, that you can select from a drop down list.

Printing and exporting

End users also have the ability to print and/or export the chart they have created. The end user accesses these features with the **Print** and **Export** buttons, shown below.



When the user selects **Print**, the chart displays in another browser window. The user prints as normal via the **File** menu or keyboard shortcut.

Export allows the user to export the chart to Excel.

Configuring the Chart component

This topic describes how to configure the **Chart** component.

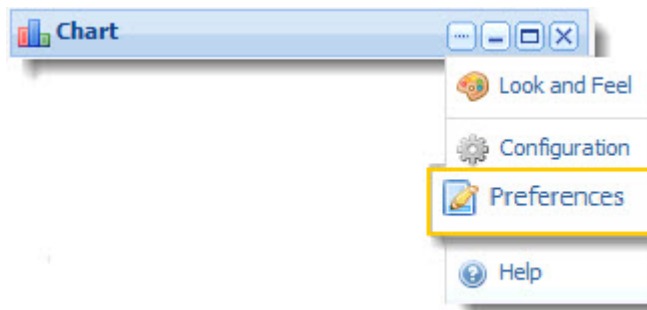
The power user can add a fixed Analytics statement to the **Chart** component, or configure a parametric Analytics statement that allows end users to see drop-down lists across the top of the chart.



Note: Unlike most components, the **Chart** component includes online help. This help, which mostly provides tips on working with Analytics statements, can be accessed from the same menu that contains the **Preferences** option. For more detailed information on the Endeca Analytics language, see the *Endeca Analytics Guide*.

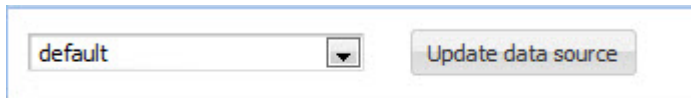
To configure the **Chart** component:

1. After adding a **Chart** component to a page, click its button and then click **Preferences**.



The **Chart** edit view appears.

2. In the **Chart** edit view, select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.



3. In the **Chart Options** pane, set the display options for the chart:

Chart Options					
Chart Height:	<input type="text" value="300"/>	Chart Width:	<input type="text" value="600"/>	Chart Select Label:	<input type="text" value="Select Chart"/>
HTML Height:	<input type="text" value="300"/>	HTML Width:	<input type="text" value="600"/>	Analytics Threshold:	<input type="text" value="1000000"/>
Analytics Threshold Exceeded Message:				<input type="text" value="Too many results to display. Filter results further to view chart"/>	

- a) In the **Chart Height** and **Chart Width** fields, enter the display height and width of the chart in pixels.
- b) In the **HTML Height** and **HTML Width** fields, enter the height and width (in pixels) of the HTML page in which the chart will be displayed. Typically, these HTML values will be the same as the Chart values. Note that the HTML values should never be smaller than the Chart values.
- c) In the **Chart Select Label** field, enter a name for the control that allows end users to choose from multiple charts when available.

- d) In the **Analytics Threshold** field, set a threshold for the number of Analytics results the query returns.
- e) In the **Analytics Threshold Exceeded Message** field, enter a message to display if the query results exceed the Analytics threshold.
4. In the **Chart** tab, enter a title for the chart in the **Chart Title** field and select the type of chart from the **Style** drop-down menu.

Chart Title: **Style:**

5. Enter the Analytics statement on which the chart will be based, then click the **Test analytics query** button to test the validity of the statement (before saving preferences). Note that labels can be configured for metrics only, using the AS functionality in an Analytics statement.

```
RETURN YearlyAverages AS SELECT
AVG("Product MSRP ($)") AS "Avg MSRP",
AVG("Product Avg National Sales Price ($)") AS "Avg National Sales
```

6. The **Metric Options**, **X Axis Options**, and **Cross tab** fields are intended for parametric Analytics charts. These settings provide options for the drop-down lists your end users will access. Cross tabs are grouped aggregations such as cross-tabulated totals over one or more dimensions.

Metric Options (one per line): **X Axis Options (one per line):** **Cross tab (one per line):**

7. If you check the **Enable Top-N Sort** box, the chart's results can be sorted in ascending or descending order. If the **Enable end user control** box is also checked, the end user can toggle the sort order.

Enable Top-N Sort: ☐

Show by

Enable end user control ☒

8. Click **Save Preferences** to save your changes.
9. After you save your preferences, the **Metric Selector** pane is shown. This pane allows you to specify an ORDER BY in the Analytics statement without including it in the display. In the example below, if you did not want to include the statement's ORDER BY "Avg Sales Price" metric in the chart, you would un-check that metric in the list. Note that the **Metric Selector** does not appear until you have tested and saved the Analytics query.

Metric Selector

☒ **Avg MSRP**

☒ **Avg National Sales Price**

☒ **Avg Sales Price**

10. If you have made any other changes (such as in the **Metric Selector**), click **Save Preferences** to save these additional changes.

11. Click **Return to Full Page** to return to the **Chart** end-user view.

Chart styles

The following chart styles are supported by the **Chart** component.

- Vertical Bar
- Stacked Vertical Bar
- Horizontal Bar
- Line
- Pie
- Line Bar Combo

Examples of each of these chart styles appear in the following topics.

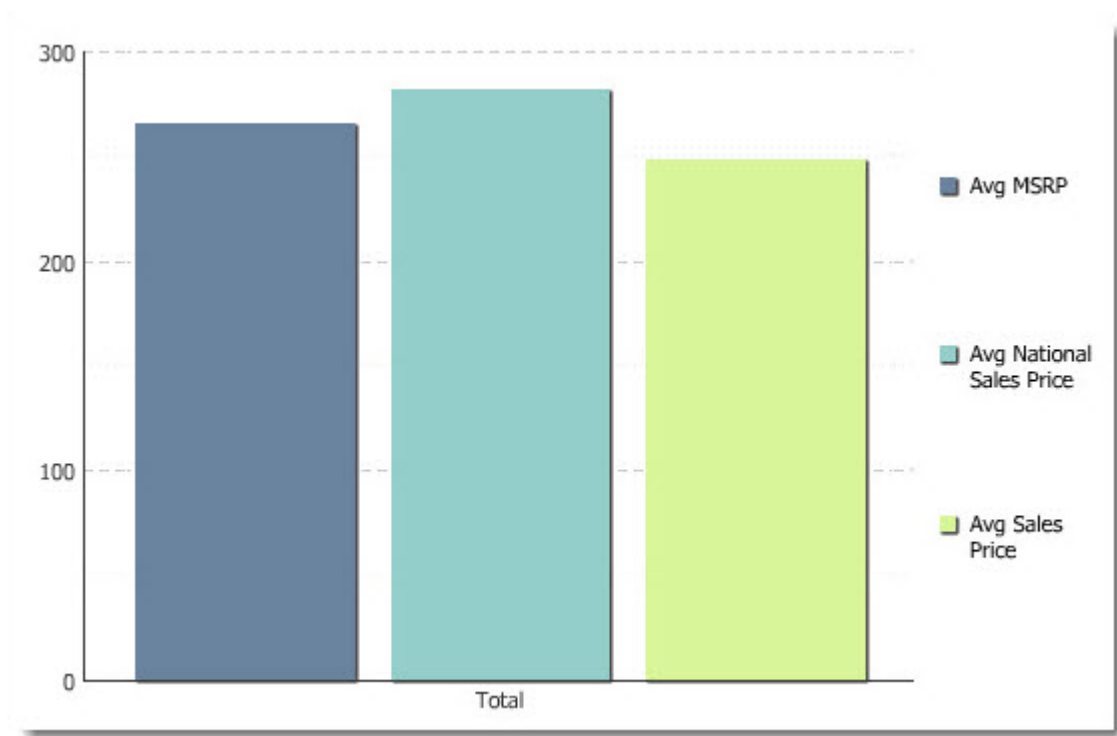
Vertical bar chart

This topic contains an example of a vertical bar chart.

The following example uses a zero GROUP BY statement with three metrics:

```
RETURN YearlyAverages AS SELECT
    AVG("Product MSRP ($)") AS "Avg MSRP",
    AVG("Product Avg National Sales Price ($)") AS "Avg National Sales
Price",
    AVG("Unit Sale Price ($)") AS "Avg Sales Price"
GROUP
```

It produces a vertical bar chart displaying “Avg MSRP”, “Avg National Sales Price”, and “Avg Sales Price” metrics:



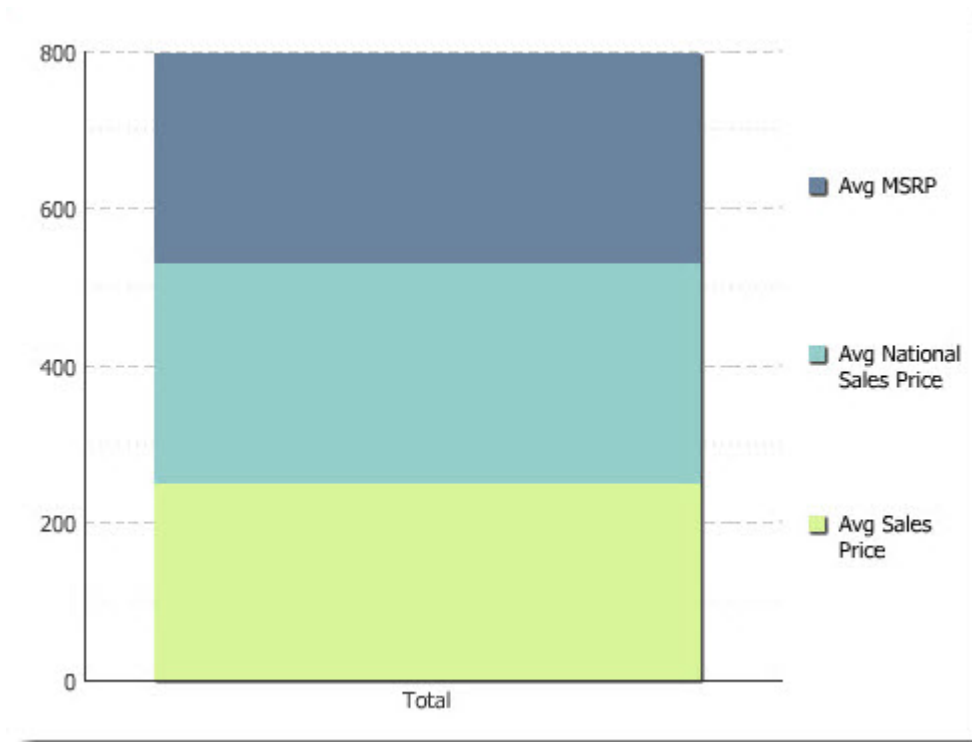
Stacked vertical bar chart

This topic contains an example of a stacked vertical bar chart.

This example uses an Analytics statement with two GROUP BY statements. One metric is summarized by a specified categorization, and the second GROUP BY is specified as the cross tab.

```
RETURN RegionalMargin AS SELECT
  SUM("Gross ($)" * "Margin (%)") / SUM("Gross ($)") AS "DollarWeightedAvgMargin"
GROUP BY "Regions", "Demographics"
```

The stacked vertical bar chart below summarizes the “DollarWeightedAvgMargin” metric for each Region (which is the first GROUP BY and therefore the primary category) and presents a cross-tabulation with each Demographic (which is the second GROUP BY and therefore the cross tab):



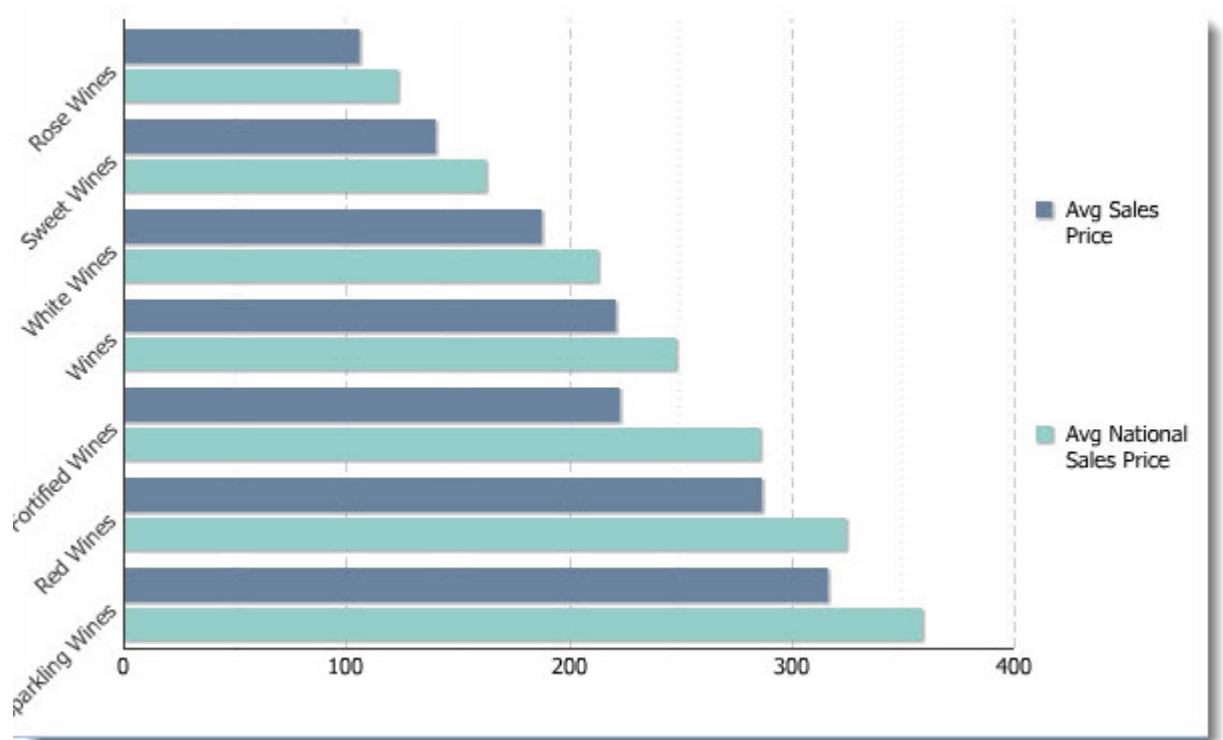
Horizontal bar chart

This topic contains an example of a horizontal bar chart.

This example uses a single GROUP BY statement with two metrics:

```
RETURN NumCases AS SELECT
  AVG("Unit Sale Price ($)") AS "Avg Sales Price",
  AVG("Product Avg National Sales Price ($)") AS "Avg National Sales Price"
GROUP BY "Varietals" ORDER BY "Avg National Sales Price"
```

The horizontal bar chart compares the "Avg Sales Price" metric to the "Avg National Sales Price" for each varietal:



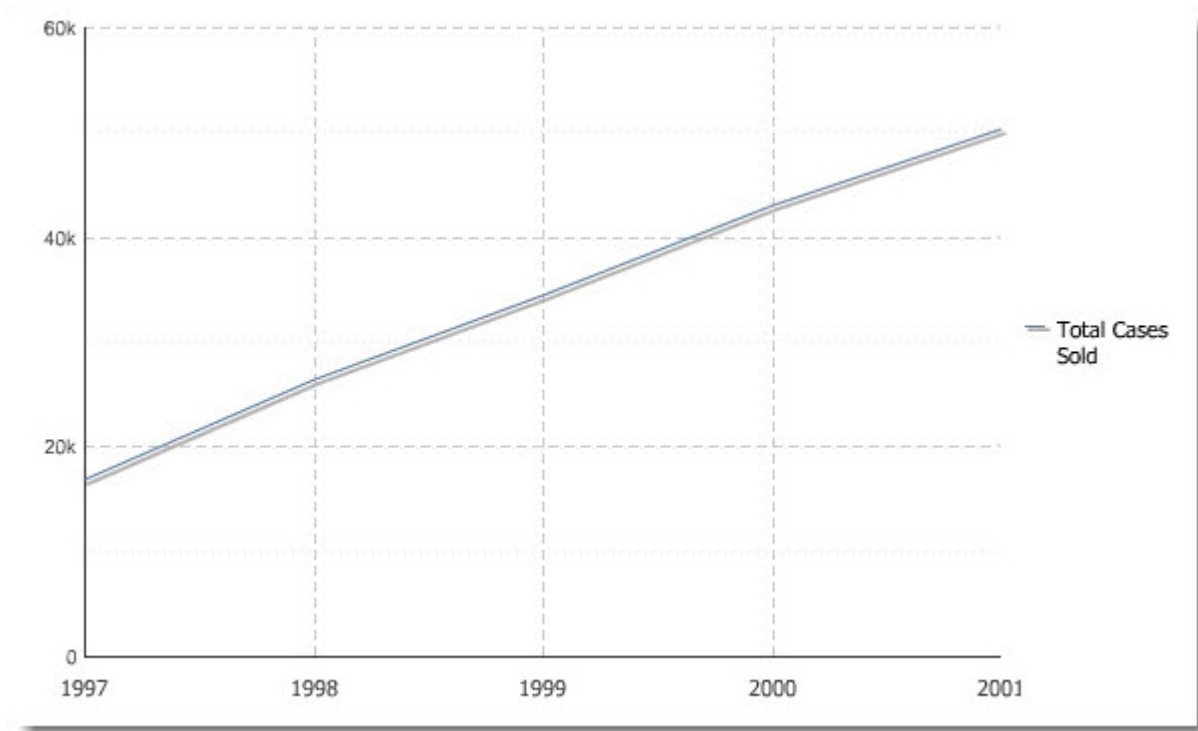
Line chart

This topic contains an example of a line chart.

The following Analytics statement has one GROUP BY and a single metric:

```
RETURN NumCases AS SELECT
  SUM("Number of Cases Sold") AS "Total Cases Sold"
GROUP BY "Booking Year"
```

It produces a line chart summarizing the "Total Cases Sold" metric for each Booking Year:



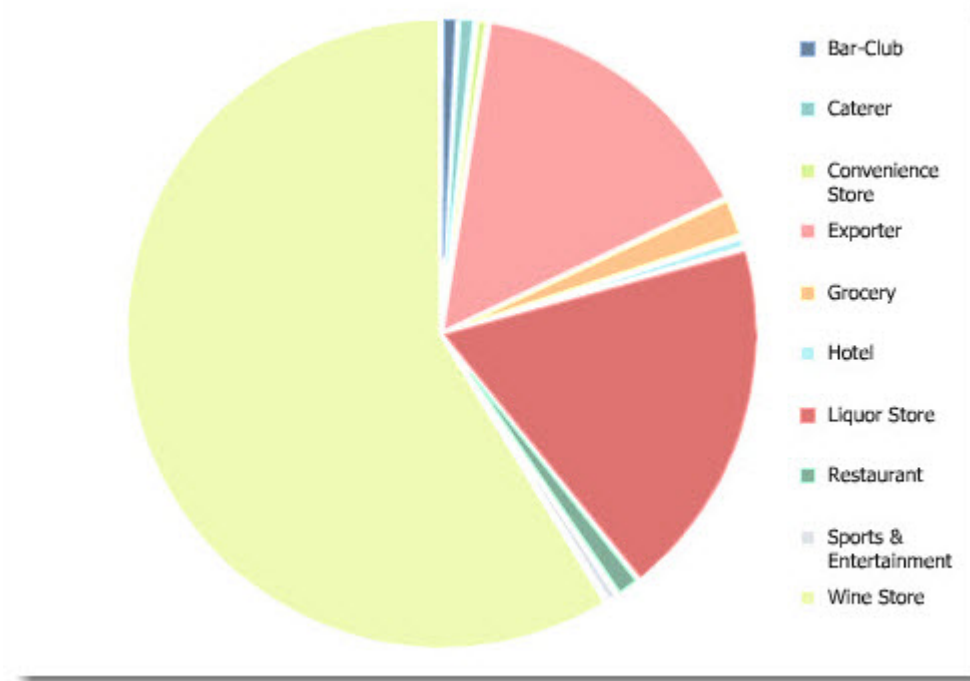
Pie chart

This topic contains an example of the pie chart type.

The following Analytics statement contains one GROUP BY and a single metric:

```
RETURN NumCases AS SELECT  
    SUM("Number of Cases Sold") AS "Total Cases Sold"  
GROUP BY "Business Types"
```

It produces a pie chart summarizing the "Total Cases Sold" metric for each Business Type:



Important: The pie chart type only supports a single GROUP BY statement. If you have multiple GROUP BYs in your statements, the chart will not display or execute drill-down as expected.

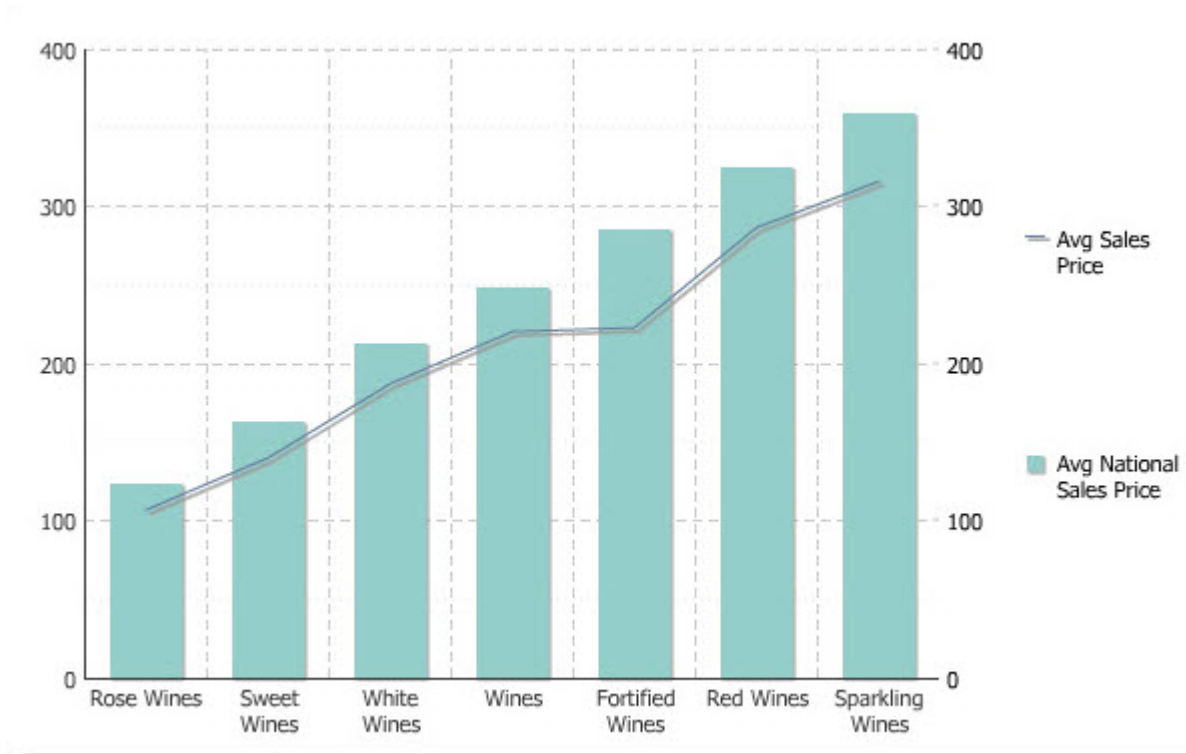
Line bar combo chart

This topic contains an example of a line bar combo chart.

The following example contains one GROUP BY statement and two metrics:

```
RETURN NumCases AS SELECT
    AVG("Unit Sale Price ($)") AS "Avg Sales Price",
    AVG("Product Avg National Sales Price ($)") AS "Avg National Sales Price"
GROUP BY "Varietals" ORDER BY "Avg National Sales Price"
```

It produces a line bar combo chart comparing the "Avg Sales Price" metric to the "Avg National Sales Price" for each varietal:



Cross Tab

The **Cross Tab** component provides a table that allows end users to perform comparisons and identify trends across several cross sections of data.

A cross tab consists of a specific configuration of rows, columns, and summary cells that together make up the table body. The values displayed in the header rows and columns represent every possible grouping value of the specified data fields. Summary cells exist at the intersection of rows and columns. The value of a given summary cell is a metric corresponding to the intersection of those two groupings.

Cross Tab												
	1997											
	First Quarter			Fourth Quarter			Second Quarter			Third Quarter		
	# Trans	avgMargin	avgGross	# Trans	avgMargin	avgGross	# Trans	avgMargin	avgGross	# Trans	avgMargin	avgGross
Austin Fyles	53	\$129.19	\$3,096.79	61	\$133.57	\$3,489.57	55	\$122.25	\$2,977.60	35	\$155.20	\$3,294.18
Jesenia Milman	71	\$133.95	\$3,246.36	95	\$125.46	\$3,386.29	80	\$118.19	\$2,886.56	60	\$126.33	\$3,144.19
Katharina Cafferky	44	\$149.95	\$3,728.25	60	\$140.72	\$4,545.75	52	\$157.50	\$4,084.33	27	\$120.89	\$3,408.31
Leida Gallander	55	\$123.77	\$3,368.03	88	\$127.51	\$3,042.26	65	\$119.46	\$3,379.75	52	\$110.20	\$2,996.39
Olevia Giardina	41	\$114.20	\$3,701.36	59	\$117.83	\$3,857.88	37	\$139.58	\$3,940.03	31	\$108.24	\$3,914.05
Total	264	\$130.47	\$3,392.61	363	\$128.60	\$3,588.54	289	\$129.06	\$3,365.21	205	\$123.71	\$3,283.51



Note: This component requires the use of Endeca Analytics.

Using Cross Tab

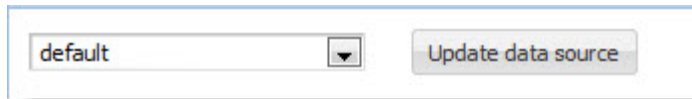
The end user views the cross tab result table, but cannot edit or modify it.

Configuring the Cross Tab component

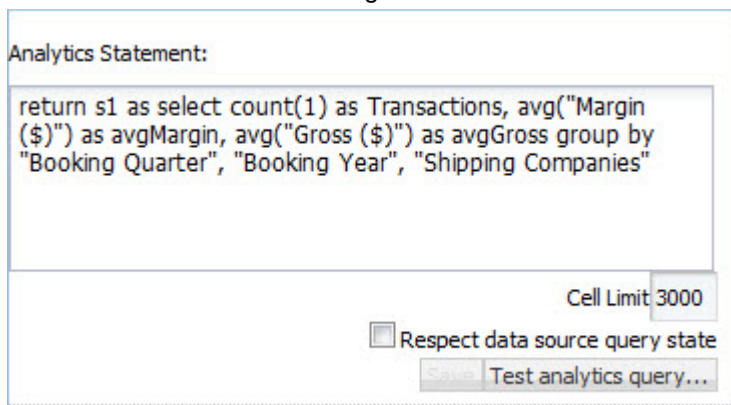
This topic describes how to configure the **Cross Tab** component.

To configure the **Cross Tab** component:

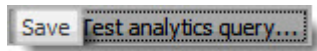
1. After adding a **Cross Tab** component to a page, click its button and then click **Preferences**.
2. In the **Cross Tab** edit view, select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.



3. In the **Analytics Statement** field, enter an Analytics statement to return cross-tab values. Note that using fields with too many values in your GROUP BY clauses may cause your **Cross Tab** results table to become too large to be useful.



4. In the **Cell Limit** field, set a threshold to prevent too large a grid from being rendered on the **Cross Tab** page. In general, you should not make this value, which represents the number of cells rendered in the **Cross Tab** results table, much larger than the default. A table with many thousands of cells is time-consuming to render and difficult for the end user to read.
5. If desired, check the **Respect data source query state** box. When this box is checked, the **Cross Tab** component is dependent on its data source query state, which means the metric values should update when the data source query state changes. When the box is unchecked, it is independent of the data source query state, so the metric values should remain constant regardless of the data source query state.
☒ Respect data source query state
6. Click the **Test analytics query** button to validate the Analytics statement before loading the metrics into the component. You are required to test the Analytics statement before loading the statement into the component.
 If the Analytics statement is valid, the **Save** button is enabled.
7. After you test and debug your Analytics query, click the **Save** button.

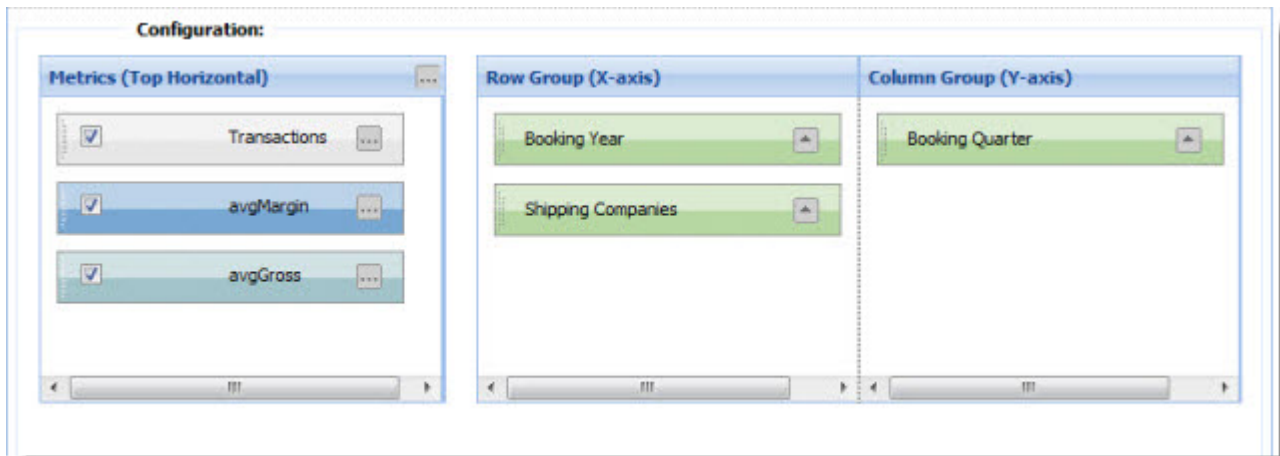


When you save, the **Configuration** section is populated with metric, row, and column information. If you wish, you can further configure the **Configuration** elements.

Adjusting the Cross Tab table

You can fine-tune the appearance of the **Cross Tab** results table by using the options in the **Configuration** section of the component's edit view.

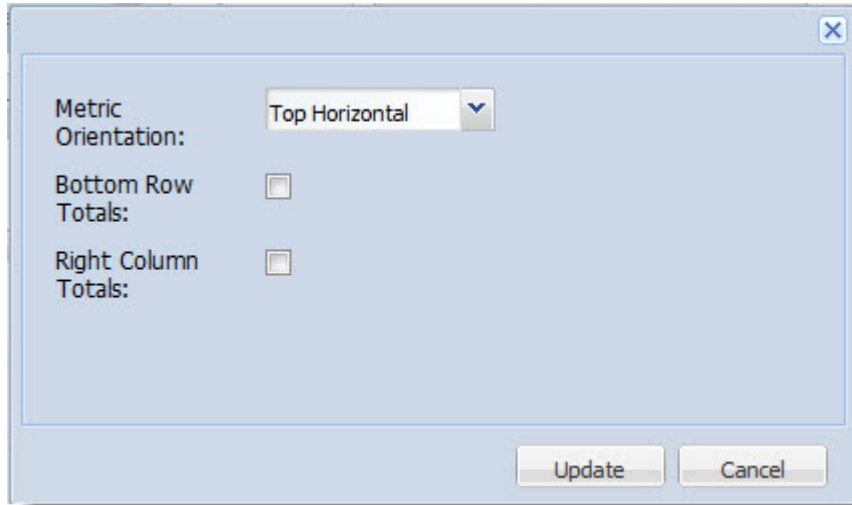
After you have initially configured the **Cross Tab** table, you can use the **Configuration** section to make adjustments to the appearance of the table.



As you make changes, you can see their effects on the table by checking the preview table at the bottom of the edit view.

To make adjustments to the **Cross Tab** table:

1. Access the **Cross Tab** edit view for your **Cross Tab** component.
The **Configuration** section is displayed, as in the example above.
2. In the **Metrics** section, you can configure general metric options by clicking the ... button in the top right corner of the panel to launch its formatting options:



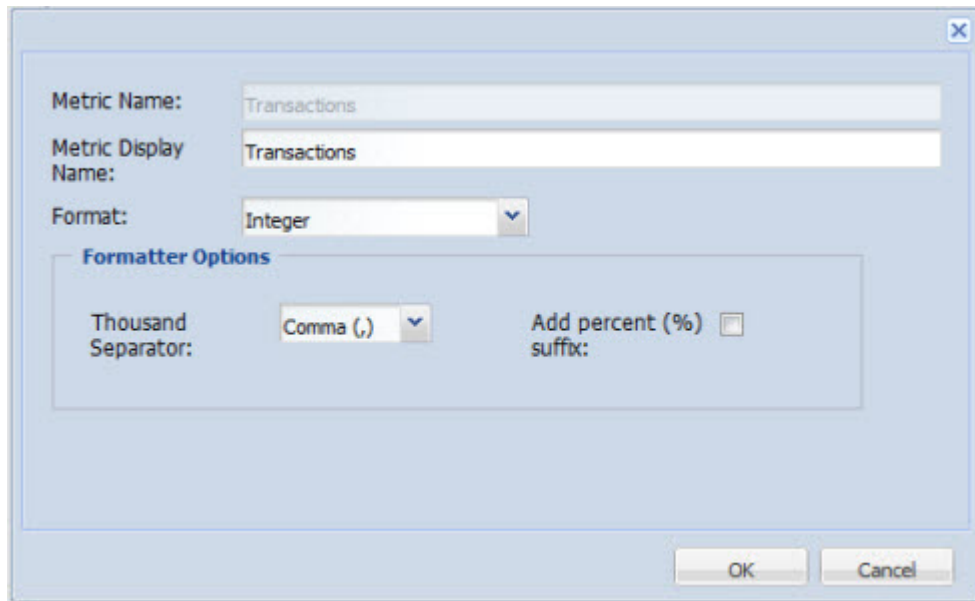
A dialog box titled "Metric Orientation" with a close button (X) in the top right corner. It contains three settings: "Metric Orientation:" with a dropdown menu set to "Top Horizontal", "Bottom Row Totals:" with an unchecked checkbox, and "Right Column Totals:" with an unchecked checkbox. At the bottom right are "Update" and "Cancel" buttons.

- **Metric Orientation** lets you change the axis on which metrics are displayed.
- **Bottom Row Totals** and **Right Column Totals** let you add metric totals to the table.



Note: Totals are computed using the same aggregating function as the original metric. For example, metrics computed using the average function (AVG) display an overall average for the Total. Because totals may be confusing in cases where the ARB function is used, you may not want to display totals in such cases.

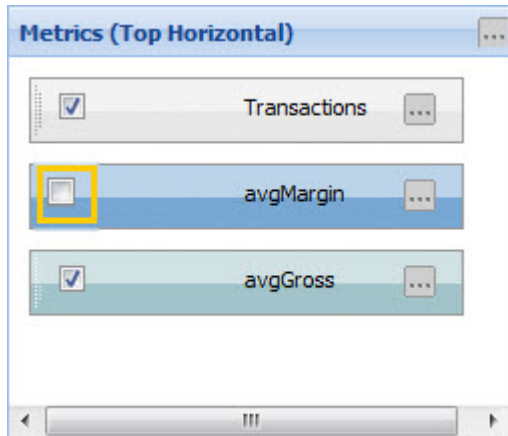
- Also in the **Metrics** section, you can configure individual metric by clicking the ... button on its label to access its formatting options:



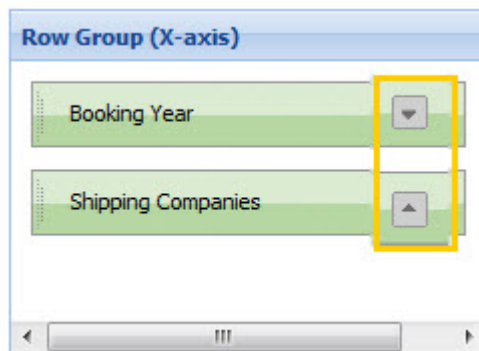
A dialog box titled "Metric Formatting" with a close button (X) in the top right corner. It contains the following fields: "Metric Name:" with a text box containing "Transactions", "Metric Display Name:" with a text box containing "Transactions", and "Format:" with a dropdown menu set to "Integer". Below these is a section titled "Formatter Options" which contains "Thousand Separator:" with a dropdown menu set to "Comma (,)" and "Add percent (%) suffix:" with an unchecked checkbox. At the bottom right are "OK" and "Cancel" buttons.

- **Metric Display Name** lets you specify the display name for the metric.
- **Format** specifies in what format the value is displayed in the table: Integer, Currency, or Decimal. The **Formatter Options** section provides output formatting options that are appropriate to the format type.

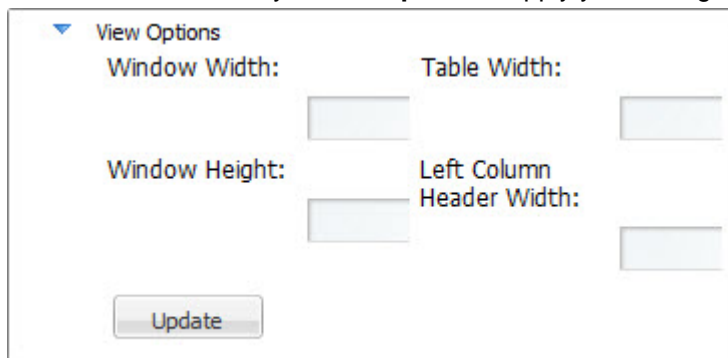
4. Another **Metrics** section feature lets you drag metrics to reorder them. You can also remove a metric from the **Cross Tab** result table display by unchecking it. Note that to improve performance, if you do not want to display a metric, it is better to remove it from the Analytics statement than it is to uncheck it.



5. In the **Row Group** and **Column Group** sections, you can drag items anywhere in the row and column panels. Each GROUP BY item contains a toggle that can be used to set ascending or descending order.



6. The **View Options** dialog lets you modify the dimensions (in pixels) of the **Cross Tab** table and its window. Make sure you click **Update** to apply your changes to the table.



Compare

The **Compare** component allows end users to view two or more records side-by-side in order to analyze differences and similarities between them.

The details are presented in a tabular format for easy comparison.

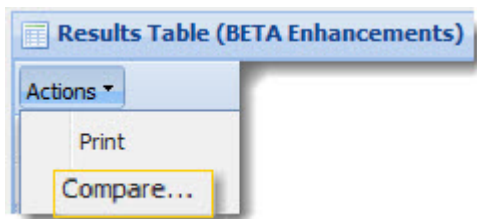


Note: The **Compare** component is accessed from either the **Results Table** component or the **Results Table (BETA Enhancements)** component.

Using Compare

This topic describes how an end user can use the **Compare** component.


In either the **Results Table** component or the **Results Table (BETA Enhancements)** component, the end user can select two or more records to compare.




Within the **Compare** component, the end user can view and compare details for the selected records.

	Record1	Record2	Record3	Record4	Record5
Ratings (2 attributes)					
Price Range	\$10 to \$20	\$20 to \$30	Under \$10	\$30 to \$40	\$10 to \$20
Review Score	20 to 30	30 to 40	80 to 90	80 to 90	70 to 80
Characteristics (3 attributes)					
Body	Crisp	Ripe	Rich, Smooth	Tight	Fresh, Soft
Flavors	Apple, Dry, Fruit,...	Citrus, Fig, Honey, ...	Apple	Apple, Fruit, Grape,...	Earthy, Melon
Drinkability	Drink now	Drink now			
Other (18 attributes)					
P_Winery	Adegas Morgadio	Hidden Cellars	Hugel	Jaffeln	Parxet
P_Price	18.000000	20.000000	9.000000	34.000000	11.000000
P_Name	Albarino Rias Bai...	Alchemy Mendocino ...	Alsace Gentil	Meursault Les Cras	Alella Marques de Al
P_DateReviewed	06/15/93	02/28/95	11/15/94	08/31/94	04/15/94
Wine Type	White	White	White	White	White
Winery	Adegas Morgadio	Hidden Cellars	Hugel	Jaffeln	Parxet
ADG Refinement Center	White	Ripe, White	White	White	Soft, White
P_Description	Dry, tart and cris...				

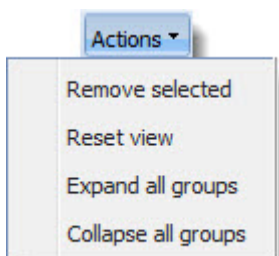
Comparing 4 record(s) to baseline Record1 0 Record(s) Currently Selected

The end user can set one record as the baseline by clicking . This locks the record's column so that several alternate records can be compared against it. Non-baseline records can be dragged left or right in the display, allowing direct, side-by-side comparison of any two records.

The end user can also drill down to details about any record by clicking . This action launches the **Record Details** component for that record.

In addition, the end user can drag and drop attribute sets to move them up or down in the display, enabling direct, side-by-side comparison of attributes in the sets. Attributes can also be reordered within their sets (though they cannot be dragged to other sets).

From the **Actions** menu, the end user can remove a selected record, reset the view to the default, or expand or collapse attribute groups.



With the **Highlight Differences** setting, the end user has control over highlighting in the display. The end user can choose to highlight all differences between records, or may choose a baseline record against which differences are calculated and highlighted, as shown in the image below:

P_DateReviewed	06/15/93	02/28/95	11/15/94	08/31/94	04/15/94
Wine Type	White	White	White	White	White
Winery	Adegas Morgadio	Hidden Cellars	Hugel	Jaffelin	Parxet
ADG Refinement Center	White	Ripe, White	White	White	Soft, White

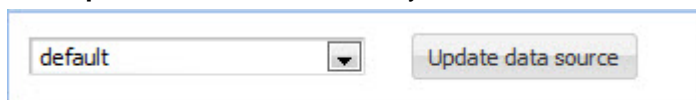
Configuring the Compare component

This topic describes the high-level steps involved in configuring the **Compare** component.

Subsequent topics describe certain individual steps in detail.

To configure the **Compare** component:

1. After adding a **Compare** component to a page, click its button and then click **Preferences**.
2. In the **Compare** edit view, select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.

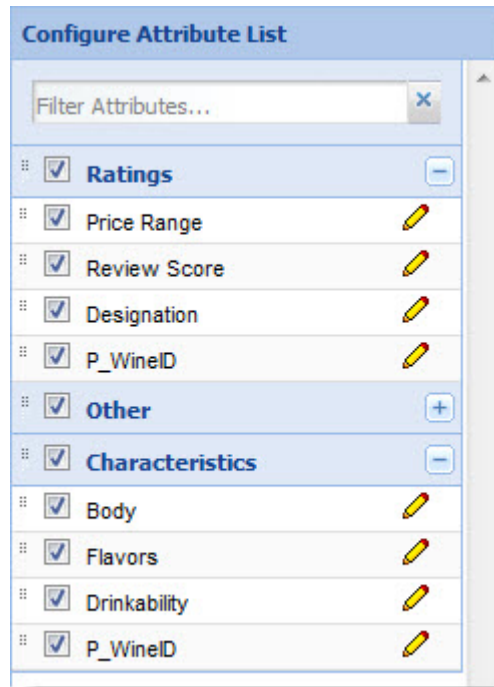



3. Configure the **Attribute List**, as described in a following topic.
4. Configure the **Grid Controls** and the **Component Controls**, as described in a following topic.
5. Click **Save Preferences** to save your changes.

Configuring the Attribute List

The **Attribute List** defines the attributes that are displayed to the end user in the **Record Details** and **Compare** tables.

The **Configure Attribute List** dialog is where you configure the attributes shown with the record in the **Record Details** table and in the **Compare** table:



The  pencil icon (next to the attribute name) is used to access the **Edit Attribute Display** dialog for that attribute.

To configure the **Attribute List**:

1. Access the **Record Details** or **Compare** edit view.
The **Configure Attribute List** dialog is one of the displayed dialogs.
2. If you need to find a specific attribute, use the **Filter Attributes** search box to search for an attribute by name (with type-ahead).
3. In the Attribute List, check the attributes and attribute sets that you want displayed in the **Record Details** or **Compare** table. Unchecked attributes or attribute sets are not displayed in the table.
4. If you want to change the formatting of an attribute, click the pencil icon for the attribute to open the **Edit Attribute Display** dialog.
See the next topic for details on formatting options.
5. Click **Save Preferences** to save your changes.

Edit Attribute Display dialog

Use this dialog box to set the output formatting for an attribute.

The **Edit Attribute Display** dialog, shown below, lets you change the default format in which the values of some attributes are output in the **Record Details** and **Compare** tables. For example, you

can specify that monetary values are prepended with dollar signs or appended percent signs to integers or decimal numerals.

Edit Attribute Display

Include attribute: ☒

Format: Integer

Formatter Options

Thousand Separator: Comma (,)

Add percent (%) suffix: ☐

Cancel OK

When changing the default formatting, you must make sure that the new formatting changes are appropriate to the data type of the attribute. For example, do not attempt to configure **Currency** formats to a string attribute. If the new formatting changes are invalid for the chosen attribute, the Discovery Framework software will revert the formatting to the defaults for the attribute's data type.

Format	Purpose	Formatter Options
Integer	Formats the output of integer values.	<ul style="list-style-type: none"> • Thousand Separator sets the thousands delimiter to a comma, period, space, or apostrophe. For example, for a value of "15000", specifying the Comma option would format the output as "15,000" while the Period option would produce a "15.000" output. • Add percent suffix appends a % character to the numeric output. For example, a rating score of "25" can be shown as "25%".
Currency	Formats the output of monetary values, such as floating point values.	<ul style="list-style-type: none"> • Thousand Separator sets the thousands delimiter to a comma, period, space, or apostrophe. • Decimal Separator sets a comma or a period as the mark between the integral and the fractional parts of a decimal numeral. • Decimal Places lets you specify the number of decimal places of a decimal numeral. • Currency Symbol either prepends or appends a dollar, pound, euro, or yen monetary symbol to the output.

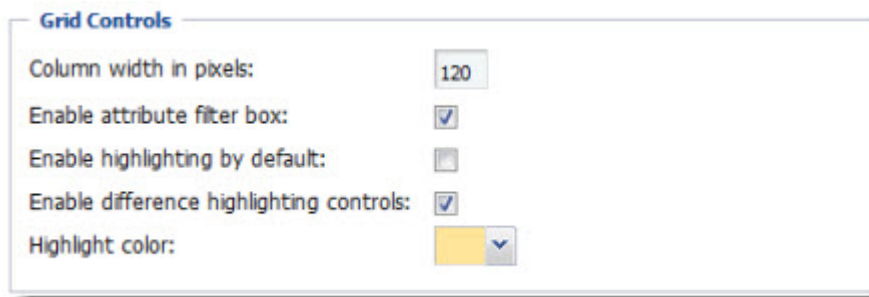
Format	Purpose	Formatter Options
Decimal	Formats the output of decimal values, such as floating point values.	<ul style="list-style-type: none"> • Thousand Separator sets the thousands delimiter to a comma, period, space, or apostrophe. • Decimal Separator sets a comma or a period as the mark between the integral and the fractional parts of a decimal numeral. • Decimal Places lets you specify the number of decimal places of a decimal numeral. • Add percent suffix appends a % character to the numeric output.
String	Formats the output of string values.	<ul style="list-style-type: none"> • Default makes no change to the string value stored in the record, which means that the original casing is preserved. • Lower Case changes all the characters to lower case. • Upper Case changes all the characters to upper case. • Title Case changes the first character of each word to lower case.

Configuring the Grid Controls and Component Controls

You can customize the appearance and behavior of the Compare table.

To configure the Grid Controls and Component Controls:

1. Access the **Compare** edit view.
2. In the **Grid Controls** panel, set the appearance and behavior of the table the end user sees:



- **Column width in pixels:** This column width applies to all record columns.
- **Enable attribute filter box:** Enabled by default. Disabling this setting causes the filter and search box to disappear from the end user view.
- **Enable highlighting by default:** Disabled by default. Enabling this setting causes the **Compare** component to highlight differences between records when they are first loaded. Because no record is in the baseline position when records are first loaded, the highlight is in "highlight with no baseline record" mode.
- **Enable difference highlighting controls:** Enabled by default. Disabling this setting causes the **Highlight differences** button to disappear from the end-user view.
- **Highlight color:** Changes the color used for the highlighting feature.

3. In the **Component Controls** panel, set controls that the end user will see and (optionally) a transition page for the Record Details component:

Component Controls

Display Name Property:
 (eg. "12345" in the title "Record 12345"):

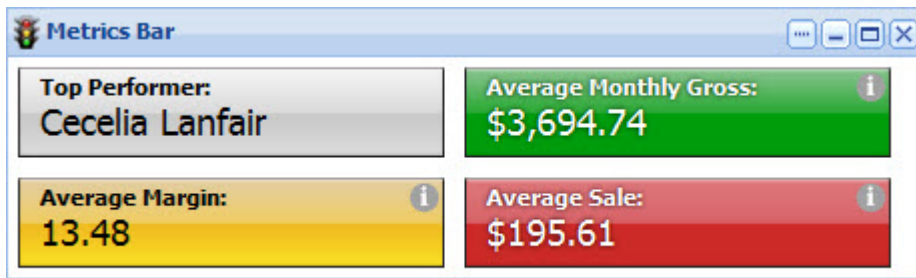
Target page for record details:

Disable back button: ☒

- **Display name property:** Set the property to be used as the display name in the end user's view of the record.
 - **Target page for record details:** Specify a separate page to display record details.
 - **Disable back button:** If checked, hides the **Back** button from the end-user view.
4. Click **Save Preferences** to save your changes.

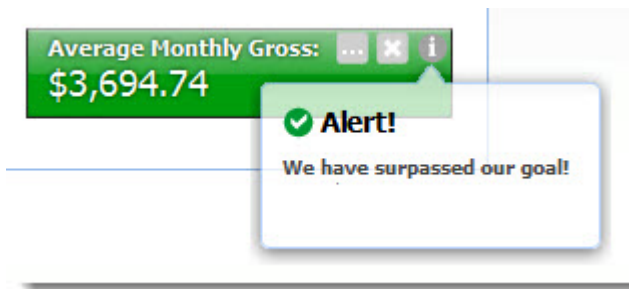
Metrics Bar

The **Metrics Bar** component allows users to quickly view metrics that summarize various aspects of the underlying data.



The **Metrics Bar** component displays metrics based on Analytics statements written by the power user.

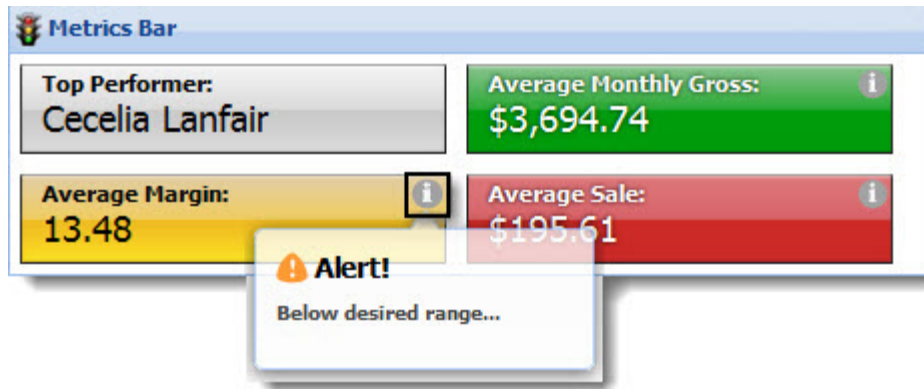
The power user is able to change the display format of the returned metric value. He or she can also set threshold violation indicators to alert the end users when a metric has reached a certain value or range of values (shown below):



Using Metrics Bar

This topic describes how an end user can use the **Metrics Bar** component.

The following image shows a **Metrics Bar** component with four metric values on display. The end user can view the metric values and mouse over the **i** symbol (highlighted below) on each metric box to view any alerts or threshold violation indicators.

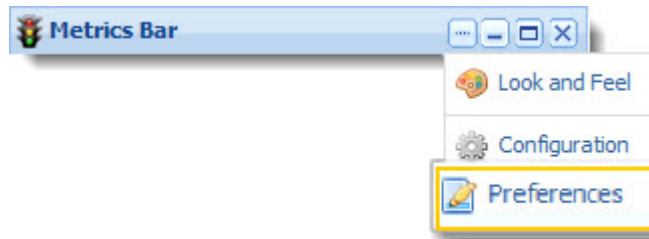


Configuring the Metrics Bar component

This topic describes how to configure the **Metrics Bar** component.

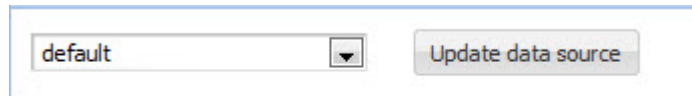
To configure the **Metrics Bar** component:

1. After adding a **Metrics Bar** component to a page, click its button and then click **Preferences**.



The **Metrics Bar** edit view appears.

2. In the **Metrics Bar** edit view, select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.



3. In the **Analytics Statement** field, enter an Analytics statement (or multiple statements) to return metric values.

Analytics Statement:

```
return s as select avg("Gross ($)") as AvgGross, avg("Margin (%)") as
AvgMarginPct group;
return s2 as select "Sales Reps (Prop)" as TopSalesRep, avg("Margin ($)") as
AvgMargin group by "Sales Reps (Prop)" order by AvgMargin desc page(0,1)
```

4. If desired, check the **Respect data source query state** box. When this box is checked, the **Metrics Bar** component is dependent on its data source query state, which means the metric values should update when the data source query state changes. When the box is unchecked, it is independent of the data source query state, so the metric values should remain constant regardless of the data source query state.

☒ Respect data source query state

5. Click the **Test analytics query** button to validate the Analytics statement before loading the metrics into the component. You are required to test the Analytics statement before loading the statement into the component.
If the Analytics statement is valid, the **Load Metrics** button is enabled.
6. Click **Load Metrics**.
7. Click **Return to Full Page** to return to the **Metrics Bar** end-user view.

Tag Cloud

The **Tag Cloud** component provides a visualization that lists textual attribute values (or tags) and their significance in the current result set. It can be configured to allow end users to filter results to only those containing a particular tag.

The visualization lists the various tags that exist for a given attribute across a result set. The relative significance of each tag is indicated by automatically-assigned tag font sizes.



Note: The **Tag Cloud** component supports base records and aggregate records. It does not support Analytics records.



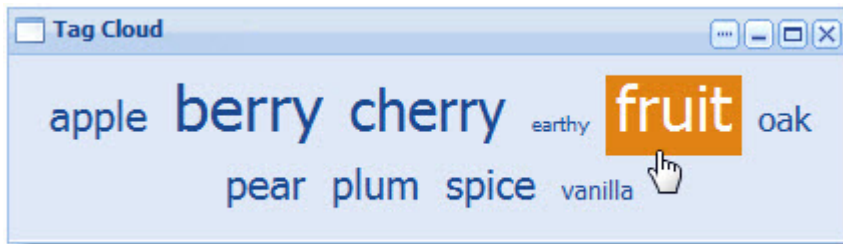
Note: In order to provide a meaningful summary, the **Tag Cloud** component must be associated with an attribute that consists of numerous textual attribute values.

Using Tag Cloud

This topic describes how an end user can use the **Tag Cloud** component.

The component can be configured so that the end user can click on a tag to refine the result set to the records associated with that textual attribute value.

In the following example, the end user clicks on the **fruit** tag:



This issues a navigation refinement, which updates the **Breadcrumbs** component, along with any other components that uses that data source. The image below shows **Tag Cloud** and a related **Results Table** component after the **fruit** tag was clicked:

Actions ▼		Choose a column set: Tab 1 Tab 2	
<input type="checkbox"/>		P_Winery	P_Flavor
<input type="checkbox"/>		Benziger	Fig Fruit Lemon Toasty
<input type="checkbox"/>		Adegas Morgadio	Apple Dry Fruit Grape Grapefr
<input type="checkbox"/>		Jaffelin	Apple Fruit Grape Mineral Pine
<input type="checkbox"/>		Quinta de Abrigada	Black Cherry Cherry Cinnamor
<input type="checkbox"/>		Zind-Humbrecht	Fruit Fruity Orange Pear Sweet
<input type="checkbox"/>		Calistoga Vineyards	Apple Cedar Fruit Fruity Melor
<input type="checkbox"/>		Nicolis	Caramel Fruit Nut Prune Sweet
<input type="checkbox"/>		Pierre Gimmonnet & Fils	Fruit
<input type="checkbox"/>		Sequoia Grove	Citrus Earthy Fruit Grape Grape
<input type="checkbox"/>		Caparzo	Berry Dried Fruit

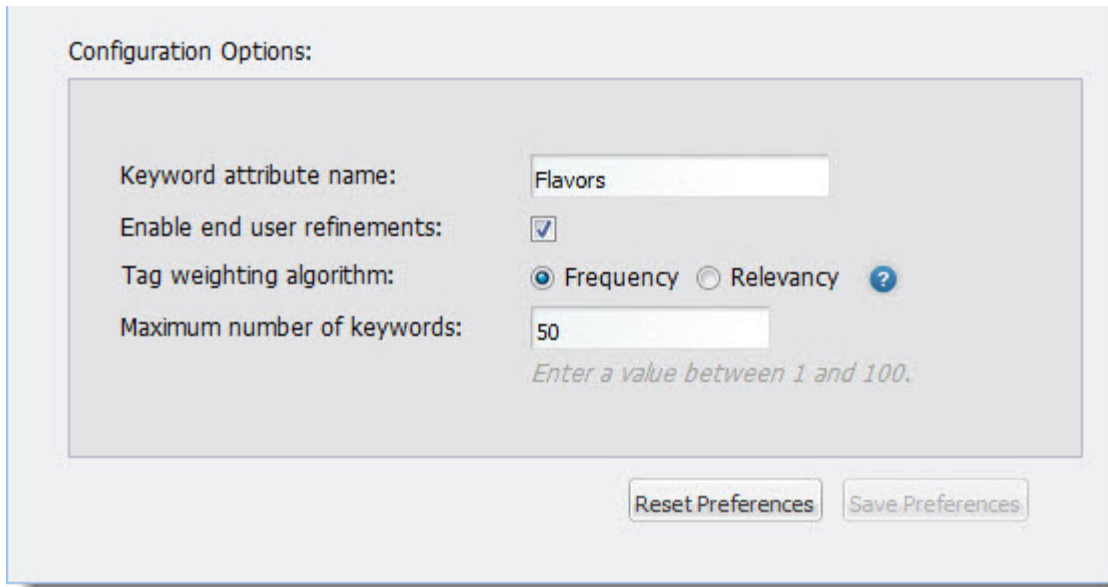
Page 1 of 1445 | Records per page ▼ | Dis

As you can see, the relative size of tags also changes, to accurately reflect the state of the underlying data.

Configuring the Tag Cloud component

This topic describes how to configure the **Tag Cloud** component.

You configure a **Tag Cloud** component via its **Configuration Options** dialog:



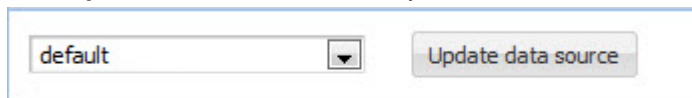
The Configuration Options dialog for the Tag Cloud component contains the following settings:

- Keyword attribute name:** A text field containing the value "Flavors".
- Enable end user refinements:** A checkbox that is checked.
- Tag weighting algorithm:** Two radio buttons: "Frequency" (selected) and "Relevancy". A blue question mark icon is located to the right of the "Relevancy" button.
- Maximum number of keywords:** A text field containing the value "50". Below this field is a hint: "Enter a value between 1 and 100."

At the bottom right of the dialog are two buttons: "Reset Preferences" and "Save Preferences".

To configure the **Tag Cloud** component:

1. After adding a **Tag Cloud** component to a page, click its button and then click **Preferences**.
2. In the **Tag Cloud** edit view, select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.



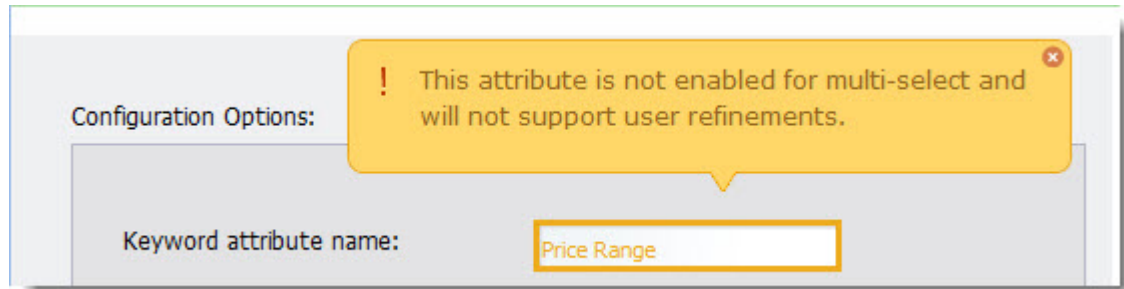
This interface shows a data source selection control. It consists of a drop-down menu with the text "default" and a small downward arrow icon. To the right of the drop-down is a button labeled "Update data source".

3. In the **Keyword attribute name** field, enter the name of the attribute the tags will be drawn from. The attribute must be enabled for multi-select. If you enter a multi-select attribute, Discovery Framework will verify it by showing a double-check icon next to the attribute name, as in this example:



This image shows a close-up of the configuration fields. The "Keyword attribute name" field contains the text "Flavors". Below it, the "Enable end user refinements" field also contains the text "Flavors" and has a small double-check icon (two overlapping checkmarks) to its right, indicating that the attribute is verified for multi-select.

If you enter an attribute that is not enabled for multi-select, it will not support user refinements, and you will see the following message:



4. Check the **Enable end user refinements** box to allow end users to drill down on tags in the tag cloud. If the attribute is not enabled for multi-select, this option is not available.
5. Select a **Tag weighting algorithm** to determine the significance of a tag:
 - **Frequency:** If this option is selected, the tags in the tag cloud are sized to represent the number of records associated with the related attribute value. The terms with the largest font size are associated with the greatest number of records.
 - **Relevancy:** If this option is selected, tags in the tag cloud are sized to represent the number of records associated with the tag, and then are weighted relative to their uniqueness in the matching record set. Terms that appear throughout the index are less relevant, and appear smaller, than those that appear only in the matching records. This strategy is recommended if certain terms appear on most records in the index, to ensure that these common terms are displayed less prominently in the tag cloud.
6. In the **Maximum number of keywords** field, specify the maximum number of attribute values to display in the **Tag Cloud** component. The default is 30.
7. Click **Save Preferences** to save your changes.



Chapter 13

Filtering Components

Filtering components allow you to search, navigate, and filter your data.

Guided Navigation

The **Guided Navigation** component provides Endeca Guided Navigation® functionality.



The **Guided Navigation** component provides contextual navigation across large volumes of structured MDEX Engine-specific data. End users can expand and close multiple dimension groups, view dimension values from multiple dimensions at a time, and select multi-OR and multi-AND dimensions. End users can also apply negative refinements to exclude dimension values from results if negative refinements are enabled by the power user.

The component shows the dimension groups that are configured in the MDEX Engine. It also displays base refinement statistics if they are enabled in the MDEX Engine.



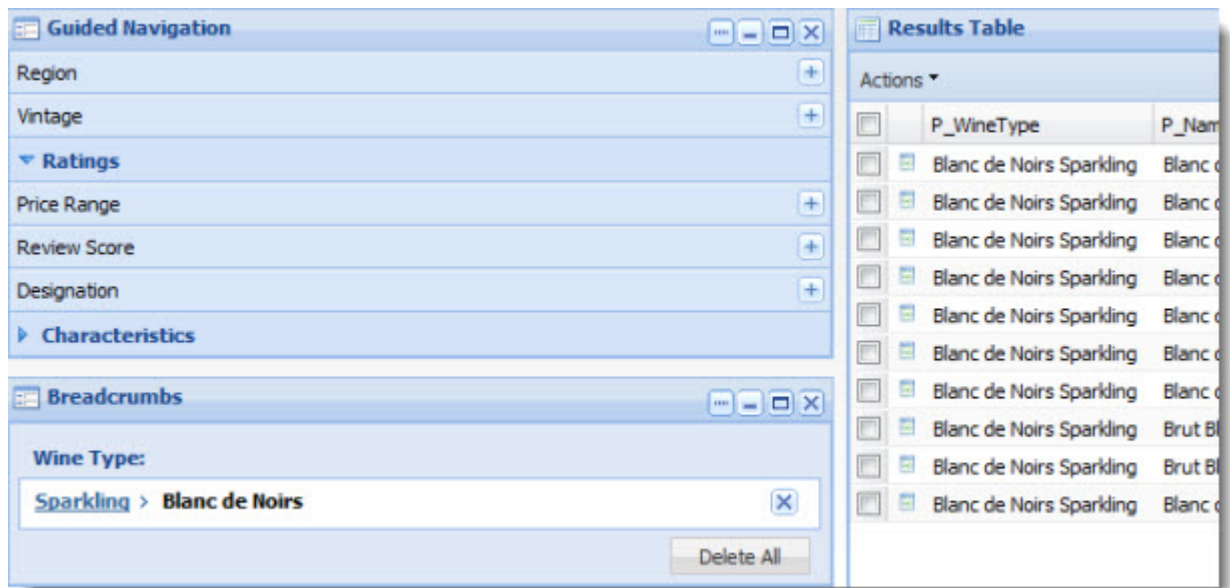
Note: Contracted or implicit dimension values and dead ends are not shown in this component.


The **Guided Navigation** component does not support multi-select for hierarchical dimensions, nor does it support tree view controls. Also, aggregated statistics are not supported in this component.


Dependencies


The **Breadcrumbs** component is required in order to view or clear **Guided Navigation** component selections.

The image below shows the interaction between the **Guided Navigation**, **Breadcrumbs**, and **Results Table** components. The navigation state, which is displayed in the **Breadcrumbs** component, determines what is displayed in the **Results Table** component. If the user navigated to a different state in the **Guided Navigation** component, the other two components would dynamically update to reflect the change.



 **Note:** The **Guided Navigation** component supports the use of aggregated records. Specifically, when the component is using a data source with a rollup key specified, its refinements display the number of aggregated records. For details about configuring Discovery Framework data sources for aggregated records, see Chapter 3 ("Configuring Data Sources") in this guide.

 **Note:** In order for the **Guided Navigation** component to generate aggregated record refinement statistics, the `--stat-abins` flag must be enabled on the MDEX Engine.

 **Note:** The **Guided Navigation** component includes a threshold on the maximum number of dimension values that can be displayed at once within a dimension. The default value is 500. This threshold is in place for performance and usability reasons, so that the end user's experience can be managed for cases where dimensions include very long lists of dimension values. This threshold can be used in conjunction with dynamic ranking and type-ahead search in the **Guided Navigation** interface, so users can still access specific values by filtering for them.

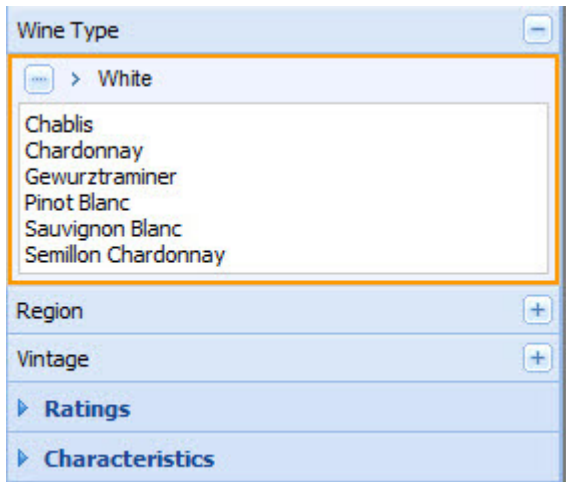
Using Guided Navigation

This topic describes how an end user can use the **Guided Navigation** component.

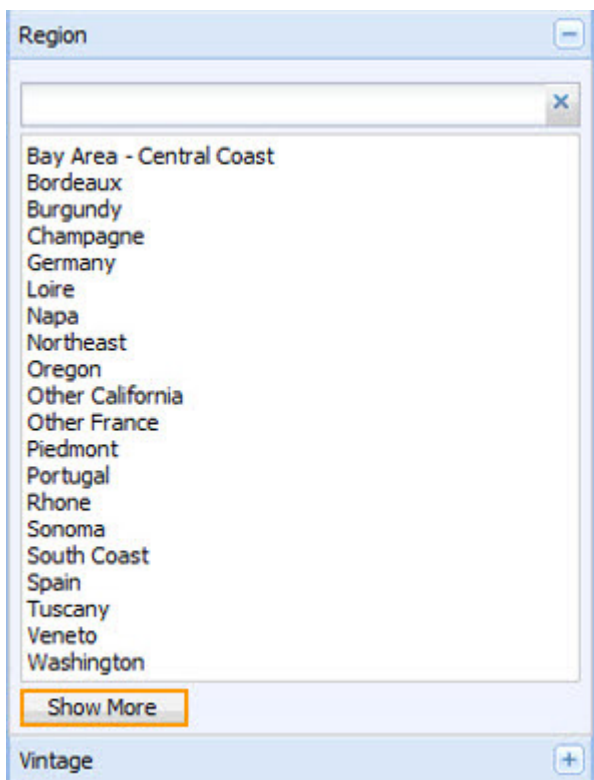


Note: The appearance of the Guided Navigation display varies depending on the dimension data and hierarchy.

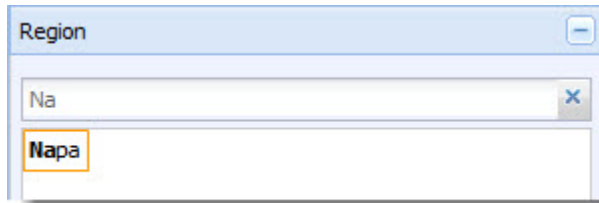
The **Guided Navigation** component can feature dimension hierarchy:



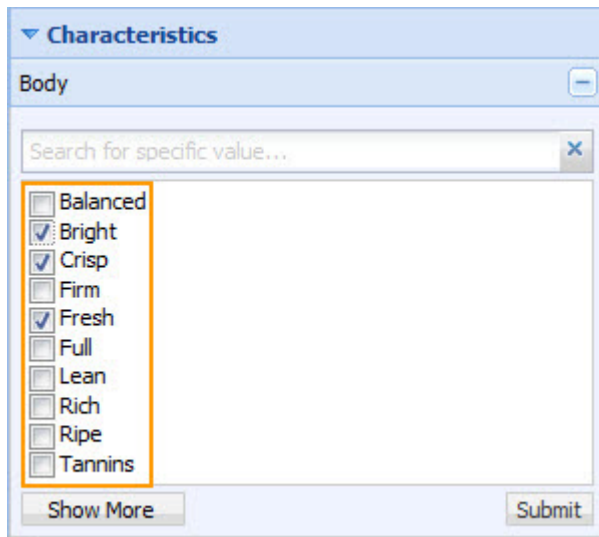
It features **Show More** functionality for large data sets:



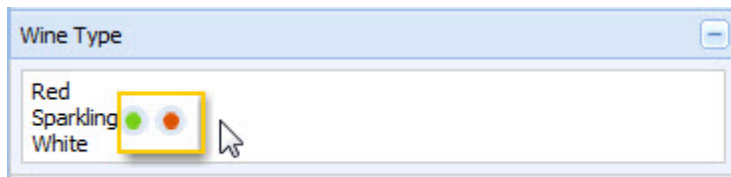
It can include type-ahead suggestions:



It can incorporate multi-select OR dimensions through a check-box interface:



It can include negative as well as positive refinements, which allow you to exclude refinements from the results displayed.



Setting the Guided Navigation configuration options

This topic describes the high-level steps involved in configuring the **Guided Navigation** component.

Before configuring type-ahead suggestions, dimension search wildcarding must be enabled in the MDEX Engine.

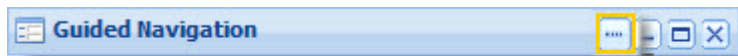
The Guided Navigation component is configured via its configuration menu:

Configuration Options

<input checked="" type="checkbox"/> Enable type-ahead	Maximum values to show in a single attribute:
Maximum type-ahead suggestions: 20	500
<input checked="" type="checkbox"/> Enable automatic attribute updates in end user view ?	Target page:

To set the **Guided Navigation** configuration options:

1. After adding a **Guided Navigation** component to a page, click its button.



2. Click **Preferences**.



3. Select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.

default ▼ Update data source

4. In the **Configuration Options** panel, check **Enable type-ahead** to enable the type-ahead search box for attributes with many values. When enabling this feature, make sure that your attributes have been configured to be wildcard searchable.

☒ Enable type-ahead

5. In the **Maximum type-ahead suggestions** field, set the maximum total number of type-ahead suggestions the MDEX Engine will offer. If fewer results are available, the number returned will be less than this maximum.

Maximum type-ahead suggestions:
20

6. In the **Maximum values to show in a single attribute** field, set the maximum number of values per attribute to be displayed. This can help prevent your end users from being overwhelmed by an excessive number of results.

Maximum values to show in a single attribute:

7. Check the **Enable automatic attribute updates in end user view** box if you want the end user's view to automatically reflect attribute changes in the MDEX Engine and in the **Attribute Settings** component.

☒ Enable automatic attribute updates in
 end user view ?

8. In the **Target page** field, set the target page for any page transition. Page transitions allow a component on one page in your Discovery Framework application to pass data to a component on another page.

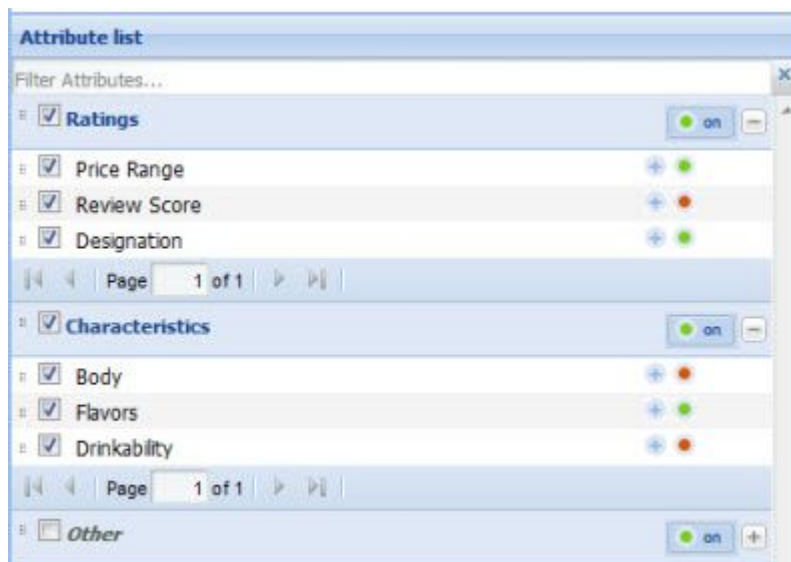
Target page:

9. You can configure the **Attribute list** now. Details are provided in the next topic.
10. Click **Save Preferences** to save your changes.

Configuring the Attribute list

The Attribute list defines the attributes that are displayed to the end user in the **Guided Navigation** component.

The **Attribute list** dialog is where you configure the attributes and attribute sets shown in the **Guided Navigation** component:



To configure the attributes shown in the **Guided Navigation** component:

1. Access the **Guided Navigation** edit view.
 The **Attribute list** dialog is one of the displayed dialogs.

2. If you need to find a specific attribute, use the **Filter Attributes** search box to search for an attribute by name (with type-ahead).
3. In the Attribute list, check the attributes and attribute sets that you want displayed in the **Guided Navigation** display. Unchecked attributes or attribute sets are hidden from the end user's view.
4. Use the **Expanded** check boxes to select attributes that will appear expanded in the end user's view.
Note that the end user may choose to collapse expanded attributes.
5. Use the **Allow Negative** check boxes to select attributes that can be negative as well as positive refinements. A negative refinement, when selected, is excluded from results.
6. Click **Save Preferences** to save your changes.

Breadcrumbs

The **Breadcrumbs** component provides breadcrumb navigation aid functionality.

The **Breadcrumbs** component summarizes any Guided Navigation selections, keyword searches, or range filters specified by the end user in a vertical stack of selected values. If multiple selections from single dimension are made, they are displayed separately in sequential order. Breadcrumbs support multi-select AND and OR refinements, but not record filters.

When used with keyword search, the **Breadcrumbs** component features spelling correction, as displayed in the following example:



The **Breadcrumbs** component supports the use of aggregated records. For details about configuring Discovery Framework data sources for aggregated records, see Chapter 3 ("Configuring Data Sources") in this guide.

Dependencies

The **Breadcrumbs** component requires a backing data source.

One of the following components must also be present: **Guided Navigation**, **Search Box**, or **Range Filter**.

Using Breadcrumbs

This topic describes how an end user can use the **Breadcrumbs** component.

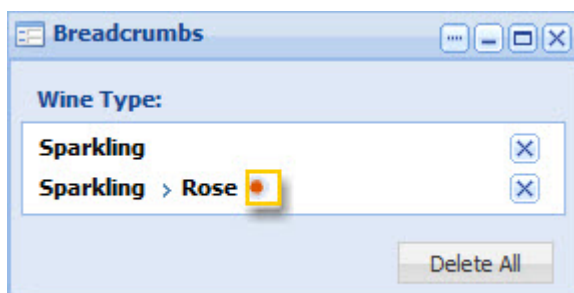
The following example is based on **Guided Navigation** selections. Note the hierarchical selection in the Wine Type dimension.



The following example is based on a **Range Filter** selection.



If a breadcrumb is based on a negative refinement (as enabled in the **Guided Navigation** component), that refinement is flagged with a red-dot icon. In the following **Guided Navigation** example, the refinement excludes Rose from the results.



End users may remove individual selections or all selections within displayed breadcrumbs, and can also select a specific ancestor within the hierarchy for display.

Configuring the Breadcrumbs component

This topic describes the high-level steps involved in configuring the **Breadcrumbs** component.

The **Breadcrumbs** component is configured via its preferences configuration menu:



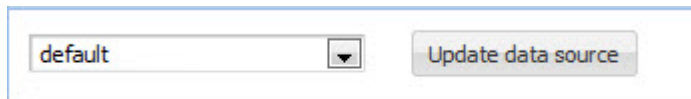
To configure the **Breadcrumbs** component:

1. After adding a **Breadcrumbs** component to a page, click its button and then click **Preferences**.



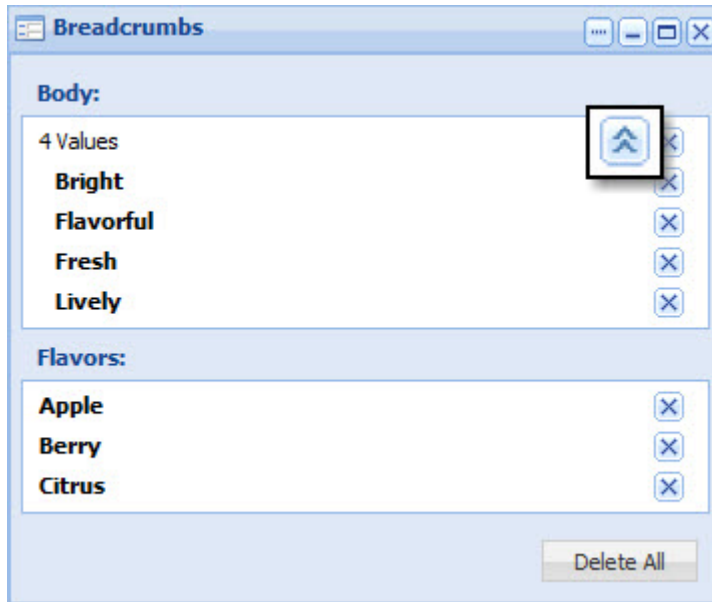
The **Breadcrumbs** preferences configuration view appears.

2. Select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.



3. In the **Multi-select collapse/expand threshold** field, set the number of dimension values after which the list can be collapsed.
4. Click **Save Preferences** to save your changes.

To illustrate how the **Multi-select collapse/expand threshold** setting works, assume that it has been set to three. In the example below, dimension values for the Body dimension can be collapsed, because there are more than three selected. The dimension values for Flavors, on the other hand, cannot be collapsed, because there are only three selected.



Search Box

The **Search Box** component provides record and dimension search functionality.



It allows end users to submit keyword searches based on available properties in the data source he or she selects from the drop-down list.

Power users can choose from multiple search interfaces and match modes when configuring this component to further control the information that is displayed to the end user. The **Search Box** component can also be configured to provide type-ahead suggestions populated with matching dimension values.

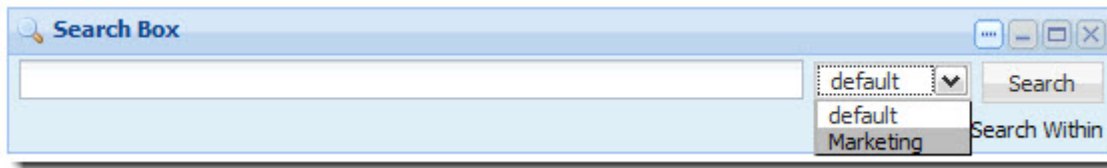
Dependencies

The **Search Box** component works in conjunction with the **Breadcrumbs** component, the **Results Table** component, and the **Guided Navigation** component.

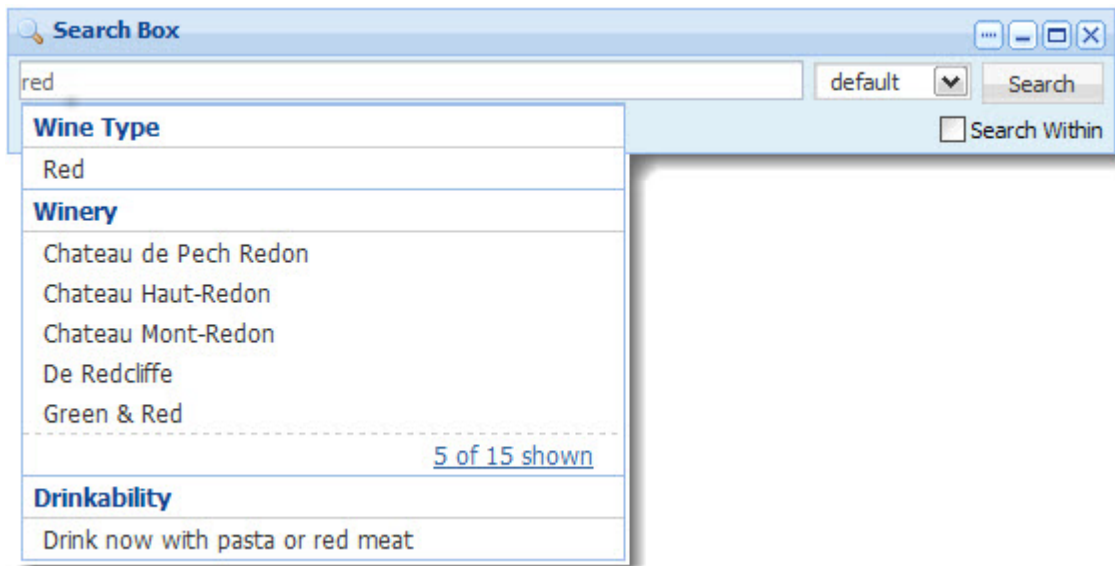
Using Search Box

This topic describes how an end user can use the **Search Box** component.

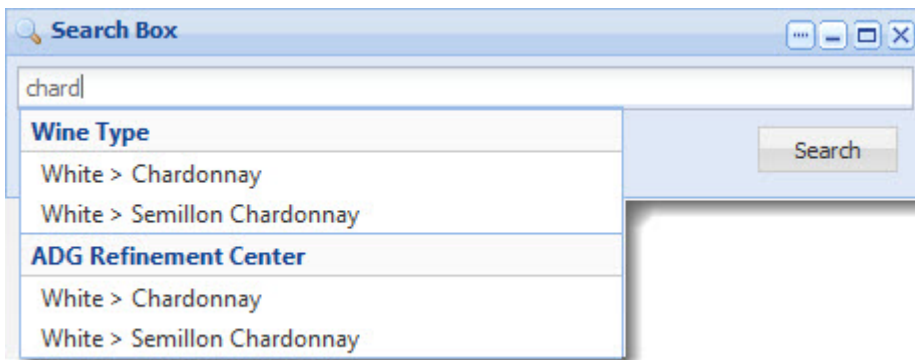
The end user can select a search configuration (based on a data source and a set of other criteria) from a list that the power user enables.



The end user can then enter a search term. In the example below, the end user typed the term *red*. The Wine Type, Winery, and Drinkability dimensions, along with matching dimension values, are made available as suggestions.



If the power user has enabled the type-ahead option, the MDEX Engine will start offering dimension value suggestions before the user has typed a complete search term.



To restrict the results displayed to those within the current navigation state, the end user can check **Search Within**.

Configuring Search Configurations

This topic describes the high-level steps involved in adding and configuring the **Search Box** component.

When you first open **Search Box** preferences, you see a list of existing search configurations:

Display Name	Default	Data Source	Search Interface	Match Mode	Target Page
default	<input checked="" type="radio"/>	default	All	all	

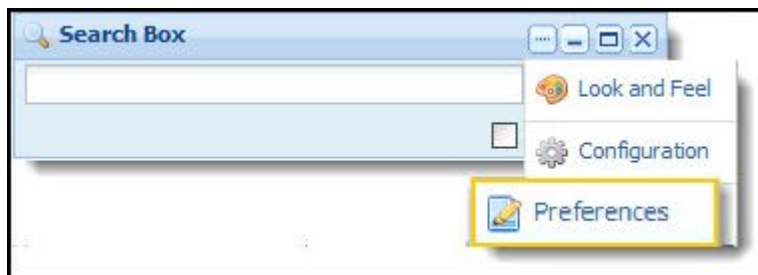
Select a search configuration to see additional options.

Save Preferences Revert Changes

From this dialog box, you add and delete configurations, as well as edit an existing search configuration. When you select a search configuration, or click **Add Configuration**, **Search Box** preferences updates to include the **Enable type-ahead suggestions** dialog. This topic describes just the **Search Configurations** dialog, while the following topic describes the type-ahead configuration procedure.

To add a search configuration to the **Search Box** component or edit an existing one:

1. After adding a **Search Box** component to a page, click its button and then click **Preferences**.



2. In the **Search Configurations** edit view, use the **Display Name** field to supply a name for the new search configuration or to change the existing name.
The **Display Name** is used in the **Search Box** component (as a selectable search configuration for the user) and in the **Breadcrumbs** component.
3. If you have two or more search configurations, use the **Default** radio button to set one of them as the default search configuration for the search box.
4. Use the **Data Source** drop-down menu to select one of the available data sources to associate with this search configuration.

Search Box

Search Configurations:

+ Add Configuration - Remove Configuration

Display Name ▲	Default	Data Source	Search Interface	Match Mode	Target Page
default	<input checked="" type="radio"/>	default	All	all	
Marketing	<input type="radio"/>	default	All	partial	

The dropdown menu for the 'default' configuration's Data Source is open, showing options: default, v7-default, and bizwine.

5. Use the **Search Interface** drop-down menu to select from the search interfaces defined in this data source. The search interface limits the end user's search, and allows you to control record search behavior for groupings of some number of dimensions, properties, attributes, and ranking strategies.

Search Box

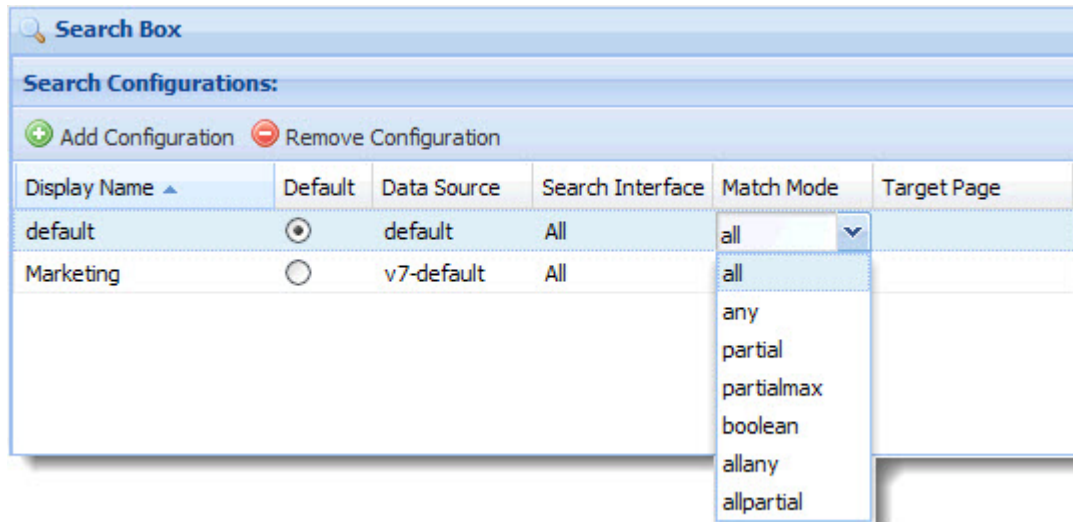
Search Configurations:

+ Add Configuration - Remove Configuration

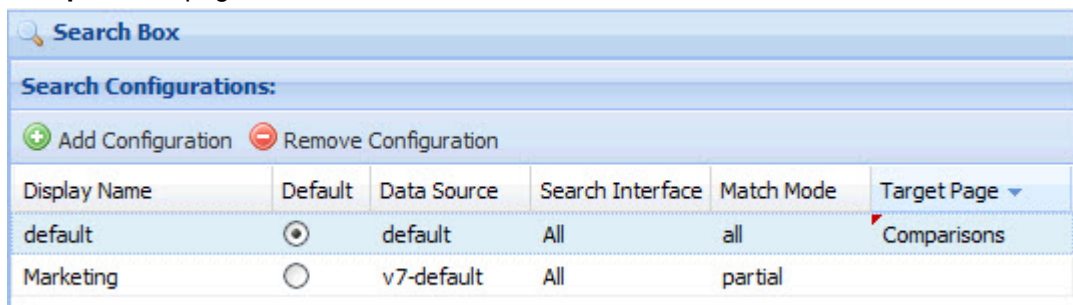
Display Name ▲	Default	Data Source	Search Interface	Match Mode	Target Page
default	<input checked="" type="radio"/>	default	All	all	
Marketing	<input type="radio"/>	v7-default	All	partial	

The dropdown menu for the 'default' configuration's Search Interface is open, showing options: All, WineAttributes, and NameAndWinery.

6. Use the **Match Mode** drop-down menu to select a match mode for the search configuration:
- `all` matches all user search terms (that is, perform a conjunctive search). This is the default mode.
 - `any` matches at least one user search term.
 - `partial` matches some user search terms.
 - `partialmax` matches a maximal subset of user search terms.
 - `boolean` matches using a Boolean query.
 - `allany` matches all user search terms if possible, otherwise matches at least one. This mode is not recommended in cases where queries can exceed two words.
 - `allpartial` matches all user search terms if possible, otherwise matches some.



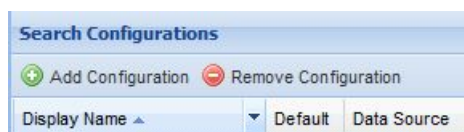
7. Optionally, use the **Target Page** field to set the target page for a page transition. Page transitions allow a component on one page in your Discovery Framework application to pass data to a component on another page. In the example below, data will be sent to a component on the **Comparisons** page.



8. By clicking in the label of any column, you can sort the contents of that column or remove columns from the **Search Configurations** display. Details are provided in the "Column Sort and Removal controls" topic.
9. Use the **Enable type-ahead suggestions** checkbox to enable or disable type-ahead suggestions. Details are provided in the "Configuring Search Box type-ahead suggestions" topic.
10. Click **Save Preferences** to save your changes.

Column Sort and Removal controls

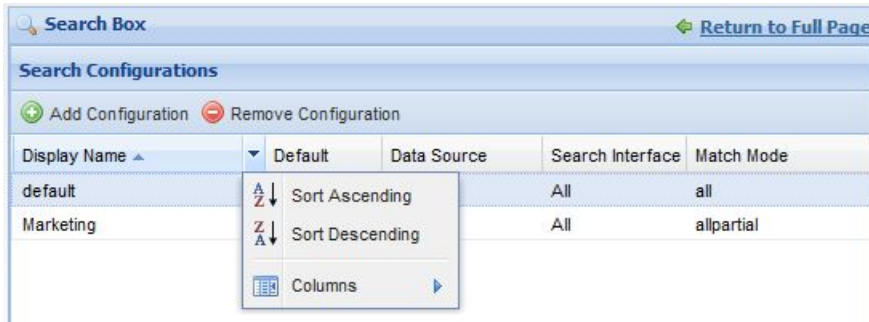
These **Search Box** controls allow you to control the sort order of columns and the removal of columns. If you click once on a column label, the label will display two arrows, as shown in this example for the **Display Name** column label:



The arrows work as follows:

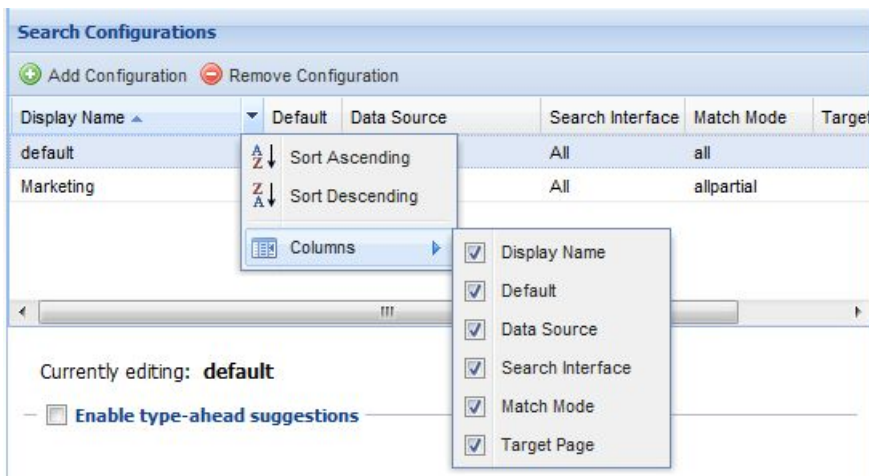
- Toggling the sort arrow next to the column name sorts the contents of the column in an ascending (up arrow) or descending (down arrow) sort order.
- Clicking on the right-most arrow in the label brings up the **Column Sort and Removal** controls.

The **Column Sort and Removal** controls look like this example:



The **Sort Ascending** and **Sort Descending** options work identically as the sort arrow described above: they sort the contents of the column in an ascending or descending sort order.

Clicking on the **Columns** option brings up the **Column Removal** dialog:



To remove a column from the **Search Configurations** edit view, uncheck its name in the **Column Removal** dialog. You can restore a removed column by displaying the **Column Removal** dialog from an existing column label and checking the column name.

Configuring Search Box type-ahead suggestions

Type-ahead suggestions are an optional feature for the **Search Box** component.

Before configuring type-ahead suggestions, dimension search wildcarding must be enabled in the MDEX Engine.

For the **Search Box** component, you can specify that the MDEX Engine start offering suggestions before the user has typed a complete search term.

The **Enable type-ahead suggestions** dialog is where you configure this feature:

Currently editing: **Marketing**

☒ **Enable type-ahead suggestions**

Minimum characters to trigger suggestions:

Maximum suggestions per dimension:

Maximum total suggestions:

Show intermediate values: ☒

Dimension Configurations		Search for a dimension...
Dimension Display Name	Included <input checked="" type="checkbox"/>	
Body	<input checked="" type="checkbox"/>	<input type="button" value="Up"/> <input type="button" value="Down"/> <input type="button" value="Refresh"/>
Region	<input checked="" type="checkbox"/>	
Review Score	<input checked="" type="checkbox"/>	
ADG Refinement Center	<input checked="" type="checkbox"/>	
Winery	<input checked="" type="checkbox"/>	
Flavors	<input checked="" type="checkbox"/>	



Note: This procedure assumes that you are editing an existing search configuration. However, the procedure can also be used for the creation of a new search configuration.

To configure type-ahead suggestions:

1. Access the **Search Configurations** edit view for a selected search configuration.
2. Check the **Enable type-ahead suggestions** checkbox to enable the type-ahead suggestions feature. Keep in mind that dimension search wildcarding must be enabled in the MDEX Engine in order for type-ahead to work.

☒ **Enable type-ahead suggestions**

3. In the **Minimum characters to trigger suggestions** field, set the minimum number of characters the user must type before type-ahead suggestions are offered.

Minimum characters to trigger suggestions:

4. In the **Maximum suggestions per dimension** field, set the maximum number of type-ahead suggestions the MDEX Engine will offer for each dimension.

Maximum suggestions per dimension:

5. In the **Maximum total suggestions** field, set the maximum total number of type-ahead suggestions the MDEX Engine will offer.

Maximum total suggestions:

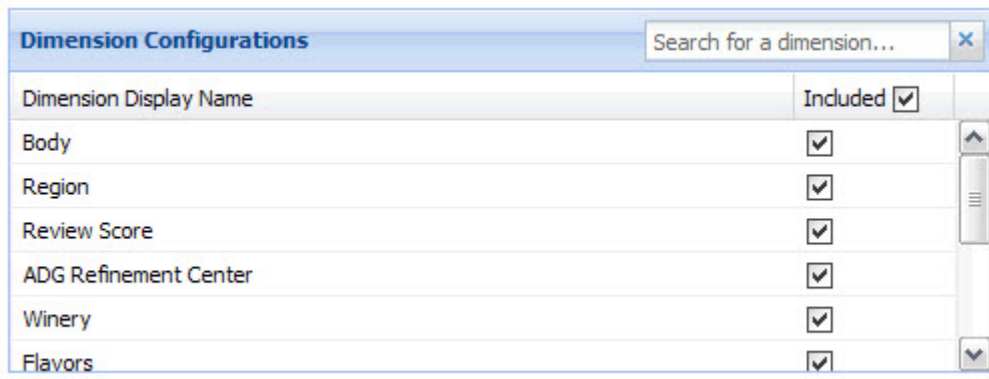
15

6. Check **Show intermediate values** to display the full path for hierarchical results.

Show intermediate values:



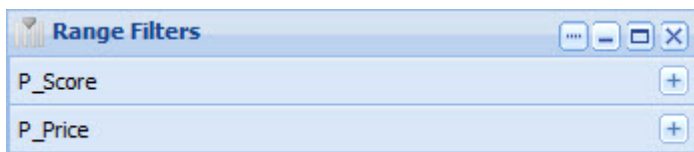
7. In the **Dimensions Configurations** dialog, check **Included** to include all dimensions in the type-ahead results. Alternatively, you can search for a dimension, or select dimensions individually, to customize which dimensions are included.



8. Click **Save Preferences** to save your changes.

Range Filter

The **Range Filter** component allows you to add and modify range filters.

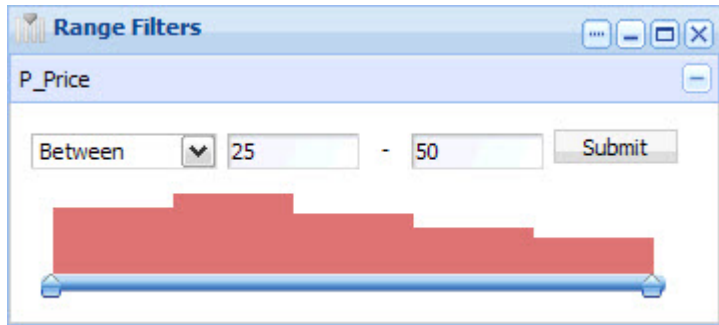


The **Range Filter** component provides numeric and date filtering, but not geocode filtering. You cannot use the **Range Filter** component to filter on dimensions.

Using Range Filter

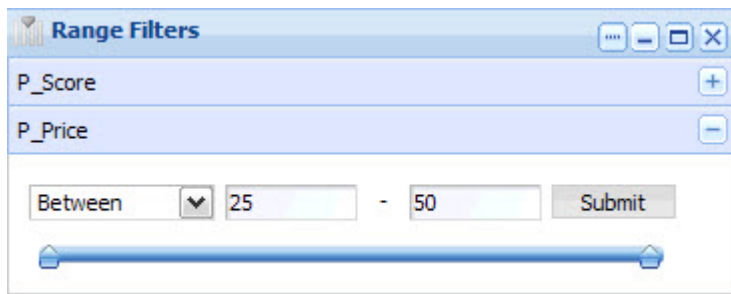
This topic describes how an end user can use the **Range Filter** component.

If the power user has configured a range filter for a property, the end user sees something like this:

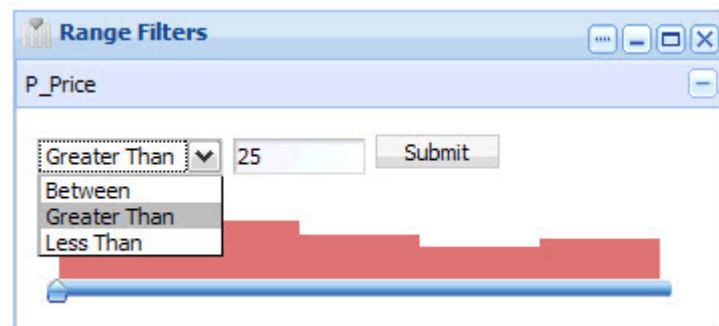


The histogram gives the end user a high-level view of the data. The end user can use the slider bar to refine the filter within the defined range.

It is also possible for the power user to configure the range filter to display the slider bar with no histogram:



The end user can also select the **Greater Than** and **Less Than** filters from the drop-down list:



Configuring range filters

You can add and edit range filters on existing properties in the data set in the **Range Filters** component.

Before adding new range filters, keep in mind that the property to be used for the range filter must already exist in the MDEX Engine.

The **Range Filters** component is configured via its configuration menu:

default [Update data source](#)

Range Filters

+ Add Range Filter - Remove Range Filter

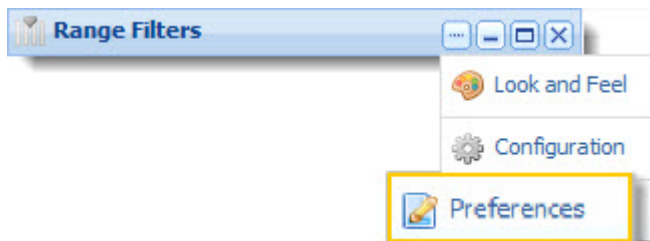
Property Name ▲	Min	Max	Type	<input type="checkbox"/> Enable Histogram	# Bin
P_Score	60.00		Numeric	Yes	30
P_Price	25.00	50.00	Numeric	Yes	5

[Save Preferences](#)

From this menu, you can add a new range filter, edit an existing range filter, or delete an existing range filter. The following procedure can be used when adding new range filters or editing existing ones.

To configure the **Range Filters** component:

1. After adding a **Range Filters** component to a page, click its button and then click **Preferences**.

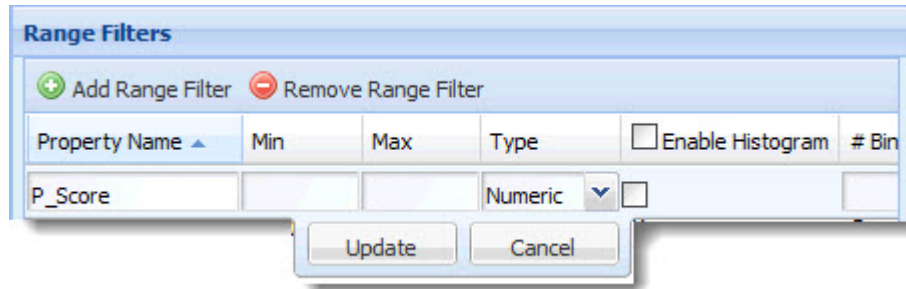


The **Range Filters** edit view appears.

2. Select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.

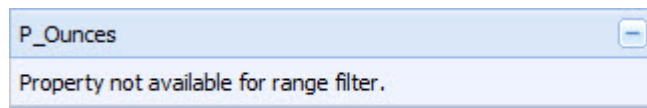
default [Update data source](#)

3. In the **Range Filters** edit view, either click **Add Range Filter** (to create a new range filter) or begin editing an existing range filter.



The 'Range Filters' dialog box has a title bar 'Range Filters'. Below the title bar are two buttons: 'Add Range Filter' (with a green plus icon) and 'Remove Range Filter' (with a red minus icon). Below these is a table with columns: 'Property Name', 'Min', 'Max', 'Type', 'Enable Histogram', and '# Bin'. The first row of the table has 'P_Score' in the 'Property Name' column, empty fields for 'Min' and 'Max', 'Numeric' in the 'Type' column, an unchecked checkbox for 'Enable Histogram', and an empty field for '# Bin'. Below the table are two buttons: 'Update' and 'Cancel'.

4. In the **Property Name** field, enter the name of the property this filter is based on.
You need to know the exact name of the property for which you want to create a filter (property names are case sensitive). If you are unsure of a property's name, you can consult the **Endeca Attribute Settings** component. If you attempt to add a range filter based on a property that does not exist, when you return to the full page, you will see the following error message:



The error message dialog box has a title bar 'P_Ounces' and a message box containing the text 'Property not available for range filter.'.

5. Enter one or two range values, depending on what type of filtering you want to provide:
 - Enter values for both **Min** and **Max** to provide a range.
 - Enter a single value for **Min** to allow **Greater Than** filtering.
 - Enter a single value for **Max** to allow **Less Than** filtering.

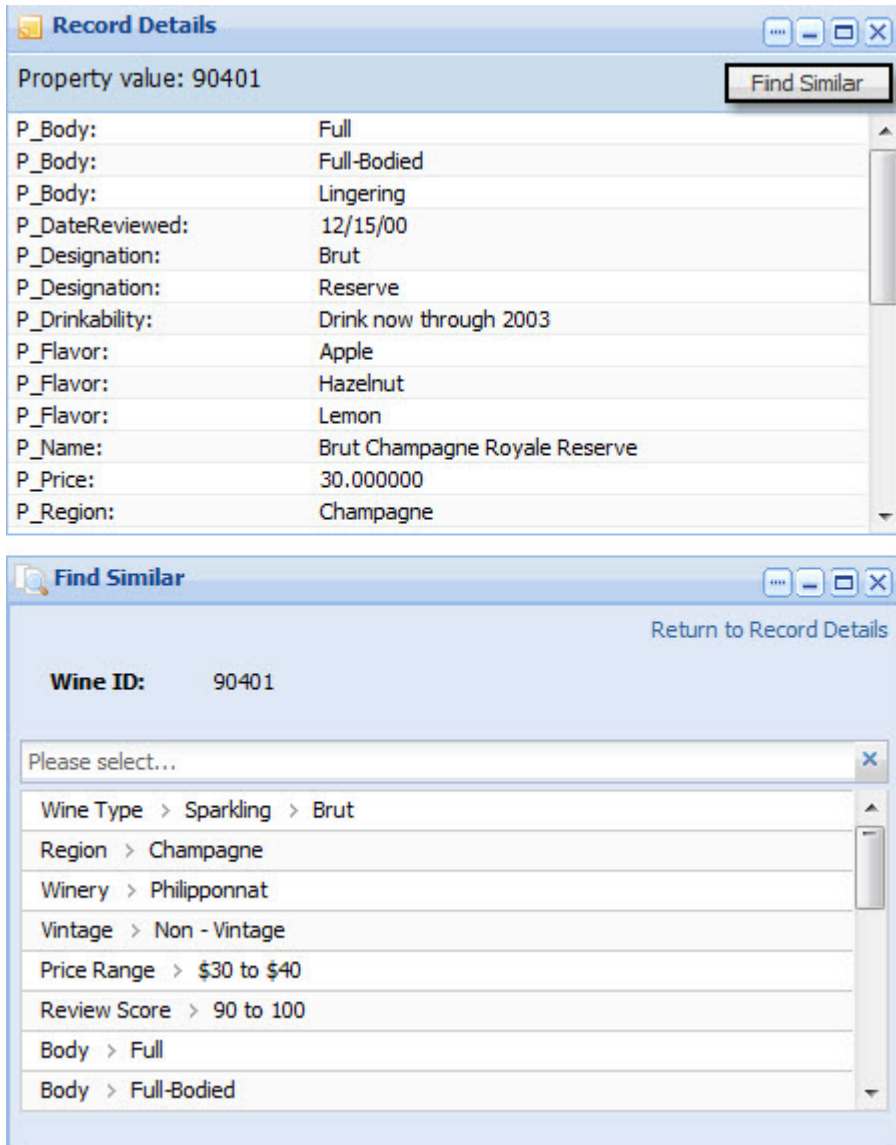
For **Date** types, **Min** and **Max** values must be given in milliseconds-since-epoch format. The **Range Filters** component does not accept formatted date strings (such as "11/22/2010") or values in standard seconds-since-epoch format.

6. From the **Type** drop-down, select the type of range filter: **Numeric** or **Date**.
7. Check **Enable Histogram** to provide the end user with an interactive histogram control. If **Enable Histogram** is not checked, the end user sees a slider bar with no associated histogram.
8. If you have enabled a histogram control, use the **# Bins** field to set the number of bars to display in the histogram.
9. Click **Update** to update the range filter configuration.
10. Click **Save Preferences** to save your changes to the component.

Find Similar

The **Find Similar** component provides end users with the opportunity to find other records that share common attributes with a specific record of interest.

Users can view specific records of interest in the **Record Details** component and then click **Find Similar** to search for records with similar attributes. The image below shows the relationship between the **Find Similar** and **Record Details** components.



The **Find Similar** component supports the use of aggregated records. For details about configuring Discovery Framework data sources for aggregated records, see Chapter 3 ("Configuring Data Sources") in this guide.

Dependencies

In order to use the **Find Similar** component, the **Record Details** component must be configured.

Using Find Similar

This topic describes how an end user uses the **Find Similar** component.



Note: The **Find Similar** component must be used in conjunction with the **Record Details** component.

Once an end user has selected a record to view in the **Record Details** component, the user clicks **Find Similar** to populate the **Find Similar** component with the record's attributes. The user then selects which attributes to search by, as shown below:

The screenshot shows the **Find Similar** window with a list of attributes and their selected values. The attributes are: Wine Type, Region, Vintage, Price Range, Review Score, Body, and Flavors. The selected values are: Red, Pinot Noir, Bay Area - Central Coast, 1997, \$20 to \$30, 80 to 90, Ripe, Tannins, Berry, and Cherry. The window also includes a 'Return to Record Details' link, a 'Clear All' button, and a 'Submit' button. A message at the bottom states: 'The search will result in 127 records.'

Attribute	Selected Value
Wine Type	Red > Pinot Noir
Region	Bay Area - Central Coast
Vintage	1997
Price Range	\$20 to \$30
Review Score	80 to 90
Body	Ripe
Body	Tannins
Flavors	Berry
Flavors	Cherry

Region: Bay Area - Central Coast

Vintage: 1997

Review Score: 80 to 90

Clear All

The search will result in 127 records.

Submit

The user can employ the search box within the **Find Similar** component to narrow the list of available attributes or to locate a specific attribute, as shown below:

The screenshot shows the search box within the **Find Similar** component. The search box contains the text 'Cherry'. Below the search box, a dropdown menu is visible, showing the attribute 'Flavors > Cherry'.

Cherry

Flavors > Cherry

When the user clicks **Submit**, the **Results Table** component displays all records with matching attributes:

	P_WineType	P_WineID	P_Score	P_Region
<input type="checkbox"/>	Merlot Red	76877	82	Bay Area - Central Coast
<input type="checkbox"/>	Merlot Red	76878	89	Bay Area - Central Coast
<input type="checkbox"/>	Merlot Red	76879	83	Bay Area - Central Coast
<input type="checkbox"/>	Sauvignon Blanc White	79904	87	Bay Area - Central Coast
<input type="checkbox"/>	Merlot Red	76880	82	Bay Area - Central Coast
<input type="checkbox"/>	Sauvignon Blanc White	79956	88	Bay Area - Central Coast
<input type="checkbox"/>	Red Syrah	80689	87	Bay Area - Central Coast
<input type="checkbox"/>	Sauvignon Blanc White	79957	90	Bay Area - Central Coast
<input type="checkbox"/>	Chardonnay White	74160	85	Bay Area - Central Coast
<input type="checkbox"/>	Pinot Noir Red	78282	89	Bay Area - Central Coast

The **Breadcrumbs** component is also populated with the search dimensions. The user can further refine these results using the **Guided Navigation** and **Search Box** components.

Configuring the Find Similar component

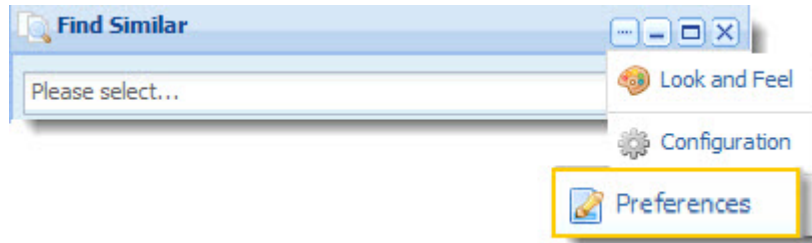
This topic describes the high-level steps involved in configuring the **Find Similar** component.

The **Find Similar** component is configured via its configuration menu:

A configuration task will be to add one or more header fields. A header field is a property or dimension value that identifies the starting record in the header of the **Find Similar** dialog. (The starting record is the record that the user is viewing in the **Record Details** component.) In this example of the **Find Similar**, the two configured header fields are the P_Name (whose display name is Name) and P_WineID (display name is WineID) properties:

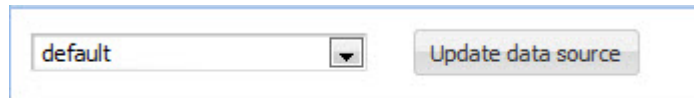
To configure the **Find Similar** component:

1. After adding a **Find Similar** component to a page, click its button and then click **Preferences**.

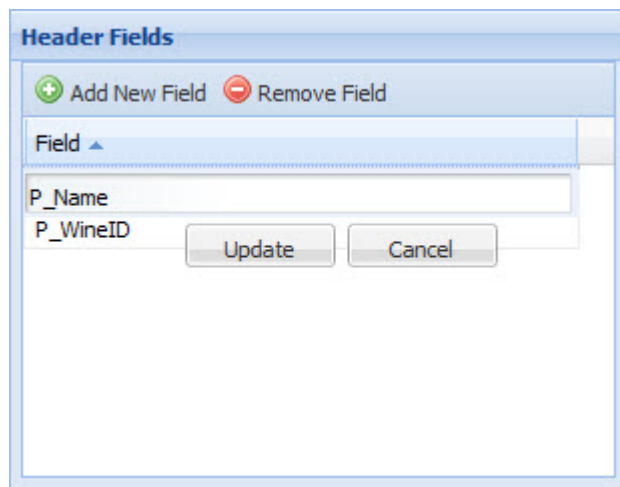


The **Find Similar** edit view appears.

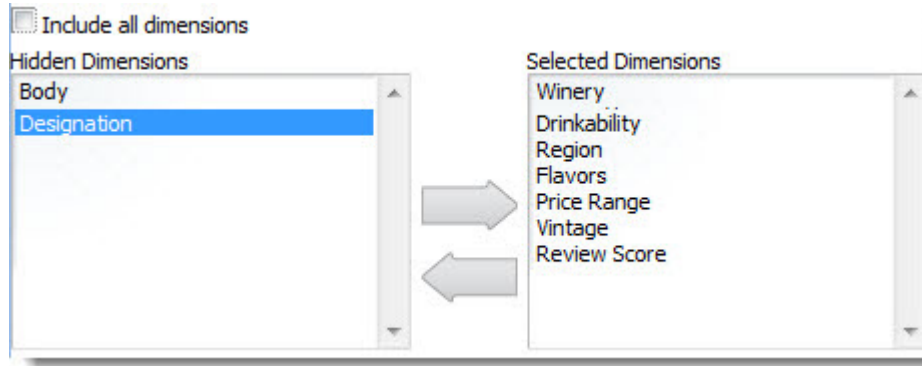
2. Select from the available data sources in the drop-down list, and then click **Update data source** to bind your selection to the component.



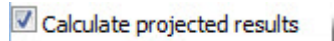
3. Add a header field:
 - a) Click **Add New Field**.
 - b) Type the name of the property or dimension value in the text field.
 - c) Click **Update**.



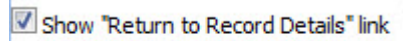
4. If desired, you can add more header fields by repeating Step 3.
5. Configure which dimensions are shown in the **Find Similar** dialog:
 - a) Check **Include all dimensions** to include all dimensions in the dialog.
 - b) If **Include all dimensions** is unchecked, the **Hidden Dimensions** and **Selected Dimensions** lists display (as shown in the following example) and you can select which dimensions to hide or include.



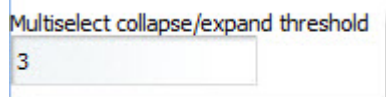
6. Check **Calculate projected results** to show how many similar results will be returned with the selected dimensions.



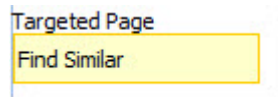
7. Check **Show "Return to Record Details" link** to show a link that allows the end user to clear the dimensions listed in the **Find Similar** dialog and return to the **Record Details** component.



8. In the **Multiselect collapse/expand threshold** field, set the number of dimension values after which the list can be collapsed.



9. Optionally, use the **Targeted Page** field to specify a separate page to display the **Find Similar** results. Keep in mind that the target page must have a **Results Table** component.



10. Click **Save Preferences** to save your changes.

To illustrate how the **Multiselect collapse/expand threshold** setting works, assume that it has been set to three. In the example below, six values are selected for the **Body** dimension. Because the threshold is set at 3, the user has the option to collapse the list (highlighted below).





Chapter 14

Incorporating Liferay Components

The Discovery Framework includes a number of Liferay components in its installation by default. These components can be integrated with Endeca components to allow you to build richer applications.

Liferay component support

The level of documentation and support Endeca provides for the Liferay components included in the Discovery Framework differs from that provided for our own standard components.

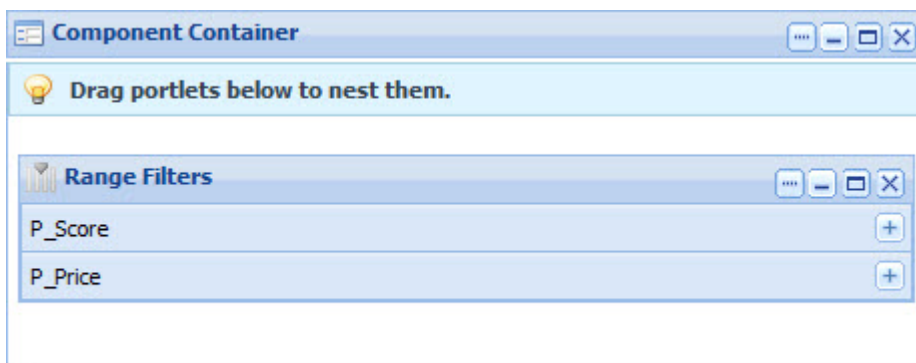
Because these components were not developed by Endeca, Endeca cannot control their interface or guarantee that they will be available in subsequent versions. Endeca provides only high-level documentation for Liferay components.

For more detailed information, consult the Liferay documentation and forums, available at <http://www.liferay.com>.

The Component Container

The **Component Container** allows power users to organize components by grouping them together in a single container.

The power user can choose from several layout templates in order to customize the display to the needs of the application.



The Languages component

The **Language** component can be used to change the locale of the server.

To select an alternate language in the **Language** component, click the flag icon associated with your target language. In the example below, we have chosen Spanish.



The Discovery Framework displays the component messages from your resource bundle in your target language. In addition, because the portal itself is also localized, menus and other portal controls also appear in your target language.



Note: For more information about localizing the Discovery Framework, see the *Discovery Framework Extension Guide*.

About Liferay Web Content Management components

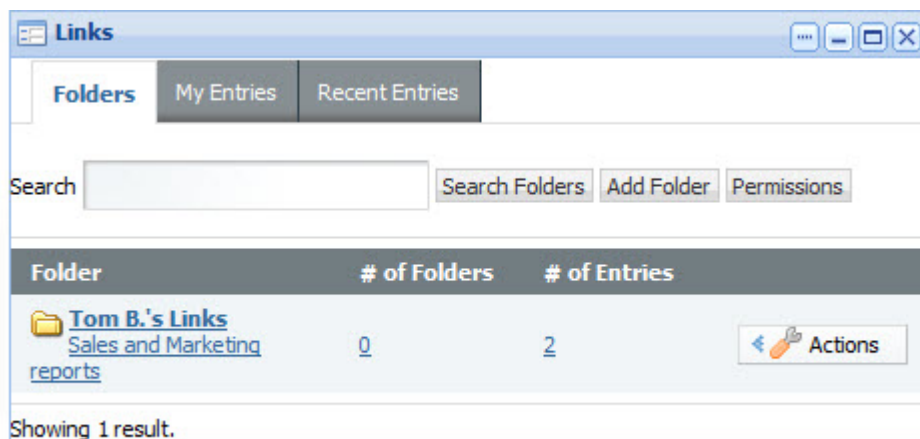
Liferay Web Content Management (WCM) components can be integrated into your application to provide document management and publishing capabilities.

These components are often used together to configure, display, and save links to Web content.

Links component

The **Links** component allows users to save and manage their own Web content links in folders.

These links can be tagged for later search and shared with other users.

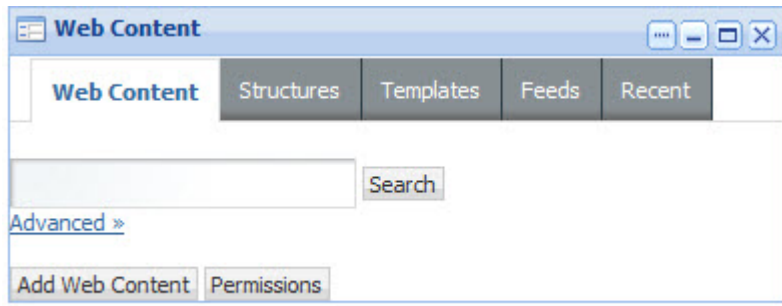


Web Content component

The **Web Content** component allows the power user to configure and manage the administrative aspects of Liferay Web Content Management.

This includes the following:

- Enabling users to write and publish articles to the site.
- Creating article templates.
- Controlling article-creation workflow and versioning.
- Managing article search and metadata.



The content is displayed in the **Web Content Display** component.

Web Content Display component

The **Web Content Display** component allows you to request and display articles in your application.

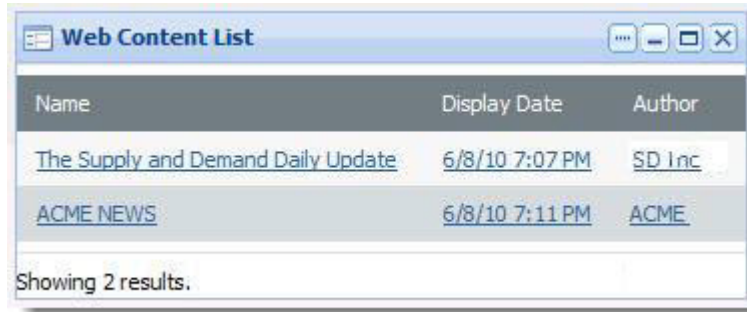
The appearance of the articles can be controlled by WCM templates, as configured in **Web Content** component.



Web Content List component

The **Web Content List** component displays a list of all Web content articles that are available in the application.

The article list is automatically updated when new articles are made available, and can be sorted by various criteria.



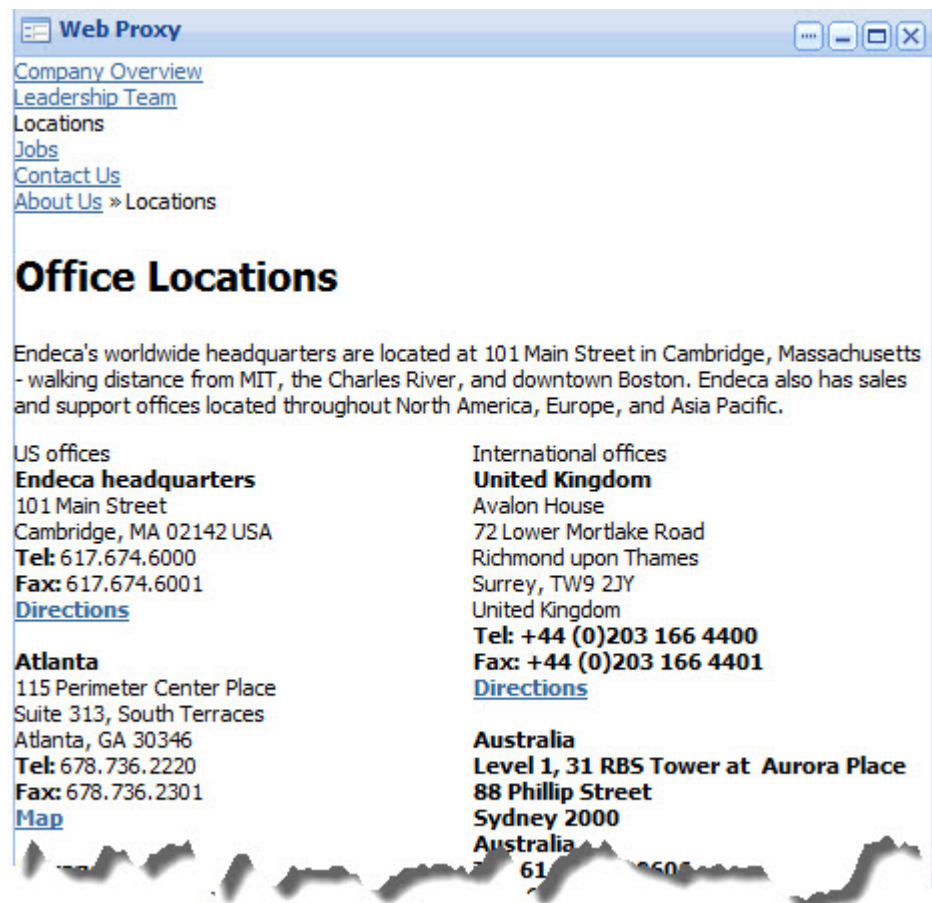
Name	Display Date	Author
The Supply and Demand Daily Update	6/8/10 7:07 PM	SD Inc
ACME NEWS	6/8/10 7:11 PM	ACME

Showing 2 results.

Web Proxy component

The **Web Proxy** component makes it possible to show any Web site as if it were a component.

The power user can configure the appearance and authentication of the content.



Web Proxy

[Company Overview](#)
[Leadership Team](#)
[Locations](#)
[Jobs](#)
[Contact Us](#)
[About Us](#) » [Locations](#)

Office Locations

Endeca's worldwide headquarters are located at 101 Main Street in Cambridge, Massachusetts - walking distance from MIT, the Charles River, and downtown Boston. Endeca also has sales and support offices located throughout North America, Europe, and Asia Pacific.

US offices Endeca headquarters 101 Main Street Cambridge, MA 02142 USA Tel: 617.674.6000 Fax: 617.674.6001 Directions	International offices United Kingdom Avalon House 72 Lower Mortlake Road Richmond upon Thames Surrey, TW9 2JY United Kingdom Tel: +44 (0)203 166 4400 Fax: +44 (0)203 166 4401 Directions
Atlanta 115 Perimeter Center Place Suite 313, South Terraces Atlanta, GA 30346 Tel: 678.736.2220 Fax: 678.736.2301 Map	Australia Level 1, 31 RBS Tower at Aurora Place 88 Phillip Street Sydney 2000 Australia 61 2 9500 6100



Chapter 15

Implementing Page Transitions

This chapter describes how to use page transitions in your Discovery Framework application.

About page transitions

Page transitions allow a component on one page in your Discovery Framework application to pass data to a component on another page. In addition, page transitions let you specify the tab state on a tabbed component on that page.

For example, you might have a **Results Table** component on one page and a **Record Details** component on another page. The **Results Table** component could be configured to target the **Record Details** component through a page transition.

The page transitions feature is implemented by your component developer, but the power user can specify transition targets that will appear in the end user's application. In the example below, the power user set the target page for the **Record Details** component to a **Find Similar** component placed on the **Comparisons** page.

▼ Configure Action Menu

Enable action menu: ☒

Action menu name:

Available actions:

Active actions:

Find similar target page:

Page transition syntax

This topic describes the syntax for Discovery Framework page transitions.

Power users can target a page using a full context path or a relative context path. If the page transition target is a relative context path, the default context, `/web/guest`, is pre-pended to the target, as is shown in the following example:

```
Target: Analyze
Redirect: /web/guest/Analyze
```

The following example uses a full context path:

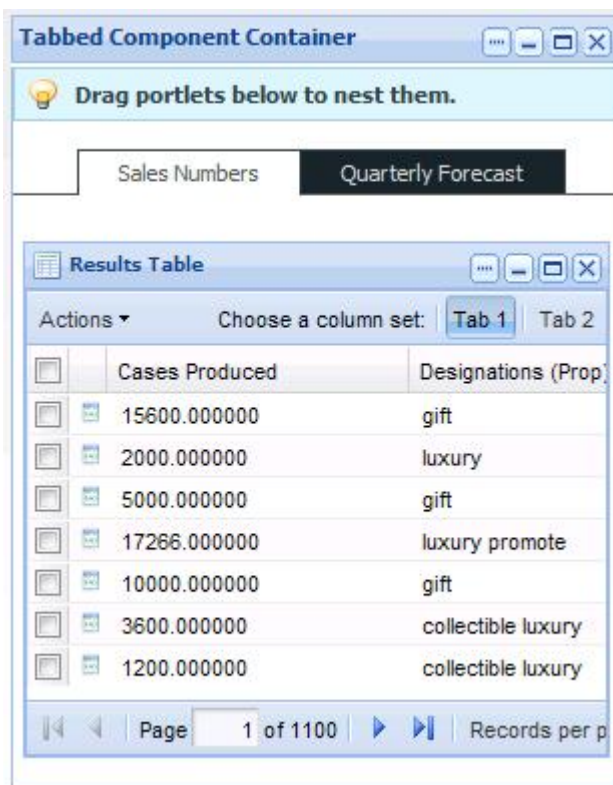
```
Target: /web/spend/Analyze
Redirect: /web/spend/Analyze
```

To set the tab state, the user appends a hash mark (#) followed by the tab name and number to the page name. In the following example, the **Sales Numbers** tabbed component is set to tab 1:

```
Target: /web/spend/Analyze#Sales Numbers[1]
Redirect: /web/spend/Analyze
```

To set multiple tabs, you delimit the additional tabs with a double colon (::). In the following example and image, the **Sales Numbers** tabbed component is set to tab 1, and the **Quarterly Forecast** tabbed component is set to tab 2:

```
Target: /web/spend/Analyze#Sales Numbers[1]::Quarterly Forecast[2]
Redirect: /web/spend/Analyze
```



Creating page transitions using component IDs

In order to implement certain kinds of page transitions, you need to use component IDs, rather than names.

Because the double colon (::) is part of the page transition syntax, you should avoid using it in your tab titles. Likewise, you should avoid multiple tabbed component containers with duplicate titles. If you cannot avoid these naming features, then you have to use a component's ID rather than its name when defining a page transition target.

To find a component's ID, hover your mouse over the tab until the URL appears at the bottom of the screen, and then extract the `p_p_id_query` parameter from the URL. In the following example, the

tabbed component with ID `nested_tabs_INSTANCE_0CbE` is set to tab 2, while the tabbed component with ID `nested_tabs_INSTANCE_Ja6E` is set to tab 1.

Target: `/web/spend/Analyze#nested_tabs_INSTANCE_0CbE[2]::nested_tabs_INSTANCE_Ja6E[1]`

Redirect: `/web/spend/Analyze`



Chapter 16

Using Deep Linking

This chapter describes how to construct URLs that transition from an external site to a Discovery Framework target page.

About deep linking

Deep linking allows users to construct URLs that take them from an external site to a target page in the Discovery Framework. In effect, these URLs act as external bookmarks to the Discovery Framework.

Pages targeted by deep linking inbound portal URLs must already exist within the Discovery Framework, and contain zero or more Endeca components bound to one or more data sources. The URLs encode the data source filter state for any or all of the data sources defined in the application. Any query filter supported by the data source JSON files is supported in the URL. Navigation by dimension values is also supported.

Users construct these URLs based on a query URL template, which define filters for a number of data sources for that user request.



Note: Only attribute keys (not display names) will work in deep links. Any deep link created using display names will fail.

Deep linking URL format

Deep linking uses the `deeplink` URL query parameter to create external bookmarks to the Discovery Framework.

The general form of the URL is:

`http://portalhost:port/path/to/page?deeplink=JSON-ARRAY-OF-DATASOURCE-STATES`

The following is a record filter example:

```
http://localhost:8080/web/guest?deeplink=[{"default":
{"queryFunctions":[{"class":"RecordFilter","recordFilter":
"Designation:BestBuy"}]}]
```

Deep linking syntax

The `deeplink` parameter value is a JSON array of data source states keyed by data source ID.

Data source state consists of query functions and `NavByValue` JSON objects. The syntax is as follows:

```
deeplink=[
  { "dataSourceId":
    {
      "queryFunctions": [data source json as defined elsewhere],
      "navByValue": {
        "dimension1": "dimval1",
        "dimension2": "dimval2",
        "dimension3": ["dimval", "dimval", "dimval"]
      }
    }
  },
  { "dataSourceId2": ... },
  { "dataSourceId3": ... }
]
```

Below is an example of the JSON array:

```
http://localhost:8080/web/guest?deeplink=[
{
  "default": {
    "queryFunctions": [
      {
        "class": "RecordFilter",
        "recordFilter": "Designation:Best Buy"
      },
      {
        "class": "RangeFilter",
        "property": "P_Price", "rangeOperator": "BTWN", "value1": "50", "value2": "100"
      }
    ],
    "navByValue": {
      "Wine Type": "Red",
      "Region": "Other France",
      "Body": ["Fresh", "Full", "Rich", "Ripe"]
    }
  },
  "v7-wine": {
    "queryFunctions": [
      {
        "class": "RecordFilter",
        "recordFilter": "Designation:Best Buy"
      }
    ],
    "navByValue": {
      "Wine Type": "Red",
      "Region": "Other France",
      "Body": ["Fresh", "Full", "Rich", "Ripe"]
    }
  }
}
]
```



Note: For more information about data source syntax in general, see Chapter 3 in this guide.

Using NavByValue filters

A `NavByValue` filter is a convenient way to refine a query using dimension values.

To do the same refinement with a `RefinementFilter`, you would need to look up the dimension value ID for each dimension value in the refinement and pass the IDs into the `RefinementFilter`. Further, there are different ways of looking up dimension value IDs depending on the version of the MDEX Engine being queried. The `NavByValue` filter handles these steps for you.

A `NavByValue` filter allows the deep link to both:

- Refine the query using dimension values instead of dimension value IDs.
- Reflect query refinements in other components on the page. This is similar to making refinements through the **Guided Navigation** component.

Deep linking examples

This topic contains a number of examples of deep linking URLs.

Record filter: The following URL applies a record filter for wines from the 2000 vintage:

```
http://localhost:8080/web/guest/my-page?deeplink=[{"default":
{queryFunctions:[{"class":"RecordFilter","recordFilter":"Vintage:2000"}]}}]
```

Range filter: The following URL applies a range filter of price between 50 and 100:

```
http://localhost:8080/web/guest/my-page?deeplink=[{"default":
{queryFunctions:[{"class":"RangeFilter","property":"P_Price",
"rangeOperator":"BTWN","value1":"50","value2":"100"}]}}]
```

Page and tab transition with a record filter: This URL combines a tab transition that points to a specific nested tab on the target page with a record filter:

```
http://localhost:8080/web/guest/my-page?com.endeca.discovery.
pageTransitionTabState=Charts[2]&deeplink=[{"default":{"queryFunctions":
[{"class":"RecordFilter","recordFilter":"Vintage:1999"}]}}]
```

Resetting data source query state

To reset a data source's query state back to its base state, you pass it an empty filter.

The following URL is an example of a `deeplink` parameter passing an empty array.

```
http://localhost:8080/web/guest/my-page?deeplink=[{"default":{}}]
```

How security is handled with deep linking

Authentication and page authorization for the deep linking feature are handled by the Liferay Portal.

- Authentication, if required, is handled before deep link processing.
- Page authorization is handled after deep link processing. Deep link processing is performed before redirecting to a URL-specified page or tab.

Index

A

- Action menu, configuring 86
- adding data sources 23
- Advanced Visualization component
 - about 97
 - configuring 99
 - saved data in bookmarks 75
 - using 97
- aggregated records, configuring 24
- Analytics
 - about 69
 - configuring Results Table for 83
 - power user requirements 69
- attribute sets
 - adding attributes 61
 - configuring for Guided Navigation component 130
 - configuring for Record Details and Compare tables 86, 116
 - creating 59
 - deleting 61
 - removing attributes from 60
- attributes
 - about 57
 - accessing Attribute Settings component 58
 - Attribute Settings dialog 58
 - configuring for Guided Navigation component 130
 - configuring for Record Details and Compare tables 86, 116
 - configuring formatting output 87, 117
 - display names 62
- authentication in deep linking 161

B

- Bookmarks component
 - about 71
 - configuring 73
 - saved data for components 75
 - saved data in bookmarks 75
 - setting up mail server 74
 - using 71
- Breadcrumbs component
 - about 131
 - configuring 132
 - saved data in bookmarks 75
 - using 131

C

- Chart
 - horizontal bar chart type 105
 - line bar combo chart type 108

- Chart (*continued*)
 - line chart type 106
 - pie chart type 107
 - stacked vertical bar chart type 105
 - vertical bar chart type 104
- Chart component
 - about 100
 - configuring 102
 - provided chart styles 104
 - saved data in bookmarks 76
 - using 100
- company logo, changing 40
- Compare component
 - about 114
 - configuring 115
 - configuring Attribute List 86, 116
 - using 114
- Component Container 151
- component IDs with page transitions, using 156
- components
 - adding 37
 - Advanced Visualization 97
 - Attribute Settings 58
 - Bookmarks 71
 - Breadcrumbs 131
 - changing data source for 24
 - Chart 100, 114
 - Cross Tab 109
 - Data Source Bindings 48
 - Data Sources 46
 - editing 39
 - Find Similar 144
 - Framework Settings 49
 - Guided Navigation 125
 - Metrics Bar 119
 - Performance Metrics 52
 - Range Filter 141
 - Record Details 84
 - renaming 39
 - Results Table 77
 - Results Table Beta enhancements 90
 - Sample Endeca Portlet 54
 - Search Box 134
 - Tabbed Component Container 65
 - Tag Cloud 121
- Control Panel
 - adjusting logging level in 41
 - overview of functions 16
- creating new QueryFunction classes 30
- Cross Tab component
 - about 109
 - configuring 110
 - using 110

D

Data Source Bindings component
 about 48
 configuring 48

data sources
 adding 23
 changing for a component 24
 configuring aggregated records 24
 connecting to secured MDEX Engine 25
 default 23
 obtaining results 31
 relationship between parent and child sources 26
 role-based security 26
 samples 21
 syntax example 22
 troubleshooting 23

Data Sources component
 using 46
 about 46

deep linking
 about 159
 examples 161
 NavByValue filters 161
 resetting data source query state 161
 security 161
 syntax 160
 URL format 159

default data source 23

Discovery Framework
 about 11
 modifying logging 41
 QueryFunction classes 27

display names for attributes 62

Dock, about 15

E

Eclipse, adding jar to 31

Endeca components, adding 37

examples, data source 22

F

Find Similar component
 about 144
 configuring 147
 saved data in bookmarks 76
 using 145

Framework Settings
 about 49
 changing 50

friendly URL name for pages 35

G

Grid Preferences Editor 79

Guided Navigation component
 what is saved in a bookmark 76

Guided Navigation component (*continued*)
 about 125
 configuring 128
 configuring attribute list 130
 using 127

H

horizontal bar chart example 105

HTTPS connection to MDEX Engine 25

I

inheritSecurity filter 26

J

JSON data source syntax 22

L

Languages component 152

layout for page, changing 36

Liferay components
 characteristics of 151
 Component Container 151
 Languages 152
 Links 152
 Web Content 153
 Web Content Display 153
 Web Content List 153
 Web Content Management 152
 Web Proxy 154

Liferay Portal
 about 15
 documentation 19

line bar combo chart example 108

line chart example 106

Links component 152

log4j.properties files 44

logging
 adjusting verbosity 41
 setting up 43
 ways to modify 41

logo, changing 40

M

mail server for Bookmarks support, setting up 74

MDEX Engine sample data sources 21

Metrics Bar component
 about 119
 configuring 120
 saved data in bookmarks 76
 using 120

modifying portal-log4j.xml 43

N

NavByValue filters for deep linking 161

O

obtaining data source results 31
obtaining more information about the MDEX Engine 13

P

page authorization in deep linking 161
page transitions
 about 155
 syntax 155
 using component IDs 156
pages
 adding 34
 applying themes 33
 changing layout 36
 deleting 35
 friendly URL name 35
parent and child data sources 26
parentDataSource filter 26
Performance Metrics
 about 52
 using 53
pie chart example 107
plugins, about 18
portal-log4j.xml, modifying 43
power users, about 11

Q

query function classes, implementing 30
QueryFunction classes
 in the Discovery Framework 27
 adding new .jar to the Eclipse build path 31
 creating your own 30
 custom 31

R

Range Filter component
 about 141
 configuring 142
 saved data in bookmarks 76
 using 141
Record Details component
 about 84
 configuring 85
 configuring Attribute List 86, 116
 saved data in bookmarks 76
Results Table (BETA Enhancements) component
 saved data in bookmarks 76
Results Table Beta enhancements component
 about 90
 editing 92

Results Table Beta enhancements component
 (continued)
 initial configuration 91
Results Table component
 using 78
 about 77
 configuring 79
 configuring for Analytics statements 83
 saved data in bookmarks 76
rolePermissions filter 26

S

sample data sources 21
Sample Endeca Portlet component 54
Search Box component
 using 134
 about 134
 configuring 135
 dependencies 134
 enabling type-ahead suggestions 139
 saved data in bookmarks 76
 sorting and removing columns 138
secured MDEX Engine, connecting to 25
security in deep linking 161
securityEnabled filter 26
securityFilters filter 26
setting up log4j 43
stacked vertical bar chart type 105
syntax
 deep linking 160
 page transitions 155

T

Tabbed Component Container
 about 65
 configuring 66
 saved data in bookmarks 76
 using 65
Tag Cloud component
 about 121
 configuring 123
 saved data in bookmarks 76
 using 122
themes, applying 33
troubleshooting data sources 23
type-ahead suggestions, configuring Search Box 139

U

URL format for deep linking 159
using
 Data Source components 46
 Results Table 78
 Chart component 100
 Find Similar 145
 Guided Navigation 127
 Range Filter 141

V

verbosity for logging 41
vertical bar chart example 104
view transitions, See page transitions

W

Web Content component 153
Web Content Display component 153
Web Content List component 153
Web Content Management in Liferay 152
Web Proxy component 154