

# Joget DX8

## Improving Your Form Design & Presentation



 <http://facebook.com/jogetworkflow>

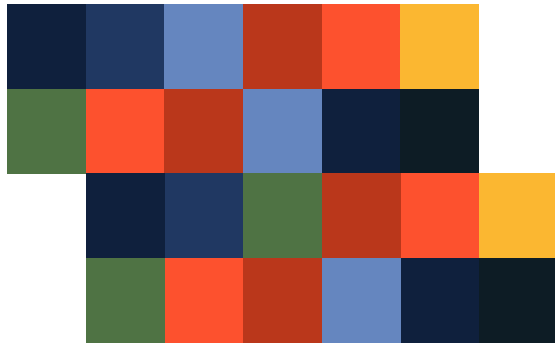
 <http://twitter.com/jogetworkflow>

# Prerequisites

1. Good understanding on the basic functionality of the Form Builder.

# Content

1. Introduction
2. Grid
3. Form Grid
4. Multirow Form Data Store
5. List Grid
6. CRUD
7. Custom HTML
8. Multiple Dynamic Cascading Drop-Down List
9. Using Advanced Tool's Permission
10. Ajax Trigger and Listening
11. Tooltip



# Chapter 1

# Introduction

# Introduction

- Learning about more Joget plugins/elements to improve your form design and presentation.
- In this module, we will be covering the following elements.
  1. Grid (Form Element)
  2. Form Grid (Form Element)
  3. Multirow Form Data Store (Form Store)
  4. List Grid (Form Element)
  5. CRUD (UI Menu)
  6. Custom HTML (Form Element)

# List of Available Elements

- Check out <https://dev.joget.org/community/display/DX8/Form+Builder> for the list of **Form** related elements (Form Element, Form Validator, Form Data Store, Form Options Data Store).
- Check out <https://dev.joget.org/community/display/DX8/UI+Builder> for the list of **UI** related elements.



# Chapter 2

## Grid

# Grid

- Grid is the most basic element available in the Form Builder in capturing multi-row data.
- Reference:  
<http://dev.joget.org/community/display/DX8/Grid>
- For your information: The other grid-like element available in Form Builder is Form Grid.



# Refresh

- Refresh your memory on what you did back in module 5 – Designing your first Form
- Take a look at your “items” field element

Configure Grid ?

Label

Items

ID \*

items

Options

FIELD ID	LABEL
name	Name
quantity	Quantity
price	Price

+

## Just In Case...

- You can download the base Purchase Requisition app – **13.jwa** to start ‘playing around’ with it.
- Completed Form Definition for “1-Submit Request” can be obtained from **13.2.1.txt**.

# Chapter Review

- Able to use the Grid element.



# Chapter 3

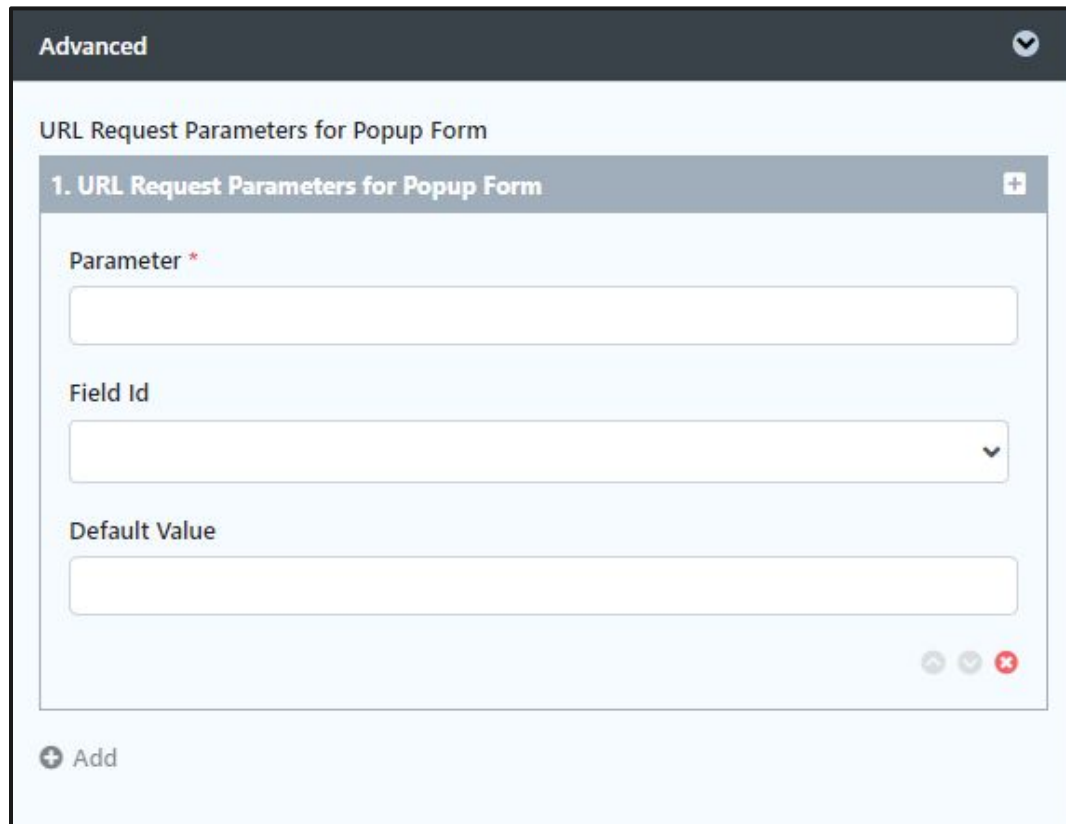
## Form Grid

# Form Grid

- Form Grid works similarly like the basic Grid.
- Instead of editing data row inline, editing is done on a full fledged Form that opens up in a dialog.
- Able to reuse validation and formatting from the selected Form.
- Reference:  
<https://dev.joget.org/community/display/DX8/Form+Grid>

# Form Grid

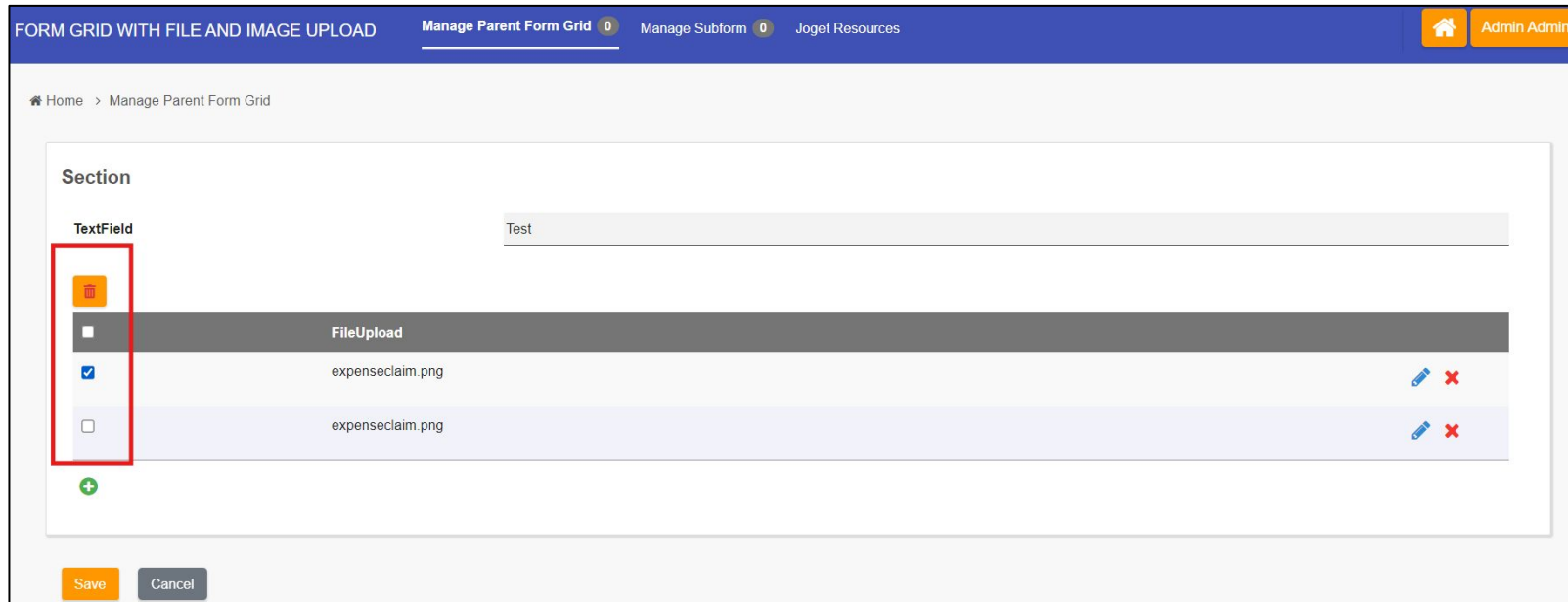
- New feature – Request URL Parameter for Pop Up Form
- Pass field value to prepopulate the form grid field



The screenshot shows the 'Advanced' configuration panel for a 'Form Grid'. The panel is titled 'Advanced' with a dropdown arrow. Below the title, the section is labeled 'URL Request Parameters for Popup Form'. Inside this section, there is a sub-header '1. URL Request Parameters for Popup Form' with a plus icon. The sub-header contains three input fields: 'Parameter \*' (a text input), 'Field Id' (a dropdown menu), and 'Default Value' (a text input). At the bottom right of the sub-header, there are three small icons: a grey circle with a plus, a grey circle with a checkmark, and a red circle with an 'X'. At the bottom left of the panel, there is a '+ Add' button.

# Form Grid

- New feature – Bulk Deletion
- Bulk delete rows of data by a simple check box, users can select the records they want to delete in bulk.



FORM GRID WITH FILE AND IMAGE UPLOAD

Manage Parent Form Grid 0 Manage Subform 0 Joget Resources





Home > Manage Parent Form Grid


Section

TextField Test

☒ ☐

FileUpload

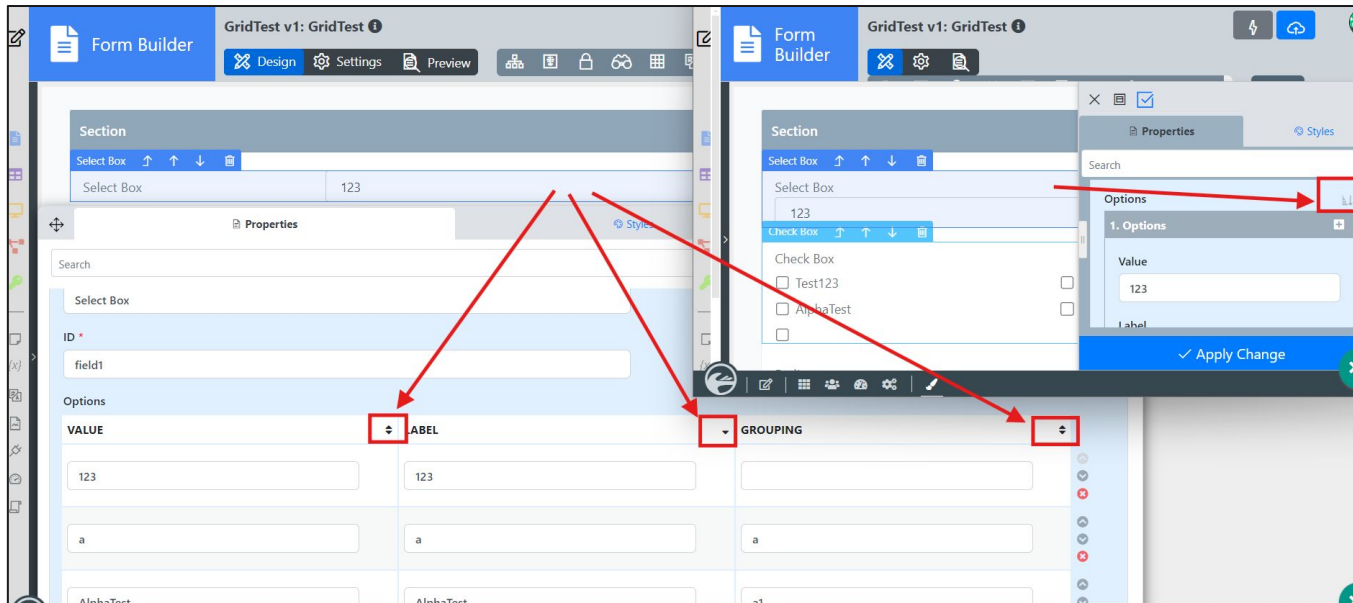
expenseclaim.png	 
expenseclaim.png	 



Save Cancel

# Form Grid

- New feature – Sorting by headers and sorting control button
- The first numerical value in the sequence having priority over alphabetical characters in the string sequence.





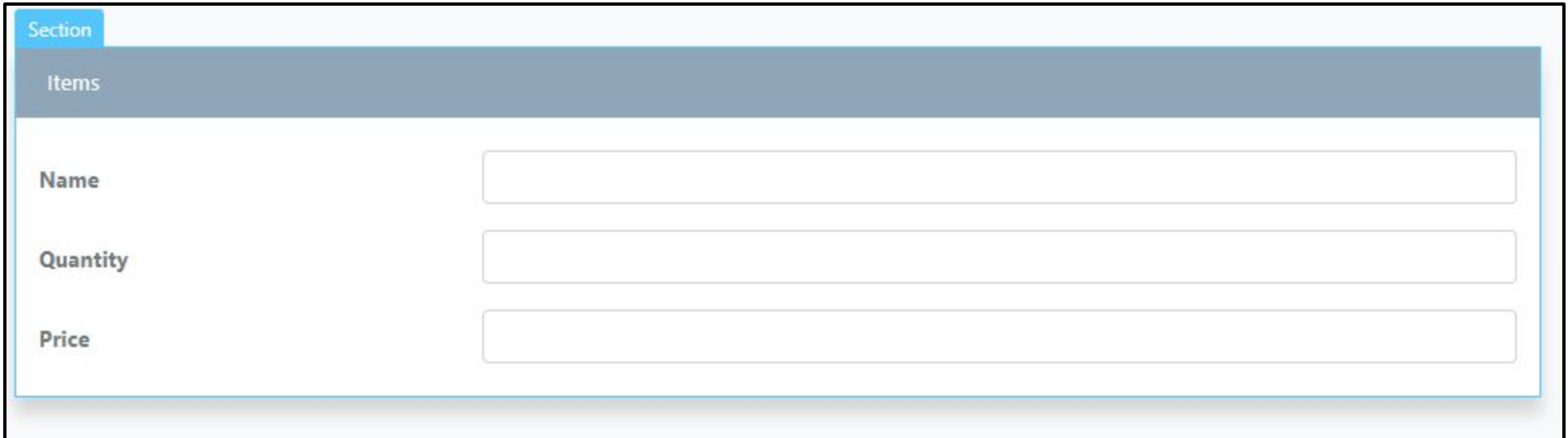
# Exercise

- Re-import the base app “**13.jwa**” into your copy of Joget, OR delete the “items” Grid element.
- Create a new Form with the following details.

FORM DETAILS	
Form ID *	items
Form Name *	Items
Table Name *	app_fd_ purchase_items
Description	

# Exercise

- Add 3 text fields with the following details:-
  - ID: name, Label: Name
  - ID: quantity, Label: Quantity
  - ID: price, Label: Price



The screenshot shows a form editor interface. At the top, there is a 'Section' tab. Below it is a header bar labeled 'Items'. The form contains three text input fields. The first field is labeled 'Name', the second is labeled 'Quantity', and the third is labeled 'Price'. Each label is positioned to the left of its corresponding text field.

- Save the form

# Exercise

- Edit the “1-Submit Request” form.
- Add a “Form Grid” wherever relevant.
- Configure accordingly.

Configure Form Grid ?

Label

ID \*

Form \*

Columns ?

FIELD ID *	LABEL	FORMAT TYPE	FORMAT	WIDTH
<input type="text" value="name"/>	<input type="text" value="Name"/>	Text ▼	<input type="text"/>	<input type="text"/>
<input type="text" value="quantity"/>	<input type="text" value="Quantity"/>	Text ▼	<input type="text"/>	<input type="text"/>
<input type="text" value="price"/>	<input type="text" value="Price"/>	Text ▼	<input type="text"/>	<input type="text"/>

# Exercise

- This is how your form design should look like.

Section

Request Details

Name

Admin Admin

Request Date \*

MM/DD/YYYY

Category

Stationery

Items

Name	Quantity	Price
<div>+</div>		

Remarks

# Exercise

- This is how the form should look like in runtime.

**PURCHASE REQUEST PROCESS - SUBMIT REQUEST**


**Request Details**

Name

Request Date  \*

Category

**Items**

Name	Quantity
	

Remarks

**Items**

Name

Quantity

Price

# Exercise

- Run a new “Submit New Request” process, and submit the form to observe.

# Materials

- "1-Submit Request" form definition can be obtained from **13.3.1.txt**
- "Items" form definition can be obtained from **13.3.2.txt**
- Complete app is available at **13.3.1.jwa**

# Chapter Review

- Being able to use the Form Grid element.
- PS: Check out Advanced Grid  
(<http://dev.joget.org/community/display/DX8/Advanced+Grid>) Form Element that performs similarly as Form Grid.





# Chapter 4

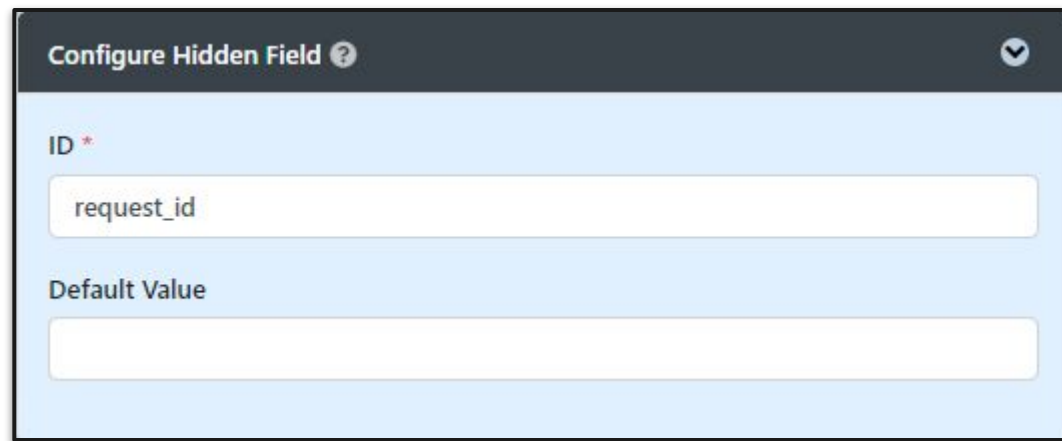
## Multiple Row Form Data Store

# Multiple Row Form

- Multiple Row Form Data Store (Formerly called Multirow Form Binder) is a Store/Load Form Data Store that is designed to treat multi-row data for grid form element.
- Rather than storing **as** traditional JSON data format in a single column cell, the Multiple Row Form saves the data into its respective tables.
- This would make data retrieval easier for sorting, statistics, and indexing/performance purpose.
- Reference:  
<https://dev.joget.org/community/display/DX8/Multiple+Form+Row>

# Exercise

- Continue to use the application from the previous chapter OR import app from **13.4.1.jwa**.
- Edit the “Items” form.
- Add a Hidden Field to the form.
- Configure accordingly.



Configure Hidden Field ?

ID \*

request\_id

Default Value

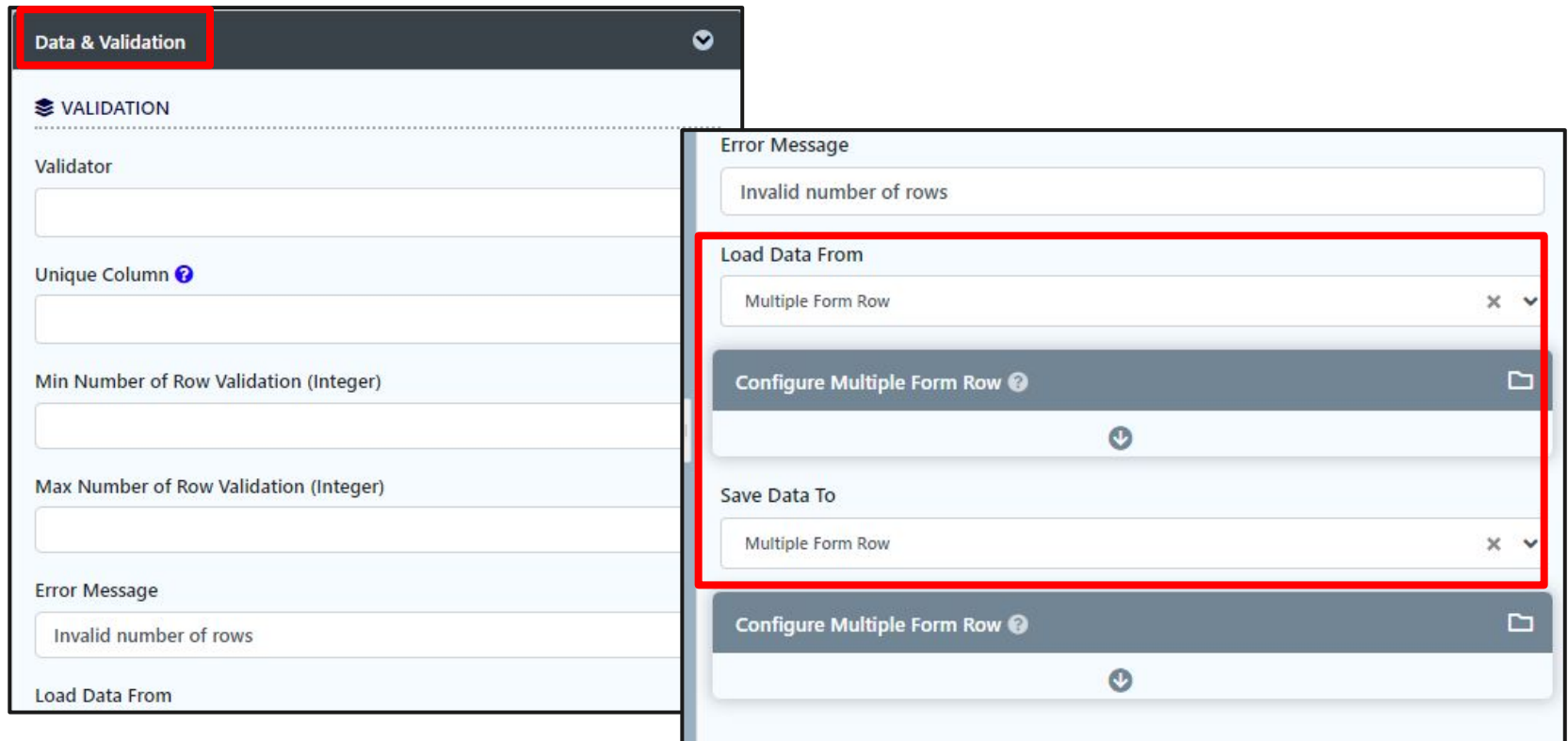
# Exercise

- This is how your “Items” form should look like.

Items	
request_id	<input type="text"/>
Name	<input type="text"/>
Quantity	<input type="text"/>
Price	<input type="text"/>

# Exercise

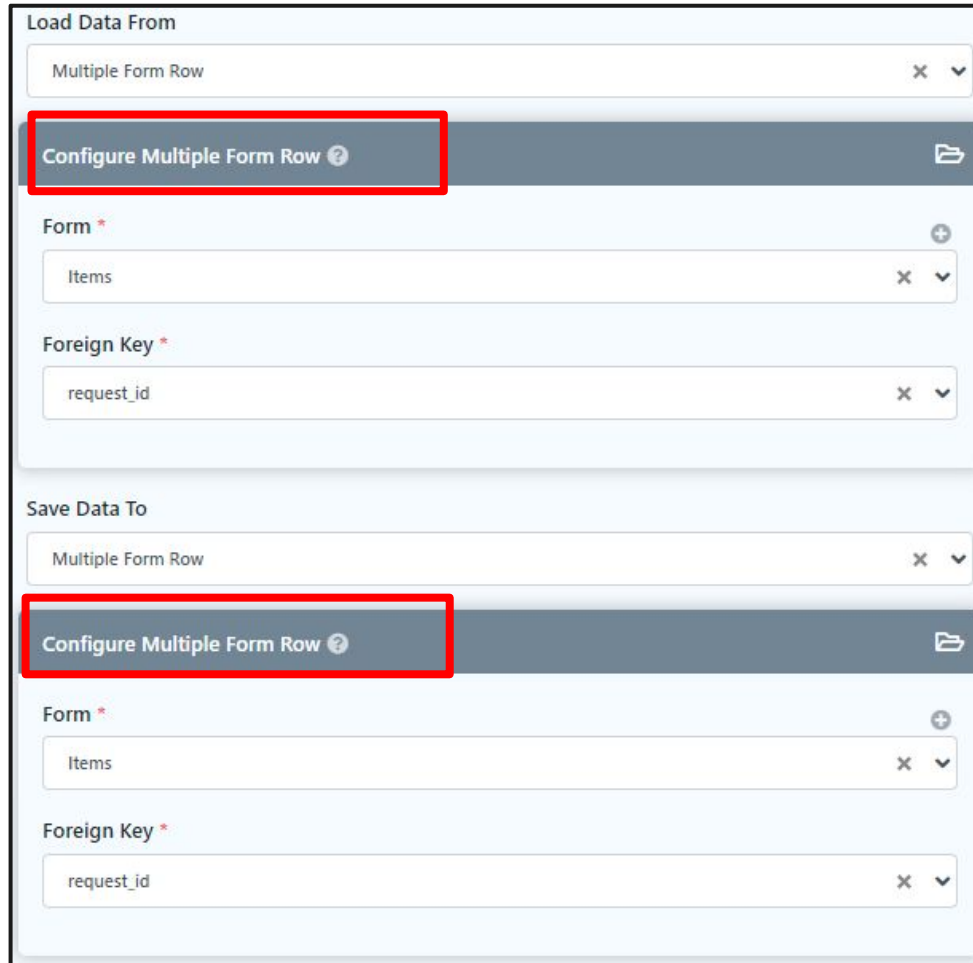
- Edit the "1-Submit Request" form.
- Configure the Form Grid element to utilize the Multiple Row Form in Data Store.



The image shows two overlapping configuration panels from the Joget interface. The left panel, titled "Data & Validation", has a red box around its header. It contains a "VALIDATION" section with fields for "Validator", "Unique Column" (with a help icon), "Min Number of Row Validation (Integer)", "Max Number of Row Validation (Integer)", "Error Message" (containing "Invalid number of rows"), and "Load Data From". The right panel, titled "Error Message", also has a red box around its "Load Data From" and "Save Data To" sections. The "Load Data From" section shows a dropdown menu with "Multiple Form Row" selected. Below it is a "Configure Multiple Form Row" button with a help icon and a downward arrow. The "Save Data To" section also shows a dropdown menu with "Multiple Form Row" selected, followed by another "Configure Multiple Form Row" button with a help icon and a downward arrow.

# Exercise

- Click next to configure the Binder.
- Configure accordingly.



The screenshot displays the Joget configuration interface for data loading and saving. It is divided into two main sections: 'Load Data From' and 'Save Data To'. Both sections have a dropdown menu set to 'Multiple Form Row'. Below each dropdown is a button labeled 'Configure Multiple Form Row' with a question mark icon, which is highlighted with a red box. Under the 'Load Data From' section, there is a 'Form' field with a plus icon and a dropdown menu set to 'Items', and a 'Foreign Key' field with a dropdown menu set to 'request\_id'. The 'Save Data To' section has identical fields.

**Load Data From**

Multiple Form Row

**Configure Multiple Form Row ?**

**Form \***

Items

**Foreign Key \***

request\_id

**Save Data To**

Multiple Form Row

**Configure Multiple Form Row ?**

**Form \***

Items

**Foreign Key \***

request\_id

# Exercise

- Run a new “Submit New Request” process, and submit the form to observe.

# Exercise – Optional

- Inspect the database table of “Items”, you will notice that rows of data is now being saved into this table rather than the parent table as JSON.

jwdb.app_fd_purchase_requests: 1 rows total (approximately)									
<span>» Next</span> <span>Show all</span> <span>▼ Sorting</span> <span>▼ Columns (13/13)</span> <span>▼ Filter</span>									
id	dateCreated	dateModified	createdBy	createdByName	modifiedBy	modifiedByName	c_name	c_request_date	c_items
a7380ecd-72eb-4eea-8560-a28aae74ba6e	2019-12-27 15:03:43	2019-12-27 15:03:43	admin	Admin Admin	admin	Admin Admin	Admin Admin	12/27/2019	(NULL)

jwdb.app_fd_purchase_items: 1 rows total (approximately)										
<span>» Next</span> <span>Show all</span> <span>▼ Sorting (1)</span> <span>▼ Columns (11/12)</span> <span>▼ Filter</span>										
id	dateCreated	dateModified	createdBy	createdByName	modifiedBy	modifiedByName	c_quantity	c_price	c_name	c_request_id
832103ee-df79-41ac-969a-2cb2870fd872	2019-12-27 15:03:43	2019-12-27 15:03:43	admin	Admin Admin	admin	Admin Admin	5	20	Pencil	a7380ecd-72eb-4eea-8560-a28aae74ba6e



# Materials

- "1-Submit Request" form definition is available at **13.4.2.txt**
- "Items" form definition is available at **13.4.3.txt**
- Complete app is available at **13.4.4.jwa**

# Chapter Review

- Understand the use case of the Multirow Form Binder and its benefits.



# Chapter 5

## List Grid

# List Grid

- **List Grid** is a grid table that populates its data from a Datalist.
- It behaves similarly like a Grid (Chapter 2) but new rows are added from a specified Datalist instead.
- It also behaves similarly like a Form Grid that allows one to open up a Form for editing.
- Reference:  
<https://dev.joget.org/community/display/DX8/List+Grid>

# Sample Use Case

## Leave Process - Submit Leave

### Leave Application Details

Name \* Admin Admin

Start Date \*

End Date \*

Reason \*

### Add Entry

10

<input type="checkbox"/>	name	contact_no
<input type="checkbox"/>	Julia	123
<input type="checkbox"/>	Jude	124

2 items found, displaying all items.  
1

### Emergency Contacts

Contact Name	Contact No
<input type="text"/>	<input type="text"/>

# Chapter Review

- Understand the List Grid element and be able to think of use cases of it.
- Able to differentiate Grid, Form Grid, and List Grid.



# Chapter 6

## CRUD

# CRUD

- CRUD is a UI Menu allows one to easily achieve the functionality of **C**reate, **R**etrieve, **U**ppdate, and **D**eleete on a data entity.
- In short, manipulate records on a specified table.
- Reference:  
<https://dev.joget.org/community/display/DX8/CRUD+Menu>



# What Is Needed For CRUD To run?

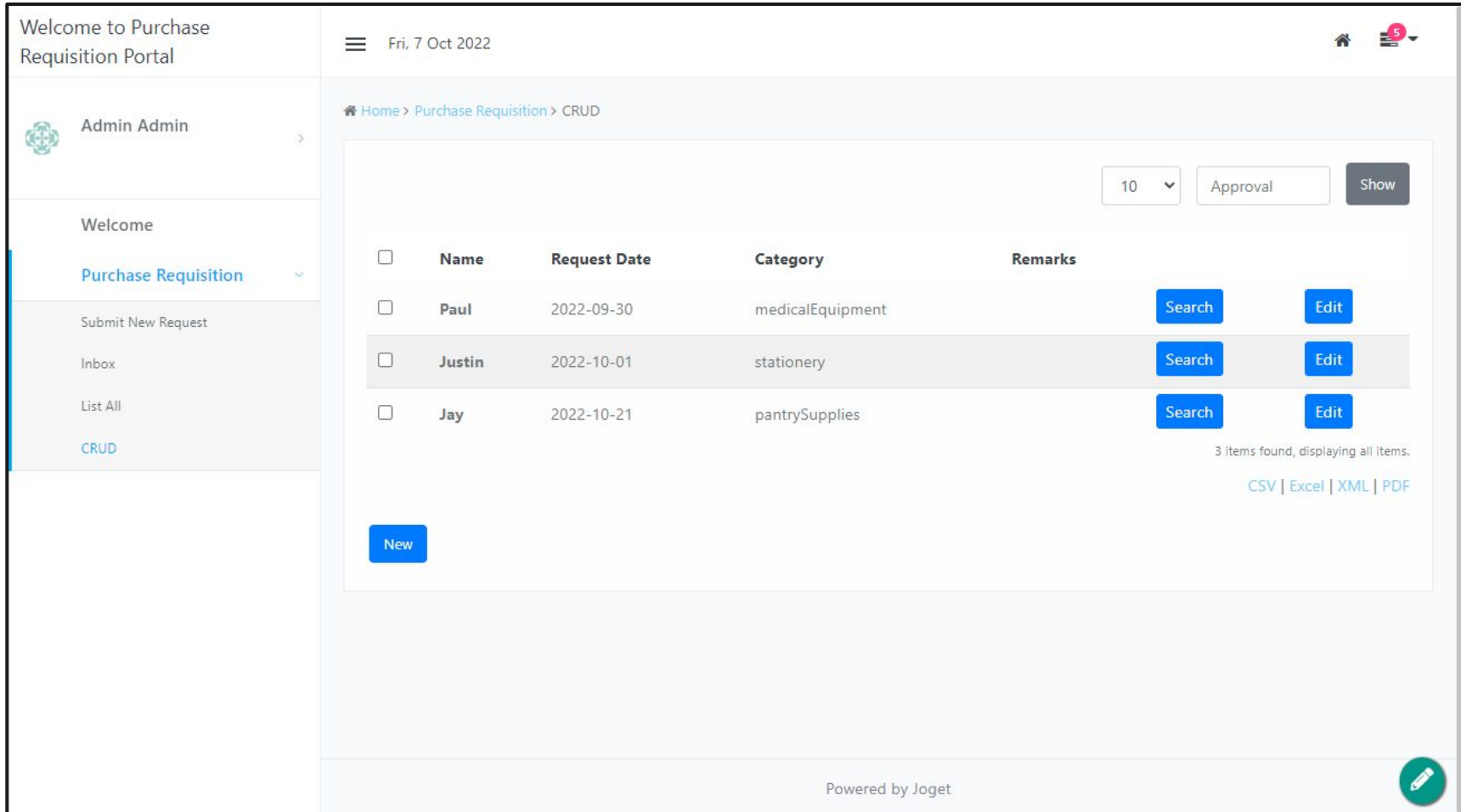
- A Form entity
- A List of the same data entity as the form
- A UI

# Refresh

- You've already done it! Refresh what you did back in Module 8 – Designing your first UI
- If you do not have the CRUD for “Request List”, ask your colleague how to, or raise hand...

# How CRUD Looks Like...

- This is how the CRUD element would look like in



The screenshot displays a web application interface for a Purchase Requisition Portal. The left sidebar contains navigation links: 'Admin Admin', 'Welcome', 'Purchase Requisition' (selected), 'Submit New Request', 'Inbox', 'List All', and 'CRUD'. The main content area shows a breadcrumb trail 'Home > Purchase Requisition > CRUD'. Below this, there's a table with columns: Name, Request Date, Category, and Remarks. The table lists three items: Paul (2022-09-30, medicalEquipment), Justin (2022-10-01, stationery), and Jay (2022-10-21, pantrySupplies). Each row has a checkbox and 'Search' and 'Edit' buttons. A 'New' button is at the bottom left. The top right shows a date 'Fri, 7 Oct 2022' and a notification icon with '5'. The bottom right has a 'Powered by Joget' logo.

Welcome to Purchase Requisition Portal

Fri, 7 Oct 2022

Home > Purchase Requisition > CRUD

10 Approval Show

<input type="checkbox"/>	Name	Request Date	Category	Remarks
<input type="checkbox"/>	Paul	2022-09-30	medicalEquipment	<a href="#">Search</a> <a href="#">Edit</a>
<input type="checkbox"/>	Justin	2022-10-01	stationery	<a href="#">Search</a> <a href="#">Edit</a>
<input type="checkbox"/>	Jay	2022-10-21	pantrySupplies	<a href="#">Search</a> <a href="#">Edit</a>

3 items found, displaying all items.

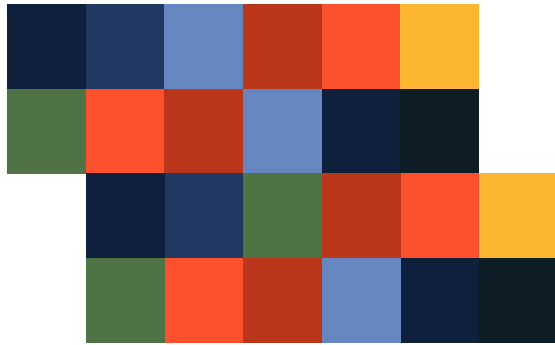
[CSV](#) | [Excel](#) | [XML](#) | [PDF](#)

[New](#)

Powered by Joget

# Chapter Review

- Able to use CRUD and understand the linkages.



# Chapter 7

## Custom HTML

# Custom HTML

**Custom HTML** in Form Builder can be used to achieve advanced form design by putting in any valid –

- **HTML**

E.g: `<b>this text is in bold</b>`

- **JavaScript** (jQuery is supported)

Remember to put in `<script`

`type="text/javascript"></script>` block

- **CSS**

Don't forget to put in `<style type="text/css"></style>` block

Reference:

<https://dev.joget.org/community/display/DX8/Custom+HTML>

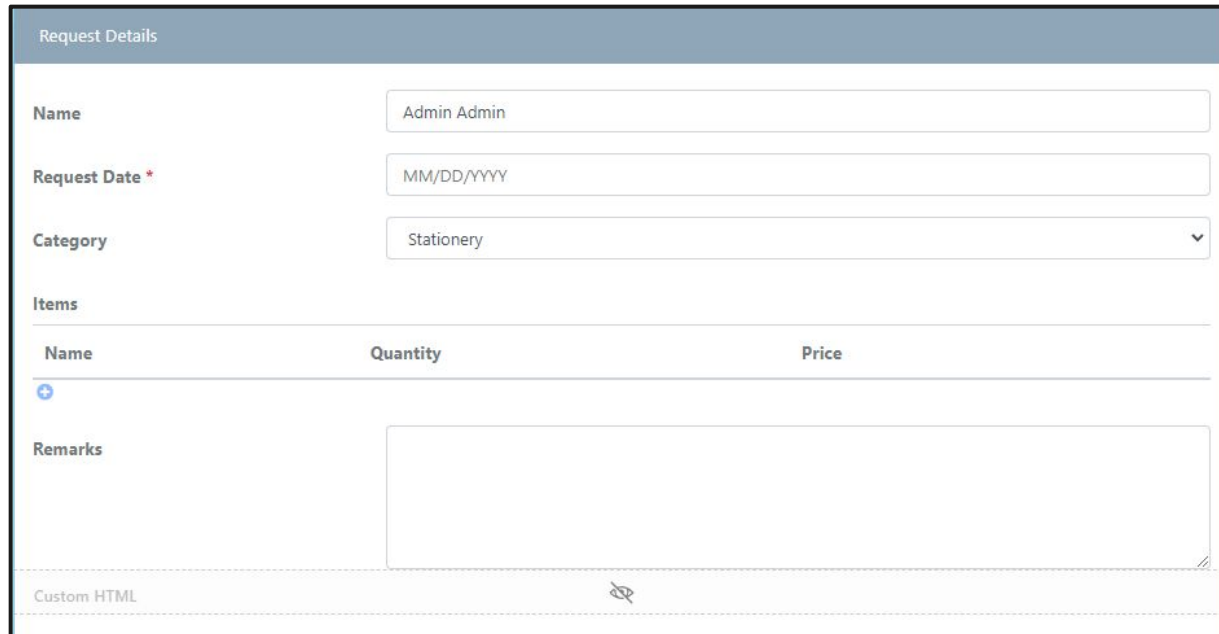
# Exercise – CSS (Optional)

- Customize the look and feel of how the form is rendered by modifying its CSS.
  - Add a **Custom HTML** form element into the bottom of the form.
  - Edit it, add the following code into **Custom HTML** property.

```
<style type="text/css">

.form-cell .label,
.subform-cell .label{
    width: 100%;
}

</style>
```



Request Details

Name Admin Admin

Request Date \* MM/DD/YYYY

Category Stationery

Items

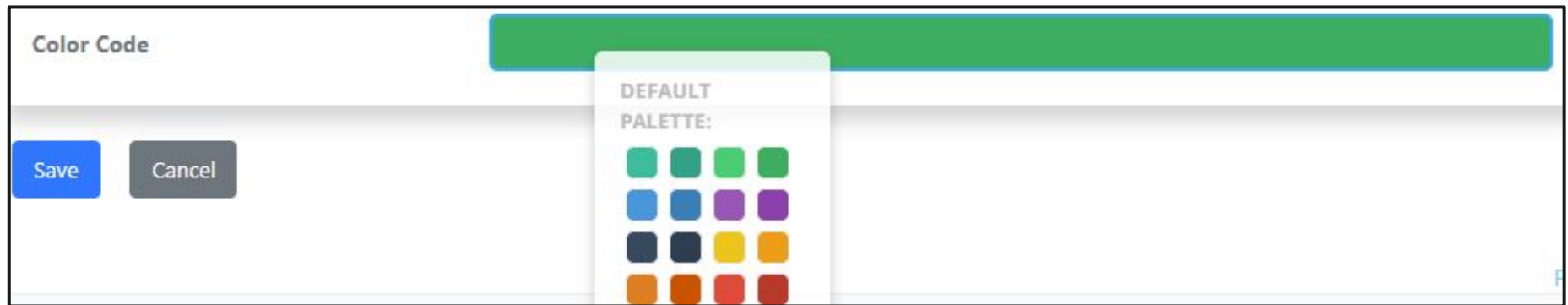
Name	Quantity	Price
+		

Remarks

Custom HTML

# Exercise – Color Picker (Optional)

- Create a text field “Color Code” with ID “color\_code”
- Create a Custom HTML. Make use of color picker library to turn text field into a color picker.





# Exercise – Color Picker – Materials

Do import these JS and CSS libraries into the app's resource from the materials folder:

- colorPick.min.css
- colorPick.min.js

# Exercise – Color Picker

```
<script src="#appResource.colorPick.min.js#"></script>
<link rel="stylesheet" href="#appResource.colorPick.min.css#">

<script type="text/javascript">
loadJSCount = 0; loadJSCountProcessed = 0;
function loadJS(FILE_URL, async = true) {
    loadJSCount += 1;
    let scriptEle = document.createElement("script");

    scriptEle.setAttribute("src", FILE_URL);
    scriptEle.setAttribute("type", "text/javascript");
    scriptEle.setAttribute("async", async);

    document.body.appendChild(scriptEle);

    // success event
    scriptEle.addEventListener("load", () => {
        console.log("File loaded");
        loadJSCountProcessed += 1;
        if(loadJSCount == loadJSCountProcessed){
            windowReady();
        }
    });
    // error event
    scriptEle.addEventListener("error", (ev) => {
        console.log("Error on loading file", ev);
        loadJSCountProcessed += 1;
        if(loadJSCount == loadJSCountProcessed){
            windowReady();
        }
    });
}

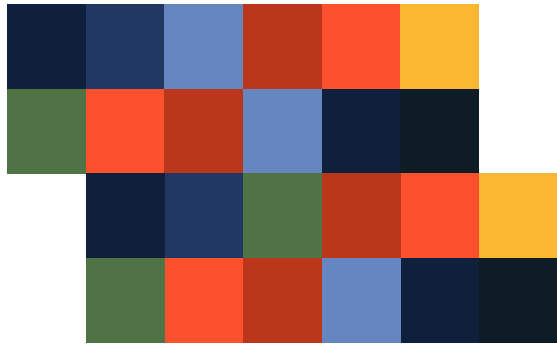
loadJS("#appResource.colorPick.min.js#", true);

//this acts as the replacement for windows.ready
function windowReady(){
    //continue with usual code execution
    $(function(){
        initialColor = FormUtil.getField("color_code").val();
        //console.log("initial " + initialColor);
        FormUtil.getField("color_code").colorPick({
            'initialColor': initialColor,
            'onColorSelected': function() {
                //console.log("The user has selected the color: " + this.color);
                FormUtil.getField("color_code").val(this.color);
                this.element.css({'backgroundColor': this.color, 'color': this.color});
            }
        });
    });
}

</script>
```

# Chapter Review

- Able to use and understand Custom HTML



# **Chapter 8**

# **Multiple**

# **Dynamic**

# **Cascading**

# **Drop-Down List**

# Multiple Dynamic Cascading Drop-Down List

- New Feature in Joget DX8
- Dynamically change the options/selections of a select box/radio button/radio based on multiple form fields value within the same form.



The screenshot displays a web form titled "Address" with a blue header bar. The form contains four input fields: "Name" (a text box), "Continent" (a dropdown menu with "Asia" selected), "Climate" (a dropdown menu with "Tropical" selected), and "Country" (a dropdown menu). The "Country" dropdown is open, showing a list of options with "Malaysia" and "Singapore" visible. At the bottom left of the form are "Save" and "Cancel" buttons. In the bottom right corner, there is a "Print" link.

# Materials

- Sample app and reference is available in <https://dev.joget.org/community/display/DX8/Multiple+Dynamic+Cascading+Drop-Down+List>

# Chapter Review

- Able to use and understand Multiple Dynamic Cascading Drop Down List



# Chapter 9

## Using Advanced Tool's Permission



# Permission Control

- The Permission in Builders Advanced Tools allows fine grain control up to field level.

**Form Builder** | Purchase Requisition v3: Request Form (Published) | Generate App | Save

Design | Settings | Preview

**PERMISSION**

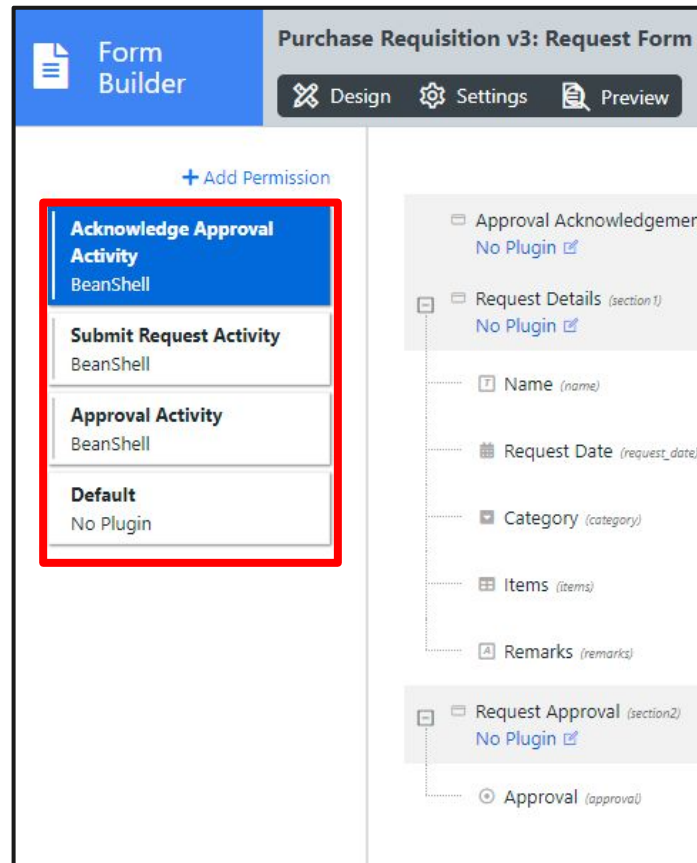
	AUTHORIZED			UNAUTHORIZED	
Approval Acknowledgement (section3) No Plugin	Visible	Readonly	Hidden	Readonly	Hidden
Request Details (section1) No Plugin	Visible	Readonly	Hidden	Readonly	Hidden
Name (name)	Visible	Readonly	Hidden	Readonly	Hidden
Request Date (request_date)	Visible	Readonly	Hidden	Readonly	Hidden
Category (category)	Visible	Readonly	Hidden	Readonly	Hidden
Items (items)	Visible	Readonly	Hidden	Readonly	Hidden
Remarks (remarks)	Visible	Readonly	Hidden	Readonly	Hidden
Request Approval (section2) No Plugin	Visible	Readonly	Hidden	Readonly	Hidden
Approval (approval)	Visible	Readonly	Hidden	Readonly	Hidden

**Left Panel:**

- Acknowledge Approval Activity** (BeanShell)
- Submit Request Activity** (BeanShell)
- Approval Activity** (BeanShell)
- Default** (No Plugin)

# Permission Control

- Typically, one would control **based on User Role** most of the time to show/hide fields.



The screenshot shows the Joget Form Builder interface for a form titled "Purchase Requisition v3: Request Form". The interface is divided into two main panels. The left panel, labeled "Form Builder", contains a list of activities: "Acknowledge Approval Activity" (highlighted with a red box), "Submit Request Activity", "Approval Activity", and "Default". Each activity is associated with a plugin, with "Acknowledge Approval Activity" and "Submit Request Activity" using "BeanShell" and the others using "No Plugin". The right panel shows the form design, which includes sections for "Approval Acknowledgement", "Request Details (section 1)", and "Request Approval (section2)". The "Request Details" section contains fields for "Name (name)", "Request Date (request\_date)", "Category (category)", "Items (items)", and "Remarks (remarks)". The "Request Approval" section contains an "Approval (approval)" field.

# Permission Control

- If the form is used part of a process flow, it is also possible to exert permission **based on activity**, rather than user role.

Form Builder

Purchase Requisition v3: Request Form (Published)

Design Settings Preview

Generate App Save

+ Add Permission

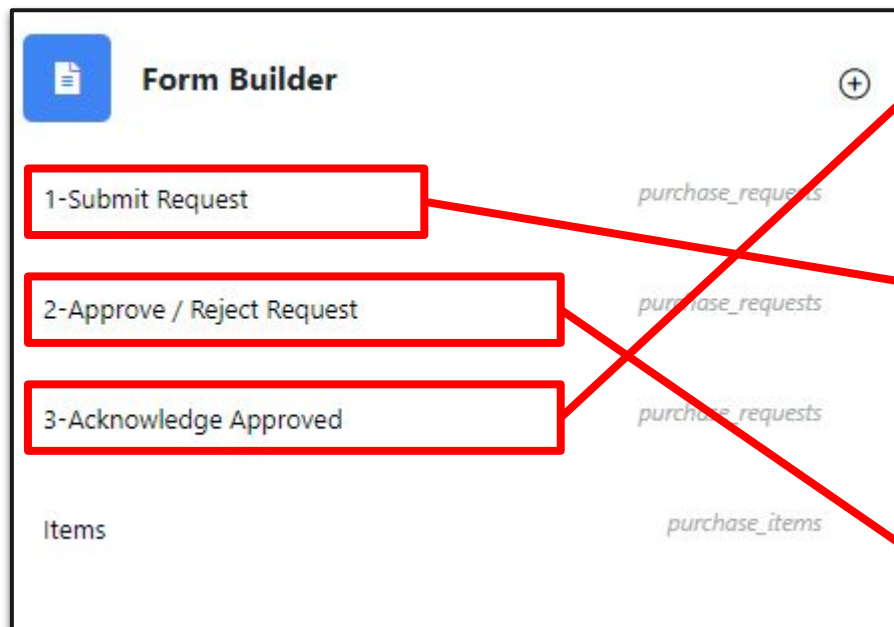
	AUTHORIZED			UNAUTHORIZED	
Approval Acknowledgement (section3) No Plugin	Visible	Readonly	Hidden	Readonly	Hidden
Request Details (section1) No Plugin	Visible	Readonly	Hidden	Readonly	Hidden
Name (name)	Visible	Readonly	Hidden	Readonly	Hidden
Request Date (request_date)	Visible	Readonly	Hidden	Readonly	Hidden
Category (category)	Visible	Readonly	Hidden	Readonly	Hidden
Items (items)	Visible	Readonly	Hidden	Readonly	Hidden
Remarks (remarks)	Visible	Readonly	Hidden	Readonly	Hidden
Request Approval (section2) No Plugin	Visible	Readonly	Hidden	Readonly	Hidden
Approval (approval)	Visible	Readonly	Hidden	Readonly	Hidden

Left sidebar permissions:

- Acknowledge Approval Activity (BeanShell)
- Submit Request Activity (BeanShell)
- Approval Activity (BeanShell)
- Default (No Plugin)

# Exercise – Advanced Tool's Permission

- *One Master Form Concept* – Try combining the 3 forms in Purchase Requisition into a single form by using the Form Builder's Advance Tool Permission Tab:
  - 1-Submit Request
  - 2-Approve / Reject Request
  - 3-Acknowledge Approved



**Form Builder**

1-Submit Request

2-Approve / Reject Request

3-Acknowledge Approved

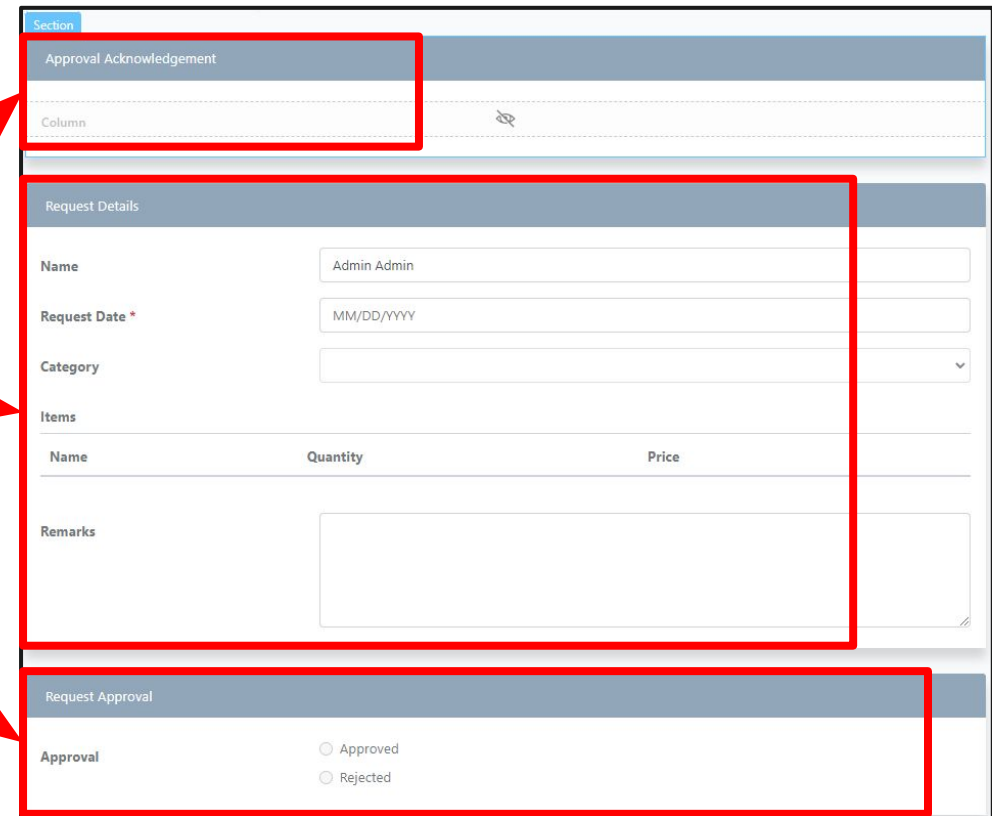
Items

purchase\_requests

purchase\_requests

purchase\_requests

purchase\_items



**Section**

Approval Acknowledgement

Column

**Request Details**

Name Admin Admin

Request Date \* MM/DD/YYYY

Category

Items

Name	Quantity	Price
------	----------	-------

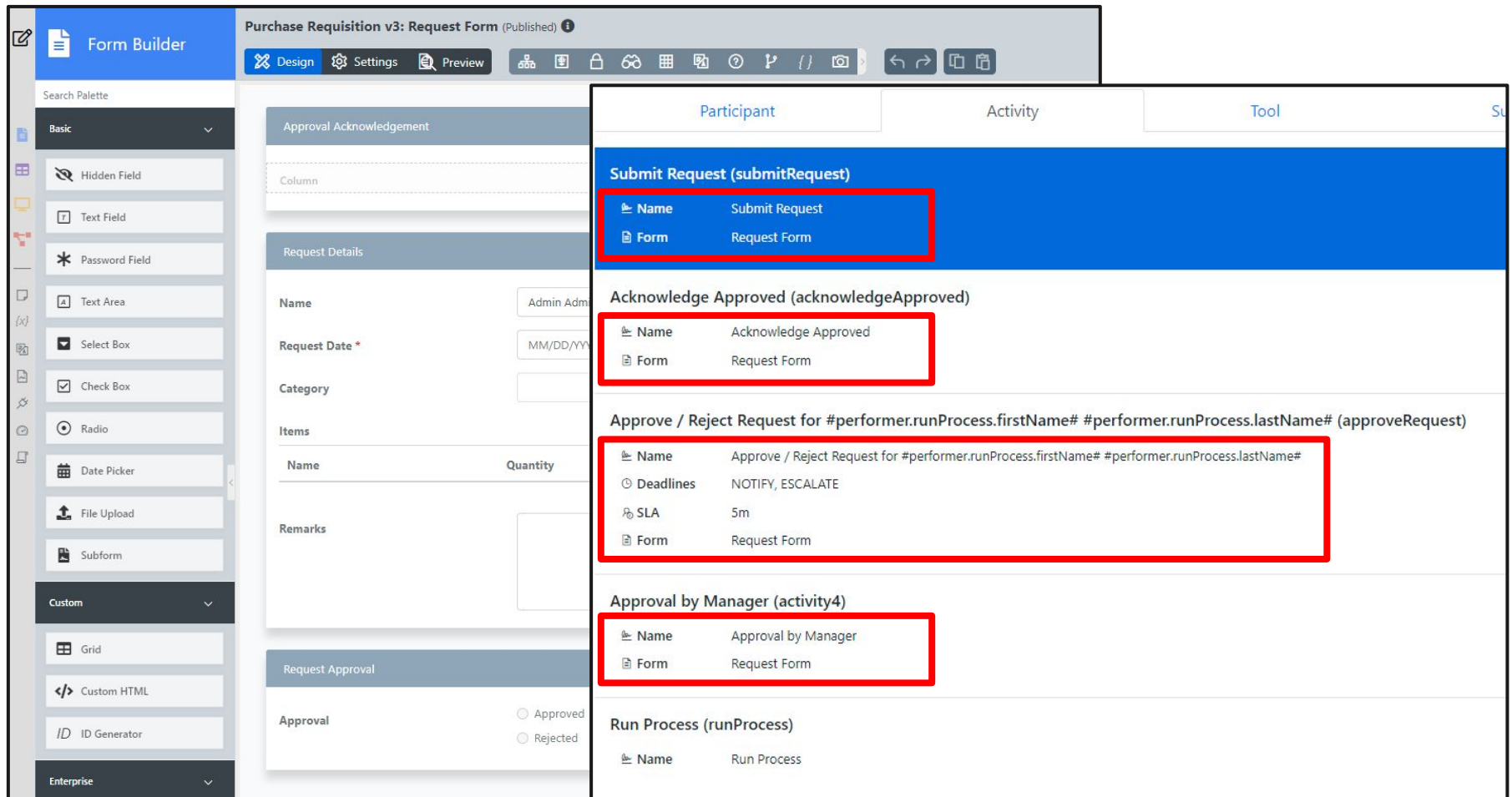
Remarks

**Request Approval**

Approval ☐ Approved ☐ Rejected

# Exercise – Advanced Tool's Permission

- The same form will be used in all activities.



The screenshot displays the Joget Form Builder interface for a 'Purchase Requisition v3: Request Form' (Published). The interface is divided into three main sections: a left sidebar with a search palette, a central design canvas, and a right-hand panel showing the form's structure and activities.

**Left Sidebar (Search Palette):**

- Basic:** Hidden Field, Text Field, Password Field, Text Area, Select Box, Check Box, Radio, Date Picker, File Upload, Subform.
- Custom:** Grid, Custom HTML, ID Generator.
- Enterprise:** (Empty)

**Central Design Canvas:**

The form is titled 'Purchase Requisition v3: Request Form' and includes sections for 'Approval Acknowledgement', 'Request Details', and 'Request Approval'. The 'Request Details' section includes fields for Name, Request Date, Category, Items (Name, Quantity), and Remarks. The 'Request Approval' section includes an 'Approval' field with radio buttons for 'Approved' and 'Rejected'.

**Right-Hand Panel (Activities and Tools):**

The right-hand panel shows the form's structure and activities. The 'Participant' tab is selected, and the 'Activity' column is highlighted. The following activities are listed, each with a 'Name' and a 'Form' associated with it:

- Submit Request (submitRequest):**
  - Name: Submit Request
  - Form: Request Form
- Acknowledge Approved (acknowledgeApproved):**
  - Name: Acknowledge Approved
  - Form: Request Form
- Approve / Reject Request for #performer.runProcess.firstName# #performer.runProcess.lastName# (approveRequest):**
  - Name: Approve / Reject Request for #performer.runProcess.firstName# #performer.runProcess.lastName#
  - Deadlines: NOTIFY, ESCALATE
  - SLA: 5m
  - Form: Request Form
- Approval by Manager (activity4):**
  - Name: Approval by Manager
  - Form: Request Form
- Run Process (runProcess):**
  - Name: Run Process

# Exercise – Advanced Tool's Permission

- In the Permission tab, we can control on what to show based on current activity.

The screenshot illustrates the configuration of a permission in Joget. On the left, the 'Submit Request Activity' is selected from a list of activities. The middle panel shows the activity's structure. The right panel, titled 'Properties', shows the configuration for the 'Submit Request Activity' permission. The 'Script' field is highlighted with a red box and contains the code: `return ("#assignment.activityDefId#".equalsIgnoreCase("submitRequest"));`. A red arrow points from the 'Submit Request Activity' box in the left panel to the 'Script' field in the right panel.

```
return ("#assignment.activityDefId#".equalsIgnoreCase("submitRequest"));
```

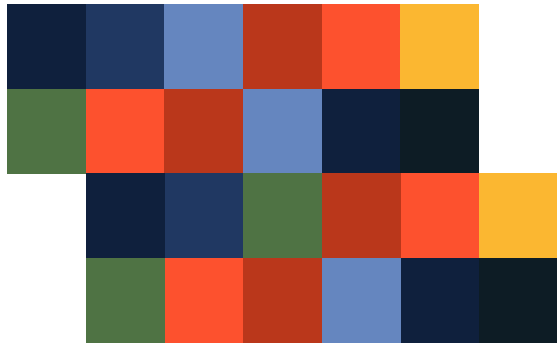
# Exercise – Advanced Tool's Permission

- Once you are done setting up, try to run through the whole flow and check if the form is showing up correctly as per the activity.

# Materials

- Complete app is available at **13.8.1.jwa**





# Chapter 10

## Ajax Trigger and Listening

# Ajax Trigger and Listening

- New feature in Joget DX8
- Enable page components to interact with each other by using Ajax event triggering and event listening

# Event Listening

- Listen to events to be trigger by page components

**AJAX & Events**

☒ Handle Component with AJAX?

**Event Listening**

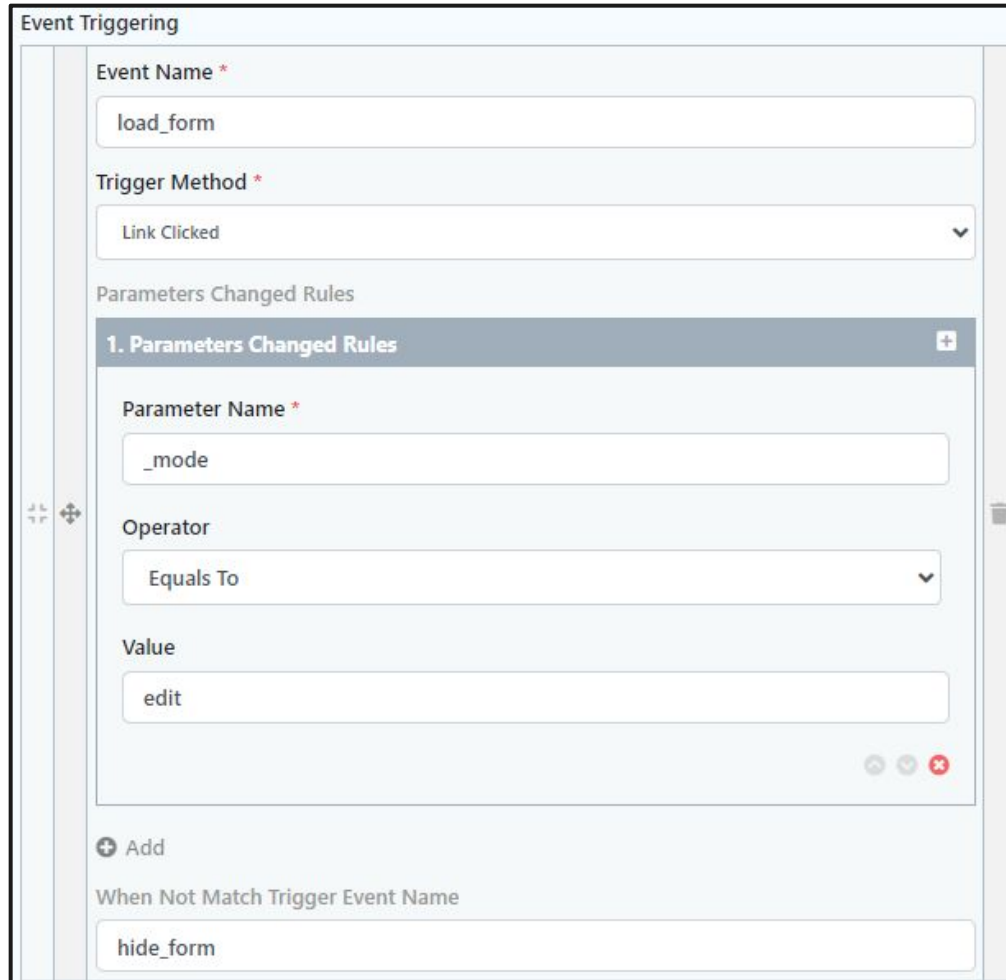
	<b>Component Object</b>	
	Form	▼
⋈	<b>Event Name *</b>	
	reload_table	🗑
	<b>Matched Action</b>	
	Show and Reload Component	▼
	<input type="checkbox"/> Disabled Scrolling to Component	

**+ Add Row**

**⋈ Collapse All Rows**

# Event Triggering

- Create trigger events to be listen by page components



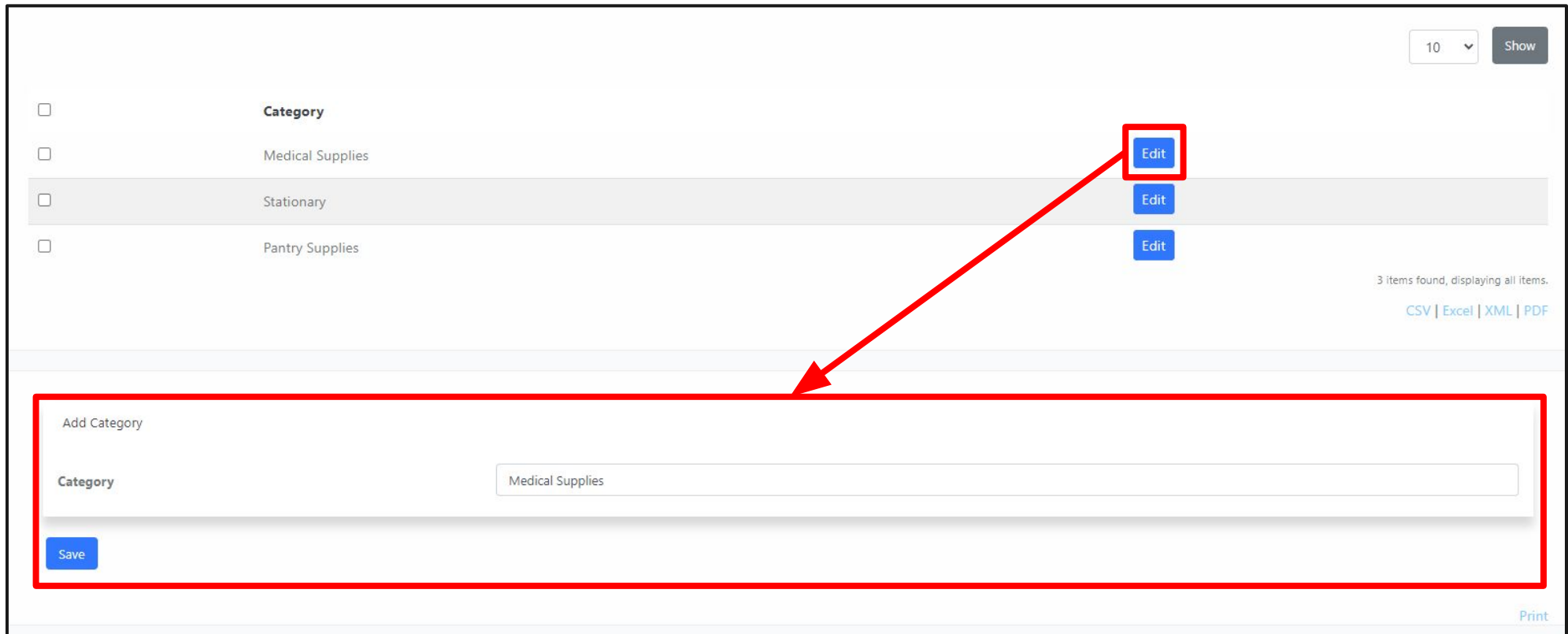
The screenshot displays the 'Event Triggering' configuration window in Joget. It is divided into several sections:

- Event Name \***: A text input field containing 'load\_form'.
- Trigger Method \***: A dropdown menu with 'Link Clicked' selected.
- Parameters Changed Rules**: A section containing a list of rules. The first rule is expanded, showing:
  - Parameter Name \***: A text input field containing '\_mode'.
  - Operator**: A dropdown menu with 'Equals To' selected.
  - Value**: A text input field containing 'edit'.
- + Add**: A button to add a new rule.
- When Not Match Trigger Event Name**: A text input field containing 'hide\_form'.

On the left side of the configuration area, there are icons for zooming in (+), zooming out (-), and a full-screen icon (⛶). On the right side, there is a trash icon (🗑).

# Example

When the Edit Button is clicked, the button triggers an event to show a form below the table for editing.



The screenshot displays a web application interface. At the top right, there is a dropdown menu set to '10' and a 'Show' button. Below this is a table with three rows. The first row is highlighted in light gray and contains a checkbox, the text 'Category', and a blue 'Edit' button. The second row contains a checkbox, the text 'Medical Supplies', and a blue 'Edit' button. The third row contains a checkbox, the text 'Stationary', and a blue 'Edit' button. The fourth row contains a checkbox, the text 'Pantry Supplies', and a blue 'Edit' button. To the right of the table, there is a status message '3 items found, displaying all items.' and links for 'CSV | Excel | XML | PDF'. Below the table, there is a form with a red border. The form has a title 'Add Category' and a label 'Category'. The text 'Medical Supplies' is entered in the input field. A blue 'Save' button is located at the bottom left of the form. A red arrow points from the 'Edit' button in the second row of the table to the form.

<input type="checkbox"/>	Category	Edit
<input type="checkbox"/>	Medical Supplies	Edit
<input type="checkbox"/>	Stationary	Edit
<input type="checkbox"/>	Pantry Supplies	Edit

3 items found, displaying all items.  
[CSV](#) | [Excel](#) | [XML](#) | [PDF](#)

Add Category

Category

# Exercise – Ajax Triggering and Listening

- Download the complete app at **13.10.1.jwa**
- Run the app › Manage Purchase Category › Edit Category › click Edit on any of the rows.



# Chapter 11

## Misc


# Exercise – Tooltip

- Make use of Advanced Tool's Tooltip to add hints into existing form "Submit Request".

**Purchase Request Process - Submit Request**


Request Details

Name ⓘ **Key in your full name**

Request Date \*  

Category  ▼

Items

Name	Quantity	Price
		

Remarks



# Module Review

1. Introduction
2. Grid
3. Form Grid
4. Multirow Form Binder
5. List Grid
6. CRUD
7. Custom HTML
8. Multiple Dynamic Cascading Drop-Down List
9. Using Advanced Tool's Permission
10. Ajax Trigger and Listening
11. Tooltip

# Learn More...

- Check out <https://dev.joget.org/community/display/DX8/Form+Builder> for the list of **Form** related elements (Form Element, Form Validator, Form Binder, Form Options Binder).
- Check out <https://dev.joget.org/community/display/DX8/UI+Builder> for the list of **UI** related elements.

# Stay Connected With Joget

- [www.joget.org](http://www.joget.org)
- [community.joget.org](http://community.joget.org)
- [twitter.com/jogetworkflow](https://twitter.com/jogetworkflow)
- [facebook.com/jogetworkflow](https://facebook.com/jogetworkflow)
- [youtube.com/jogetworkflow](https://youtube.com/jogetworkflow)