AAVN UI Component

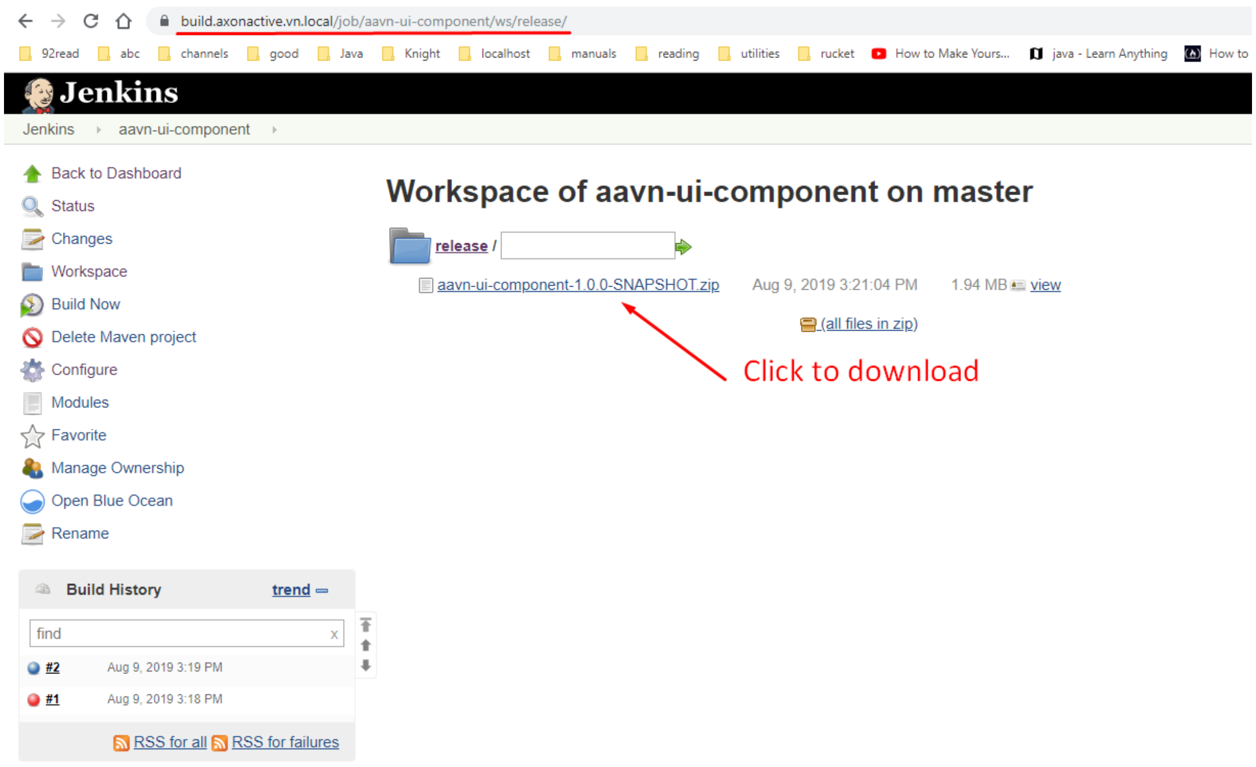
(User Guide)

Contents

I. Download

You can get *aavn-ui-component-<VERSION>.zip* from Jenkin workspace:

https://build.axonactive.vn.local/job/aavn-ui-component/ws/release/


build.axonactive.vn.local/job/aavn-ui-component/ws/release/
92read abc channels good Java Knight localhost manuals reading utilities rucket How to Make Yours... java - Learn Anything How to
Jenkins
Jenkins › aavn-ui-component ›
Back to Dashboard
Status
Changes
Workspace
Build Now
Delete Maven project
Configure
Modules
Favorite
Manage Ownership
Open Blue Ocean
Rename
Build History trend
find
#2 Aug 9, 2019 3:19 PM
#1 Aug 9, 2019 3:18 PM
RSS for all RSS for failures
Workspace of aavn-ui-component on master
release /
aavn-ui-component-1.0.0-SNAPSHOT.zip Aug 9, 2019 3:21:04 PM 1.94 MB view
(all files in zip)
Click to download

II. Install

Extract “src” folder of installation package, then overwrite the “src” folder in your Java EE project. The following folders and files will be added/overwritten:

```
\---src
    \---main
        \---webapp
~~          |   mini-example-of-using-dialog-template.xhtml~~
~~          |   mini-example-of-using-page-template.xhtml~~
            |
            +---resources
            |   \---library
            |       +---css
            |       |       aavn-form-components.min.css
            |       |       aavn-nova-light-theme.min.css
            |       |
            |       +---font-awesome
            |       |
            |       +---fonts
            |       |   \---roboto-condensed
            |       |
            |       +---js
            |       |       aavn-script.min.js
            |       |
            |       \---materialize
            |           \---css
            |
            \---WEB-INF
                    dialog-template.xhtml
                    page-template.xhtml
```

* If you want to uninstall the aavn-ui-component, you can remove the files and folders by yourself.

III. Convention coding UI

- Create a new special css/js file for your screen/feature
- About Icon: using icon of Font-Awesome (fa fa-icon)
- About Grid CSS: using of primefaces grid (ui-g)

IV. How to use?

1. Import css/js into your xhtml page by extending our templates:

Stylesheet and Javascript of aavn-ui-component already imported to the following files:

- /src/main/webapp/WEB-INF/dialog-template.xhtml

- /src/main/webapp/WEB-INF/page-template.xhtml

When creating a new page, you can extend the page-template.xhtml as following example:

new-page.xhtml

```
--------------------------------------------------------------------------------

<ui:composition xmlns="http://www.w3.org/1999/xhtml"
            xmlns:h="http://java.sun.com/jsf/html"
            xmlns:f="http://java.sun.com/jsf/core"
            xmlns:ui="http://java.sun.com/jsf/facelets"
            xmlns:p="http://primefaces.org/ui"
            template="/WEB-INF/page-template.xhtml">

        <ui:define name="head">
                <!-- define your head section here -->
        </ui:define>
        <ui:define name="body">
                <!-- define your body section here -->
        </ui:define>
        <ui:define name="script">
                <!-- define your script section here -->
        </ui:define>
</ui:composition>
--------------------------------------------------------------------------------
```

(Note: the library attribute based on you structure project)

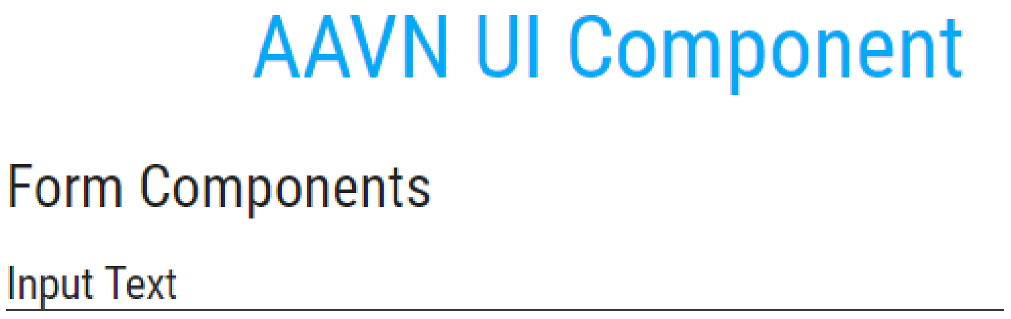
When creating a new dialog-framework-page, you can extend the dialog-template.xhtml as following example:

new-dialog-framework-page.xhtml

```
------------------------------------------------------------------------------------------

<ui:composition xmlns="http://www.w3.org/1999/xhtml"
              xmlns:h="http://java.sun.com/jsf/html"
              xmlns:f="http://java.sun.com/jsf/core"
              xmlns:ui="http://java.sun.com/jsf/facelets"
              xmlns:p="http://primefaces.org/ui"
              template="/WEB-INF/dialog-template.xhtml">

        <ui:define name="head">
                <!-- define your head section here -->
        </ui:define>
        <ui:define name="body">
                <!-- define your body section here -->
        </ui:define>
        <ui:define name="script">
                <!-- define your script section here -->
        </ui:define>
</ui:composition>
------------------------------------------------------------------------------------------
```

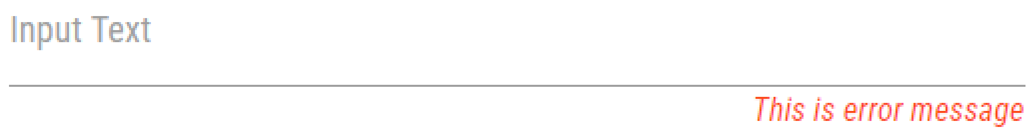
2. Heading:

- Page heading: `<h2 class="aavn-page-heading">AAVN UI Component</h2>`
- Heading lv1: `<h4 class="aavn-heading-lv1">Form Components</h4>`
- Heading lv2: `<h5 class="aavn-heading-lv2">Input field</h5>`



3. Using class “aavn-form-component” for the form to apply the CSS design for form component.

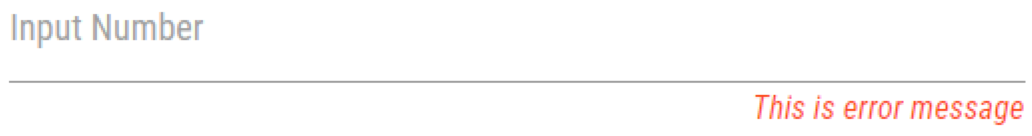
4. Form component: Input Text



```
<p:outputPanel styleClass="input-field aavn-input-field">
        <p:inputText id="inputText" styleClass="aavn-input-text" value=""></p:inputText>
        <label class="aavn-input-text-label active" for="inputText">Input Text</label>
        <h:outputText styleClass="aavn-error-msg-input-field" value="This is error message"></h:outputText>
</p:outputPanel>
```

(Note: in case you want to use h:message to show the error message, use the same CSS class "aavn-error-msg-input-field")

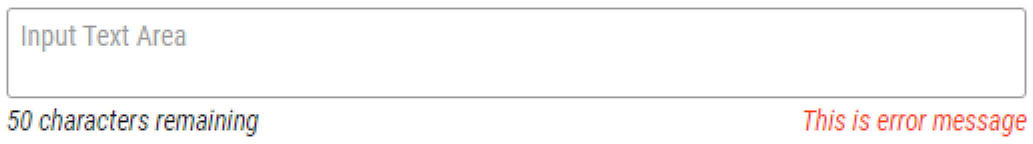
5. Form component: Input Number



```
<p:outputPanel class="input-field aavn-input-field">
        <p:inputNumber id="inputNumber" styleClass="aavn-input-number" value=""></p:inputNumber>
        <label class="aavn-input-number-label active" for="inputNumber">Input Number</label>
        <h:outputText styleClass="aavn-error-msg-input-field" value="This is error message"></h:outputText>
</p:outputPanel>
```

(Note: in case you want to use h:message to show the error message, use the same class "aavn-error-msg-input-field")

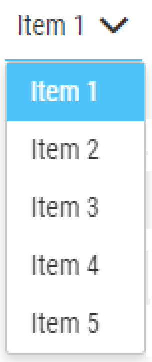
6. Form component: Input TextArea



```
<p:outputPanel styleClass="input-field aavn-input-field">
        <p:inputTextarea styleClass="aavn-text-area" counter="textArea-remaining-message"
counterTemplate="{0} characters remaining" maxlength="50" placeholder="Input Text Area"></p:inputTextarea>
        <h:outputText id="textArea-remaining-message" styleClass="aavn-remaining-msg-input-text-
area"></h:outputText>
        <h:outputText styleClass="aavn-error-msg-input-text-area" value="This is error message"></h:outputText>
</p:outputPanel>
```

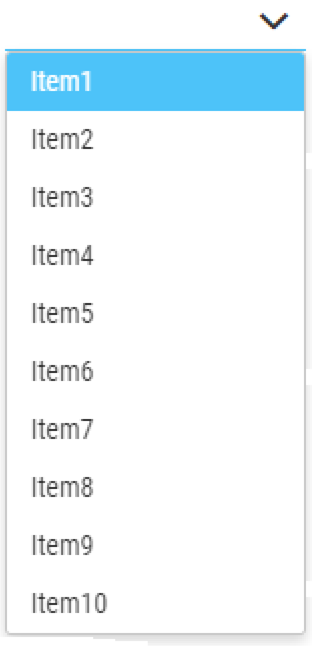
(Note: in case you want to use h:message to show the error message, use the same CSS class "aavn-error-msg-input-field")

7. Form component: Select Box



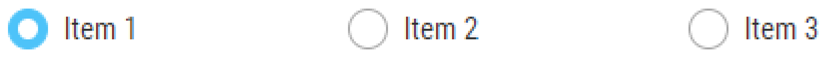
```
<p:selectOneMenu styleClass="aavn-select-box">
            <f:selectItem itemLabel="Item 1" itemValue="1" />
            <f:selectItem itemLabel="Item 2" itemValue="2" />
            <f:selectItem itemLabel="Item 3" itemValue="3" />
</p:selectOneMenu>
```

8. Form component: AutoComplete



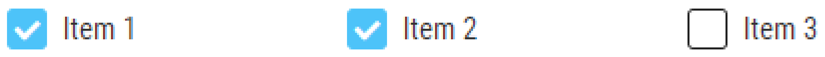
```
<p:autoComplete styleClass="aavn-auto-complete" dropdown="true"
value="#{formComponentBean.selectedAutoComplete}"
completeMethod="#{formComponentBean.completeItem}">
</p:autoComplete>
```

9. Form component: Radio Box



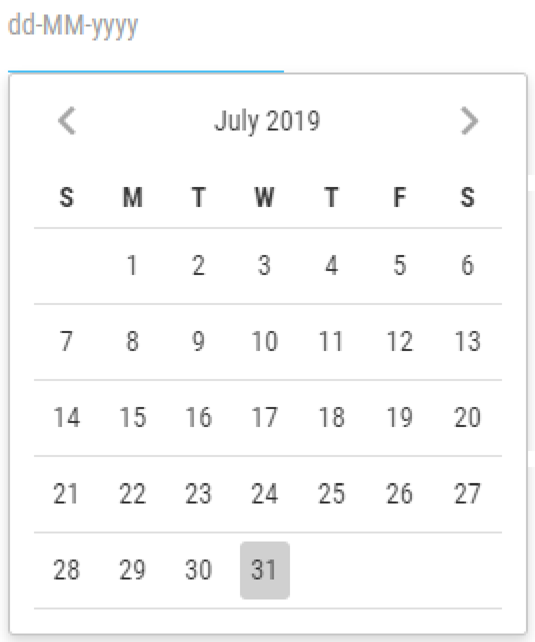
```
<p:selectOneRadio styleClass="aavn-radio-box">
            <f:selectItem itemLabel="Item 1" itemValue="1" />
            <f:selectItem itemLabel="Item 2" itemValue="2" />
            <f:selectItem itemLabel="Item 3" itemValue="3" />
</p:selectOneRadio>
```

10. Form component: Check Box



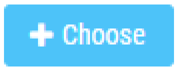
```
<p:selectManyCheckbox styleClass="aavn-checkbox">
            <f:selectItem itemLabel="Item 1" itemValue="1" />
            <f:selectItem itemLabel="Item 2" itemValue="2" />
            <f:selectItem itemLabel="Item 3" itemValue="3" />
</p:selectManyCheckbox>
```

11. Form component: Calendar



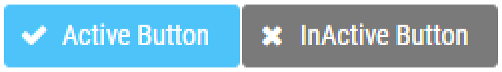
```
<p:calendar styleClass="aavn-calendar" pattern="dd-MM-yyyy" placeholder="dd-MM-yyyy"></p:calendar>
```

12. Form component: File Upload



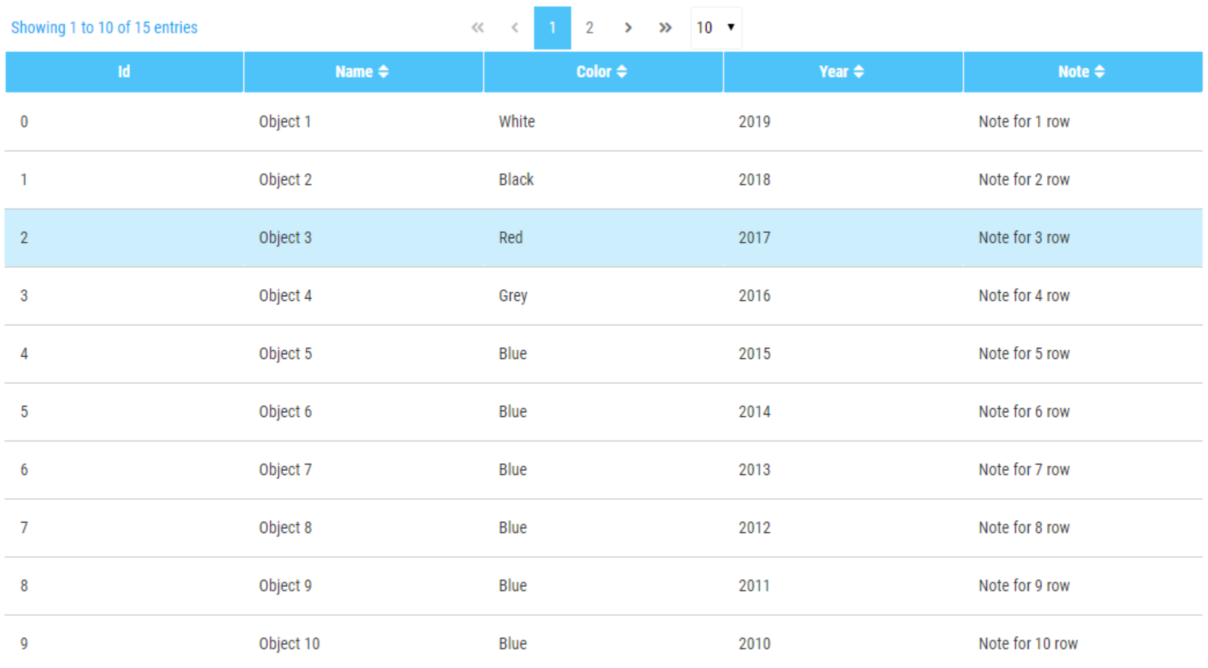
```
<p:fileUpload fileUploadListener="#{formComponentBean.handleFileUpload}" mode="advanced" styleClass="aavn-file-upload" auto="true" fileLimit="1" sizeLimit="100000" allowTypes="/(\.|\/)(gif|jpe?g|png)$/"></p:fileUpload>
```

13. Form component: CommandButton



```
<p:commandButton styleClass="aavn-active-button" icon="fa fa-check" value="Active Button"></p:commandButton>
<p:commandButton styleClass="aavn-inactive-button" icon="fa fa-times" value="InActive
Button"></p:commandButton>
```

14. Datatable

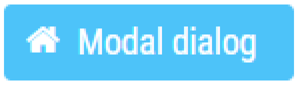

Showing 1 to 10 of 15 entries

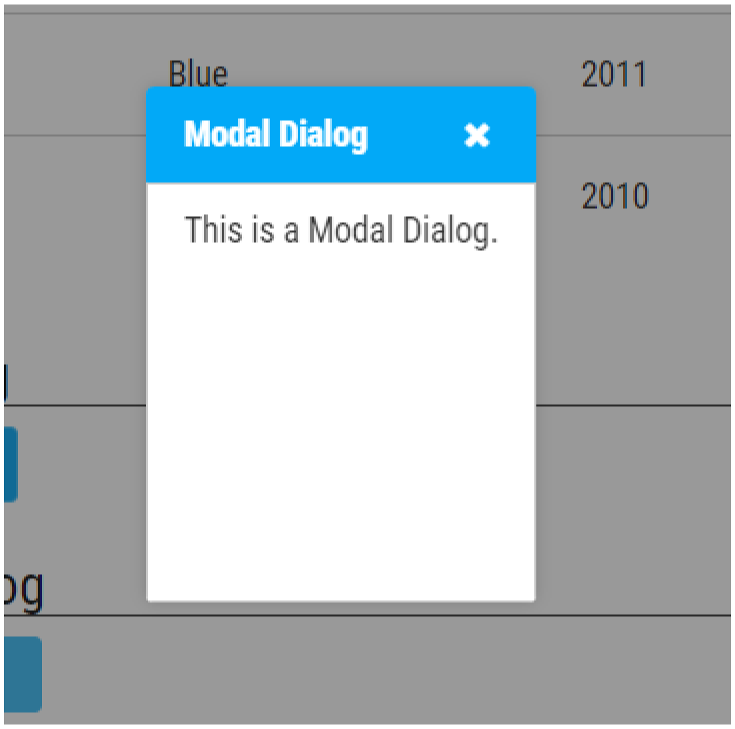
« ‹ 1 2 › » 10 ▾

Id	Name	Color	Year	Note
0	Object 1	White	2019	Note for 1 row
1	Object 2	Black	2018	Note for 2 row
2	Object 3	Red	2017	Note for 3 row
3	Object 4	Grey	2016	Note for 4 row
4	Object 5	Blue	2015	Note for 5 row
5	Object 6	Blue	2014	Note for 6 row
6	Object 7	Blue	2013	Note for 7 row
7	Object 8	Blue	2012	Note for 8 row
8	Object 9	Blue	2011	Note for 9 row
9	Object 10	Blue	2010	Note for 10 row

```
<p:dataTable styleClass="aavn-data-table"
        value="#{formComponentBean.dataTableObjs}" var="dataTableRow"
        rows="10" rowIndexVar="rowIndex" paginator="true" paginatorPosition="top"
        paginatorTemplate="{CurrentPageReport} {FirstPageLink} {PreviousPageLink} {PageLinks} {NextPageLink}
{LastPageLink} {RowsPerPageDropdown}"
        rowsPerPageTemplate="10,20,50,100"
        currentPageReportTemplate="Showing {startRecord} to {endRecord} of {totalRecords} entries"
        emptyMessage="No records found">

        <p:column headerText="Id">
                <h:outputText value="#{rowIndex}"></h:outputText>
        </p:column>

        <p:column headerText="Name" sortBy="#{dataTableRow.name}">
                <h:outputText value="#{dataTableRow.name}"></h:outputText>
        </p:column>

        <p:column headerText="Color" sortBy="#{dataTableRow.color}">
                <h:outputText value="#{dataTableRow.color}"></h:outputText>
        </p:column>

        <p:column headerText="Year" sortBy="#{dataTableRow.year}">
                <h:outputText value="#{dataTableRow.year}"></h:outputText>
        </p:column>

        <p:column headerText="Note" sortBy="#{dataTableRow.note}">
                <h:outputText value="#{dataTableRow.note}"></h:outputText>
        </p:column>
</p:dataTable>
```

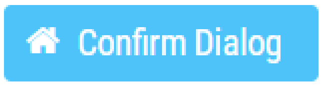
15.Modal Dialog

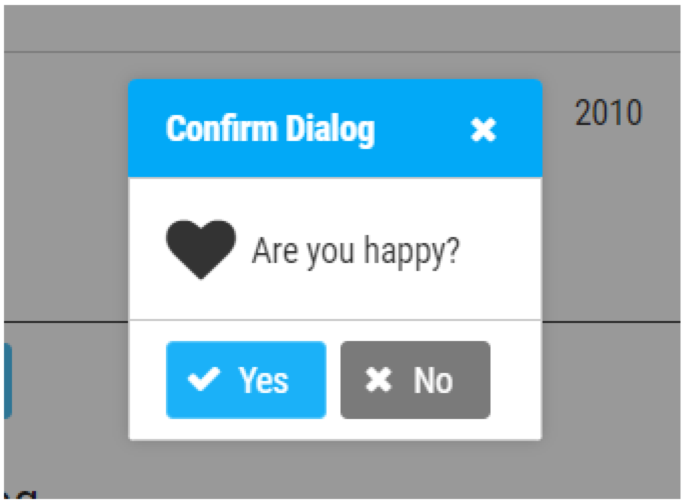




```
<p:commandButton styleClass="aavn-active-button" icon="fa fa-home" value="Modal dialog" onclick="PF('modal-
dialog').show()"></p:commandButton>
<p:dialog styleClass="aavn-modal-dialog" header="Modal Dialog" widgetVar="modal-dialog" modal="true"
height="150" closeOnEscape="true">
        <h:outputText value="This is a Modal Dialog."></h:outputText>
</p:dialog>
```

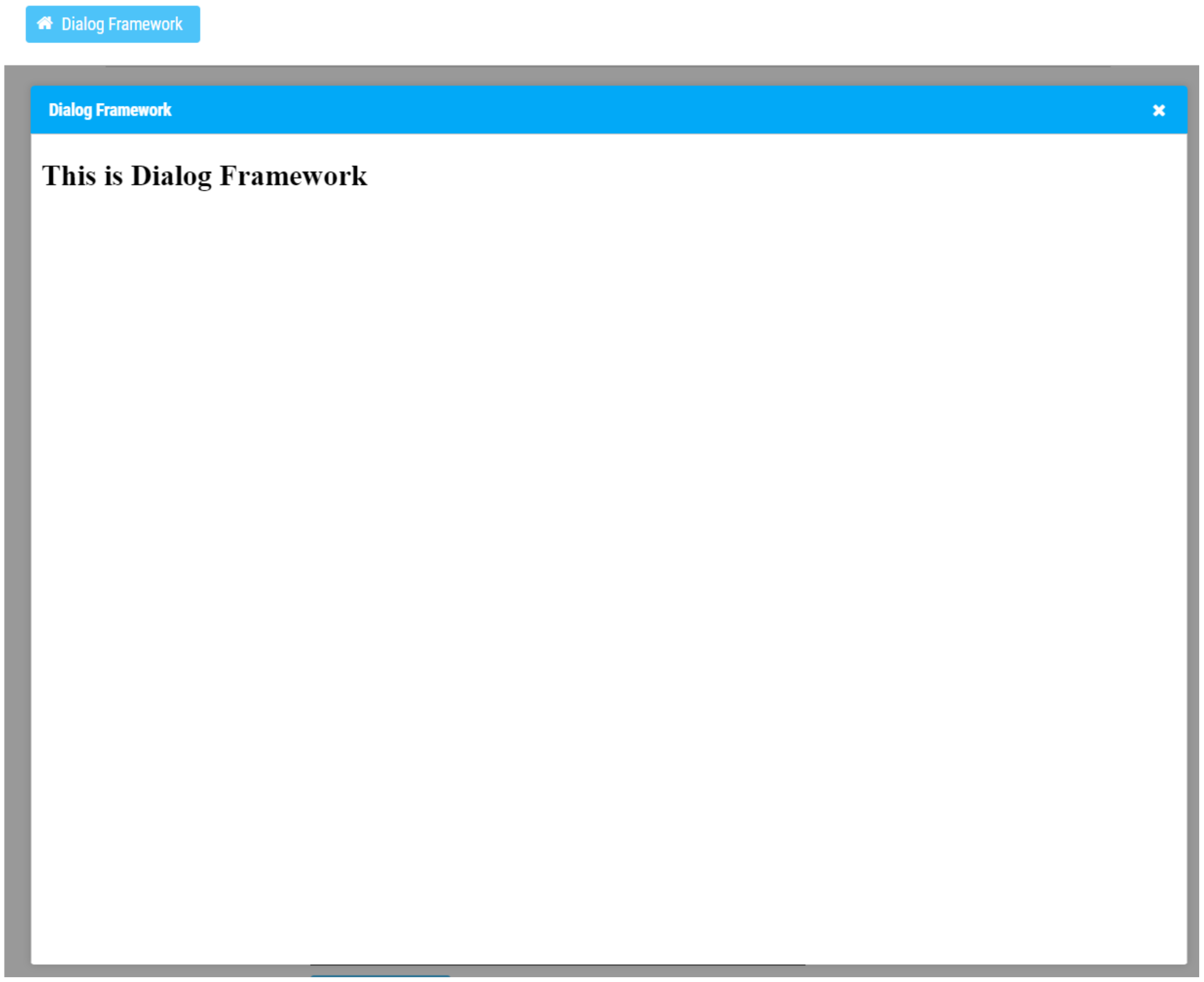
16. Confirm Dialog





```
<p:commandButton styleClass="aavn-active-button" icon="fa fa-home" value="Confirm Dialog">
        <p:confirm header="Confirm Dialog" message="Are you happy?" icon="fa fa-heart"></p:confirm>
</p:commandButton>
<p:confirmDialog styleClass="aavn-confirm-dialog" global="true" closeOnEscape="true">
        <p:commandButton value="Yes" type="button" styleClass="aavn-active-button" icon="fa fa-check" />
        <p:commandButton value="No" type="button" styleClass="aavn-inactive-button" icon="fa fa-times" />
</p:confirmDialog>
```

17. Dialog Framework



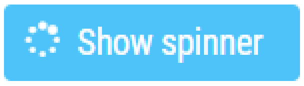
This is Dialog Framework

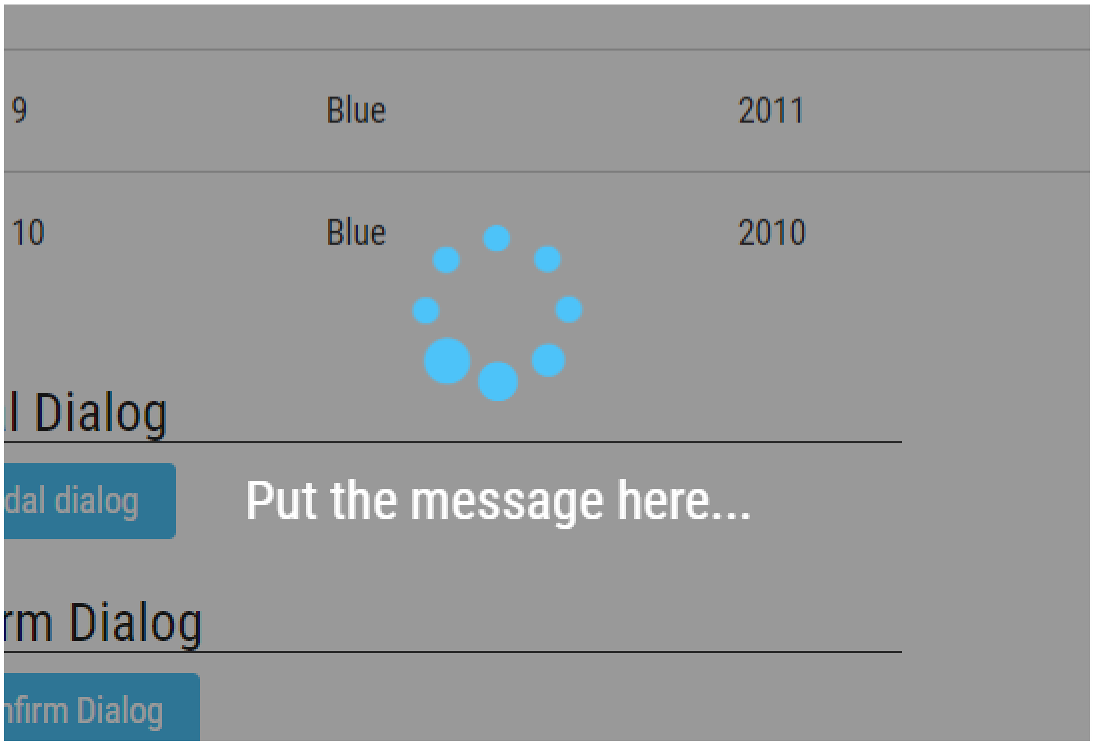
```
<p:commandButton styleClass="aavn-active-button" icon="fa fa-home" value="Dialog Framework"
action="#{formComponentBean.showDialogFramework}" ></p:commandButton>
```

In Bean:

```
public void showDialogFramework() {

        Map<String, Object> options = new HashMap<>();

        options.put("modal", true);

        options.put("width", "950px");

        options.put("height", "70vh");

        options.put("contentWidth", "100%");

        options.put("contentHeight", "100%");

        options.put("resizable", true);
        options.put("draggable", true);

        options.put("closable", true);

        options.put("closeOnEscape", true);


        Map<String, List<String>> params = new HashMap<>();

        PrimeFaces.current().dialog().openDynamic("DialogFramework.xhtml", options, params);

}
```

18. Spinner

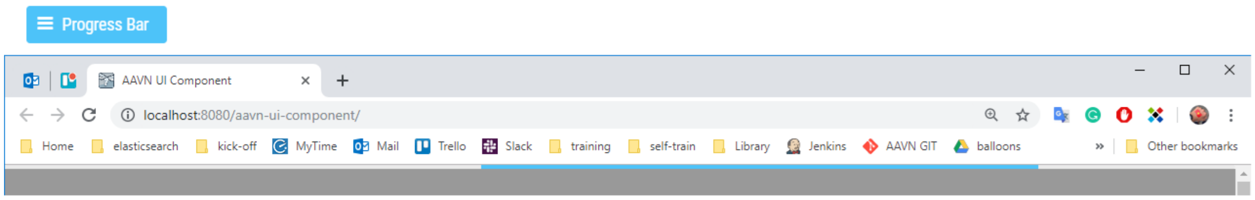




```
<p:commandButton styleClass="aavn-active-button" icon="fa fa-spinner fa-pulse" value="Show spinner"
onstart="showBackDrop('aavn-spinner')"></p:commandButton>

<div class="aavn-spinner">
        <div class="content">
                <i class="fa fa-spinner fa-pulse fa-5x fa-fw"></i>
                <div class="message">Put the message here...</div>
        </div>
</div>
```

(Note: incase you want to stop the snipper, use hideBackDrop('aavn-spinner') javascript method)

19. Progress Bar



```
<p:commandButton styleClass="aavn-active-button" icon="fa fa-bars" value="Progress Bar"
action="#{formComponentBean.toggleProgressBar()}" update="aavn-progressBar"
onstart="showBackDrop('aavn-progress-bar-container')" ></p:commandButton>

<p:outputPanel id="aavn-progressBar">
        <p:progressBar styleClass="aavn-progress-bar" mode="indeterminate"
rendered="#{formComponentBean.active}" ></p:progressBar>
</p:outputPanel>
<div class="aavn-progress-bar-container"></div>
```

In Bean:

```
@Getter private Boolean isActive;
public void toggleProgressBar() {
        this.isActive = !this.isActive;
}
```

(Note:

- Incase you want to stop the progress bar, use hideBackDrop('aavn-progress-bar-container') javascript method
- The method toggleProcessBar() in formComponentBean is change the value of active variable from false -> true and reverse)

This is all of common component, for more detail, you can run the aavn-ui-component project and access to the main.xhtml page to see the showcase.