TrakSYS™ 11 Training Lab Manual Day 2

Revised August 23, 2019

Parsec Automation Corp.
180 North Riverview, No. 300
Anaheim, CA 92808 USA
Phone +1 714 996 5302
Fax +1 714 996 1845
www.parsec-corp.com

Lab 05 | Content Pages and Properties....................3
Lab 06 | Values Dictionary and Visual Pages....................4
Lab 07 | Content Parts....................11
Lab 08 | Content Part Features and Permissions....................14

This document is for training purposes only. Parsec Automation makes no warranties, express or implied, in this summary.

Lab 05 | Content Pages and Properties

Overview

Traksys has a powerful built-in user interface for discrete systems called the **System Discrete Real Time Overview** (abbreviated as **SDRO**). In this assignment you will learn how to add a **Content Page Definition** to your implementation**, configure its properties** and adjust how the page looks and feels. This lab will focus upon the **SDRO**, but the patterns can be expanded and repeated with the other standard content.

Estimated Time to Complete This Lab

30 Minutes

Details

The following sections describe the detailed steps to be completed for this lab.

1. Configure a System Overview Content Page

The **SDRO** is a standard TrakSYS **Content Page** designed to act as a real-time landing page for the plant floor operator. Add an **SDRO** page as a **Hub 3** under the **Training Section, Day 2, Lab 5**.

- Navigate to **Developer | Page Definitions**.
- Locate and select the **Training** Section in slice 2 to reveal a list of **Hub 1**s. Select **Day 2** from the list.
- Select the **Hubs** option in the **Related** menu in slice 3. Slice 2 should now contain a list of **Hub 2**s.
- Select **Lab 5** and select the **+ New Hub 3** menu option**.** Add a new **Content Page** of type the **TrakSYS/Application/Overview/System Discrete Real-Time Overview** with the following properties...

Name	Line 0
Key	Line0
Refresh Key	L0
System (Properties Tab)	123

To view your newly created page, navigate to **Training | Day 2 | Lab 5 | Line 0**. We will be adjusting the look and feel of this page, so keep it open in a separate tab.

2. Adjust the Page's Visibility Properties

Content Pages have the ability **hide/show certain parts** and **resize** based upon the screen width. The available options depend on the individual page, and the SDRO page has more options than most. Explore these options and their impacts by removing some **standard navigational links**.

- Navigate to **Developer | Page Definitions**.
- Locate and click the **Training Section** in the list of **Sections** in slice 2.
- Select **Day 2** from the list.

- Select the **Hubs** option in the **Related** menu in slice 3. Slice 2 should contain a list of **Hubs 2**s.
- Select the **Lab 5** Hub 2.
- Select **Line 0** and click **Edit** in slice 3. Set the following properties:

Title Visible (Visibility Tab)	False
Tree Visible (Visibility Tab)	False
Breadcrumbs Visible (Visibility Tab)	False

Use the **Preview** menu option from the item slice to **inspect the changes** made to Line 0 Overview. Or Navigate to the page using the sections and hubs.

3. Remove Content

Each page features additional visibility options based upon the content of the page. Navigate back to the **Line 0's Page Definition** in the **Developer** section and make the following changes…

- Navigate to **Developer | Page Definitions**.
- Locate and click the **Training Section** in the list of **Sections** in slice 2 to reveal its Hub 1s. Select **Day 2** from the list.
- Select the **Hubs** option in the **Related** menu in slice 3. Slice 2 should now contain a list of Hub 2s.
- Select **Lab 5** Hub 2 which will reveal your newly created **Line 0** Overview page.
- Select **Line 0** and click **Edit** in slice 3. Set the following properties:

Tasks (Content Tab)	No
Journals (Content Tab)	No
Production State (Charts Tab)	No

Use the **Preview** menu option from the item slice to **inspect the changes** made to Line 0 Overview.

4. Resize Content

Entire sections of the page can also be adjusted through simple configuration. Navigate back to the **Line 0's Page Definition** in the **Developer** section and make the following changes…

Center 1-12 (Properties Tab)	10
Right 1-12 (Properties Tab)	0

The numbers represent values used by a framework known as **Bootstrap**. This will be covered in more detail in a later lab. Navigate back to Line 0 Overview and **inspect the changes**. Use the **Preview** menu option from the item slice to **inspect the changes** made to Line 0 Overview.

Conclusion

In this exercise, you started creating a user interface for the **OCV** implementation. There is now a dedicated Operator Interface for viewing information on the line.

A Standard Content Page was added to your site showing how user interface pages can be quickly configured and implemented. You now have a better understanding of page development and the type of options available on Content Pages. It is recommended to become familiar with the pattern of navigation within this section. While this exercise was focused upon navigation within front-end development, it follows the same patterns that are used by all entities in the software.

Common Mistakes

None

Further Exploration

Below are optional activities that can be completed to further your understanding.

User Simulation

Line 0 is a simulation-based line that contains custom configuration for special manual interactions. Navigate to **Training | Day 2 | Lab 5 | Simulation** to see the tags used in this Line. From here, you can control the entire line, including the simulation of **Events**, **Counters**, **Jobs**, **Tasks**, and more.

The way the software reacts to certain changes in Tags may not make sense to you at this time without a further understanding of the configuration. Therefore, this screen may also be very useful to revisit after more topics have been covered.

User Interactions

On your SDRO page, try clicking through the various screens and buttons that are presented. Some of the functionality requires the **User Simulation** from the step above, but portions of it can be completed without needing to leave the standard pages. See the various pages that are linked to your SDRO and how a User can interact with the information that is shown.

Create and Start a **Job** using the **Planned** button in the top left.

Start and End an **Event** using the **User Simulation** screen. Then, **acknowledge** the event.

Trigger a **Task** by setting the **TR.L0.TASK.QC_OVERRIDE** tag to 1. Complete the Task.

Simulate KPI Counts by incrementing the various **TR.L0.COUNTER.<CounterName>** tags. This will take up to a minute to update on the main page if done correctly.

Lab 06 | Values Dictionary and Visual Pages

Overview

This assignment will explore two of the fundamental components used in building extensible user interfaces in TrakSYS, **Visual Page Definitions** and the **Values Dictionary**.

Most TrakSYS implementation require **solution-specific** user interfaces. The Visual Page Definition allows for a designer-based layout experience to select and arrange **Content Part**. The Content Parts can then be adjusted through **properties** in a similar pattern to Content Pages. This exercise focuses upon some of the basic tools used to start building **Visual Pages**.

Understanding the benefits of using Values Dictionary as a **communications repository** between Content Parts and script is necessary to understanding the value of the Visual Page toolset. Additionally, mastering the fundamental techniques for **viewing and interacting** with values from the dictionary will provide developers with significant development capabilities. This lab will emphasize both skillsets.

Estimated Time to Complete This Lab

30 Minutes

Details

The following sections describe the detailed steps to be completed for this lab.

1. Configure a Visual Page Definition

To complete the remaining steps in the lab, you will need to create a Visual Page Definition to work with. Create a testing and learning **Page** called **Hello World** as a **Hub 1** under the **Packaging Section**.

- Navigate to **Developer | Page Definitions | Training | Day 2**.
- Select the **Hubs** option from slice 3. Slice 2 should contain a list of **Hub 2**s.
- Select **Lab 6-8** and click **+ New Hub 3**. Select the **Visual** tab then choose the **Default Page**. Set the following properties:

Name	Hello World
Key	HelloWorld

Use the **Preview** menu option from the item slice to view your new page. You should see an empty page with your Page Title.

2. Configure a Grid Layout

Before adding **Content Parts**, rows and columns must be established that will structure the layout of the **Page**. Open the **Grid** editor for the **Hello World Page** and create the following layout…

2	8	2
2	8	2

- While viewing the **Hello World Page**, click the **Parts Editor** shortcut button in the lower left corner of the bottom bar in the browser (it looks like nine boxes). Once the Parts Editor is displayed, click the **Grid Editor** option in the top menu.
- Select the row **R1** (grey) and add 1 rows beneath it using the **+ Add Row Below** menu option in the right margin.
- Select the column **C1-12** in **R1** and **add 2 more columns** using the **+ Add Column Right** menu option.
- The widths of the column should match the Grid shown above (2-8-2).
- Repeat the last step and apply it to **R2** (Row 2).

Use the **Preview** menu option from the item slice to view your new page. Use should not see any visible changes to the page.

As a part of **Further Exploration**, you may view the HTML that is being created to match the grid layout that you configured. To do this, you will want to utilize the **Developer Tools** built into the browser (Press F12 to open the Developer Tools).

3. Add a Header Content Part

Once the Grid layout is complete, the Page is ready to have Content Parts added. Use the **Parts Editor** to add a basic **Header Part** to the only column 1 in row 1. Name the part **HeaderTop** and provide a dynamic header using **Expression Syntax.**

- From the **Grid** editor, click the **Parts** editor option in the top menu.
- Select the **Column 1** of **Row 1** and click **+ Add Part** menu option in the right margin.
- Select the **Header** entry in the tree view of **Content Parts** catalog (**bottom of catalog**).
- Set the following properties…

Part ID	HeaderTop
Text	Hello World

Once the part is added, view the page by selecting the **Preview** option in the top menu. The words “Hello World” should be shown in the top left of the page.

4. Make the Header Dynamic

Change the Header to accept a **Values Key** so that the text can be changed by using the **Values Dictionary**. While this initial change will not have any impact, the next steps will utilize this new Key to allow for a dynamically changing text.

- Open the **Parts** editor.
- Select the **HeaderTop** part and change the following property…

Text	**{CustomText[Hello World]}**

Once the part is changed, view the page by selecting the **Preview** option in the top menu. Notice “CustomText” **does not appear** on screen because it is being evaluated as Expression Syntax.

Since no value has been provided for the Key “CustomText”, its default value “Hello World” is displayed.

5. Add to the Values Dictionary using the QueryString

A common way for variables to be passed into a page is called the URL **QueryString**. Pass a value into the **Hello World Page** using the QueryString. This value will be used to populate the **Text** property of the **Header Content Part**.

- From the **Hello World Page**, edit the URL in the browser to include a **QueryString** key/value pair for “CustomText”. The new **URL** should be something like this…

  TS/pages/ocv/training/day02/lab6x8/helloworld/**?CustomText=Dynamic Hello World**

- Click enter in the browser **URL** bar to refresh the page.

Preview your page to see the result. Your header should now read **Dynamic Hello World**. This was done by mapping the **Key** of **CustomText** to your header. Now that you have defined “CustomText”, your header will no longer use its default value of “Hello World”.

6. View the Values Dictionary using Trace

Use the **Trace** display to examine the **Values Dictionary** for the **Hello World Page** and confirm that the **CustomText** variable has been added and is available for use.

- From the tab showing the end-user view of the **Hello World Page**, click the **eyeball icon** in the lower left corner of the page in the bottom bar. The **Page** should refresh and the **Trace** contents should show at the bottom.
- Scroll down and verify that there is an entry with a category of “**- -** Ets.Values” and a value of “ CustomText = 'Dynamic Hello World' ”.

7. Add to the Values Dictionary using Script

Another common technique for adding variables to the **Values Dictionary** is to retrieve and populate them programmatically from the **Script** editor. From the **Script** editor, add code to populate the **Values Dictionary** with the key **CustomIcon**, and the value **fa-thumbs-up**. The code should be placed in the **ContentPage_Init** method. This value will be used later to populate the **Icon** property of the **Header Content Part**.

- Navigate back to the tab with the **Part Editor** and open the **Script** editor. Alternatively, while looking at the **Hello World Page**, open the **Script Editor** by using the shortcut button (**C#**) in the lower left corner of the bottom bar in the browser. Be sure to open it in a different tab to avoid completing development in the **Preview Window**.
- Locate the **ContentPage_Init** method and add the following line of code…

```
protected override bool ContentPage_Init()
{
    this.Ets.Values["CustomIcon"] = "fa-thumbs-up";
    return true;
}
```

- Save the changes using the **Save** button in the **Script** editor menu bar.

To verify this, you may refresh the page and verify that the new “CustomIcon” value is being added to the Values Dictionary using the same step as before.

8. Map from the Values Dictionary to a Header Content Part

In some cases, setting **Content Part** properties with a fixed value at design time is insufficient. Variable values need to be mapped from various sources. Open the properties of the **HeaderTop Content Part** and set the **Icon** property to retrieve its value from the **Values Dictionary** as listed below…

- While viewing the **Hello World Page**, click the **Parts** editor shortcut button in the lower left corner of the bottom bar in the browser (it looks like nine boxes).
- Edit the **HeaderTop** part and set the following property…

Property	Value	Key
Icon		CustomIcon

- After setting the properties, **Save** the changes.
- Open the end user view of the **Page** by selecting the **Preview** option in the top menu.
- Re-edit the **URL** in the browser to include the **?CustomText=Dynamic Hello World QueryString** entry and refresh.

If successful, the **Hello World Page** should now render and the label should show the text of **From QueryString = Dynamic Hello World** and have a **thumbs up** icon.

Conclusion

In this exercise, you have learned how to create a basic **Visual Page**, arrange its **Grid** layout and add/configure a **Content Part**. You also explored a few ways to add variables to the **Values Dictionary**, examine its contents using **Trace**, and map entries to properties of a **Header Content Part**.

It is highly recommended that you explore and practice these basic techniques as they are the foundation and building blocks for future exercises. Understanding the **Values Dictionary**, mastering the use of the editors and navigating the hierarchy in the **Developer** section is critical to creating and managing TrakSYS user interfaces efficiently and effectively.

Common Mistakes

None

Further Exploration

Explore the HTML

For those interested in the web development aspect, try looking at the HTML that is being created by the various editors. To do this, you will want to utilize the **Developer Tools** for the browser.

Explore the Page's Trace

The Page's Trace will output a large amount of data. Since there are limitations to development capabilities when using the web, especially when using a complex set of interweaving scripts, troubleshooting tools are limited. When completing page development, the Page **Trace** will be your primary tool for debugging.

The following are included in the Page Trace:

- Loading of Keys from predefined sources (such as User and Device information)
- Navigational information (such as Site, Folder, and Page URLs)
- Site information (such as the Site's root URL and time)
- Full page execution order, including all parts and their outputs
- Control Tree information
- Session, Form, Querystring, and Device information

While not all this information may be useful to you now, it is all the information that the TrakSYS developers believed to be useful during troubleshooting. It is designed to help you or anything helping you to understand how the page is functioning and where issues may occur.

Lab 07 | Content Parts

Overview

A common report or dashboard pattern is to create pages that contain some type of filtering controls (Date/Time, entity pickers, etc…), a data retrieval part that fetches the filtered data from the database, and finally one or more display parts that render the data to the user interface.

In this assignment, you will arrange a more complex **Visual Page** with several different **Content Parts** that will pass data between them. This page will contain a Date/Time picker **Part** to allow user selection, a Data Table **Part** to retrieve **Events** based on the selected range, and 2 display **Parts**, a Table and a Chart which will both be fed from the same Data Table source.

Using the **Page** editors and **Preview** functionality that you have learned in previous labs, you will expand the **Hello World Page** to contain the new **Content Parts** and functionality. Use your discretion to add the **Parts** to appropriate columns on the **Page**.

Estimated Time to Complete This Lab

30 Minutes

Details

The following sections describe the detailed steps to be completed for this lab.

1. Add a Date/Time Filter and Refresh Button

Add a **Date/Time Range Part** to the **Page**. To allow for a user-refresh of the page after altering the date range, add a **Refresh Button Part** to the page as well.

- Create a **Date/Time Range** filter part with the properties as below. Since the sample data in the database is in **January of 2018**, set the default range values so that the range does not have to be changed every time the page is called up.
- Use **Row 1, Column 1** for this part, beneath the existing **Header** part.
- Add a new Part by clicking on the column and using the **+ Add Part** menu option.
- Both parts can be found in the **TrakSYS\Filters** folder in the tree-view picker.

Part ID	DateTimeRange
Initial Start Value	YEARSTART - 1
Initial End Value	TODAY

- Now add the **Refresh Button** and set the properties as shown below.
- Place **beneath** the **Data Time Range** Filter part.

Part ID	ButtonRefresh

2. Add an Event Data Table

To retrieve aggregated **Event** data for display, an **Event Grouped Data Table Part** would normally be added to the **Page**. This **Part** contains properties that allow filtering for a specific **System** and other query related settings to control which **Events** are to be retrieved. However, since there is no data yet recorded for the **Packaging Line 1 System,** a simulation Data Table part will be used to return fabricated **Event** data rows for testing.

- Add part to **Row 2**, **Column 2** using the **+ Add Part** menu option**.**
- The **Chart Grouped Value Data (Simulation) Part** can be found in the **TrakSYS\Data Tables\Chart** folder in the tree-view picker.
- Set the **Chart Grouped Value Data (Simulation)** properties as below.

Part ID	DataEventGrouped
Target Values Key (Data Tab)	Data.EventGrouped

3. Add a Table View and bind it to the Data Table Results

Display the results of the **Event** Data Table in a tabular grid using a **Table View** Part. Configure the columns for the table to display only the **GroupValue** (as **ID**), **GroupLabel** (as **Name**) and **Value** (as **Value**) fields from the Data Table.

- Add part to **Row 1**, **Column 2** using the **+ Add Part** menu option**.**
- This **Part** can be found in the **TrakSYS\Lists** folder in the tree-view picker.
- Set the **Table View** properties as below.

Part ID	TableEventGrouped
Source Values Key (Data Tab)	Data.EventGrouped

- In addition, configure the **Columns** property as below (in the **General Tab**)…

Caption	Content
ID	{GroupValue}
Name	{GroupLabel}
Value	{Value}

4. Add a Chart and bind it to the Data Table Results

As an alternative visualization, display the results of the **Event** Data Table in a chart using a **Column Chart** Part.

- Add part to **Row 1**, **Column 2** above the Table View using the **+ Add Part Above** menu option (You must single click TableEventGrouped for + Add Part Above to appear).
- This **Part** can be found in the **TrakSYS\Charts\JavaScript (Responsive)** folder in the tree-view picker. Add it above the Table View from the previous step.
- Set the **Column Chart** properties as below.

Part ID	ChartEventGrouped
Source Values Key (Data Tab)	Data.EventGrouped

5. View the final Page Results

After all the **Parts** are added, navigate to the end-user view of the **Hello World Page** and verify that the Event data for **January 2019** is displayed in the table and chart. You can also use the **Preview Button** to view your page. Note that the filter will not actually affect the Chart and Table since you are using **Simulation Data**.

Conclusion

In this exercise, you have created a dashboard user interface containing multiple **Content Parts** communicating via the **Values Dictionary** throughout the **Content Page** lifecycle.

Selected Date/Time values are automatically placed in the **Values Dictionary** during Initialization Order 2. The Data Table part automatically retrieves the range values and queries for the appropriate **Events**, placing the resulting DataTable object in the **Values Dictionary** during Initialization Order 5. Finally, the Table and Chart parts retrieve the Data Table and render the visible user interface in **Initialization Order 10**.

While this example is somewhat simplified for the lab, the concepts can be used to build very complex and useful **Pages**. It is recommended that you practice adding and configuring **Parts**, as well as exploring the other types of filter, Data Table and display type elements available in the **Part** catalog.

Common Mistakes

None

Further Exploration

More Charts and Mapping

While this is just the beginning of web development within TrakSYS, it does set the foundation for various core patterns. With the knowledge you gained from this section, you can repeat these patterns for any other part in the TrakSYS catalogue.

Explore some of the other Chart options. Some of them may be more difficult, but most Charts will follow the same patterns as the **Column Chart** that you configured. Some of them may even require different **Data Sources**. You can add other **Simulation Data Table** to your page to feed those charts in the same manner.

Explore **Values Key** mapping. In the same way you changed the Text and Icon of the Header, you can change the values of any properties that accept values keys. You can use the Query String to set the color of the Columns in your Column Chart or use the Script to set the maximum and minimum values of the Y-Axis. Try completing some mapping other properties of your existing parts.

Lab 08 | Content Part Features

Overview

Clicking on **Page** elements and navigating off page (or even back to the same page) is a common pattern in user interface design within TrakSYS. In this assignment, you will continue to enhance the **Hello World Page** with interactive **Part** features like click navigation, selected row highlighting, and selected row detail information display.

You will also be creating **Users**, **Roles**, and assign **Page Permissions.** So far we have added multiple pages to your implementation. Now let's give users a way to access the pages and set the correct permissions.

Estimated Time to Complete This Lab

30 Minutes

Details

The following sections describe the detailed steps to be completed for this lab.

1. Configure the Table View URL Click

Configure the **Table View** to navigate back to the same **Page** when a row is clicked. The navigation should pass the ID (**GroupValue**) of the clicked row back on the **QueryString** as **SelectedID** so that the **Page** can identify which row to highlight (in a later step).

- Edit **TableEventGrouped** (Table View) and set the properties as below.

URL (General Tab, Under Click)	self:SelectedID={GroupValue}

2. Configure the Chart URL Click

Configure the **Chart** to navigate back to the same **Page** when a **row** is clicked. The navigation should pass the ID (**GroupValue**) of the clicked bar back on the **QueryString** as **SelectedID** so that the **Page** can identify which row to highlight (in a later step).

- Edit **ChartEventGrouped** (Column Chart) and set the properties as below.

URL (Click Tab)	self:SelectedID={GroupValue}

3. Configure the Data Table to emit the Selected Row Data

The Data Table Parts contain a feature which allows them to write the selected row's column values to the **Values Dictionary** based on some matching criteria. Values are written with keys constructed with the following format **<Target Values Key>.Selected.<Column Name>**. So in the case of the Data Table being used in the lab, for the **GroupValue** column, the key would be **Data.EventGrouped.Selected.GroupValue**.

Configure the Data Table to emit the selected row based on matching the **QueryString** value **SelectedID** to the row's **GroupValue** column.

- Edit **DataEventGrouped** (Data Table) and set the properties as below.

Property	Value	Key
When Field (Data Tab)	GroupValue	
Equals Value (Data Tab)		SelectedID

4. Configure the Table View to highlight Selected Row

Configure the **Table View** to highlight the row whose **GroupValue** column matches the **QueryString** value of **SelectedIndex**. This can be done easily by instructing the **Part** to look at the **Selected** row index from the mapped **Data Source** using the **Use Data Source Selected Index** property.

- Edit **TableEventGrouped** (Table View) and set the properties as below.

Use Data Source Selected Index (General Tab)	Yes

5. Configure the Chart to highlight the Selected Row

Configure the **Chart** to highlight the row whose **GroupValue** column matches the **QueryString** value of **SelectedIndex**. This can be done easily by instructing the **Part** to look at the **Selected** row index from the mapped **Data Source** using the **Use Data Source Selected Index** property under the **Point Tab**.

- Edit **ChartEventGrouped** (Column Chart) and set the properties as below.

Use Data Source Selected Index (Point Tab)	Yes

6. Add a Details Part to the Page

When a row on the Table View or Chart is selected, it may be desirable to show more details about the selected item. Add a **Details** part to the **Page**.

- Add this Part on **Row 1, Column 3** using the **+ Add Part** menu option.
- Find this part at the **Bottom** of the **Content Parts catalog** (near the Header part).
- Set the **Details** properties as below (**more properties set in Step 7**).

Part ID	DetailsEvent

7. Configure the Details Part to display the Selected Row Data

The **Details Part** can be used to map **Values Dictionary** items to a name/value information display. Configure the **Detail Items** property of the **Part** with the following mappings.

Caption (General Tab)	Content
ID	{Data.EventGrouped.Selected.GroupValue[]}
Name	{Data.EventGrouped.Selected.GroupLabel[]}
Value	{Data.EventGrouped.Selected.Value[]}

Navigate back to **Hello World** and **inspect the changes** you've made before moving on.

8. Configure Roles

Now that we have multiple pages let's create some **Users**, **Roles** and **Permissions** to access those pages. There will be two main types of **Users** in the **Packaging** production area in **OCV**, **Packaging Operators** and **Packaging Supervisors**. Create two **Roles** to hold **User** and **Permission** assignments.

- Navigate to **Administration | Roles**.
- Select the **+ New** menu option under **Roles** to add a new **Role** using the following properties.

Name
Packaging Operators
Packaging Supervisors

9. Assign Page Permissions to Roles

The **Packaging Operators Role** should only have permissions to access pages under the **Training Section** of the user interface. Members of the **Packaging Supervisors Role** must have access to the **Configuration Section** in order to manage **Product** profile information. Assign the following **Page Permissions** to the **Packaging Roles**…

- Navigate to **Administration | Roles**.
- Select the desired **Role** from the list in slice 1 to reveal the item slice and details for the **Role** in slice 3.
- Select the **Page Permissions** option in the **Related** menu in slice 3. The item details for the **Role** should shift to slice 1 and slice 2 should contain an empty list of **Page Permissions**.
- Click the **+ Assign** menu option under **Page Permissions** and add the permissions configuration as described in the table below.

Role	Site	Section	Ability
Packaging Operators	OCV	Training	Yes to All
Packaging Supervisors	OCV	Configuration	Yes to All
Packaging Supervisors	OCV	Training	Yes to All

10. Create Users

User records must be created to allow operations and supervisory staff in the OCV Packaging area to login and access the TrakSYS application. Create the following **Users**…

- Navigate to **Administration | Users**.
- Select **> TrakSYS User** from the list, and then select the **+ New** menu option under **Users** to add a new **User**.
- Note that the **Locale** property can be set from the **Advanced** tab in the Properties Page.

Type	Name	Login	Password	Locale (Advance Tab)

TrakSYS User	Alvar Hanso	al	Abc!23	
TrakSYS User	Edward Ashford	ed	Abc!23	
TrakSYS User	Bob Bradley	bob	Abc!23	fr

11. Assign Users to Roles

Once **Users** are configured, they must be assigned to their respective **Roles**. Assign the **Users** to the **Roles** as listed below…

- Navigate to **Administration | Roles**.
- Select the desired **Role** from the list in slice 1.
- In slice 2, select the **+ Assign** menu option under **Users** to assign new **Users**.

Role	Name
Packaging Operators	Alvar Hanso Edward Ashford Bob Bradley
Packaging Supervisors	Bob Bradley

12. Login as an Operator

To test the **User** access, you must logout of the application as **administrator** and login as an **Operator**, and then as a **Supervisor**. Verify that the proper user interface **Sections** are displayed for each **User**.

To logout as **administrator**…

- Click the user name in the upper right corner of the top menu bar and select **Log Off**.

To login as a **TrakSYS User**…

- Click the **Log On** link in the upper right corner of the top menu bar.
- Enter the **TrakSYS Users** login and password as needed.

To log back in as **administrator**…

- **Log Off** the currently logged in **User**.
- Click the **Log On** link in the upper right corner of the top menu bar.
- Use “**Administrator**” as Login name, **no password is needed**.
- After typing in Administrator, click the **Sign In** button**.**

Conclusion

In this exercise, you have added interactive and data highlighting features to a dashboard report.

Clicking a **Table View** row or a bar in a **Chart** was configured to navigate back to the current **Page** URL, merging the identifier of the clicked row into the **QueryString**.

Display elements (such as **Table Views** and **Charts**), were configured to highlight a specific row based on items in the **Values Dictionary**. A **Details Part** was configured to display additional information for the selected row from a Data Table.

In addition, you have learned about and created **Users** to allow access to individual operators, and **Roles** to define different types of business capabilities within the application. **Users** were assigned to their respective **Roles**. **Page Permissions** were allocated allowing members of each Role to access specific areas of the interface.

Make note of the fact that each Part was configured to retrieve any dynamic property settings from the **Values Dictionary**, there were no direct **Part to Part references** in the configuration. For example, the Table View and Chart do not care where the Event data comes from, only that a Data Table exists called **Data.EventGrouped**. This makes it easier to adjust implementation details and change components out as project requirements change.

These techniques are common user interface patterns and can be leveraged to build a wide variety of dashboard and report **Pages** in TrakSYS.

Common Mistakes

None

Further Exploration

Below are optional activities that can be completed to further your understanding.

Content Page Rebuild: Example

Content Pages, while efficient to deploy, are limited in their options. For example, the colors and charts on a SDRO page cannot be changed outside of the default options. Should you want to change those colors, you would need to rebuild the page and make the appropriate changes yourself.

As an example, a Visual Page equivalent for the SDRO has been provided. Navigate to **Training | Day 2 | Lab 6-8 | Further Exploration**. It will provide some basic information regarding the SDRO page and considerations taken when rebuilding. Then, navigate to the Spoke (in the top right) called **System Discrete Real-Time Overview**. This is a complete rebuild of the Content Page you previously. On this page, you can view the **Grids**, **Parts** and **Scripts** used to make the page. While it is not an identical copy of its Content Page counterpart, this Visual Page provides a valuable window into how Visual Pages can be set-up.