Wakanda Widgets HTML Structure

In this manual, we describe each widget and how it is built in HTML so that you can add a Wakanda widget to your web page without using the GUI Designer. Most widgets are created in tags; however, many of Wakanda's widgets are in standard HTML tags for their type, e.g., the Text Input widget is defined in an <input> tag.

In your HTML page, you can define a datasource that can be used with the Wakanda widget so that you can interact with data on a Wakanda Server through one of Wakanda's widgets.

In the **Creating a widget instance in JavaScript** section, you can find out how to create a widget dynamically in JavaScript.

Accordion

Wakanda's **Accordion** widget is defined in a tag and is composed of multiple widgets (**Container** and **Text**) as shown below:

```
<div id="accordion1" data-type="accordion" data-lib="WAF" class="waf-widget waf-accordion " data-expand-several="true
<div id="container1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-0 waf-l
    <div id="container2" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-1 w
        <div id="richText1" data-type="richText" data-lib="WAF" class="waf-widget waf-richText waf-accordion-title wa
    </div>
    <div id="container3" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-1 w
</div>
<div id="container4" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-0 waf-l
    <div id="container5" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-1 w
        <div id="richText2" data-type="richText" data-lib="WAF" class="waf-widget waf-richText waf-accordion-title wa
    </div>
    <div id="container6" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-1 w
</div>
<div id="container7" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-0 waf-l
    <div id="container8" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-1 w
        <div id="richText3" data-type="richText" data-lib="WAF" class="waf-widget waf-richText waf-accordion-title wa
    </div>
    <div id="container9" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-1 w
</div>
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Accordion widget's properties are the following:

Property	Description
data-expand-several	True/False = More than one section opened
data-duration	Duration of time to expand/collapse a section
data-header-height	Header height
data-content-height	Height of the section's content container
data-linked-tag	Widgets linked to the main widget

Auto Form

Wakanda's **Auto Form** widget is defined in a tag.

```
<div id="autoForm1" data-type="autoForm" data-lib="WAF"
    data-binding="employee" data-draggable="true" data-resizable="true"
    data-withoutTable="true" data-resize-each-widget="true" data-resize-optimal-each-widget="true"
    data-display-error="true" data-errorDiv="errorDiv1"
    data-column-attribute="firstName,lastName,employer,salary,birthday,photo,married"
    data-column-name="First Name,Last Name,Employer,Salary,Birthday,Photo,Married"
    class="waf-widget waf-autoForm">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Auto Form widget's properties are the following:

Property	Description
data-binding	Datasource to bind to the widget
data-column-name	A string containing the titles (delimited by commas) for the attributes to display
data-column-attribute	A string containing the attribute names (delimited by commas) to use
data-withoutTable	True/False = With included widgets
data-resize-each-widget	True/False = Allow resizing of each widget
data-resize-optimal-each-widget	True/False = Auto-fit widgets
data-draggable	True/False = draggable by default
data-resizable	True/False = resizable by default
data-display-error	True/False = Allow Wakanda to manage the server errors
data-errorDiv	Widget ID that will contain the error messages sent by the server

Button

Wakanda's **Button** widget is built in a standard HTML <button> tag.
The following Button widget uses a datasource and an automatic action to save the current entity.

```
<button id="button1" data-type="button" data-lib="WAF"
    data-action="save" data-target="_blank"
    data-binding="company" data-text="Save"
    class="waf-widget waf-button">
</button>
```

This Button widget opens a URL in the browser:

```
<button id="button2" data-type="button" data-lib="WAF"
    data-action="simple" data-text="Wakanda"
    data-link="http://www.wakanda.org" data-target="_blank"
    class="waf-widget waf-button">
</button>
```

Below is the code for a standard Button widget:

```
<button id="button3" data-type="button" data-lib="WAF"
    data-action="simple" data-text="Click here" tabindex="6"
    class="waf-widget waf-button">
</button>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Button widget's properties are the following:

Property	Description
data-text	Button text
data-binding	When using any of the automatic actions, bind a datasource to the Button
data-action	The automatic action if you define a datasource for the Button. The automatic actions are: "Save", "Previous", "Next", "Create", "First", "Last", or "Remove". The Simple action is for simple Buttons or Button widgets that have a URL defined in the **data-link** property
data-link	URL to open for the Button widget
data-target	If a URL is defined, the options are "_blank" (to open in a new window) or "_self" (to open in the same window)
tabindex	Widget's tabindex

Calendar

Wakanda's **Calendar** widget is defined in a tag as shown below:

```
<div id="calendar1" data-type="calendar" data-lib="WAF"
    data-binding="employee.birthday" data-save="true" data-format="dd/mm/yy"
    data-calendars="2" data-start="7"
    class="waf-widget waf-calendar">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Calendar widget's properties are the following:

Property	Description
data-save	Automatically save (which automatically saves the change to the datasource)
data-format	Format for the date, by default it's "dd/mm/yy"
data-binding	Datasource to bind to the widget
data-mode	Single, multiple, or range
data-calendars	Number of views
data-start	Day of the week to start the Calendar (1=Monday, 2=Tuesday,...7=Sunday)

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Canvas

Wakanda's **Canvas** widget is defined in a <canvas> tag as shown below:

```
<canvas id="canvas1" data-type="canvas" data-lib="WAF"
    data-binding="sketchbook.drawing"
    data-src="/images/bg.png"
    class="waf-widget waf-canvas">
</canvas>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Canvas widget's properties are the following:

Property	Description
data-binding	Datasource to bind to the widget
data-src	Background image for the Canvas widget

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Chart

Wakanda's **Chart** widget is defined in a tag and is linked with a **Text** and two **Container** widgets:

```
<div id="container1" data-type="container" data-lib="WAF"
    class="waf-widget waf-container default waf-container-level-0 waf-chart-container">
    <div id="richText1" data-type="richText" data-lib="WAF"
        data-target="_blank" data-overflow="Hidden" data-plainText="true" data-autoWidth="false"
        class="waf-widget waf-richText default waf-chart-title ">
        Employee Pie Chart</div>
    <div id="container2" data-type="container" data-lib="WAF"
    class="waf-widget waf-container default waf-container-level-1 waf-level-1 waf-chart-legendary "></div>
    <div id="chart1" data-type="chart" data-lib="WAF"
        data-binding="employee" data-oldSource="employee" data-axisLabel="firstName"
        data-linked-tag="container1,richText1,container2"
        data-chartType="Pie Chart" data-chartLineType="basic"
        data-tooltipDisplay="true" data-tooltipType="blob"
        data-tooltipAngle="90" data-limitlength="100" data-draggable="false"
        data-yinterval="Step Count"
        data-labelAngle="90" data-labelAlign="middle"
        data-column="[{'sourceAttID':'salary','colID':'salary','width':'90','format':'$##,##0.00','title':'salary','c
        class="waf-widget waf-chart">
    </div>
</div>
```

The Chart's legendary is in a Container widget and must be linked with the Chart widget and must have waf-chart-legendary as one of its CSS classes.

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

This widget's properties are the following:

Property	Description
data-binding	Datasource to bind to the widget
data-axisLabel	Image for the hover state
data-chartType	Chart type (Pie Chart, Line Chart, or Bar Chart)
data-chartLineType	Line chart type ("basic"=Basic bar, "percentage"=Stacked percentage column, or "stacked"=Stacked column)
data-linked-tag	The widgets linked to the Chart widget (two Containers, one of which is the Legendary, and one Text widget, which is the title)
data-column	An array with an object that defines the columns in the chart: sourceAttID (datasource attribute), colID (column attribute), width, format, title, and color (expressed as an HTML color, e.g., "blue", a HEX value, e.g., "#CCC" or "#39C488", or a RGB value, e.g., "rgb (112, 0, 0)")
data-tooltipDisplay	True/False = display tool tip
data-tooltipAngle	Tooltip angle for Line and Bar charts
data-tooltipType	Tool tip type: blob, tag, label, popup, or flag
data-limitlength	Limit length property (defining the number of entities to display)
data-labelAngle	Axis label angle for the Bar chart
data-labelAlign	Axis label alignment for the Bar chart
data-axisLabel	The Axis label attribute
data-ymin	The y-axis minimum value
data-ymax	The y-axis maximum value
data-ystepvalue	The y-axis step value or range if data-yinterval property is "Range"
data-yinterval	The y-axis interval (range or steps)

Checkbox

Wakanda builds the **Checkbox** widget in a tag:

```
<div id="checkbox1" data-type="checkbox" data-lib="WAF"
    data-checked="false" data-binding="employee.married"
    data-errorDiv="errorDiv1" tabindex="0"
    class="waf-widget waf-checkbox">
</div>
```

Here is the Checkbox widget with different icons for its states:

```
<div type="checkbox" id="checkbox1" data-type="checkbox" data-lib="WAF"
    data-icon-default="/images/002_01.png" data-icon-hover="/images/002_02.png"
    data-icon-active="/images/002_04.png" data-icon-selected="/images/onebit_43.png"
    class="waf-widget waf-checkbox">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Checkbox widget's properties are the following:

Property	Description
data-binding	Datasource to bind to the widget
data-checked	True/False = defines if the widget is checked by default
data-errorDiv	Error ID linked to this widget
data-icon-default	Icon for the default state
data-icon-hover	Icon for the hover state
data-icon-active	Icon for the active state
data-icon-selected	Icon for the selected state
tabindex	The widget's tabindex

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Combo Box

Wakanda's **Combo Box** widget is defined in a tag. If there are static values, each one is defined in an <option>...</option> tag in the <select> tag contained in the .

The example below is for a Combo Box bound to a datastore class (to display the values) and bound to an attribute that accepts the selected value (defined by the **data-binding-key** property).

```
<div id="combobox1" data-type="combobox" data-lib="WAF"
    data-binding="company" data-binding-key="company"
    data-binding-out="employee.employer" data-binding-options="[name] "
    data-autoDispatch="true" data-editable="true" data-limit="20" tabindex="1"
    class="waf-widget waf-combobox" >
    <select><option value="">[name] </option></select>
</div>
```

The following Combo Box was created by defining static values:

```
<div id="combobox2" data-type="combobox" data-lib="WAF"
     data-limit="20" data-editable="true"
    class="waf-widget  waf-combobox">
    <select>
      <option value="New  York">New York</option>
      <option value="Boston">Boston</option>
      <option value="San  Francisco">San Francisco</option>
    </select>
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Combo Box widget's properties are the following:

Property	Description
data-limit	Number of entities to display in the Combo Box widget.
data-checked	True/False = defines if the widget is checked by default
data-editable	True = Combo Box; False = Dropdown menu (the **Autocomplete** property in the GUI Designer)
data-format	Format to use for displaying data
data-binding	Name of the datasource (datastore class or array) that will supply the values to display
data-binding-out	Name of the attribute to receive the selected value from the widget
data-binding-options	Name of the attribute to display in the widget
data-binding-key	The attribute from the datasource defined in the data-binding property to be returned
data-autoDispatch	True = Wakanda automatically sends the query to modify the entity collection
tabindex	The widget's tabindex
data-errorDiv	Widget ID that will contain the error messages sent by the server

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library

for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Component

Wakanda's **Component** widget is defined in a tag:

```
<div id="component1" data-type="component" data-lib="WAF"
    data-path="/login.waComponent"
    data-draggable="true" data-resizable="true" data-modal="true"
    class="waf-widget waf-component">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Component widget's properties are the following:

Property	Description
data-path	Path to the Web Component in your project's WebFolder.
data-resizable	True/False = resizable by default
data-draggable	True/False = draggable by default
data-modal	True/False = The Component widget will be used as a modal dialog

Container

Wakanda's **Container** widget is defined in a tag:

```
<div id="container1" data-type="container" data-lib="WAF"
    data-resizable="true" data-draggable="true"
    class="waf-widget waf-container">
</div>
```

If the Container is split either vertically or horizontally, it will be contain multiple Container widgets defined as shown below:

```
<div id="container1" data-type="container" data-lib="WAF"
    class="waf-widget waf-container default">
    <div id="container2" data-type="container" data-lib="WAF"
    class="waf-widget waf-container waf-split-container">
        <!--place widgets here-->
    </div>
    <div id="container3" data-type="container" data-lib="WAF"
    class="waf-widget waf-container waf-split-container">
        <!--place widgets here-->
    </div>
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Container widget's properties are the following:

Property	Description
data-resizable	True/False = resizable by default
data-draggable	True/False = draggable by default
data-hideSplitter	True/False = display splitters (if Container widget was split vertically or horizontally)
data-popup-display-button	Button widget's ID that appears when the Left area of a split Container widget is hidden either programmatically or automatically when the page (mobile or desktop) is resized and the left Container no longer fits

If using the Container widget for a section in the **Section** widget, there are a few additional properties:

Property	Description
data-rowType	This row type can be one of the following: "Label", "Blank", "Switch", "Input", or "NavBar"
data-rowIndex	The number of the section in order of its placement in the Section widget, starting at 0

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-	Margin (in pixels) from the widget defined by the **for** property

margin	
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Dialog

Wakanda's **Dialog** widget is defined in a tag and is composed of multiple widgets (**Container**, **Image**, **Button**, and **Text**):

```
<div id="dialog1" data-type="dialog" data-resizable="true" data-modal="true" data-load="/includeDialog.waPage/" data-
    <div id="container1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-1 w
        <div id="richText1" data-type="richText" data-plainText="true" data-overflow="Hidden" data-lib="WAF" data-aut
        <div id="image1" data-type="image" data-target="_blank" data-src="/walib/WAF/widget/dialog/images/alert.png"
        <button id="button1" data-type="button" data-text="+" data-lib="WAF" data-action="simple" class="waf-widget w
        <button id="button2" data-type="button" data-text="-" data-lib="WAF" data-action="simple" class="waf-widget w
        <button id="button3" data-type="button" data-text="x" data-lib="WAF" data-action="simple" class="waf-widget w
    </div>
    <div id="container2" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-1 w
    <div id="container3" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-1 w
        <button id="button4" data-type="button" data-text="Cancel" data-lib="WAF" data-action="simple" class="waf-wid
        <button id="button5" data-type="button" data-text="OK" data-lib="WAF" data-action="simple" class="waf-widget
    </div>
</div>
```

The main DOM node for the Dialog widget is the following:

```
<div id="dialog1" data-type="dialog" data-lib="WAF"
    data-load="/includeDialog.waPage/"
    data-linked-tag="container1,richText1,image1,button1,button2,button3,container2,container3,button4,button5"
    data-hideOnOutsideClick="true" data-modal="true"
    data-front="true" data-draggable="true" data-resizable="true"
    class="waf-widget waf-dialog">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Dialog widget's properties are the following:

Property	Description
data-load	Path to the HTML file to include
data-linked-tag	Widget IDs linked to the main widget (delimited by commas)
data-hideOnOutsideClick	Hide the widget when the user clicks outside of the widget
data-modal	Open the Dialog widget as a modal dialog
data-front	Float the Dialog widget on the top
data-draggable	True/False = draggable by default
data-resizable	True/False = resizable by default

Display Error

Wakanda's **Display Error** widget is defined in a tag:

```
<div id="errorDiv1" data-type="errorDiv" data-lib="WAF"
    class="waf-widget waf-errorDiv">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Display Error widget has no additional properties.

File Upload

Wakanda's **File Upload** widget is defined in a tag and is composed of multiple widgets (**Container** and **Button**):

```
<div id="fileUpload1" data-userAction="Ask the user" data-type="fileUpload" data-binding="employee.photoPath" data-te
    <button id="button1" data-type="button" data-text="..." data-lib="WAF" data-action="simple" class="waf-widget waf
    <button id="button2" data-type="button" data-text=" " data-lib="WAF" data-action="simple" class="waf-widget waf-b
    <button id="button3" data-type="button" data-text="n" data-lib="WAF" data-action="simple" class="waf-widget waf-b
    <button id="button4" data-type="button" data-text="x" data-lib="WAF" data-action="simple" class="waf-widget waf-b
    <div id="container1" data-type="container" data-linked-tag="fileUpload1" data-lib="WAF" class="waf-widget waf-con
</div>
```

The File Upload's main DOM node is the following:

```
<div id="fileUpload1" data-type="fileUpload" data-lib="WAF"
    data-binding="employee.photoPath" data-text="Drag your file(s) here"
    data-notification="true" data-listStyle="menu"
    data-maxfiles="2" data-maxfilesize="500" data-maxfilesize-unity="KB"
    data-action="false" data-folder="tmp" data-userAction="Ask the user"
    data-autoUpload="true"
    data-linked-tag="button1,button2,button3,button4,container1"
    class="waf-widget waf-fileUpload">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The File Upload widget's properties are the following:

Property	Description
data-binding	The datasource to store the uploaded file (if defined to be saved as a binary object)
data-text	Text to display in the File Upload's widget area
data-notification	True/False = Display notification when file is uploaded
data-listStyle	Can either be a Popup or List ("popup" or "menu")
data-maxfiles	Maximum number of files to upload (if you pass -1, it's unlimited)
data-maxfilesize	Maximum file size (if you pass -1, it's unlimited)
data-maxfilesize-unity	Unity for the file size: MB, KB, or byte ("MB", "KB", or "bytes")
data-action	True/False = save as a binary object
data-folder	Folder or temporary folder to upload the file ("tmp" by default)
data-userAction	Action to occur if there is a conflict with the file uploaded ("Ask the user", "Replace", or "Rename")
data-autoUpload	Automatically upload the file after it is selected
data-linked-tag	Widget IDs linked to the main widget (delimited by commas)

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget

data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Frame

Wakanda's **Frame** widget is defined in a standard HTML tag.
If you define a URL to display in the Frame, the following HTML is created:

```
<div id="frame1" data-type="frame" data-lib="WAF"
    data-src="http://www.wakanda.org/"
    class="waf-widget waf-frame">
</div>
```

If you bind a datasource to the Frame, the HTML is as follows:

```
<div id="frame2" data-type="frame" data-lib="WAF"
    data-binding="company.url"
    class="waf-widget waf-frame">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

Here are the Frame widget's properties:

Property	Description
data-src	A URL to display
data-binding	Datasource to bind to this widget

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.
Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Google Maps

Wakanda's **Google Maps** widget is defined in a tag.
Here is a Google Maps widget in dynamic mode bound to a datasource:

```
<div id="googleMaps1" data-type="googleMaps" data-lib="WAF"
data-mtype="dynamic" data-binding="company" data-address="company.fullAddress"
data-mapType="hybrid" data-zoom="15" data-language="English"  data-binding-tooltip="fullAddress"
data-marker-icon-selected="/images/002_01.png" data-marker-icon="/images/002_02.png"
data-infowindow="ID+:true+,name+:true+,address+:true+,city+:true+,country+:true+,fullAddress+:true"
data-infowindow-display="true" data-streetView="true" data-zoomControl="true"
data-scaleControl="true" data-panControl="false" data-marker-number="1"
data-marker-color="red" data-marker-size="mid" data-marker-label="A"
class="waf-widget waf-googleMaps" data-key=""></div>
```

This is a static Google Maps widget with a datasource bound to it:

```
<div id="googleMaps2" data-type="googleMaps" data-lib="WAF"
data-mtype="static" data-binding="company" data-address="company.fullAddress"
data-mapType="hybrid" data-zoom="15" data-language="English"
data-marker-color="red" data-marker-size="mid" data-marker-label="A"
class="waf-widget waf-googleMaps" ></div>
```

The following HTML code defines a static Google Maps widget with a defined position:

```
<div id="googleMaps3" data-type="googleMaps" data-lib="WAF"
data-mtype="static" data-position="los angeles, ca, usa"
data-zoom="16" data-marker-size="mid" data-marker-label="A"
data-marker-color="red" data-mapType="hybrid" data-language="English"
class="waf-widget waf-googleMaps"></div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

For both types of the Google Maps widget, the following properties are common to both:

Property	Description
data-mtype	"static" or "dynamic"
data-binding	Datasource to bind to this widget (datastore class)
data-address	Datasource defining address to bind to this widget (datastore class and attribute)
data-zoom	Zoom value from 1-20
data-mapType	Map type: satellite, hybrid, roadmap, or terrain
data-language	Define which language, by default it's "English"

The dynamic widget's properties are the following:

Property	Description
data-infowindow	Definition of the bubble information per attribute, e.g., "ID+:true+,name+:true"
data-infowindow-display	True/False = Automatically display the bubble
data-streetView	True/False = In the map's controls, display the street view
data-zoomControl	True/False = In the map's controls, display the zoom
data-scaleControl	True/False = In the map's controls, display the scale
data-panControl	True/False = In the map's controls, display the pan
data-marker-number	In the Marker section, define number of markers
data-key	The Google key that will allow you to use the API in your Web application

The static Google Maps widget's properties are the following:

Property	Description
data-marker-color	Color of the marker
data-marker-size	Size of the marker ("tiny", "mid", or "small")
data-marker-label	The marker's one-character label. If left blank, it's a bullet

Grid

Wakanda's **Grid** widget is defined in a tag.

```
<div id="dataGrid1" data-type="dataGrid" data-lib="WAF"
    data-binding="employee"
    data-column="[{'sourceAttID':'ID','colID':'ID','width':'70','title':'ID'},
    {'sourceAttID':'firstName','colID':'firstName','width':'150','title':'First Name'},
    {'sourceAttID':'lastName','colID':'lastName','width':'150','title':'Last Name'},
    {'sourceAttID':'birthday','colID':'birthday','width':'100','title':'Birthday','readOnly':'true'},
    {'sourceAttID':'salary','colID':'salary','width':'90','title':'Salary','format':'$###,##0.00'}]"
    data-selection-mode="multiple" data-readOnly="true"
    data-errorDiv="errorDiv1" data-display-error="true"
    data-draggable="true" data-resizable="true" data-header-hide="true"
    data-footer-hide="true" data-footer-text="item(s)"
    class="waf-widget waf-dataGrid">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Grid widget's properties are the following:

Property	Description
data-binding	Datasource to bind to this widget
data-column	An array containing an object to define the attributes. The properties are: "sourceAttID" (attribute name), "colID" (attribute name), "width", "title", "format", and "readOnly" (true or false).
data-selection-mode	Selection mode either "single" or "multiple"
data-readOnly	True/False = Read only
data-display-error	True/False = Allow Wakanda to manage the server errors
data-errorDiv	Widget ID that will contain the error messages sent by the server
data-draggable	True/False = draggable by default
data-resizable	True/False = resizable by default
data-header-hide	True/False = Hide header
data-footer-hide	True/False = Hide footer
data-footer-text	Text to display in the footer along with the number of entities displayed

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property

class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Icon

Wakanda's **Icon** widget is defined in a tag as shown below with an <img> tag for each of the icon's images for each state:

```
<span id="icon1" data-type="icon" data-lib="WAF"
    data-image-state4="/images/002_04.png" data-image-state3="/images/002_03.png"
    data-image-state2="/images/002_02.png" data-image-state1="/images/002_01.png"
    data-icon-type="images"
    class="waf-widget waf-icon">
    <img src="/images/002_01.png" class="waf-icon-state1"/><img src="/images/002_02.png" class="waf-icon-state2"/>
    <img src="/images/002_03.png" class="waf-icon-state3"/><img src="/images/002_04.png" class="waf-icon-state4"/>
</span>
```

If you define the image type to be "sprite", the tag is the following:

```
<span id="icon2" data-type="icon" data-lib="WAF"
    data-image-state1="/images/images.jpg" data-icon-type="sprite"
    class="waf-widget waf-icon">
</span>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

This widget's properties are the following:

Property	Description
data-image-state1	Image for the default state
data-image-state2	Image for the hover state
data-image-state3	Image for the active state
data-image-state4	Image for the disabled state
data-icon-type	Type of icon ("images" or "sprite") If the type is "images", each of the images defining the different states are defined in <img> tags with the appropriate class, i.e., "waf-icon-state2"

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Image

Wakanda's **Image** widget is defined in a tag as shown below:

```
<div id="image1" data-type="image" data-lib="WAF"
    data-binding="employee.photo" data-fit="4"
    class="waf-widget waf-image">
</div>
```

If you define an image and a URL for the Image widget, its DOM node is as follows:

```
<div id="image2" data-type="image" data-lib="WAF"
    data-src="/images/images.jpg" data-fit="2"
    data-link="http://www.wakanda.org" data-target="_blank"
    class="waf-widget waf-image">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Image widget's properties are the following:

Property	Description
data-binding	Datasource to bind to this widget
data-src	Image URL
data-fit	Fit to properties for the Image widget (0="to container", 1="to widgth", 2="to height", 3="container to image", or 4="to container (proportionately)")
data-link	URL to open for this widget
data-target	If a URL is defined, the options are "_blank" (to open in a new window) or "_self" (to open in the same window)

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Image Button

Wakanda's **Image Button** widget is built in a standard HTML tag.

The following Image Button widget uses a datasource and an automatic action to save the current entity. The Image Button widget is linked with two other widgets: **Icon** and Label.

```
<div id="imageButton1" data-lib="WAF" data-type="buttonImage"
    data-linked-tag="icon1" data-binding="company"  data-action="save"
    class="waf-widget waf-buttonImage">
        <span id="icon1" data-type="icon" data-lib="WAF"
            data-linked-tag="imageButton1"
            data-label-position="right" data-label="Save"
            data-image-state4="/images/002_04.png" data-image-state3="/images/002_03.png"
            data-image-state2="/images/002_02.png" data-image-state1="/images/002_01.png"
            data-icon-type="images"
            class="waf-widget waf-icon">
                <img src="/images/002_01.png" class="waf-icon-state1"/><img src="/images/002_02.png" class="waf-icon-
                <img src="/images/002_03.png" class="waf-icon-state3"/><img src="/images/002_04.png" class="waf-icon-
        </span>
        <label id="label1" data-lib="WAF" data-type="label"
            for="icon1" data-valign="middle" data-margin="5"
            class="waf-widget waf-label waf-label-icon">
            Save
        </label>
</div>
```

This widget opens a URL in the browser:

```
<div id="imageButton1" data-lib="WAF" data-type="buttonImage"
    data-linked-tag="icon1" data-link="http://www.wakanda.org/" data-target="_blank"
    class="waf-widget waf-buttonImage">
        <span id="icon1" data-type="icon" data-lib="WAF"
            data-linked-tag="imageButton1"
            data-label-position="right" data-label="Wakanda"
            data-image-state4="/images/002_04.png" data-image-state3="/images/002_03.png"
            data-image-state2="/images/002_02.png" data-image-state1="/images/002_01.png"
            data-icon-type="images"
            class="waf-widget waf-icon">
                 <img src="/images/002_01.png" class="waf-icon-state1"/><img src="/images/002_02.png" class="waf-icon
                 <img src="/images/002_03.png" class="waf-icon-state3"/><img src="/images/002_04.png" class="waf-icon
        </span>
        <label id="label1" data-lib="WAF" data-type="label"
            for="icon1" data-valign="middle" data-margin="5"
            class="waf-widget waf-label waf-label-icon">
            Wakanda
        </label>
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

This widget's properties are the following:

Property	Description
data-linked-tag	Icon widget's ID
data-binding	Datasource to apply to any of the automatic actions
data-action	The automatic action if you define a datasource for the Image Button. The automatic actions are: Save, Previous, Next, Create, First, Last, or Remove.
data-link	URL to open for this widget.
data-target	If a URL is defined, the options are "_blank" (to open in a new window) or "_self" (to open in the same window)

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

List View

Wakanda's **List View** widget is defined in a tag and is composed of multiple widgets (**Matrix**, **Container**, **Image**, **Button**, and **Text**):

```
<div id="listView1" data-type="listView" data-lib="WAF" data-inset="true" data-image="true" data-number="true" data-m
 <div id="container1" data-type="matrix" data-lib="WAF" class="waf-widget waf-matrix waf-list-matrix " data-fit="fals
      <div id="row1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-list-row ">
           <div id="mainText1" data-type="richText" data-lib="WAF" class="waf-widget waf-richText waf-list-mainLabel
           <button id="goTo1" data-type="button" data-lib="WAF" class="waf-widget waf-button waf-list-buttonGoTo " da
           <div id="secondText2" data-type="richText" data-lib="WAF" class="waf-widget waf-richText waf-list-secondLa
           <div id="numberValue2" data-type="richText" data-lib="WAF" class="waf-widget waf-richText waf-list-number
           <div id="sideImage2" data-type="image" data-lib="WAF" class="waf-widget waf-image waf-list-image " data-fi
       </div>
</div>
```

This widget's DOM node is the following:

```
<div id="listView1" data-type="listView" data-lib="WAF"
    data-binding="employee" data-inset="true"
    data-image="true" data-number="true" data-multi="true" data-nextButton="true"
    data-linked-tag="container1,row1,mainText1,goTo1,secondText2,numberValue2,sideImage2"
    class="waf-widget waf-listView widget-list-inset">
    <!--place other widgets here-->
</div>
```

In this widget's tag, the other widgets are placed in a Container widget, which is placed in the Matrix widget.

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The List View widget's properties are the following:

Property	Description
data-binding	Datasource to bind to this widget
data-inset	True/False = Inset with label
data-image	True/False = Include side image
data-number	True/False = Include side image
data-multi	True/False = Include a second text
data-nextButton	True/False = Include a next button
data-linked-tag	Widget IDs linked to the main widget (delimited by commas)

Only the **data-binding** and **data-linked-tag** properties are mandatory for this widget to function. The other properties are used if you open this widget in the GUI Designer.

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")

data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Login Dialog

Wakanda's **Login Dialog** widget is defined in a tag as shown below:

```
<div id="login1" data-type="login" data-lib="WAF"
    data-login-title="Login Dialog"
    data-user-label="User: " data-user-display="Signed in as "
    data-no-user-display="User has not yet logged in"
    data-password-label="Password: " data-logout-action="Logout"
    data-login-button="Login" data-login-action="Login"
    class="waf-widget waf-login">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

This widget's properties are the following:

Property	Description
data-user-label	User field label
data-password-label	Password field label
data-login-button	Title for the Login button in the dialog
data-login-title	Title for the username and password dialog
data-user-display	Text to display when user has signed in
data-no-user-display	Text to display when the user has not signed in
data-login-action	Title for the Login link
data-logout-action	Title for the Logout link

Matrix

Wakanda's **Matrix** widget is defined in a tag as shown below:

```
<div id="matrix1" data-type="matrix"  data-lib="WAF"
    data-draggable="false" data-resizable="true" data-margin="15"
    data-scrolling="vertical" data-scrollbar="true" data-fit="true"
    class="waf-widget waf-matrix">
    <!--place other widgets here-->
</div>
```

To add other widgets inside the Matrix widget, you can do so where we have inserted the "place other widgets here" comment.

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

This widget's properties are the following:

Property	Description
data-draggable	True/False = Draggable by default
data-resizable	True/False = Resizable by default
data-margin	The margin around the widget inside the Matrix
data-scrolling	The scrollbars can either be "horizontal" or "vertical"
data-scrollbar	True/False = Display scrollbars
data-fit	True/False = Automatically resize the Matrix widget in relation to the widgets inside of it

Menu Bar

Wakanda's **Menu Bar** widget is built in a standard HTML <ul> tag.
The Menu Bar widget is composed of multiple widgets: Menu Item, **Icon** (if an image is added to a menu item), Label (for the Icon), and **Text**.

```
<ul id="menuBar1" data-type="menuBar" data-tab-margin="6" data-subMenuShow="click" data-lib="WAF" data-level="0" data
    <li id="menuItem1" data-type="menuItem" data-linked-tag="richText1" data-lib="WAF" class="waf-widget waf-menuItem
        <div id="richText1" data-type="richText" data-plainText="true" data-overflow="Hidden" data-linked-tag="menuIt
        <ul id="menuBar2" data-type="menuBar" data-subMenuShow="hover" data-lib="WAF" data-level="1" data-display="ve
            <li id="menuItem4" data-type="menuItem" data-linked-tag="richText4" data-lib="WAF" class="waf-widget waf-
                    <div id="richText4" data-type="richText" data-lib="WAF" class="waf-widget waf-richText" data-auto
                    </li>
            <li id="menuItem5" data-type="menuItem" data-linked-tag="richText5" data-lib="WAF" class="waf-widget waf-
                <div id="richText5" data-type="richText" data-plainText="true" data-overflow="Hidden" data-linked-tag
            </li>
            <li id="menuItem6" data-type="menuItem" data-linked-tag="richText6" data-lib="WAF" class="waf-widget waf-
                <div id="richText6" data-type="richText" data-plainText="true" data-overflow="Hidden" data-linked-tag
            </li>
        </ul>
    </li>
    <li id="menuItem2" data-type="menuItem" data-linked-tag="richText2" data-lib="WAF" class="waf-widget waf-menuItem
        <div id="richText2" data-type="richText" data-plainText="true" data-overflow="Hidden" data-linked-tag="menuIt
    </li>
    <li id="menuItem3" data-type="menuItem" data-linked-tag="richText3" data-lib="WAF" class="waf-widget waf-menuItem
        <div id="richText3" data-type="richText" data-plainText="true" data-overflow="Hidden" data-linked-tag="menuIt
    </li>
</ul>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Menu Bar widget's properties in the <ul> tag whose data-type is "menuBar" are the following:

Property	Description
data-tab-margin	Margin for the menu items
data-subMenuShow	Show menu items either on "hover" or on "click".
data-display	Display the menu either horizontally ("horizontal") or vertically ("vertical").
data-level	Level in the Menu Bar. The main menu bar is "0" and a submenu is "1", for example.

The Menu Item widget's properties in the <li> tag and the data-type property is "menuItem" are the following:

Property	Description
data-icon	Path to the image to include (which will also be defined in the Icon widget)
data-linked-tag	If you have associated an image for your menu item, the Icon widget's ID will be defined here

If you enter only text, you will have to create a **Text** widget to be included in the <li>... tags. If you want to include an image for the Menu Item, you have to create the **Icon** widget to be included in the <li>... tags.

Navigation View

Wakanda's **Navigation View** widget is defined in a tag and is composed of multiple widgets (**Container**, **Text**, and **Button**):

```
<div id="navigationView1" data-type="navigationView" data-lib="WAF" data-views="[{'Description':'New View', 'Index':'
<div id="viewManager1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-navigationView-conten
    <div id="view1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-navigationView-view waf-
        <div id="header1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-navigationView-hea
            <div id="title1" data-type="richText" data-lib="WAF" class="waf-widget waf-richText waf-navigationView-ti
            <button id="backButton1" data-type="button" data-lib="WAF" class="waf-widget waf-button waf-navigationVie
        </div>
        <div id="body1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-widget-body waf-navi
    </div>
</div>
<div id="footer1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-widget-footer waf-navigati
</div>
```

The main widget's DOM node is the following:

```
<div id="navigationView1" data-type="navigationView" data-lib="WAF"
    data-views="[{'Description':'New View', 'Index':'1'}]"
    data-linked-tag="viewManager1,footer1,view1,header1,title1,body1"
    class="waf-widget waf-navigationView waf-navigationView-level-0 waf-level-0 ">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Navigation View widget's properties are the following:

Property	Description
data-views	An array defining each view in an object (which has two properties: Description and Index)
data-linked-tag	The IDs of the Containers for each of the views

Popover

Wakanda's **Popover** widget is built in a tag as shown below:

```
<div id="popover1" data-type="popover" data-lib="WAF"
data-linked-tag="content1,header1,footer1" data-hideonload="true"
data-header="false" data-footer="false" data-button="button1"
class="waf-widget waf-popover">
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Popover widget's properties are the following:

Property	Description
data-linked-tag	Widget IDs linked to the main widget (delimited by commas)
data-header	True/False = Show header in the widget
data-footer	True/False = Show footer in the widget
data-button	Linked button widget's ID

Progress Bar

Wakanda's **Progress Bar** widget is build in a tag:

```
<div id="progressBar1" data-type="progressBar" data-lib="WAF"
data-showstop="false" data-no-empty-display="false" data-progressinfo="progressIndicator"
class="waf-widget waf-progressBar"></div>

<label id="label1" for="progressBar1" data-type="label" data-lib="WAF"
data-valign="middle" data-margin="5"
class="waf-widget waf-label waf-label-progressBar">
Progress on {curValue} out of {maxValue}</label>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Progress Bar widget's properties are the following:

Property	Description
data-progressinfo	Progress reference
data-showstop	True/False = Show stop button
data-no-empty-display	True/False = Hide if inactive

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Query Form

Wakanda's **Query Form** widget is defined in a tag:

```
<div id="queryForm1" data-type="queryForm" data-lib="WAF"
    data-binding="company"
    data-column-name="ID,Name,Address,City" data-column-attribute="ID,name,address,city"
    data-withoper="true" data-draggable="true" data-resizable="true"
    class="waf-widget waf-queryForm">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Query Form widget's properties are the following:

Property	Description
data-binding	Datasource to bind to this widget
data-column-name	A string containing the titles (delimited by commas) for the attributes to display
data-column-attribute	A string containing the attribute names (delimited by commas) to use
data-withoper	True/False = Show or hide the operators
data-draggable	True/False = draggable by default
data-resizable	True/False = resizable by default

Radio Button Group

Wakanda's **Radio Button Group** widget is defined in a <ul> tag. For static values, each one is defined in an <li>... tag with a standard <input> tag per radio button and a <label> for each radio button's label.

The example below is for a Radio Button Group bound to a datastore class (to display the values) and bound to an attribute that accepts the selected value (defined by the **data-binding-key** property).

```
<ul id="radioGroup1" data-type="radioGroup" data-lib="WAF"
    data-binding-out="employee.department"  data-binding-key="ID"
    data-binding="department" data-autoDispatch="true"
    data-display="vertical" data-binding-options="[name] "
    class="waf-widget waf-radioGroup">
    <li>
        <input type="radio" value="name" radioGroup1="radioGroup1-0"/>
        <label for="radioGroup1-0">[name]</label>
    </li>
</ul>
```

The following Radio Button Group was created by defining static values:

```
<ul id="radioGroup2" data-type="radioGroup" data-lib="WAF"
    data-binding-out="company.city" data-binding-options="[name] " data-binding-key="province"
    data-autoDispatch="true" data-display="vertical"
    class="waf-widget waf-radioGroup">
    <li>
        <input type="radio" value="San Francisco" name="radioGroup2" id="radioGroup2-0"/>
        <label for="radioGroup2-0">San Francisco</label>
    </li>
    <li>
        <input type="radio" value="Los Angeles" name="radioGroup2" id="radioGroup2-1" checked="checked"/>
        <label for="radioGroup2-1">Los Angeles</label>
    </li>
    <li>
        <input type="radio" value="Miami" name="radioGroup2" id="radioGroup2-2"/>
        <label for="radioGroup2-2">Miami</label>
    </li>
    <li>
        <input type="radio" value="New York" name="radioGroup2" id="radioGroup2-3"/>
        <label for="radioGroup2-3">New York</label>
    </li>
</ul>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Radio Button Group widget's properties are the following:

Property	Description
data-binding	Name of the datasource (datastore class or array) that will supply the values to display
data-binding-out	Name of the attribute to receive the selected value from the widget
data-binding-options	Name of the attribute to display in the widget
data-binding-key	The attribute from the datasource defined in the data-binding property to be returned
data-autoDispatch	True = Wakanda automatically sends the query to modify the entity collection
data-format	Format to use for displaying data
tabindex	The widget's tabindex

For each of the radio buttons created in <li>... tags, there is an <input> tag.

Property	Description
id	ID made up of the widget's ID plus a unique number offset by a hyphen
type	In this case, it's "radio"
name	Either the attribute's name or the Radio Button Group ID if static values
value	Used only if the Radio Button Group has static values
radioGroupID	This property, which is the same as the Radio Button Group's ID and only appears in the case of this widget being bound to a datastore class. Its value is the widget's ID.
checked	For static values, to define a particular radio button as selected, enter "checked" as this property's value

In the <li> definition of the radio button, you must also define a <label> for it. The only parameter **for** is used to link the <input> tag to the <label>.

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Section

Wakanda's **Section** widget is defined in a tag and is composed of multiple widgets (**Container**, **Image**, **Button**, and **Text**).
The following example defines a section for each type:

```
<div id="section1" data-type="section" data-linked-tag="container1,container3,container5,container7,container9" data-
    <div id="container1" data-type="sectionNavigation" data-rowType="Label" data-showImage="true" data-rowIndex="0" d
        <div id="richText1" data-type="richText" data-plainText="true" data-overflow="Hidden" data-lib="WAF" data-inl
        <div id="image1" data-type="image" data-src="/images/001_57.png" data-lib="WAF" data-fit="0" class="waf-widge
    </div>
    <div id="container3" data-type="sectionNavigation" data-rowType="Switch" data-showImage="false" data-rowIndex="1"
        <div type="checkbox" id="switchbox1" data-type="switchbox" data-binding="employee.married" data-on="ON" data-
        <div id="richText3" data-type="richText" data-plainText="true" data-overflow="Hidden" data-lib="WAF" data-inl
    </div>
    <div id="container5" data-type="container" data-rowType="Input" data-rowIndex="2" data-linked-tag="textField1,ric
        <input type="text" id="textField1" data-type="textField" data-binding="employee.birthday" data-multiline="fal
        <div id="richText5" data-type="richText" data-plainText="true" data-overflow="Hidden" data-lib="WAF" data-lab
    </div>
    <div id="container7" data-type="sectionNavigation" data-rowType="NavBar" data-showImage="false" data-rowIndex="3"
        <button id="button1" data-type="button" data-text=" " data-lib="WAF" data-action="simple" class="waf-widget w
        <div id="richText7" data-type="richText" data-overflow="Hidden" data-lib="WAF" data-inline="true" data-autoWi
    </div>
    <div id="container9" data-type="container" data-rowType="Blank" data-rowIndex="4" data-lib="WAF" class="waf-widge
    </div>
</div>
```

Below is the DOM node for a Section widget with one Label section:

```
<div id="section1" data-type="section" data-lib="WAF"
     data-linked-tag="container1" data-items="[{'type':'Label'}]"
    class="waf-widget waf-section">
     <div id="container1" data-type="container" data-lib="WAF"
    data-rowIndex="0" data-rowType="Label" data-linked-tag="richText1"
    class="waf-widget waf-container waf-section-label waf-element-first waf-element-last">
         <div id="richText1" data-type="richText" data-lib="WAF"
        data-autoWidth="true" data-overflow="Hidden" data-plainText="true" data-inline="true"
        class="waf-widget waf-richText">Label</div>
     </div>
</div>
```

Each section in this widget is defined in a **Container** widget. The label used in sections of type "Label", "Input", and "Switch" are **Text** widgets; however, you can use the Label widget if you prefer.

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Section widget's properties are the following:

Property	Description
data-linked-tag	Widget IDs linked to the main widget (delimited by commas)
data-items	An array of objects defining the type of sections to create. The different types are: "Label", "Blank", "Switch", "Input", or "Navigation"

The **data-linked-tag** property is mandatory for this widget to function.
For each **Container** widget that defines a section in the Section widget, there are a few additional properties:

Property	Description
data-rowType	This row type can be one of the following: "Label", "Blank", "Switch", "Input", or "NavBar"
data-rowIndex	The number of the section in order of its placement in the Section widget, starting at 0
data-showImage	True/False = Display a side image for sections of type "Label", "Switch", and "Navigation" (If you set this property to True, you must add an Image widget inside the section's Container and link to it by using the **data-linked-tag** property)

***Note**: For the "Label", "Switch", and "Navigation" section types, you must set the **data-type** property to "sectionNavigation" in the tag.*

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Select

Wakanda's **Select** widget is defined in a tag. If there are static values, each one is defined in an <option>...</option> tag in the <select> tag contained in the .

The example below is for a Select widget bound to a datastore class (to display the values) and bound to an attribute that accepts the selected value (defined by the **data-binding-key** property).

```
<div id="select1" data-type="select" data-lib="WAF"
    data-binding="company" data-binding-out="employee.employer"
    data-binding-options="[name] " data-binding-key="company"
    data-autoDispatch="true" data-limit="20" tabindex="1"
    class="waf-widget waf-select">
    <select>
        <option value="">[name] </option>
    </select>
</div>
```

The following Select widget was created by defining static values:

```
<div id="select2" data-type="select" data-lib="WAF"
    data-limit="20" data-binding-out="employee.city"
    class="waf-widget  waf-select">
    <select>
        <option value="New  York">New York</option>
        <option  value="Boston">Boston</option>
        <option value="San  Francisco">San Francisco</option>
    </select>
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Select widget's properties are the following:

Property	Description
data-binding	Name of the datasource (datastore class or array) that will supply the values to display
data-binding-out	Name of the attribute to receive the selected value from the widget
data-binding-options	Name of the attribute to display in the widget
data-binding-key	The attribute from the datasource defined in the data-binding property to be returned
data-autoDispatch	True/False = Wakanda automatically sends the query to modify the entity collection
data-limit	Number of entities to display in the Select widget.
data-format	Format to use for displaying data
data-errorDiv	Widget ID that will contain the error messages sent by the server

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-	Text alignment for the widget ("middle", "top", or "bottom")

valign	
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Slider

Wakanda's **Slider** widget is defined in a tag as shown below:

```
<div id="slider1" data-type="slider" data-lib="WAF"
    data-binding="employee.salary"
    data-minValue="0" data-maxValue="100"
    data-step="1" data-orientation="horizontal" data-range="min"
    class="waf-widget waf-slider">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Slider widget's properties are the following:

Property	Description
data-binding	Datasource to bind to this widget
data-step	Slider's step
data-range	Defines where the range will be shown on either the minimum side, maximum side, or not at all ("min", "max", or "none")
data-orientation	Slider's orientation ("horizontal" or "vertical")
data-minValue	Slider's minimum value
data-maxValue	Slider's maximum value
data-errorDiv	Error ID linked to this widget

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Split View

Wakanda's **Split View** widget is defined in a tag and is composed of multiple widgets (**Navigation View**, **Container**, **Text**, and **Button**):

```
<div id="splitView1" data-type="splitView" data-lib="WAF"  class="waf-widget waf-splitView waf-splitView-level-0 waf-
    <div id="container1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level-0 w
        <div id="container2" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level
            <div id="navigationView1" data-type="navigationView" data-lib="WAF"  data-views="[{'Description':'New Vie
    <div id="viewManager1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-navigationView-co
        <div id="view1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-navigationView-view
            <div id="header1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-navigationView
                <div id="title1" data-type="richText" data-lib="WAF" class="waf-widget waf-richText waf-navigationVie
                <button id="backButton1" data-type="button" data-lib="WAF" class="waf-widget waf-button waf-navigatio
            </div>
            <div id="body1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-widget-body waf-
        </div>
    </div>
    <div id="footer1" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-widget-footer waf-navi
</div>
        </div>
        <div id="container3" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-container-level
            <div id="navigationView2" data-type="navigationView" data-lib="WAF"  data-views="[{'Description':'New Vie
    <div id="viewManager2" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-navigationView-co
        <div id="view2" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-navigationView-view
            <div id="header2" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-navigationView
                <div id="title2" data-type="richText" data-lib="WAF" class="waf-widget waf-richText waf-navigationVie
                <button id="backButton2" data-type="button" data-lib="WAF" class="waf-widget waf-button waf-navigatio
            </div>
            <div id="body2" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-widget-body waf-
        </div>
    </div>
    <div id="footer2" data-type="container" data-lib="WAF" class="waf-widget waf-container waf-widget-footer waf-navi
</div>
            <button id="button1" data-type="button" data-lib="WAF" class="waf-widget waf-button waf-splitView-menuBut
        </div>
    </div>
</div>
```

The widget's main DOM is the following:

```
<div id="splitView1" data-type="splitView" data-lib="WAF"
    data-linked-tag="container1,container2,container3"
    class="waf-widget waf-splitView waf-splitView-level-0  waf-level-0">
    <div id="container1" data-type="container" data-lib="WAF"
      data-hideSplitter="true"  data-popup-display-button="button1"
      class="waf-widget waf-container waf-container-level-0 waf-level-0">
            <div id="container2" data-type="container" data-lib="WAF"
                class="waf-widget waf-container waf-container-level-0 waf-level-0 waf-split-container">
                <!--left container in which to put other widgets-->
            </div>
            <div id="container3" data-type="container" data-lib="WAF"
                class="waf-widget waf-container waf-container-level-0 waf-level-0 waf-split-container ">
                <!--main container in which to put other widgets-->
            </div>
    </div>
</div>
```

The Split View widget's properties are the following:

Property	Description
data-linked-tag	The IDs of the Containers for each of the views

For each of the Containers (which are views in this widget), the "waf-split-container" class must also be included in the Container's **class** property.

Switch

Wakanda's **Switch** widget is build in a tag:

```
<div id="switchbox1" data-type="switchbox"  data-lib="WAF"
data-binding="employee.married" data-on="ON" data-off="OFF"
data-checked="true" data-errorDiv="errorDiv1"
class="waf-widget waf-switchbox"></div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Switch widget's properties are the following:

Property	Description
data-binding	Datasource to bind to this widget
date-on	The text to display for the "on" value
date-off	The text to display for the "off" value
data-checked	True/False = Set the value to "on" by default
data-errorDiv	Error ID linked to this widget

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Tab View

Wakanda's **Tab View** widget is defined in a tag and is composed of multiple widgets (**Text**, **Menu Bar**, and **Container**):

```
<div id="tabView1" data-type="tabView" data-padding="5" data-menu-position="top left" data-linked-tag="menuBar1,conta
 <ul id="menuBar1" data-type="menuBar" data-tab-margin="5" data-subMenuShow="hover" data-linked-tag="tabView1" data-l
      <li id="menuItem1" data-type="menuItem" data-linked-tag="richText1,container1" data-lib="WAF" class="waf-widget
           <div id="richText1" data-type="richText" data-plainText="true" data-overflow="Hidden" data-linked-tag="men
       </li>
      <li id="menuItem2" data-type="menuItem" data-linked-tag="richText2,container2" data-lib="WAF" class="waf-widget
           <div id="richText2" data-type="richText" data-plainText="true" data-overflow="Hidden" data-linked-tag="men
       </li>
      <li id="menuItem3" data-type="menuItem" data-linked-tag="richText3,container3" data-lib="WAF"  class="waf-widge
           <div id="richText3" data-type="richText" data-plainText="true" data-overflow="Hidden" data-linked-tag="men
       </li>
  </ul>
 <div id="container1" data-type="container" data-linked-tag="menuItem1" data-lib="WAF" class="waf-widget waf-containe
 <div id="container2" data-type="container" data-linked-tag="menuItem2" data-lib="WAF" class="waf-widget waf-containe
 <div id="container3" data-type="container" data-linked-tag="menuItem3" data-lib="WAF" class="waf-widget waf-containe
</div>
```

This widget's main DOM node is the following:

```
<div id="tabView1" data-type="tabView" data-lib="WAF"
    data-padding="5" data-menu-position="top left"
    data-linked-tag="menuBar1,container1,container2,container3"
    class="waf-widget waf-tabView waf-tabView-level-0 waf-level-0">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Tab View widget's properties are the following:

Property	Description
data-padding	Padding (in pixels) around the widget
data-menu-position	Widget IDs linked to the main widget (delimited by commas)
data-linked-tag	The IDs of the Containers for each tab as well as the IDs for the Tab View's menu bar
data-closable-tabs	True/False = Closable tabs
data-draggable	True/False = draggable by default
data-resizable	True/False = resizable by default

Text

Wakanda's **Text** widget is built in a standard HTML tag.
The following Text widget uses a datasource that will be displayed.

```
<div id="richText1" data-type="richText" data-lib="WAF"
    data-binding="employee.salary" data-format="$###,##0.00"
    data-plainText="true" data-overflow="Hidden" data-autoWidth="true"
    class="waf-widget waf-richText">
</div>
```

The following Text widget has static HTML text defined:

```
<div id="richText2" data-type="richText" data-lib="WAF"
    data-plainText="false" data-overflow="Horizontal" data-autoWidth="false"
    class="waf-widget waf-richText">
    This is my &lt; b &gt; formatted &lt; /b &gt; text.
</div>
```

This Text widget is used to display a URL:

```
<div id="richText3" data-type="richText" data-lib="WAF"
    data-plainText="true" data-link="http:www.wakanda.org" data-target="_blank"
    data-autoWidth="true"
    class="waf-widget waf-richText">
    wakanda.org
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Text widget's properties are the following:

Property	Description
data-binding	Datasource to bind to this widget
data-format	Format for the data to be displayed, e.g., "$###,##0.00" or "M dd, yy"
data-plainText	True/False = if set to False, translate HTML tags.
data-link	URL to open for this widget.
data-target	If a URL is defined, the options are "_blank" (to open in a new window) or "_self" (to open in the same window)
data-autoWidth	True/False = automatically resize the widget based on the text displayed.
data-overflow	Display the scrollbars ("Horizontal", "Vertical", "Hidden" or "Both").

If a static text is included, it must be defined between the ... tags.
For the HTML tags, the "<" and ">" characters must be translated to "& lt ;" and "& gt ;".

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.
Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"

data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Text Input

Wakanda's **Text Input** widget is defined in a standard HTML <input> tag.
The example below is a Text Input widget bound to an attribute in a datasource with a placeholder text:

```
<input data-type="textField" id="textField1" data-lib="WAF"
data-binding="employee.firstName"  placeholder="Enter your first name" tabindex="1"
class="waf-widget waf-textField"/>
```

This Text Input example is bound to an attribute of type Date in a datasource:

```
<input data-type="textField" id="textField2" data-lib="WAF"
data-binding="employee.birthday" data-format="M d, yy"
data-datapicker-on="true" data-datapicker-icon-only="true"
class="waf-widget waf-textField" />
```

This example shows how to make the Text Input widget into a Password field:

```
<input data-type="textField" id="textField3" data-lib="WAF"
data-binding="passwordVar" data-password="true"
class="waf-widget waf-textField" />
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Text Input widget's properties are the following:

Property	Description
data-binding	Datasource to bind to this widget
data-format	Format for the data to be displayed, e.g., "$###,##0.00" or "M dd, yy"
placeholder	Placeholder text to display in the widget
data-password	True/False = Set the widget type to Password, which means that the text typed is replaced by bullet characters
data-readOnly	True/False = Set the widget to Read Only mode
data-multiline	True/False = Allow the widget to have scrollbars if the text surpasses the size of the widget
data-picker-on	True/False = Display the date picker when the user tabs into it
data-picker-icon-only	True/False = Display the date picker icon
data-errorDiv	Widget ID that will contain the error messages sent by the server
tabindex	Widget's tabindex

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.
Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-	Margin (in pixels) from the widget defined by the **for** property

margin	
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Video

Wakanda's **Video** widget is defined in a tag.
Below is the definition for a Video widget with multiple video sources:

```
<div id="video1" data-type="video" data-lib="WAF"
    data-video-urls="[{'video-url':'/videos/RUN001_AddingRelatedData.mp4'},
    {'video-url':'/videos/RUN001_AddingRelatedData.ogv'}]"
    data-video-controls="true" data-video-autohide="true" data-video-loop="true"
    data-video-autoplay="true" data-video-start="10"
    class="waf-widget waf-video">
</div>
```

Below is a Video widget whose datasource is a datastore class attribute:

```
<div id="video2" data-type="video" data-lib="WAF"
    data-binding="videos.video"
    class="waf-widget waf-video">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Video widget's properties are the following:

Property	Description
data-binding	Datasource to bind to this widget
data-video-start	Number of seconds after which to start the video
data-video-autoplay	True/False = Autoplay the video
data-video-loop	True/False = Loop the video
data-video-controls	True/False = Display controls
data-video-autohide	True/False = Autohide controls
data-video-urls	Array of objects in which each object defines the video source in the video-url property (only HTML5 video types are accepted)

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.
Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

WYSIWYG Editor

Wakanda's **WYSIWYG Editor** widget is built in a standard HTML tag.

```
<div id="wysiwyg1" data-type="wysiwyg" data-lib="WAF"
    data-binding="employee.resume"
    data-toolbar-location="top" data-toolbar-align="left" data-save="true"
    data-elements="[{'key':'newdocument','value':'New Document'},{'key':'save','value':'Save'},{'key':'|','value':'Se
    {'key':'italic','value':'Italic'},{'key':'underline','value':'Underline'},{'key':'|','value':'Separator'},{'key':
    {'key':'justifycenter','value':'Justify To Center'},{'key':'justifyright','value':'Justify To Right'},{'key':'jus
    {'key':'|','value':'Separator'},{'key':'cut','value':'Cut'},{'key':'copy','value':'Copy'},{'key':'|','value':'Sep
    {'key':'redo','value':'Redo'},{'key':'|','value':'Separator'},{'key':'forecolor','value':'Text Color'}]"
    data-draggable="true" data-resizable="true"
    class="waf-widget waf-wysiwyg">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The WYSIWYG Editor widget's properties are the following:

Property	Description
data-binding	Datasource to bind to this widget
data-toolbar-location	Display the toolbar either on the top or bottom by specifying either "top" or "bottom"
data-toolbar-align	Align the toolbar either to the left or right ("left" or "right")
data-save	True/False = automatically save the data entered to the datasource
data-draggable	True/False = draggable by default
data-resizable	True/False = resizable by default
data-elements	An array that defines an object containing each of the elements to display. Each element is defined by the "key" and the "value" to display.

Below are the keys that you can use in the **data-elements** property:

Key	
bold	image
italic	cleanup
underline	help
strikethrough	code
justifyleft	hr
justifycenter	removeformat
justifyright	formatselect
justifyfull	fontselect
bullist	fontsizeselect
numlist	styleselect
outdent	sub
indent	sup
cut	forecolor
copy	backcolor
paste	forecolorpicker
undo	backcolorpicker
redo	charmap
link	visualaid
unlink	anchor
newdocument	blockquote
separator	

Before including the Loader.js file, you must also include the following script when including the WYSIWYG Editor on your Page:

```
<script type="text/javascript" src="/waLib/WAF/lib/tiny_mce/tiny_mce.js"></script>
```

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Yahoo! Weather

Wakanda's **Yahoo! Weather** widget is built in a standard HTML tag.

```
<div id="yahooWeather1" data-type="yahooWeather" data-lib="WAF"
    data-binding="company.city" data-unit="c"
    class="waf-widget waf-yahooWeather">
</div>
```

Widget's DOM node

Below is a description of this widget's DOM node. The following properties are common for all widgets:

Property	Description
id	Widget's unique ID
data-type	Widget's type (see **Data-type property** for more information)
data-lib	WAF library
class	CSS classes that define this widget (see **Default CSS styles** for more information), including widget roles like "waf-role-cancelButton" (see **Widget Roles** for more information).
data-hideonload	True/False = Hide widget on load of page. By default, this property is set to False even if you do not include it.

The Yahoo! Weather widget's properties are the following:

Property	Description
data-binding	Datasource to bind to this widget
data-unit	Pass "c" for Celsius and "f" for Fahrenheit

Label widget

This widget has a Label widget attached to it if the **data-label** property is defined. It is created automatically in Wakanda Studio; however, if you are coding yourself, you'll have to add it manually.

Wakanda uses the standard <label> tag for the Label widget with the text for the label defined before the </label> tag:

```
<label id="label1" data-type="label" data-lib="WAF"
    for="widgetID" data-valign="middle"
    data-margin="5"
    class="waf-widget waf-label">
    Label Text
</label>
```

The Label widget's properties are:

Property	Description
id	Widget's unique ID
data-type	Widget's type, which is "label"
data-lib	WAF library
for	The ID of the widget that is attached to this Label widget
data-valign	Text alignment for the widget ("middle", "top", or "bottom")
data-margin	Margin (in pixels) from the widget defined by the **for** property
class	CSS classes that define this widget. After "waf-widget waf-label default", you must also include a CSS class for the Label widget based on the widget it is linked to. To create the CSS class for the widget's label, use "waf-label-" and add the widget's **data-type** as the suffix, e.g., "waf-label-icon".

For more information on the widget's **data-type** property, refer to the **Data-type property** for more information.

Creating a widget instance in HTML

You can use any of Wakanda's widgets on your HTML page. The steps are the same for both desktop and mobile devices with the exception of an additional <meta> tag that is required for mobile devices.

Desktop development

To create a widget in HTML to be used on the desktop:

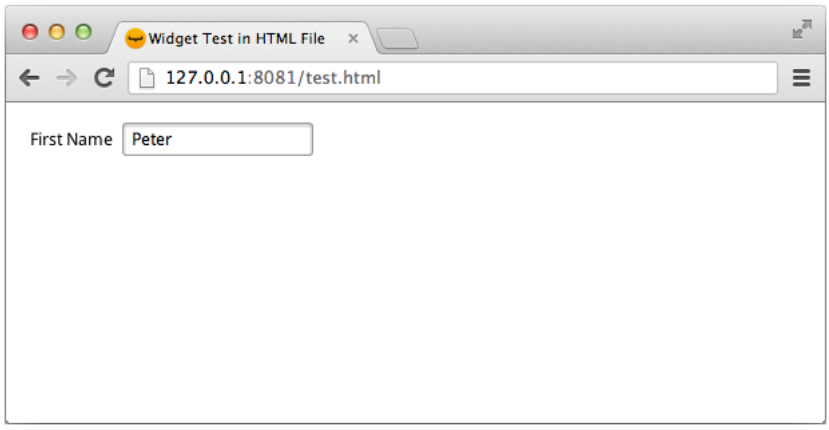
1. Declare any datasources that you want to use in a <meta> tag in the header of your HTML file (for more information, refer to **Datasources**),
2. Define the CSS styles to position your widgets (either in your HTML file or in an external CSS file),
3. Add one or more widgets to the <body> of your HTML file (based on each widget's DOM node as defined in this manual), and
4. Include Wakanda's Loader.js file in the <body> of your HTML file.

1. Datasource
2. Positioning CSS
3. Widgets
4. Loader.js file

```
<!DOCTYPE html>
<html>
<head>
    <title>Widget Test in HTML File</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <meta name="WAF.config.datasources" data-type="dataSource" data-source-type="dataClass" data-source="Employee"
     data-lib="WAF" data-id="employee" data-autoLoad="true" />

    <style type="text/css">
        #textField1 { width: 174px; height: 23px; top: 20px; left: 79px; position: absolute;}
        #label1 { width: 100px; height: 14px; top: 22px; left: -30px; position: absolute; text-align: right; min-width: 100px; }
    </style>
</head>
<body>

    <label id="label1" for="textField1" data-valign="middle" data-type="label" data-lib="WAF"
     class="waf-widget waf-label waf-label-textField">First Name</label>
    <input data-lib="WAF" data-type="textField" id="textField1"  data-binding="employee.firstName"
     placeholder="Enter your first name" class="waf-widget waf-textField" />

    <script type="text/javascript" src="/waLib/WAF/Loader.js"></script>
</body>
</html>
```

Desktop example

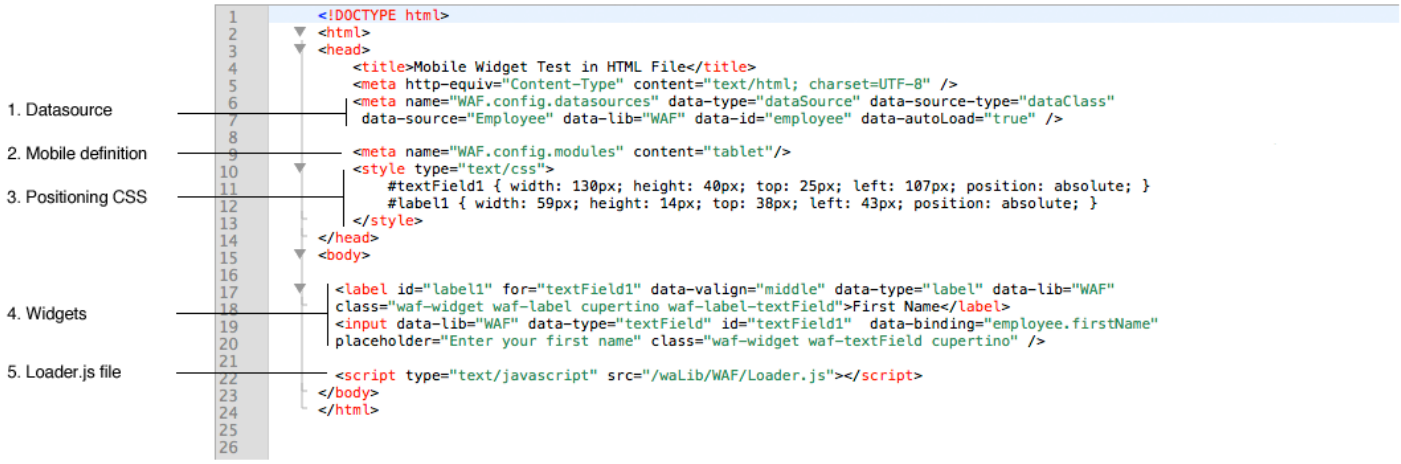
The example HTML file defined above contains a **Text Input** widget bound to the Employee datastore class datasource and a Label widget. This HTML page is published as shown below in a Web browser:



Mobile development

To create a widget in HTML to be used on any mobile devices:

1. Declare any datasources that you want to use in a <meta> tag in the header of your HTML file (for more information, refer to **Datasources**),
2. Include a <meta> tag that Wakanda requires (),
3. Define the styles in CSS for your widgets (either in your HTML file or in an external CSS file),
4. Add one or more widgets to the <body> of your HTML file (based on each widget's DOM node as defined in this manual), and
5. Include Wakanda's Loader.js file in the <body> of your HTML file.




```
<!DOCTYPE html>
<html>
<head>
    <title>Mobile Widget Test in HTML File</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <meta name="WAF.config.datasources" data-type="dataSource" data-source-type="dataClass"
     data-source="Employee" data-lib="WAF" data-id="employee" data-autoLoad="true" />

    <meta name="WAF.config.modules" content="tablet"/>
    <style type="text/css">
        #textField1 { width: 130px; height: 40px; top: 25px; left: 107px; position: absolute; }
        #label1 { width: 59px; height: 14px; top: 38px; left: 43px; position: absolute; }
    </style>
</head>
<body>

    <label id="label1" for="textField1" data-valign="middle" data-type="label" data-lib="WAF"
    class="waf-widget waf-label cupertino waf-label-textField">First Name</label>
    <input data-lib="WAF" data-type="textField" id="textField1"  data-binding="employee.firstName"
    placeholder="Enter your first name" class="waf-widget waf-textField cupertino" />

    <script type="text/javascript" src="/waLib/WAF/Loader.js"></script>
</body>
</html>
```

Using a Widget Role

If you want to use a Widget Role defined in your Project, you can do so by specifying it in the **class** property:

```
<button id="button1" data-type="button" data-lib="WAF" data-action="simple"
class="waf-widget waf-button waf-role-squareCorners"></button>
```

For more information about widget roles, refer to **Widget Roles**.

HTML source of a Page

There are other <meta> tags that are necessary when creating a Page (its view's HTML file). For more information, refer to **HTML File** in the **Architecture of Wakanda Applications** manual.

Caveat

In this manual, we describe the properties for each widget. There are additional properties that the GUI Designer adds for its own purposes, like positioning, that we have not described here.

If you open your HTML page in Wakanda, please be aware that some properties might be added that are not necessary when displaying your page; however, are necessary for the GUI Designer to render the widget correctly on the Page.

Creating a widget instance in JavaScript

You can create widgets dynamically in JavaScript by following these steps:

- Create a datasource (if needed by the widget)
- Create the HTML element on the Page for the widget to define how it is displayed (position and CSS classes)
- Create the instance of the widget (defining the behavior of the widget)

Creating a datasource

You can create a datasource in JavaScript if your widget needs one; however, if a datasource already exists for your Page, you won't need to recreate it for your newly created widget.

To learn how to create a datasource dynamically, refer to the "**Creating a datasource dynamically**" chapter.

Creating the HTML element

To create your widget dynamically, you must first create the HTML element on the Page.

In the HTML element, you define all the attributes for the widget. In the createElement() function, you pass the actual HTML tag for the widget. Most of Wakanda's widgets are defined in tags, but some of them, like the Text Input is defined in a <input> HTML tag. You must then define the attributes for the widget, like ID (which links the HTML element and the widget's instance), styles, and classes (including the ones used by default in Wakanda).

In our example below, we create the HTML element for a Button widget:

```
var buttonElement = document.createElement('button'); //HTML tag
buttonElement.setAttribute('id','myNewButton'); // ID that links to the widget constructor ID
buttonElement.setAttribute('style','width:120px;height:30px;left:224px;top:160px;position:absolute');
buttonElement.setAttribute('class','waf-widget waf-button default inherited');
document.body.appendChild(buttonElement);
```

In jQuery, you can write the following:

```
var buttonElement = $("<button>"); //HTML tag
buttonElement.attr('id','myNewButton'); // ID that links to the widget constructor ID
buttonElement.attr('style','width:120px;height:30px;left:224px;top:160px;position:absolute');
buttonElement.attr('class','waf-widget waf-button default inherited');
$('body').append(buttonElement);
```

Creating an instance of the widget

After you create the HTML element, you can then create the instance of the widget. For each widget, you must know its properties, which are described in this manual. You can look up each one individually. Some of the properties are mandatory for all widgets, like the ID, while others are specific to each widget.

The var that you create can be anything you'd like. For the new instance of the class, you must write WAF.widget.*widgetType*. For *widgetType*, which is the widget's class, please refer to the **Data-type property** section.

In the definition of the widget, you must define the widget's properties. The most important property is the **id** (which is the same id as the one you indicated in the HTML element). For each widget, there are specific properties that must be defined.

Continuing with our Button widget, below is the code to define the instance of the widget:

```
var myNewButton = new WAF.widget.Button({
    'id': 'myNewButton', // ID
    'data-text': 'My New Button' // title for the button
 });
```

***Note**: You must also add a reference to this widget in your Page's package.json file as discussed in the **Package.json File** chapter.*

Example

In the example below, we create a Next button bound to the company datasource so that we can apply the "next" automatic action. Once the button appears, you can then click it to go to the next entity in the company datasource.

```
button1.click = function button1_click (event) {

//    create the datasource
       WAF.dataSource.create({
         'id': 'company', // id
         'binding': 'Company' // name of the datastore class
       });

    // create the HTML element on our page
        var buttonElement = document.createElement('button'); //HTML tag
        buttonElement.setAttribute('id','nextButton'); // ID that links to the widget constructor ID
        buttonElement.setAttribute('style','width:92px;height:22px;left:128px;top:511px;position:absolute');
        buttonElement.setAttribute('class','waf-widget waf-button default inherited');
        document.body.appendChild(buttonElement);

    // create an instance of our Button widget
        var nextButton = new WAF.widget.Button({
            'id': 'nextButton', // ID
            'data-text': 'Next', // title for the button
            'data-action': 'next', // action
            'data-binding': 'company' // datasource ID
            });

        // resolve datasource
        sources.company.resolveSource();

};
```

Data-type property

The **data-type** property defines the type of widget to create in HTML while its class is used when you create it in JavaScript. Below are Wakanda's widgets, their types, and their classes:

Widget	Type	Class
Accordion	accordion	Accordion
Auto Form	autoForm	AutoForm
Button	button	Button
Calendar	calendar	Calendar
Canvas	canvas	Canvas
Chart	chart	Chart
Checkbox	checkbox	Checkbox
Combo Box	combobox	Combobox
Component	component	Component
Container	container	Container
Dialog	dialog	Dialog
Display Error	errorDiv	ErrorDiv
File Upload	fileUpload	FileUpload
Frame	frame	Frame
Google Maps	googleMap	GoogleMap
Google Maps v3	googleMaps	GoogleMaps
Grid	dataGrid	DataGrid
Icon	icon	Icon
Image	image	Image
Image Button	buttonImage	ButtonImage
Label	label	Label
List View	list	List
Login Dialog	login	Login
Matrix	matrix	Matrix
Menu Bar	menuBar	MenuBar
Menu Item	menuItem	MenuItem
Navigation View	navigationView	NavigationView
Popover	popover	Popover
Progress Bar	progressBar	ProgressBar
Query Form	queryForm	QueryForm
Radio Button Group	radioGroup	RadioGroup
Section	section	Section
Select	select	Select
Slider	slider	Slider
Split View	splitView	SplitView
Switch	switchbox	SwitchBox
Tab View	tabView	TabView
Text	richText	RichText
Text Input	textField	TextField
Video	video	Video
WYSIWYG Editor	wysiwyg	TinyMCE
Yahoo! Weather	yahooWeather	YahooWeather

Default CSS styles

For all of Wakanda's widgets, you must define its CSS classes in the **class** property of its DOM node.

For both desktop and applications, you must include the following two CSS classes: "waf-widget" and the specific class for the widget (see table below).

```
class="waf-widget waf-accordion"
```

Widget's CSS classes

Each widget has its own default CSS class:

Widget	Type
Accordion	waf-accordion
Auto Form	waf-autoForm
Button	waf-button
Calendar	waf-calendar
Canvas	waf-canvas
Chart	waf-chart
Checkbox	waf-checkbox
Combo Box	waf-combobox
Component	waf-component
Container	waf-container
Dialog	waf-dialog
Display Error	waf-errorDiv
File Upload	waf-fileUpload
Frame	waf-frame
Google Maps	waf-googleMaps
Google Maps v1	waf-googleMap
Google Maps v3	waf-googleMaps
Grid	waf-dataGrid
Icon	waf-icon
Image	waf-image
Image Button	waf-buttonImage
Label	waf-label
List View	waf-list
Login Dialog	waf-login
Matrix	waf-matrix
Menu Bar	waf-menuBar
Navigation View	waf-navigationView
Popover	waf-popover
Progress Bar	waf-progressBar
Query Form	waf-queryForm
Radio Button Group	waf-radioGroup
Section	waf-section
Select	waf-select
Slider	waf-slider
Split View	waf-splitView
Switch	waf-switchbox
Tab View	waf-tabView
Text	waf-richText
Text Input	waf-textField
Video	waf-video
WYSIWYG Editor	waf-wysiwyg
Yahoo! Weather	waf-yahooWeather

CSS classes for complex widgets

For many of the widgets that are made up of multiple widgets, like the Accordion, Chart, Dialog, Menu Bar, Navigation View, Split View, and Tab View, there are additional CSS classes to include. Refer to each widget's example for more information.

waf-level CSS classes

For certain widgets, like the Accordion, Chart, Dialog, Menu Bar, Navigation View, Split View, and Tab View, there are also classes that help define the level, like "waf-level-0".