How to develop a Gantt Chart UI Menu

Then, let us create a tem

- 1. What is the problem?
- 2. What is your idea to solve the problem?
- 3. What is the input needed for your plugin?
- 4. What is the output and expected outcome of your plugin?
- 5. Is there any resources/API that can be reuse?
- 6. Prepare your development environment
- 7. Just code it!
    - a. Extending the abstract class of a plugin type
    - b. Implement all the abstract methods
    - c. Manage the dependency libraries of your plugin
    - d. Make your plugin internationalization (i18n) ready
    - e. Register your plugin to Felix Framework
    - f. Build it and testing
- 8. Take a step further, share it or sell it

In this tutorial, we will following the Guideline for Developing a Plugin to develop our Gantt Chart Userview Menu plugin. Please also refer to the very first tutorial How to develop a Bean Shell Hash Variable for more details steps.

1. What is the problem?

I want to display collected form data in a gantt chart view. I can choose to use Jasper Reports Menu to achieve it but It does not look nice and lack of interactive control. Then, I found this Jquery Gantt Chart library developed by Tait Brown under MIT license which look nicer and work better than viewing a gantt chart using Jasper Report. I want to use it to display my collected form data.

2. What is your idea to solve the problem?

We can develop a Userview Menu Plugin to use the Jquery Gantt Chart library to display the collected form data.

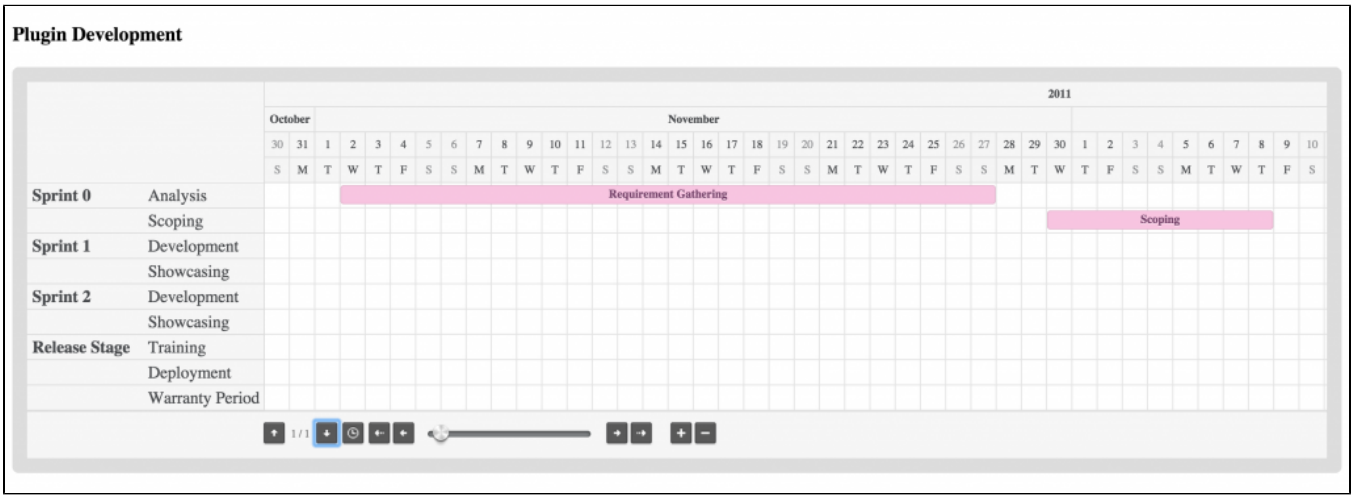
3. What is the input needed for your plugin?

To develop a Gantt Chart Userview Menu plugin, we can consider to provide the following as input.

1. Data Binder : We can reuse datalist binder to retrieve data
2. Data Mapping : Map the retrieve data from datalist binder to the data format of the library
3. Formatting options : Options to format and customise the gantt chart.

4. What is the output and expected outcome of your plugin?

By referring to the library demo, we can quickly come out a static HTML page like the picture below. As this is a Joget plugin tutorial, we will not go into the detail on coding the static HTML page. You can refer to static.zip. We will expect our userview page can display our collected data as the static HTML page.



5. Is there any resources/API that can be reuse?

If you are not familiar with FreeMarker syntax, you should have a look on their document before proceed.

6. Prepare your development environment

We need to always have our Joget Workflow Source Code ready and builded by following this guideline.

The following of this tutorial is prepared with a Macbook Pro and Joget Source Code version 8.0-Snapshot. Please refer to Guideline for Developing a Plugin for other platform command.

Let said our folder directory as following.

```
- Home
  - joget
    - plugins
    - jw-community
```

The "plugins" directory is the folder we will create and store all our plugins and the "jw-community" directory is where the Joget Workflow Source code stored.

Run the following command to create a maven project in "plugins" directory.

```
cd joget/plugins/
~/joget/jw-community/wflow-plugin-archetype/create-plugin.sh org.joget.tutorial gantt_chart_menu 8.0-Snapshot
```

Then, the shell script will ask us to key in a version for your plugin and ask us for confirmation before generate the maven project.

```
Define value for property 'version':  1.0-SNAPSHOT: : 8.0-Snapshot
[INFO] Using property: package = org.joget.tutorial
Confirm properties configuration:
groupId: org.joget.tutorial
artifactId: gantt_chart_menu
version: 5.0.0
package: org.joget.tutorial
Y: : y
```

We should get "BUILD SUCCESS" message shown in our terminal and a "gantt_chart_menu" folder created in "plugins" folder.

Open the maven project with your favourite IDE. I will be using NetBeans.

7. Just code it!

a. Extending the abstract class of a plugin type

Create a "GanttChartMenu" class under "org.joget.tutorial" package. Then, extend the class with **org.joget.apps.userview.model.UserviewMenu** abstract class. Please refer to Userview Menu Plugin.

b. Implement all the abstract methods

As usual, we have to implement all the abstract methods. We will using AppPluginUtil.getMessage method to support i18n and using constant variable MESSAGE_PATH for message resource bundle directory.

Implementation of all basic abstract methods

```
package org.joget.tutorial;

import org.joget.apps.app.service.AppPluginUtil;
import org.joget.apps.userview.model.UserviewMenu;

public class GanttChartMenu extends UserviewMenu {

    private final static String MESSAGE_PATH = "messages/GanttChartMenu";

    public String getName() {
        return "Gantt Chart Menu";
    }
    public String getVersion() {
        return "5.0.0";
    }

    public String getClassName() {
        return getClass().getName();
    }

    public String getLabel() {
        //support i18n
        return AppPluginUtil.getMessage("org.joget.tutorial.GanttChartMenu.pluginLabel", getClassName(),
MESSAGE_PATH);
    }

    public String getDescription() {
        //support i18n
        return AppPluginUtil.getMessage("org.joget.tutorial.GanttChartMenu.pluginDesc", getClassName(),
MESSAGE_PATH);
    }

    public String getPropertyOptions() {
        return AppUtil.readPluginResource(getClassName(), "/properties/ganttChartMenu.json", null, true,
MESSAGE_PATH);
    }

    @Override
    public String getCategory() {
        return "Marketplace";
    }
    @Override
    public String getIcon() {
        //sorry, i am reuse the icon of other plugin here
        return "/plugin/org.joget.apps.userview.lib.HtmlPage/images/grid_icon.gif";
    }

    @Override
    public boolean isHomePageSupported() {
        return true; // Can use as first page of the userview
    }

    @Override
    public String getDecoratedMenu() {
        return null; // using default
    }

    @Override
    public String getRenderPage() {
        throw new UnsupportedOperationException("Not supported yet.");
    }
}
```

Then, we have to do a UI for admin user to provide inputs for our plugin. In getPropertyOptions method, we have already specifed our Plugin Properties Options definition file is locate at "/properties/ganttChartMenu.json". Let us create a directory "resources/properties" under "gantt_chart_menu/src/main" directory. After create the directory, create a file named "ganttChartMenu.json" in the "properties" folder.

In the properties definition options file, we will need to provide options as below. Please note that we can use "@@message.key@@" syntax to support i18n in our properties options.

```
 [{
    title : '@@userview.ganttchart.config@@',
    properties : [{
        name : 'id',
        label : 'Id',
        type : 'hidden'
    },
    {
        name : 'customId',
        label : '@@userview.ganttchart.customId@@',
        type : 'textfield',
        regex_validation : '^[a-zA-Z0-9_]+$',
        validation_message : '@@userview.ganttchart.invalidId@@'
    },
    {
        name : 'label',
        label : '@@userview.ganttchart.label@@',
        type : 'textfield',
        required : 'True'
    },
    {
        name : 'title',
        label : '@@userview.ganttchart.title@@',
        type : 'textfield'
    },
    {
        name : 'binder',
        label : '@@userview.ganttchart.binder@@',
        type : 'elementselect',
        required : 'True',
        options_ajax : '[CONTEXT_PATH]/web/property/json/getElements?classname=org.joget.apps.datalist.model.
DataListBinderDefault',
        url : '[CONTEXT_PATH]/web/property/json[APP_PATH]/getPropertyOptions'
    },
    {
        label : '@@userview.ganttchart.mapping@@',
        type : 'header'
    },
    {
        name : 'category',
        label : '@@userview.ganttchart.mapping.category@@',
        type : 'textfield',
        required : 'True'
    },
    {
        name : 'task',
        label : '@@userview.ganttchart.mapping.task@@',
        type : 'textfield',
        required : 'True'
    },
    {
        name : 'activity',
        label : '@@userview.ganttchart.mapping.activity@@',
        type : 'textfield',
        required : 'True'
    },
    {
        name : 'fromDate',
        label : '@@userview.ganttchart.mapping.fromDate@@',
        type : 'textfield',
        required : 'True'
    },
    {
        name : 'toDate',
        label : '@@userview.ganttchart.mapping.toDate@@',
        type : 'textfield',
        required : 'True'
    },
    {
```

```
        name : 'dateFormat',
        label : '@@userview.ganttchart.mapping.dateFormat@@',
        type : 'textfield',
        required : 'True',
        value : 'yyyy-MM-dd'
    },
    {
        name : 'taskId',
        label : '@@userview.ganttchart.mapping.taskId@@',
        type : 'textfield'
    },
    {
        name : 'cssClass',
        label : '@@userview.ganttchart.mapping.cssClass@@',
        type : 'textfield'
    }]
},
{
    title : '@@userview.ganttchart.advanced@@',
    properties : [
    {
        name : 'itemsPerPage',
        label : '@@userview.ganttchart.itemsPerPage@@',
        type : 'textfield',
        required : 'True',
        value : '20'
    },
    {
        name : 'navigate',
        label : '@@userview.ganttchart.navigate@@',
        type : 'selectbox',
        required : 'True',
        value : 'scroll',
        options : [{
            value : 'buttons',
            label : '@@userview.ganttchart.navigate.buttons@@'
        },
        {
            value : 'scroll',
            label : '@@userview.ganttchart.navigate.scroll@@'
        }]
    },
    {
        name : 'scale',
        label : '@@userview.ganttchart.scale@@',
        type : 'selectbox',
        required : 'True',
        value : 'days',
        options : [{
            value : 'hours',
            label : '@@userview.ganttchart.scale.hours@@'
        },
        {
            value : 'days',
            label : '@@userview.ganttchart.scale.days@@'
        },
        {
            value : 'weeks',
            label : '@@userview.ganttchart.scale.weeks@@'
        },
        {
            value : 'months',
            label : '@@userview.ganttchart.scale.months@@'
        }]
    },
    {
        name : 'maxScale',
        label : '@@userview.ganttchart.maxScale@@',
        type : 'selectbox',
        required : 'True',
        value : 'months',
```

```
        options : [{
            value : 'hours',
            label : '@@userview.ganttchart.scale.hours@@'
        },
        {
            value : 'days',
            label : '@@userview.ganttchart.scale.days@@'
        },
        {
            value : 'weeks',
            label : '@@userview.ganttchart.scale.weeks@@'
        },
        {
            value : 'months',
            label : '@@userview.ganttchart.scale.months@@'
        }]
    },
    {
        name : 'minScale',
            label : '@@userview.ganttchart.minScale@@',
        type : 'selectbox',
        required : 'True',
        value : 'hours',
        options : [{
            value : 'hours',
            label : '@@userview.ganttchart.scale.hours@@'
        },
        {
            value : 'days',
            label : '@@userview.ganttchart.scale.days@@'
        },
        {
            value : 'weeks',
            label : '@@userview.ganttchart.scale.weeks@@'
        },
        {
            value : 'months',
            label : '@@userview.ganttchart.scale.months@@'
        }]
    },
    {
        name : 'useCookie',
        label : '@@userview.ganttchart.useCookie@@',
        type : 'checkbox',
        options : [{
            value : 'true',
            label : ''
        }]
    },
    {
        name : 'scrollToToday',
        label : '@@userview.ganttchart.scrollToToday@@',
        type : 'checkbox',
        options : [{
            value : 'true',
            label : ''
        }]
    },
    {
        name : 'onItemClick',
        label : '@@userview.ganttchart.onItemClick@@',
        type : 'codeeditor',
        mode : 'javascript',
        value : '//console.log(data); //data obj hold all the columns value of a row'
    },
    {
        name : 'onAddClick',
        label : '@@userview.ganttchart.onAddClick@@',
        type : 'codeeditor',
        mode : 'javascript',
        value : '//console.log(datetime); //the DateTime in ms for the clicked Cell\n//console.log(rowId);
```

```
//the row ID of clicked Cell'
    },
    {
        name : 'onRender',
        label : '@@userview.ganttchart.onRender@@',
        type : 'codeeditor',
        mode : 'javascript',
        value : '//console.log("chart rendered");'
    },
    {
        name : 'customHeader',
        label : '@@userview.ganttchart.customHeader@@',
        type : 'codeeditor',
        mode : 'html'
    },
    {
        name : 'customFooter',
        label : '@@userview.ganttchart.customFooter@@',
        type : 'codeeditor',
        mode : 'html'
    }]
}]
```

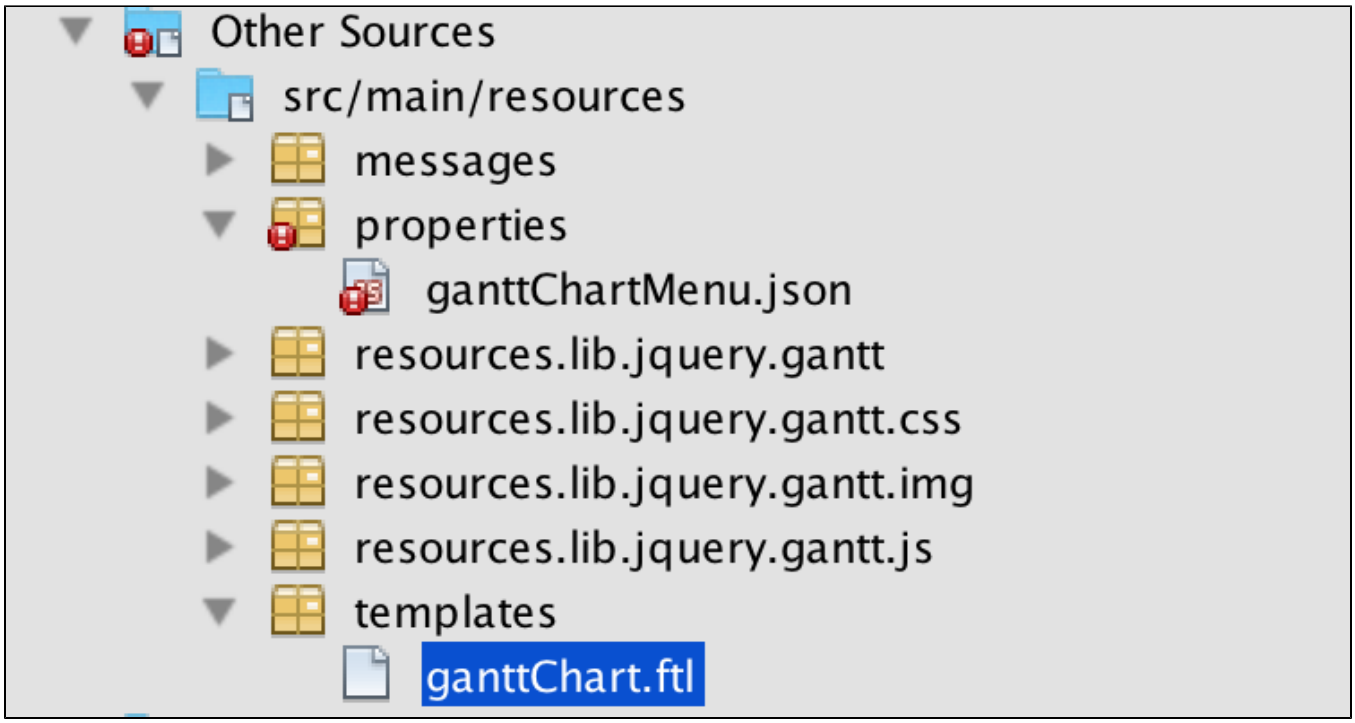
After done the properties option to collect input, we can work on the main method of the plugin which is getRenderPage method. Normally, what I will do before go into detail to populate the data to the view, I will first put the static content for the getRenderPage to build and test the plugin first. It everything is fine, then only we try to add data to the view.

```
    @Override
    public String getRenderPage() {
        Map model = new HashMap();
        model.put("request", getRequestParameters());
        model.put("element", this);

        PluginManager pluginManager = (PluginManager)AppUtil.getApplicationContext().getBean("pluginManager");
        String content = pluginManager.getPluginFreeMarkerTemplate(model, getClass().getName(), "/templates
/ganttChart.ftl", null);
        return content;
    }
```

Then, let us create a template file locate at "/templates/ganttChart.ftl". Let us create a directory "resources/templates" under "gantt_chart_menu/src/main" directory. After create the directory, create a file named "ganttChart.ftl" in the "templates" folder.

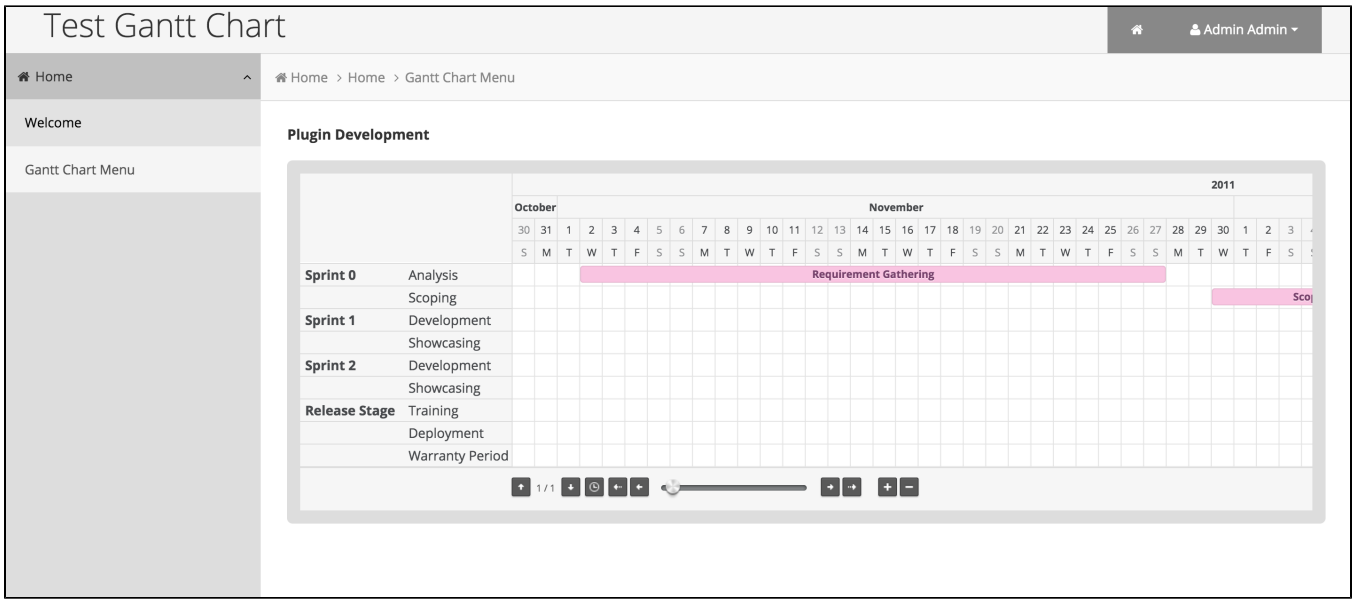
Put the static HTML we create previously into the template file as below. Remember to put all the dependencies javascript library and images under "gantt_chart_menu/src/main/resources/resources" and change the url as below accordingly. You project directory should look like the image below now.




```
<link href="${request.contextPath}/plugin/org.joget.tutorial.GanttChartMenu/lib/jquery/gantt/css/style.css"
rel="stylesheet" type="text/css" />
<script src="${request.contextPath}/plugin/org.joget.tutorial.GanttChartMenu/lib/jquery/gantt/js/jquery.fn.
gantt.min.js"></script>
<div class="gantt_chart_menu_body">
    <h3>Plugin Development</h3>
    <div class="gantt"></div>
    <script>
        $(function() {
            "use strict";
            $(".gantt").gantt({
                source: [{
                    name: "Sprint 0",
                    desc: "Analysis",
                    values: [{
                        from: "/Date(1320192000000)/",
                        to: "/Date(1322401600000)/",
                        label: "Requirement Gathering",
                        customClass: "ganttRed"
                    }]
                },{
                    name: " ",
                    desc: "Scoping",
                    values: [{
                        from: "/Date(1322611200000)/",
                        to: "/Date(1323302400000)/",
                        label: "Scoping",
                        customClass: "ganttRed"
                    }]
                 },{
                    name: "Sprint 1",
                    desc: "Development",
                    values: [{
                        from: "/Date(1323802400000)/",
                        to: "/Date(1325685200000)/",
                        label: "Development",
                        customClass: "ganttGreen"
                    }]
                },{
                    name: " ",
                    desc: "Showcasing",
```

```
                    values: [{
                        from: "/Date(1325685200000)/",
                        to: "/Date(1325695200000)/",
                        label: "Showcasing",
                        customClass: "ganttBlue"
                    }]
                },{
                    name: "Sprint 2",
                    desc: "Development",
                    values: [{
                        from: "/Date(1326785200000)/",
                        to: "/Date(1325785200000)/",
                        label: "Development",
                        customClass: "ganttGreen"
                    }]
                },{
                    name: " ",
                    desc: "Showcasing",
                    values: [{
                        from: "/Date(1328785200000)/",
                        to: "/Date(1328905200000)/",
                        label: "Showcasing",
                        customClass: "ganttBlue"
                    }]
                },{
                    name: "Release Stage",
                    desc: "Training",
                    values: [{
                        from: "/Date(1330011200000)/",
                        to: "/Date(1336611200000)/",
                        label: "Training",
                        customClass: "ganttOrange"
                    }]
                },{
                    name: " ",
                    desc: "Deployment",
                    values: [{
                        from: "/Date(1336611200000)/",
                        to: "/Date(1338711200000)/",
                        label: "Deployment",
                        customClass: "ganttOrange"
                    }]
                },{
                    name: " ",
                    desc: "Warranty Period",
                    values: [{
                        from: "/Date(1336611200000)/",
                        to: "/Date(1349711200000)/",
                        label: "Warranty Period",
                        customClass: "ganttOrange"
                    }]
                }],
                navigate: "scroll",
                maxScale: "hours",
                itemsPerPage: 10
            });
        });
    </script>
</div>
```

Now, for testing purpose, we can skip to c. Manage the dependency libraries of your plugin, d. Make your plugin internationalization (i18n) ready, e. Register your plugin to Felix Framework and f. Build it and testing then continue the below after testing it. You will get something similar to below in your userview.



After verify the static HTML is working in our plugin, we can further enhance it by adding data to the view. Now, modify your getRenderPage method and ganttChart.ftl template as below.

```
    @Override
    public String getRenderPage() {
        Map model = new HashMap();
        model.put("request", getRequestParameters());
        model.put("element", this);

        //populate data in JSON format
        model.put("data", getData());

        PluginManager pluginManager = (PluginManager)AppUtil.getApplicationContext().getBean("pluginManager");
        String content = pluginManager.getPluginFreeMarkerTemplate(model, getClass().getName(), "/templates
/ganttChart.ftl", MESSAGE_PATH);
        return content;
    }

    protected String getData() {
        String json = "[]";

        try {
            DataListCollection data = null;
            String idColumn = getPropertyString("taskId");

            //get the binder
            Object binderData = getProperty("binder");
            if (binderData != null && binderData instanceof Map) {
                Map bdMap = (Map) binderData;
                if (bdMap != null && bdMap.containsKey("className") && !bdMap.get("className").toString().
isEmpty()) {
                    PluginManager pluginManager = (PluginManager) AppUtil.getApplicationContext().getBean
("pluginManager");
                    DataListBinder binder = (DataListBinder) pluginManager.getPlugin(bdMap.get("className").
toString());

                    if (binder != null) {
                        Map bdProps = (Map) bdMap.get("properties");
                        binder.setProperties(bdProps);

                        data = binder.getData(null, bdProps, new DataListFilterQueryObject[0], null, null,
null, null);
                        if (idColumn.isEmpty()) {
                            idColumn = binder.getPrimaryKeyColumnName();
                        }
                    }
                }
```

```
            }

            JSONArray dataArry = new JSONArray();
            String dateFormat = getPropertyString("dateFormat");
            SimpleDateFormat sdf = new SimpleDateFormat(dateFormat);

            if (data != null && !data.isEmpty()) {
                String currentCategory = null;
                int cat_count = 0;
                String currentTask = null;
                int task_count = 0;
                int act_count = 0;

                JSONObject taskObj = null;
                JSONArray taskValuesArray = null;
                for (Object r : data) {
                    String id = getValue(r, idColumn);
                    String cat = getValue(r, getPropertyString("category"));
                    String task = getValue(r, getPropertyString("task"));
                    String act = getValue(r, getPropertyString("activity"));
                    String formDate = getValue(r, getPropertyString("fromDate"));
                    String toDate = getValue(r, getPropertyString("toDate"));
                    String status = getValue(r, getPropertyString("cssClass"));

                    if (currentTask == null || !currentTask.equals(task) || (currentTask.equals(task) && !cat.
equals(currentCategory))) {
                        currentTask = task;
                        if (taskObj != null) {
                            taskObj.put("values", taskValuesArray);
                            dataArry.put(taskObj);
                        }
                        taskObj = new JSONObject();
                        taskValuesArray = new JSONArray();
                        taskObj.put("desc", task);
                        taskObj.put("id", id);
                        task_count++;
                    }

                    if (currentCategory == null || !currentCategory.equals(cat)) {
                        currentCategory = cat;
                        taskObj.put("name", cat);
                        cat_count++;
                    }

                    JSONObject valueObj = new JSONObject();
                    JSONObject actObj = new JSONObject();
                    act_count++;

                    actObj.put("taskId", id);
                    actObj.put("category", cat);
                    actObj.put("task", task);
                    actObj.put("activity", act);
                    actObj.put("formDate", formDate);
                    actObj.put("toDate", toDate);
                    actObj.put("status", status);

                    valueObj.put("dataObj", actObj);
                    valueObj.put("label", act);
                    valueObj.put("from", "/Date("+sdf.parse(formDate).getTime()+")/");
                    valueObj.put("to", "/Date("+sdf.parse(toDate).getTime()+")/");
                    valueObj.put("customClass", "cat_"+cat_count+" task_"+task_count+" act_"+act_count+"
"+status.replace(" ", "_"));
                    taskValuesArray.put(valueObj);
                }
                if (taskObj != null) {
                    taskObj.put("values", taskValuesArray);
                    dataArry.put(taskObj);
                }
            }

            return dataArry.toString();
```

```
        } catch (Exception e) {
            LogUtil.error(GanttChartMenu.class.getName(), e, json);
        }

        return json;
    }

    protected String getValue(Object o, String name) {
        String value = "";

        try {
            Object v = LookupUtil.getBeanProperty(o, name);
            if (v != null) {
                return v.toString();
            }
        } catch (Exception e) {
            LogUtil.error(GanttChartMenu.class.getName(), e, name);
        }

        return value;
    }
```

```
<link href="${request.contextPath}/plugin/org.joget.tutorial.GanttChartMenu/lib/jquery.gantt/css/style.css"
rel="stylesheet" type="text/css" />
<script src="${request.contextPath}/plugin/org.joget.tutorial.GanttChartMenu/lib/jquery.gantt/js/jquery.fn.
gantt.min.js"></script>
<div class="gantt_chart_menu_body">
    <#if element.properties.title?? ><h3>${element.properties.title!}</h3></#if>
    ${element.properties.customHeader!}
    <div class="gantt"></div>
    <script>
        $(function() {
            "use strict";
            $(".gantt").gantt({
                source: ${data!},
                months: [@@userview.ganttChart.months.label@@],
                dow: [@@userview.ganttChart.dow.label@@],
                itemsPerPage: ${element.properties.itemsPerPage!},
                navigate: "${element.properties.navigate!}",
                scale: "${element.properties.scale!}",
                maxScale: "${element.properties.maxScale!}",
                minScale: "${element.properties.minScale!}",
                waitText: "@@userview.ganttChart.waitText@@",
                onItemClick: function (data) {
                    ${element.properties.onItemClick!}
                },
                onAddClick: function(datetime, rowId) {
                    ${element.properties.onAddClick!}
                },
                onRender: function() {
                    ${element.properties.onRender!}
                },
                useCookie: <#if element.properties.useCookie! == 'true'>true<#else>false</#if>,
                scrollToToday: <#if element.properties.scrollToToday! == 'true'>true<#else>false</#if>
            });
         });
    </script>
    ${element.properties.customFooter!}
</div>
```

c. Manage the dependency libraries of your plugin

Our plugin is using some libraries, we have to add all of them to our POM file.

```
        <!-- Change plugin specific dependencies here -->
        <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>jsp-api</artifactId>
            <version>2.0</version>
        </dependency>
        <dependency>
            <groupId>org.json</groupId>
            <artifactId>json</artifactId>
            <version>20080701</version>
        </dependency>
        <dependency>
            <groupId>displaytag</groupId>
            <artifactId>displaytag</artifactId>
            <version>1.2</version>
            <exclusions>
                <exclusion>
                    <artifactId>slf4j-api</artifactId>
                    <groupId>org.slf4j</groupId>
                </exclusion>
                <exclusion>
                    <artifactId>jcl104-over-slf4j</artifactId>
                    <groupId>org.slf4j</groupId>
                </exclusion>
                <exclusion>
                    <groupId>org.slf4j</groupId>
                    <artifactId>slf4j-log4j12</artifactId>
                </exclusion>
            </exclusions>
        </dependency>
        <!-- End change plugin specific dependencies here -->
```

d. Make your plugin internationalization (i18n) ready

We are using i18n message key in getLabel and getDescription method. We also used i18n message key in our properties options definition as well. So, we will need to create a message resource bundle properties file for our plugin.Create directory "resources/messages" under "gantt_chart_menu/src/main" directory. Then, create a "GanttChartMenu.properties" file in the folder. In the properties file, let add all the message keys and its label as below.

```
org.joget.tutorial.GanttChartMenu.pluginLabel=Gantt Chart Menu
org.joget.tutorial.GanttChartMenu.pluginDesc=Display data in a Gantt Chart view
userview.ganttchart.config=Configure Gantt Chart Menu
userview.ganttchart.customId=Custom ID
userview.ganttchart.invalidId=Only alpha-numeric and underscore characters allowed
userview.ganttchart.label=Label
userview.ganttchart.title=Page Title
userview.ganttchart.binder=Data Binder
userview.ganttchart.mapping=Column to Data Mappings
userview.ganttchart.mapping.category=Category (column ID)
userview.ganttchart.mapping.task=Task (column ID)
userview.ganttchart.mapping.activity=Activity (column ID)
userview.ganttchart.mapping.fromDate=Activity From Date (column ID)
userview.ganttchart.mapping.toDate=Activity To Date (column ID)
userview.ganttchart.mapping.dateFormat=Date format for Activity From/To Date
userview.ganttchart.mapping.taskId=Task Id (column ID)
userview.ganttchart.mapping.cssClass=Status (column ID, use as CSS class for styling)
userview.ganttchart.advanced=Advanced
userview.ganttchart.itemsPerPage=Item pre page
userview.ganttchart.navigate=Navigator
userview.ganttchart.navigate.buttons=Buttons
userview.ganttchart.navigate.scroll=Scroll
userview.ganttchart.scale=Default Scale
userview.ganttchart.scale.hours=Hours
userview.ganttchart.scale.days=Days
userview.ganttchart.scale.weeks=Weeks
userview.ganttchart.scale.months=Months
userview.ganttchart.maxScale=Maximum Scale
userview.ganttchart.minScale=Minimum Scale
userview.ganttchart.useCookie=Use cookie for storing chart states
userview.ganttchart.scrollToToday=Auto scroll to today after rendered
userview.ganttchart.onItemClick=On Item Clicked Event (Javascript)
userview.ganttchart.onAddClick=On Empty Space Clicked Event (Javascript)
userview.ganttchart.onRender=On Rendered Event (Javascript)
userview.ganttchart.customHeader=Custom Header (HTML)
userview.ganttchart.customFooter=Custom Footer (HTML)
userview.ganttChart.months.label="January", "February", "March", "April", "May", "June", "July", "August",
"September", "October", "November", "December"
userview.ganttChart.dow.label="S", "M", "T", "W", "T", "F", "S"
userview.ganttChart.waitText=Loading...
```

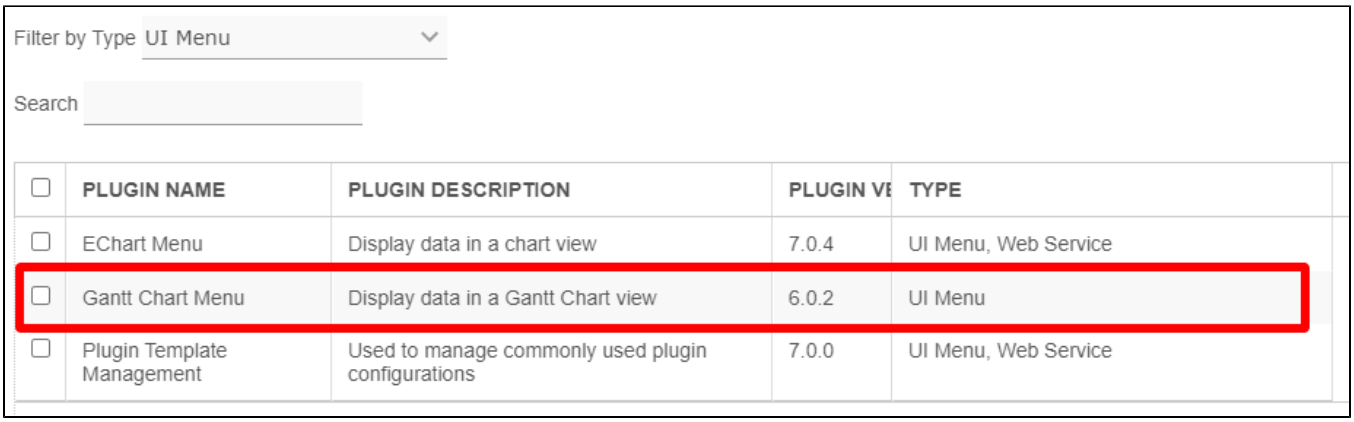
e. Register your plugin to Felix Framework

We will have to register our plugin class in Activator class (Auto generated in the same class package) to tell Felix Framework that this is a plugin.

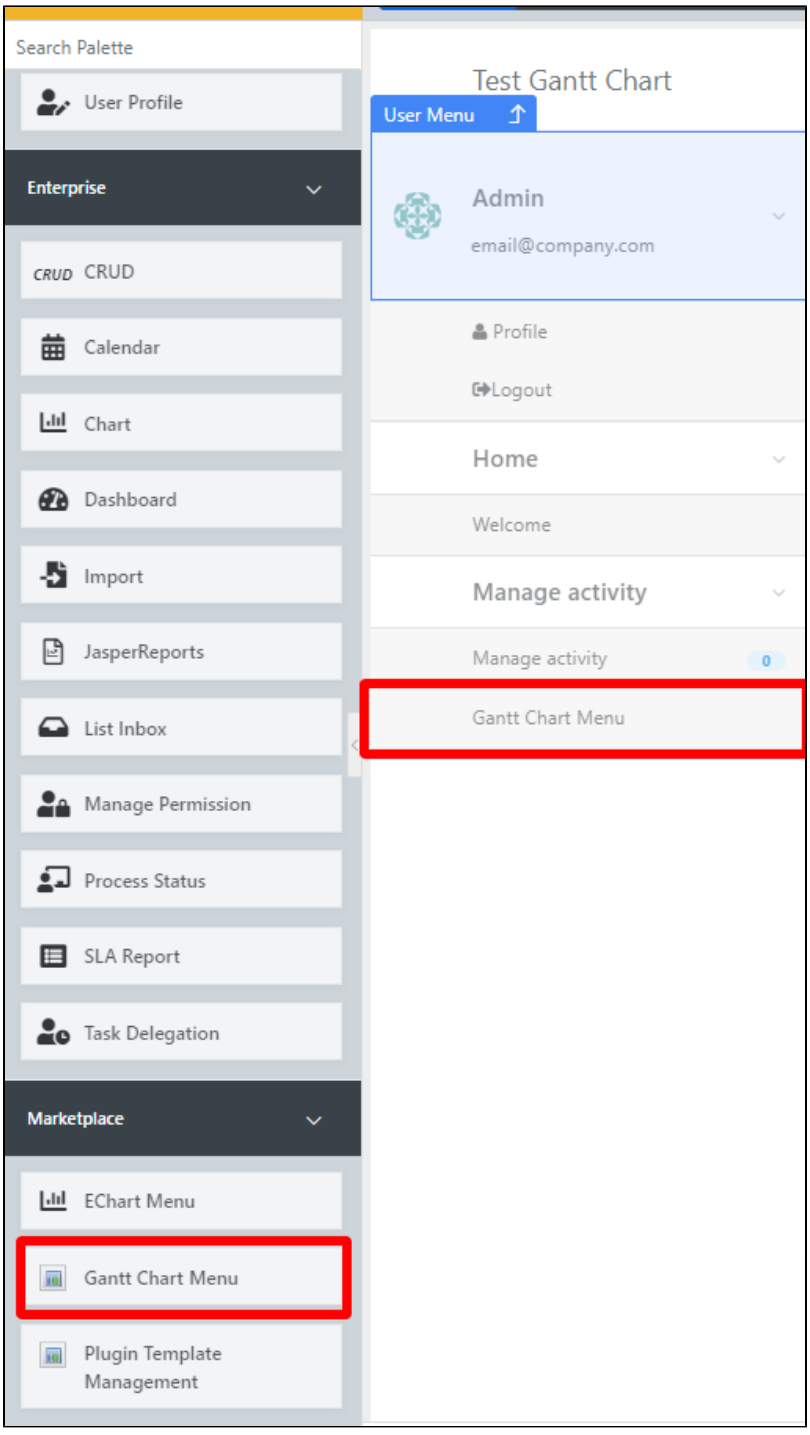
```
    public void start(BundleContext context) {
        registrationList = new ArrayList<ServiceRegistration>();
        //Register plugin here
        registrationList.add(context.registerService(GanttChartMenu.class.getName(), new GanttChartMenu(),
null));
    }
```

f. Build it and testing

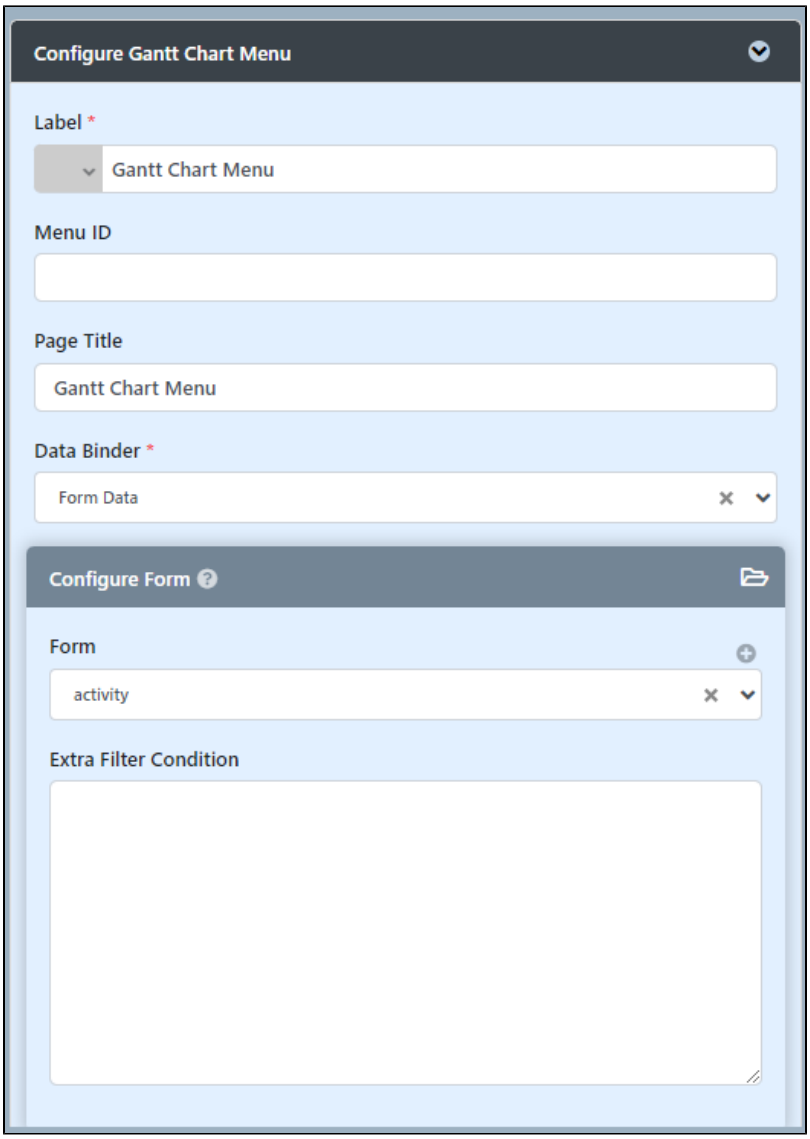
Let build our plugin. Once the building process is done, we will found a "gantt_chart_menu-5.0.0.jar" file is created under "gantt_chart_menu/target" directory.Then, let upload the plugin jar to Manage Plugins. After upload the jar file, double check the plugin is uploaded and activated correctly.

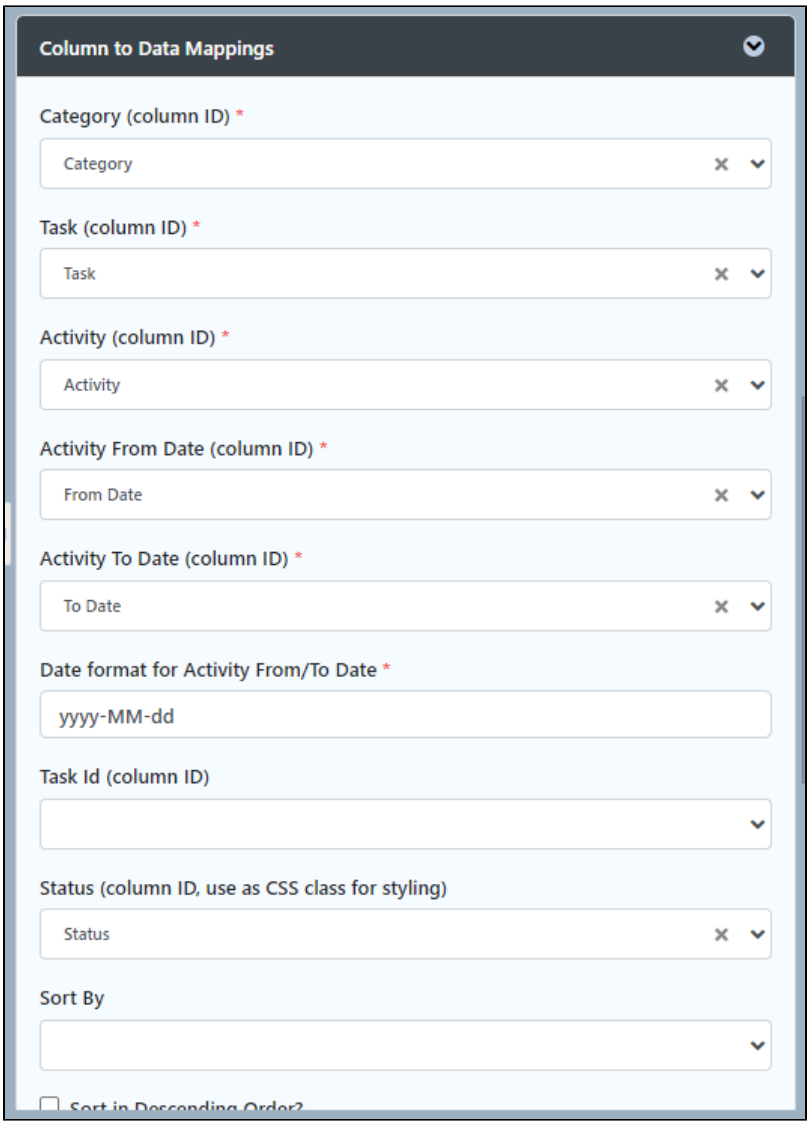


Open a userview, you will see the new plugin is added under "Marketplace". Drag it to one of your UI Category.

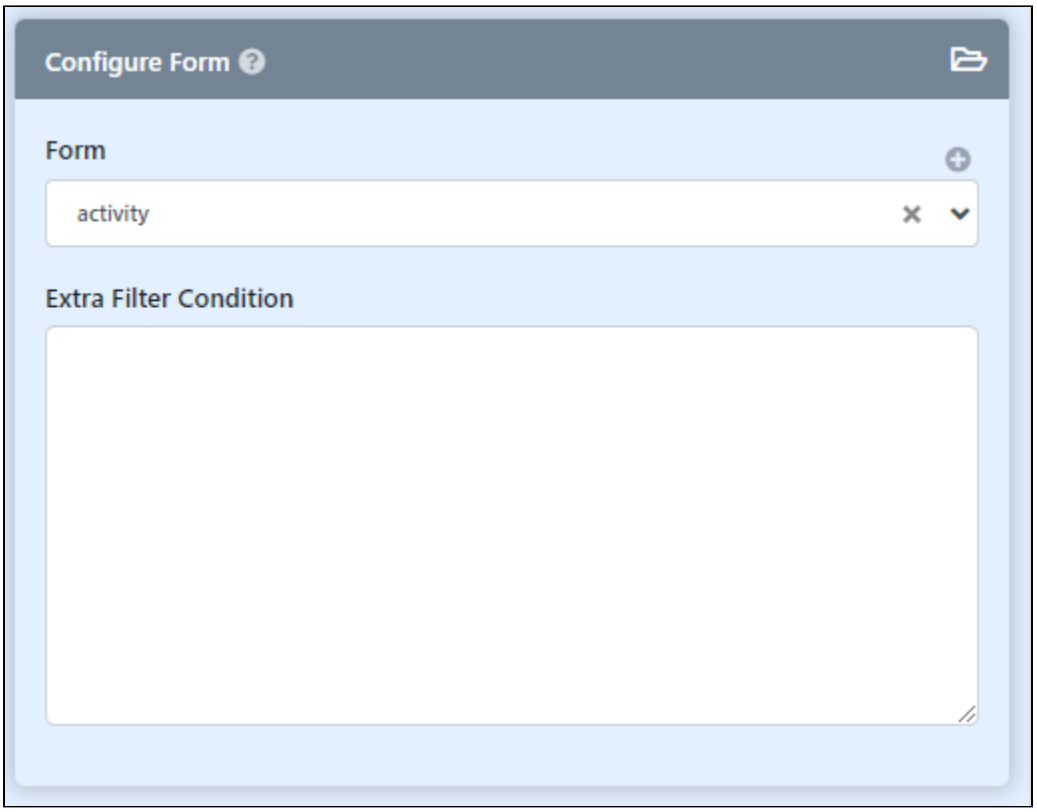


Edit the properties of the Gantt Chart Menu.

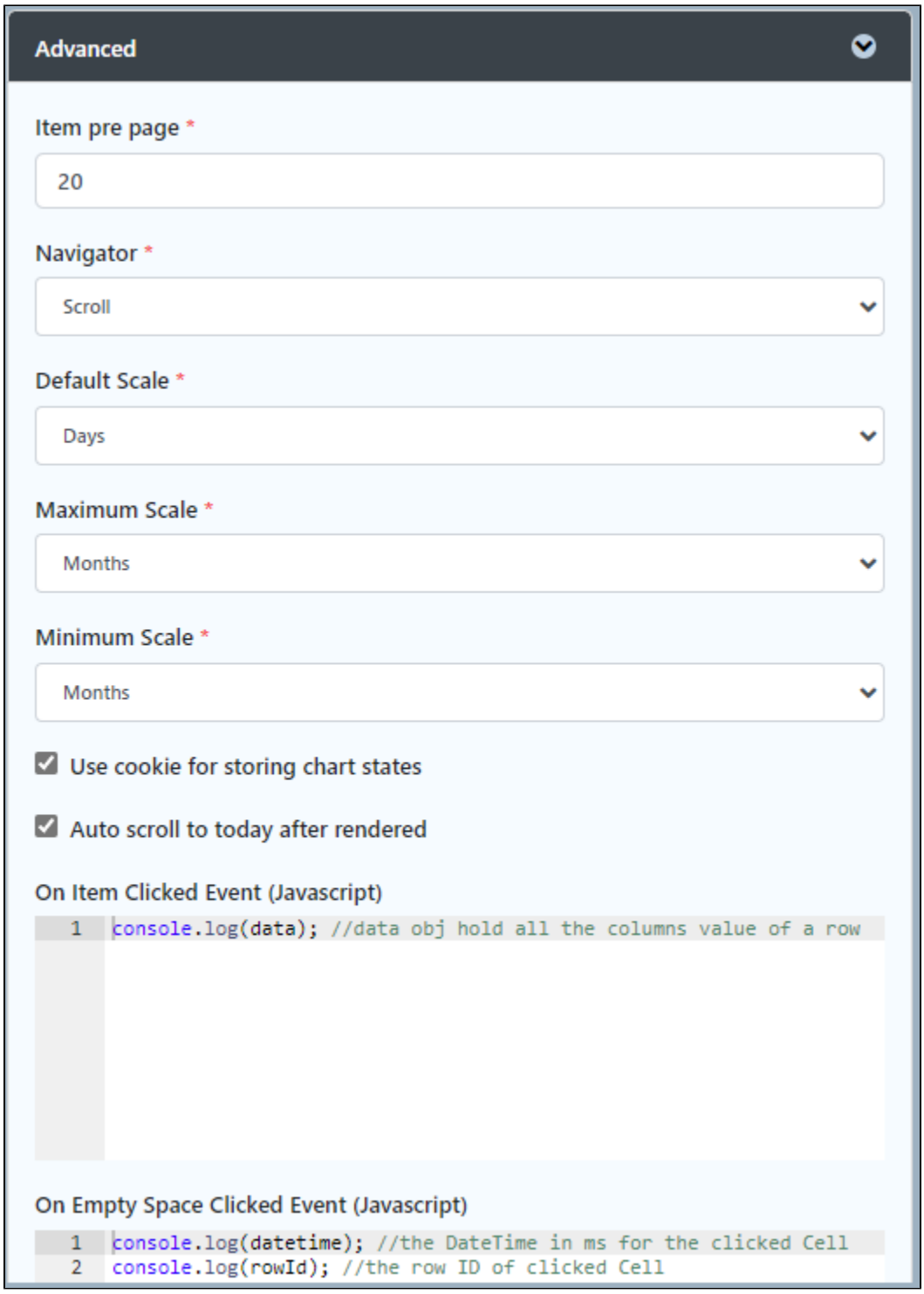

Configure Gantt Chart Menu
Label *
Gantt Chart Menu
Menu ID
Page Title
Gantt Chart Menu
Data Binder *
Form Data
Configure Form
Form
activity
Extra Filter Condition



I selected "Form Data Binder" as "Data Binder" for testing. Fill all the mappings to corresponding Field Id/Column Id.



Configure binder.



Advanced setting to configure the gantt chart.

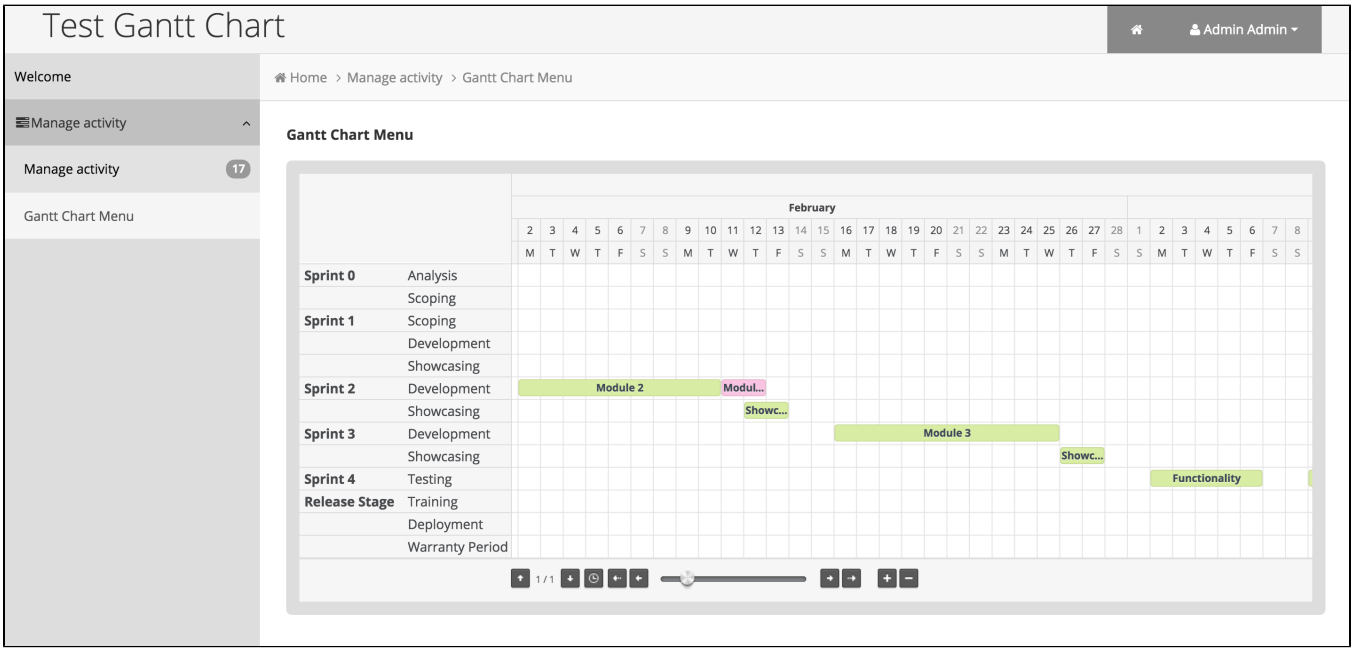
Custom Footer (HTML)

```
<style>
.fn-gantt .Done {
    background-color: #D8EDA3;
}
.fn-gantt .Pending {
    background-color: #FCD29A;
}
.fn-gantt .Aborted {
    background-color: #F9C4E1;
}
</style>
```

Writing some css styling in "Custom Footer (HTML)" option to give different colors for different status.

	Category	Task	Activity	From Date	To Date	Status	
☐	Sprint 0	Analysis	Requirement Gathering	2015-01-05	2015-01-12	Done	Edit
☐	Sprint 0	Scoping	Scoping	2015-01-13	2015-01-16	Done	Edit
☐	Sprint 1	Scoping	Scoping	2015-01-19	2015-01-21	Done	Edit
☐	Sprint 1	Development	Project Setup	2015-01-22	2015-01-23	Done	Edit
☐	Sprint 1	Development	Module 1	2015-01-26	2015-01-28	Done	Edit
☐	Sprint 1	Showcasing	Showcasing	2015-01-29	2015-01-30	Done	Edit
☐	Sprint 2	Development	Module 2	2015-02-02	2015-02-10	Done	Edit
☐	Sprint 2	Development	Module 3	2015-02-11	2015-02-12	Aborted	Edit
☐	Sprint 2	Showcasing	Showcasing	2015-02-12	2015-02-13	Done	Edit
☐	Sprint 3	Development	Module 3	2015-02-16	2015-02-25	Done	Edit

Populate some data to the form for testing.



The end result.

8. Take a step further, share it or sell it

You can download the source code from gantt_chart_menu.zip.

The test app for this tutorial is APP_testGanttChart-1-20151106194035.jwa.

To download the ready-to-use plugin jar, please find it in Gantt Chart Plugin.