HITACHI
Inspire the Next

Developing with Pentaho CTools

HITACHI
Inspire the Next

Change log:

Date	Version	Author	Changes
10/12/2017	1.0	Miguel & Megan	First edition

Contents

This page intentionally left blank.

Overview

This document covers some best practices on using Pentaho Community Tools (CTools) to create dashboards from mockups.

Our intended audience is business analysts, web designers, and others who are interested in displaying their data in a way that is simple, yet informative.

The intention of this document is to speak about topics generally; however, these are the specific versions covered here:

Software	Version(s)
Pentaho	6.x, 7.x, 8.0
Updated Web Browser	Internet Explorer, Chrome, Firefox, Safari
Developer Tools on Browser	Current

The Components Reference in Pentaho Documentation has a complete list of supported software and hardware.

Before You Begin

Before beginning, use the following information to prepare for the procedures described in the main section of the document.

Terms You Should Know

Here are some terms you should be familiar with:

- **CDF**: The Community Dashboard Framework opens in your browser and allows you to use JavaScript and CSS to build your dashboard. Using CDF directly gives you more flexibility, but also requires you to have more coding knowledge.
- **CDA**: The Community Data Access plugin acts as an interface between data connections and the CDF and/or CDE, provides data to dashboards, and can output many different data types.
- **CDE**: The Community Dashboard Editor, part of the Pentaho User Console (PUC), is a plugin graphical tool for creating, editing, and previewing dashboards. To install CDE, you must also install CDF, because CDE is built on top of CDF.
- **CCC**: The Community Charts Components, built on Protovis, is a chart library that lets you customize your dashboard beyond the basics.

Other Prerequisites

This document assumes that you have basic knowledge about JavaScript, JQuery, and Cascading Style Sheets (CSS). It also assumes that you have already installed Pentaho and enabled CTools.

© Hitachi Vantara Corporation 2017. All Rights Reserved

Use Cases

Use cases for CTools could include the following:

- *A healthcare company needs highly customized dashboards to track health data across countries for research for a new blood pressure medication.*
- *A corporation needs flexible dashboards to visualize sales information in different global territories to see if new products are succeeding in the marketplace.*
- *A city government needs to see utility usage data on the fly to load-balance electrical transformers and respond to potential service interruptions.*

© Hitachi Vantara Corporation 2017. All Rights Reserved

Getting Ready to Work with CTools

Creating a dashboard involves both preparation and development. You will need a thorough understanding of your data and a visualized outcome to create an effective dashboard using CTools.

We recommend building a mockup of your dashboard first, then doing a functional breakdown to tie the pieces together.

Details on these topics appear in the following sections:

- Building the Mockup
- Functional Breakdown of Dashboard

Building the Mockup

To build your mockup, first list what your initial requirements are, including the purpose, intended use, and audience of the dashboard.

Gathering Requirements

During your requirements gathering, make note of the answers to the following questions, as well as any other details that may relate to your situation:

1. Who will be using this dashboard?
    a. What role(s) do the user(s) have?
    b. What information do they need to be able to see and work with?
2. Is this dashboard replacing any system or reports that are currently being used?
    a. Can you see the system or reports currently in use to compare?
    b. Should you respect the existing user interface (UI)?
    c. If this is a completely new dashboard, is something already drawn or rendered to use as a starting point?
3. What information should be visible on the dashboard?
    a. At what level or granularity should the data be displayed?
    b. If several detail levels are required, might performance be negatively impacted?
4. How is the data being filtered?
    a. What filters are necessary for the dashboard?
    b. Which sections of the dashboard need filtering?
5. What is the best visualization for the data, so the user can understand it?

Mockup Recommendations

Now that you have all the available background information about your dashboard, try creating a mockup.

We recommend you have user experience (UX) experts create the mockup, and/or split efforts between a design team and development team, or a front-end person and a back-end person.

© Hitachi Vantara Corporation 2017. All Rights Reserved

- Use anything from pencil and paper sketches up to building a wireframe model that shows what each click does.
- Keep the interface as simple as possible, so that users will be able to identify what they are looking at and how to use it to find what they are looking for. Make the dashboard as visually appealing as possible, without overcomplication.
- Seek feedback from people who will be using the dashboard.
- Once you have created the mockup, consider each piece of the dashboard in turn, and try to match the piece to a function of CTools that already exists. A functional breakdown can help with that.

Functional Breakdown of Dashboard

Next, create a functional breakdown of the dashboard, to help you connect the front end and back end of your operation. Where the mockup reflected a business need, you now must consider funneling that business need through the technological environment to reach your goal. Create a functional breakdown by determining what goes where, and how.

The main advantages to creating a functional breakdown and sharing it early are:

- **Anticipating change requests**, because you will be able to identify any elements that were overlooked during the first scoping for the dashboard.
- **Saving time and effort** by determining which of the parts of your dashboard can be created with out of the box components from CTools, and which ones may require customization of some out of the box elements, or may even require custom elements from scratch. The more items on your dashboard that you create using unmodified or even modified out of the box components, the better, because the implementation will be easier.

For example:

Table 1: Potential CTools Components for Dashboard Functional Breakdown

Item	Components and Parameters
1	**Navigation bars**
Type:	Multi button component
Priority:	5
Listeners:	None
Parameters Written:	`navigationBarParam`
Datasource/Values array:	`navigationBarQuery`
Notes:	click: opens a link to another dashboard passing the `topicWeekParam`
2	**Topic of the week info**
Type:	Template component
Priority:	5
Listeners:	None
Parameters Written:	`topicWeekParam`
Datasource/Values array:	`topicWeekInfoQuery`

© Hitachi Vantara Corporation 2017. All Rights Reserved

Item	**Components and Parameters**
Notes:	
3	**Topic of the week chart**
Type:	Chart
Priority:	5
Listeners:	None
Parameters Written:	
Datasource/Values array:	`topicWeekChartQuery`
Notes:	query filtered by `topicWeek`
4	**Discover Research**
Type:	Table component
Priority:	5
Listeners:	None
Parameters Written:	
Datasource/Values array:	`discoverResearchTableQuery`
Notes:	Custom add-in that will insert an image whose URL may be retrieved in the query

Functionality Considerations

Consider the functionality of your dashboard from a mobile standpoint, as well as from a regular desktop.

Creating clickable items that are too small may cause your dashboard to be inoperable from a mobile device.

The more purposeful time and effort you spend in the planning phases of the mockup and the functional breakdown, the less time and effort you will need to expend on the development part of your dashboard.

© Hitachi Vantara Corporation 2017. All Rights Reserved

CTools

CTools are a set of tools that allows you to create dashboards and use visualizations configurable through flexible and customizable options, performing every action and fulfilling every need for your organization. Details on these topics appear in the following sections:

- Installation
- Key Features
- Filters
- Charts
- Tables
- Add-Ins
- Template Component
- New Map Component
- CCC Visualizations
- CDA
- Embedding Your Dashboard
- Debugging

Installation

CTools comes installed in Pentaho, but disabled. Enable CTools using the Pentaho documentation. Further information is at the Pentaho Marketplace, which you can access either from that link or from your Pentaho User Console (PUC).

Key Features

- With CTools, you can completely customize the behavior and look and feel of your dashboards. You will need more time for development compared to when using drag-and-drop tools, but you will have the ability to fully customize the dashboards.
- CTools lets you create applications that take advantage of full stack and achieve results you cannot get with any other tools.
- CTools makes use of Pentaho **multi-tenancy** capabilities, the ability to serve multiple tenants on a unique server. This means that CTools dashboards can be accessed and manipulated by different groups of users with different needs and points of access.
- On CTools dashboards, **state transition** is mostly driven by parameter changes. Once the parameters have been altered, the components on the dashboard are updated there.
- With CTools, you can create visualizations by representing information from multiple data sources in a clear way.

© Hitachi Vantara Corporation 2017. All Rights Reserved

Filters

CTools contains out of the box filter components to use in your dashboard to let users choose subsets of information and drill down to more and more specific levels of data. Filter result values can also be used in cascading filters, allowing further scrutiny of the results. Several different types of filters are available, including:

Table 2: Potential CTools Components for Dashboard Functional Breakdown

Filter Type	Details
Select component	The user selects an option from a dropdown list, which focuses the component on a specific subset of data. Alternatively, put in a multi-select component, which allows the user to select multiple options at the same time.
Filter component	Filters can also enable a user to select one or more options from a list, but they also let the user choose “all” or “none” – that is, if nothing is selected, the filter is simply not applied. The filter component is more extensible and user-friendly than the select component.
Multi-button component	This component can be used as a selector. Set a data source or fixed values, and allow or disallow multiple selections as needed.
Date range component	This component lets you select a start and end date for the information you want to view, or choose a common view like previous month, or month to date.

Charts

Many kinds of charts are available as out of the box CTools options, including bar charts and line charts. Combining the chart types with selectors and filters makes the charts dynamic and useful for deep dives into information.

The charts are highly configurable, leading to a wide variety of possibilities for your chart displays.

When you are creating your charts, use the custom dashboards site to test your charts.

A more flexible option is for you to test them on the CCC documentation page. There, you can choose your chart type and alter any parameters or styles you want, and see what the outcome is.

We recommend you choose a chart that looks the closest to your desired outcome, and then tweak that chart until your results match your desired outcome.

Clicking on properties brings you to the documentation page which will show you what options are available for that property. For example, clicking on a color brings you to the documentation page that shows you how to use hexadecimal, RGB, or RGBA codes, or even SVG color names, to set your color options.

Extension Points appear further down on the CCC documentation page, allowing you to change properties like `textAngle` and `textDecoration`.

© Hitachi Vantara Corporation 2017. All Rights Reserved

Tables

With the right setup and styling, you can do nearly anything with a table, much more than simply listing items. Even if it is just a one-row table, it already has formatting, so it can be convenient for displaying a great deal of different types of information. Tables are also the go-to item for sorting columns.

Standard tables are simple to navigate. They usually include a search as you type function, short and long pagination controls, page length change, and server-side page processing. Column types (add-ins) allow you to run code while drawing your table. More information on existing add-ins on the BI server is available at Public > Plugin Samples > CDE > Require Samples > Add-Ins.

Add-Ins

If standard options suit your use case, you will not need add-ins. However, they can be a powerful customization feature to fine-tune your dashboard if you need to do so. You can usually identify an add-in through its properties, because `addIn` will appear in some part of the property.

Add-ins are available for tables, template, and filter components:

Table 3: Add-Ins for Table Component

Add-In	Details
`sparkline`	This is an implementation of the JQuery Sparkline plugin. Values separated by "," are represented in a small chart.
`pvSparkline`	A simple implementation of sparklines in Protovis that provides only lines, no bars or pie charts.
`dataBar`	Turns a numeric cell into a bar. By default, `maximum` is the column's absolute max and `minimum` is `0`.
`trendArrow`	Adds a trend arrow to a numeric cell.
`hyperlink`	Turns a cell into a clickable link. Specific styling can be formatted with CSS. If a `regexp` is given, it will only be applied if the text matches it.
`circle`	Turns a cell into a circle that can display data as either circle radius or color. By default, this displays a black, 4px radius circle. The `radius` and `color` properties can be changed either to fixed values or functions.
`formattedText`	A general-purpose text-formatting add-in that defaults to a simple format that preserves the `colFormat` formatting.
`cccBulletChart`	Renders a CCC bullet chart inside the table cell. Please refer to the CCC documentation for particulars on the chart's configuration.
`groupHeaders`	Groups rows using the value of the `addIn` column and creates a header row over the group. The `addIn` column is hidden by default.

© Hitachi Vantara Corporation 2017. All Rights Reserved

Table 4: Add-Ins for Template Component

Add-In	Details
`clippedText`	Use this when you have text larger than the space available. It will show a tooltip with full text when a user hovers over the element.
`trendArrow`	Displays up or down arrows in red or green depending on the value and options applied.
`formatted`	This add-in has many options for custom formatting.
`sparkline`	An implementation of the JQuery Sparkline plugin. Values separated by "`,`" are represented in a small chart.
`hyperlink`	Provides links to any URL that is recognized by the HTML and the browser.
`localized`	Uses CDF/CDE internationalization/localization capabilities.
`bubble`	Draws a bubble where the size of the bubble is the relationship between the value and the minimum and maximum values of all rows.
`bulletChart`	Creates a CCC bullet chart.
`cccChart`	Allows any CCC chart to be displayed. Use caution: it is tricky to use and may cause your dashboard to become slower.

Table 5: Add-Ins for Filter Component

Add-In	Details
`notification SelectionLimit`	Shows a notification that the selection limit has been reached.
`sumSelected`	Calculates the sum value for the list of selected items.
`selectedOnTop`	Keeps selected item(s) on top.
`insertionOrder`	Keeps the insertion order
`sortByLabel`	Sorts items alphabetically by their labels.
`sortByValue`	Sorts items numerically by their values.
`sumValues`	Calculates the sum value for the item values.
`template`	Applies a Mustache template for you to customize the item's HTML.
`accordion`	Causes a group's filters to behave as an accordion.

You apply an add-in to a slot, which is part of filter processing that comes between `postFetch` and `postExecution` functions. Each add-in has one or more slots to which it can be applied:

© Hitachi Vantara Corporation 2017. All Rights Reserved

Table 6: Add-Ins and Available Slots

Add-In	Post Update	Root Header	Root Selection	Group Selection	Item Selection	Root Footer	Group Sorting	Item Sorting
`notificationSelectionLimit`			X					
`sumSelected`			X	X				
`selectedOnTop`							X	X
`insertionOrder`							X	X
`sortByLabel`							X	X
`sortByValue`							X	X
`sumValues`			X	X				
`template`	X		X	X	X	X		
`accordion`	X							

Alter specifics of the add-ins using calls with `setAddInOptions` on the `preExecution` callback of the component: `this.dashboard.setAddInOptions`.

Template Component

The template component is a clean page where you can create your own HTML to customize visualizations more specifically. Though you can still use the query component instead, you might want to use the template component in a situation where you want to minimize the amount of custom code you must write. Another advantage to the template component is that you are able to maintain the HTML template component independently, and update it separately, so the UX people do the HTML, and another person takes care of the logic aspect.

There are two different template engines to use:

- Mustache: uses tags instead of logic
- Underscore: more complex, can use logic, and has more flexibility than Mustache

Use Mustache for a simpler project, and Underscore if you require more complexity.

New Map Component

Add a new map component to your dashboard using Google Maps or Open Layers. Maps render information using either shapes or markers, not both at the same time.

© Hitachi Vantara Corporation 2017. All Rights Reserved

Table 7: Shapes and Markers

Shapes	Markers
• Render geographical information • Render nongeographical information • Require creation of shape file	• Pinpoint actual items • Easier to use

Whether you use maps in Analyzer or CTools, they may require a valid API key.

Community Charts Components (CCC) Visualizations

Using CCC charts on dashboards is a great way to add flexibility to your CDF/CDE projects, because it lets you customize chart properties to increase usability and aesthetics at the same time. You can experiment with code by using the CCC website, updating code, and then applying it to see if your changes are valid code and if they make your charts appear the way you want them to.

Use a CDF/CDE created CCC chart on a dashboard by adding the correct object names and types to the code. Here are the CCC components usable on CDF/CDE dashboards, along with examples of what some of them look like. Remember, the look of each component is customizable:

Table 8: CCC Charts

CCC Component	Chart Type
`cccAreaChartComponent`	Area chart
`cccBarChartComponent`	Bar chart
`cccBoxplotChartComponent`	Boxplot chart
`cccBulletChartComponent`	Bullet chart
`cccDotChartComponent`	Dot chart
`cccHeatGridChartComponent`	Heatgrid chart
`cccLineChartComponent`	Line chart
`cccMetricDotChartComponent`	Metric dot chart
`cccMetricLineChartComponent`	Metric line chart
`cccNormalizedBarChartComponent`	Normalized bar chart
`cccParCoordChartComponent`	Parallel coordinates chart
`cccPieChartComponent`	Pie chart
`cccStackedAreaChartComponent`	Stacked area chart
`cccStackedDotChartComponent`	Stacked dot chart
`cccStackedLineChartComponent`	Stacked line chart
`cccSunburstChartComponent`	Sunburst chart
`cccTreemapAreaChartComponent`	Treemap area chart
`cccWaterfallAreaChartComponent`	Waterfall area chart

© Hitachi Vantara Corporation 2017. All Rights Reserved

Make sure to set the mandatory properties for your CCC charts. These include:

- `name`: The name of your chart
- `dataSource`: The datasource for your chart
- `htmlObject`: Specifies where on the dashboard the components will be rendered, an element of your HTML
- `height`: Specifies the height of the object in pixels

It may also be useful for you to set:

- `seriesInRows`: If true, `seriesInRows` reads the series from the rows; otherwise it reads from the columns.
- `crosstabMode`: Set `false` if your query is dependent upon one variable, and `true` if it is dependent on multiple variables.
- `timeseries`: Set `true` to present values over time (continuous) and `false` to present discrete values.

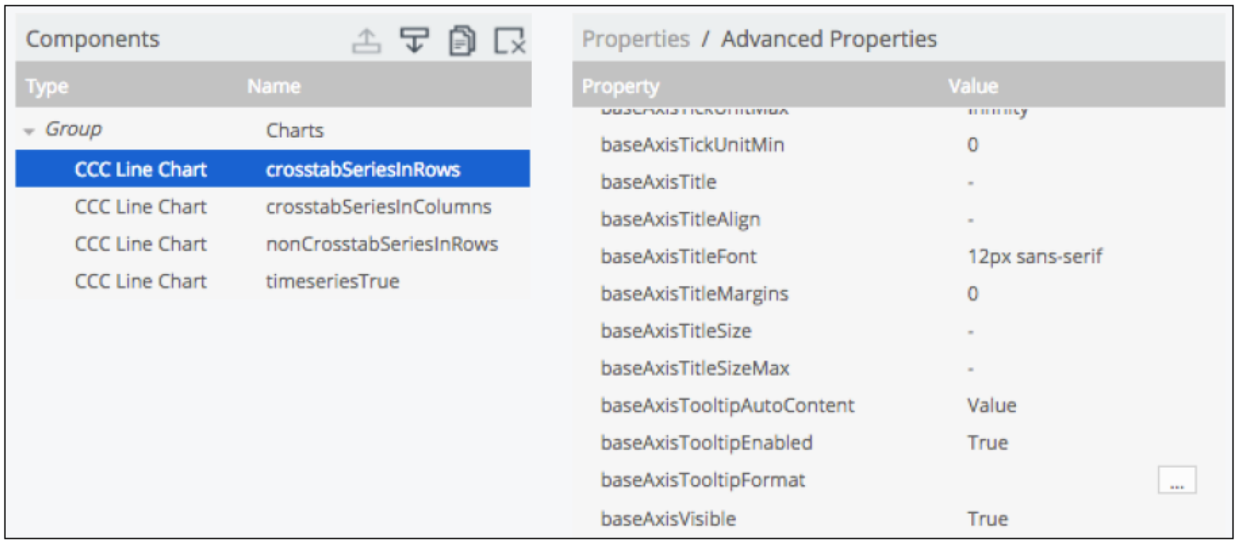
CCC can receive data as relational or crosstab, and will translate received crosstab data to a relational structure:

Table 9: `crosstabMode` and `seriesInRows` Settings

crosstabMode	**seriesInRows**	**Behavior**
`true`	`true`	The column titles will act as category values, while the series values are represented as data points of the first column.
`true`	`false`	The column titles will act as series values, while the category/category values are represented as data points of the first column.
`false`	`true`	A row corresponds to exactly one data point. Two or three columns can be returned. When three columns are returned, they are in the order: category, series, data.
`false`	`false`	A row corresponds to exactly one data point. Two or three columns can be returned. When three columns are returned, they are in the order: series, category, data.

© Hitachi Vantara Corporation 2017. All Rights Reserved

You can also use CCC in the Community Dashboard Editor (CDE), which may not require you to use any code at all, depending on your customization needs. If the available value choices fit your requirements, no further coding will be necessary:



Components

Type	Name
Group	Charts
CCC Line Chart	crosstabSeriesInRows
CCC Line Chart	crosstabSeriesInColumns
CCC Line Chart	nonCrosstabSeriesInRows
CCC Line Chart	timeseriesTrue

Properties / Advanced Properties

Property	Value
baseAxisTickUnitMin	0
baseAxisTitle	-
baseAxisTitleAlign	-
baseAxisTitleFont	12px sans-serif
baseAxisTitleMargins	0
baseAxisTitleSize	-
baseAxisTitleSizeMax	-
baseAxisTooltipAutoContent	Value
baseAxisTooltipEnabled	True
baseAxisTooltipFormat	...
baseAxisVisible	True

Figure 1: CCC Charts in CDE

Extension Points

Extension points are additional options that are part of Protovis, but are not in CCC. Each chart has different extension points available to modify it, listed in the CCC documentation, below each chart type.

While building your chart, click on a visual element name. This brings you to a page where you can find the correct extension points to add in order to change what you want in the chart.

Community Data Access (CDA)

As one of the first CTools created, CDA was originally an interface between data connections and the CDF, although now you can use CDA in Report Designer as well. CDA is a flexible way for you to collect data for your dashboard, as it outputs data in many different types and has other configurable options that improve performance.

CDA uses a cache for data requests, and will only query final data sources if the answer to your query is not found in its cache (or if the cache is not enabled).

Creating CDA Files

Create CDA files by:

- Creating and editing an XML file
- Using the GUI in CDE to create and edit your file

© Hitachi Vantara Corporation 2017. All Rights Reserved

Table 10: CDA Data Sources

Data Source	Details
SQL databases	Get data from any source that uses SQL and can be reached using a JNDI connection or JDBC driver (`sql.jndi` or `sql.jdbc`).
Mondrian cubes	MDX connection types include: • `mondrian.jdbc` • `mondrian.jndi` • `olap4j.defaultolap4j`
Pentaho metadata	Use a Pentaho metadata schema and provide the metadata type, domain, path, filename, and a metadata query that will transfer the data to the dashboard.
Kettle transformations	Getting data from a Kettle transformation is easier if you are using data from multiple types or sources. Use CDA to move the data from a step in your Kettle transformation to your dashboard.
Scriptable data sources	Use dummy data sources to develop your dashboard so that you do not have to wait for your back-end team to provide a query.
XPath over XML	Get specific nodes from an XML file using this type of connection.
Compound queries	Use compound queries to join or make a union of two different queries.

CDA File Parameters

Use parameters to make queries more flexible. For example, instead of querying:

```
select * from customers where country in ('USA');
```

you can instead query:

```
select * from customers where country in (${country});
```

If you specify USA for `country` in the second query, the results of both queries will be the same, but now if you want to choose a different country, you can do so easily without having to write another whole query.

Set this up by using XML syntax similar to the following (more than one parameter can be defined within these tags):

```
<Parameters>
  <Parameter default="USA" name="country" type="String"/>
</Parameters>
```

This syntax lets you set up the parameter(s) and set default values for them for situations where no value is specified in a query.

It is even easier to set up parameters in MDX queries than in SQL, because any tool you need can pass a query as a parameter for Mondrian to execute. Mondrian's security will prevent any results being issued to unauthorized people making queries.

© Hitachi Vantara Corporation 2017. All Rights Reserved

You can also pass parameters to a Kettle transformation, by mapping dashboard parameters to their appropriate names within the transformation. For example:

```
<?xml version="1.0" encoding="utf-8"?>
<CDADescriptor>
<DataSources>
     <Connection id="1" type="kettle.TransformFile">
       <KtrFile>sample-trans.ktr</KtrFile>
       <variables datarow-name="myRadius"/>
       <variables datarow-name="ZipCode" variable-name="myZip"/>
     </Connection>
   </DataSources>
<DataAccess id="1" connection="1" type="kettle" access="public"
cache="true">
   <Name>Sample query on SteelWheelsSales</Name>
       <Query>Report Columns</Query>
       <Parameters>
         <Parameter name="myRadius" type="Integer" default="30"/>
         <Parameter name="ZipCode" type="Integer" default="32771"/>
       </Parameters>
   </DataAccess>
</CDADescriptor>
```

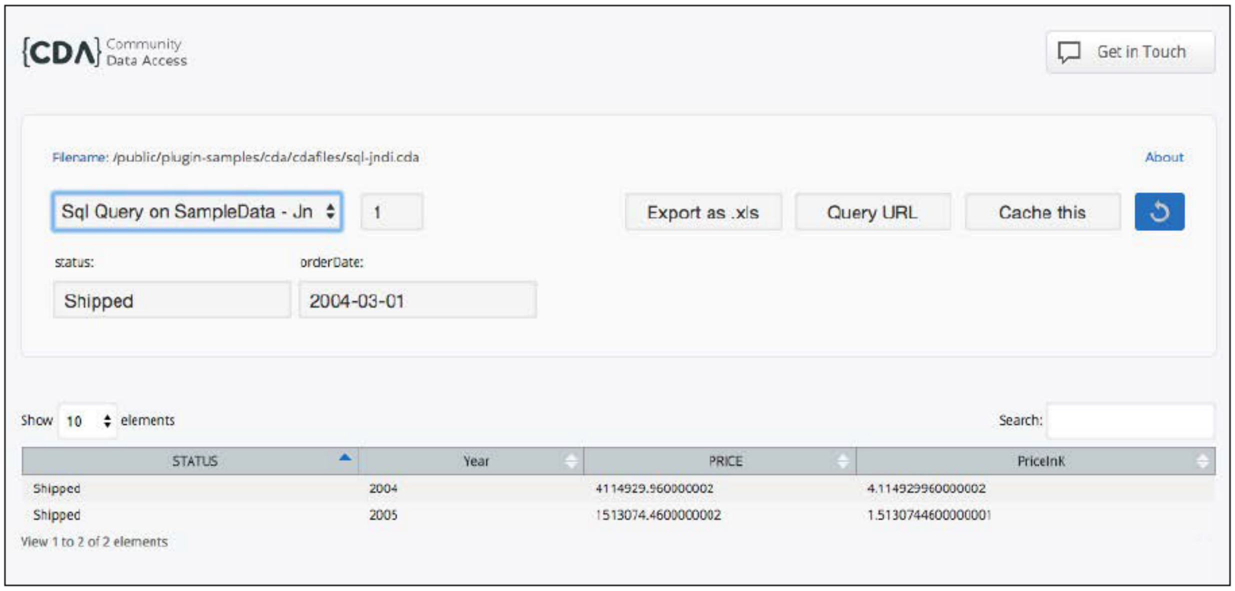
You can define parameters as private, where it will be set to a default even if it is specified as something else by the query:

```
<Parameters>
     <Parameter default="${[security:principalName]}" name="username"
type="String" access="private"/>
</Parameters>
<Query>SELECT * from Employee where id=${username}</Query>
```

© Hitachi Vantara Corporation 2017. All Rights Reserved

Previewing CDA Files

Once you have created and uploaded your CDA file, preview the file using the Pentaho User Console (PUC):



Filename: /public/plugin-samples/cda/cdafiles/sql-jndi.cda

Sql Query on SampleData - Jn | 1

status: Shipped

orderDate: 2004-03-01

Show 10 elements

Search:

STATUS	Year	PRICE	PriceInK
Shipped	2004	4114929.960000002	4.114929960000002
Shipped	2005	1513074.4600000002	1.5130744600000001

View 1 to 2 of 2 elements

Figure 2: Previewing a CDA File Using PUC

You can also preview the file using the **Query URL** provided to you after you execute a query.

If you decide not to use CDE or CDF to build your dashboard, you can use CDA to move data into an external application by requesting using Web API.

Use the URL `$BASE_URL/$WEBAPP/plugin/cda/api/`, where `$BASE_URL` is your protocol, hostname, and port; `$WEBAPP` is your web application name used on Tomcat; and the default web app is `pentaho`.

The following endpoints are available for your use:

Table 11: Add-Ins for Template Component

Add-In	Details
`getCdaList`	Gets a list of all CDA files available in the repository. No need to specify parameters.
`listQueries`	Lists all queries available in CDA file. **Required**: specify `path` parameter for the path to where the CDA file is from which the queries will come. **Optional**: specify `outputType` parameter (default is `json`, specify for `xml`).
`listParameters`	Lists all parameters defined in a particular query. **Required**: • Specify `path` for the path to where the CDA file is from which the query will come.

© Hitachi Vantara Corporation 2017. All Rights Reserved

Add-In	Details
	• Specify `dataAccessId` for the query you want to use. **Optional**: specify `outputType` parameter (default is `json`, specify for `xml`).
`doQuery`	Makes a call to a query and returns the result. **Required**: • Specify `path` for the path to where the CDA file is from which the query will come. • Specify `dataAccessId` for the query you want to use. **Optional**: • Specify `outputType` parameter (default is `json`, specify for `xml`, `csv`, `xls`, or `html`). • Specify `paginateQuery` with a Boolean value. If `true`, also specify `pageStart` (which page to start with) and `pageSize` (how many rows). • Specify `bypassCache` as `true` to bypass the cache and force a new request to the database. Use `sortBy` and a column list if necessary.
`clearCache`	Clears CDA cache.
`previewQuery`	Opens CDA previewer.
`editFile`	Opens CDA editor for a query. Required: specify `path` parameter for the path to where the CDA file is from which the query will come.
`manageCache`	Opens the cache manager, allowing you to change aspects of how the cache is used.

Embedding Your Dashboard

It is common to want to embed a dashboard in a third-party application, and you can do that with your CDF and CDE dashboards by using RequireJS. Keep in mind the Pentaho best practices for embedding content into third-party web applications.

For a CDE dashboard, you will need to create it before you embed it, but for a CDF dashboard, you can create and embed it at the same time.

First, you must make sure you avoid any cross-domain request issues. To do that, simply add this XML property to `settings.xml` in either a CDE or CDF dashboard:

```
<allow-cross-domain-resources>true</allow-cross-domain-resources>
```

© Hitachi Vantara Corporation 2017. All Rights Reserved

Embedding from Community Dashboard Framework (CDF)

Embed a CDF dashboard in an HTML page simply by including a script that requests CDF from a Pentaho Server that has CDF installed:

```
<script type="text/javascript"
src=http://<server>/<webapppath>/plugin/pentaho-cdf/api/cdf-embed.js>

</script>
```

- `<server>` should be the server name or IP address, and include the port if it is not 80.
- `<webapppath>` should be the web app name. It defaults to Pentaho, but can be changed.

Then, because a CDF dashboard is just JavaScript code, you can insert that code into your page and it will work. Your `.xcdf` file can be called as a script.

Embedding from Community Dashboard Editor (CDE

To embed a CDE dashboard, request it in this way:

```
<script type= "text/javascript"
src=http://localhost:8080/pentaho/plugin/pentaho-cdf-dd/api/renderer/cde-
embed.js>
```

Then, embed it by using RequireJS to include the dashboard as a module, in one of these two ways:

1. Directly point to the `getDashboard` endpoint from CDF:

```
../pentaho/plugin/pentaho-cdf-
dd/api/renderer/getDashboard?path=/public/dash/sample.wcdf
```

2. Use the `dash!` RequireJS loader plugin:

```
../dash!/public/dash/sample.wcdf
```

Debugging

Here are some tips and tricks for testing out your dashboard prior to deploying it.

- Use a site like Can I Use to see whether each feature of your dashboard will work on the browsers necessary.
- Try out developer tools for browsers to inspect and edit CSS, HTML, JavaScript, and network requests:
  - Chrome
  - Firefox
  - Microsoft
  - Safari
- Make sure you add the debug parameter to your dashboard so you can check the code, as `?debug=true` if you have no other parameters, or as `&debug=true` if you do have other parameters.

© Hitachi Vantara Corporation 2017. All Rights Reserved

Related Information

Further training and resources are available on the topic of CTools and what you can do with it.

- Pentaho Courses available:
  - CT1000: CTools Fundamentals (Pentaho University)
  - CT2000: CTools Advanced (Pentaho University)
- Learning Pentaho CTools by Miguel Gaspar (Scribd link)

Here is some other information that you may find helpful while using this best practices document:

- Developer tools
  - Chrome
  - Firefox
  - Microsoft
  - Safari
- Omnipotent
  - Sparkline
- Pentaho
  - Activating CDE
  - BA Content Embedding
  - CCC documentation
  - CDE Dashboard Overview
  - CDE Quick Start Guide
  - CTools Overview
  - Custom dashboards
  - Installation
  - Marketplace

© Hitachi Vantara Corporation 2017. All Rights Reserved