

# Form Tags

Please make sure you have read the [Tag Syntax](#) document and understand how tag attribute syntax works.

Within the form tags, there are two classes of tags: the form tag itself, and all other tags, which make up the individual form elements. The behavior of the form tag is different than the elements enclosed within it.

## Form Tag Themes

As explained in [Themes and Templates](#), the HTML Tags (which includes Form Tags) are all driven by templates. Templates are grouped together to create themes. The framework bundles three themes in the distribution.

<b>simple</b>	Sometimes <i>too</i> simple	
<b>xhtml</b>	Extends simple	(default)
<b>ajax</b>	Extends xhtml	

The predefined themes can be used "as is" or customized.

xhtml layout

The xhtml theme renders out a two-column table. If a different layout is needed, do *not* write your own HTML. Create a new theme or utilize the simple theme.

## Simple theme caveats

The downside of using the simple theme is that it doesn't support as many of the attributes that the other themes do. For example, the `label` attribute does nothing in the simple theme, and the automatic display of error messages is not supported.

## Common Attributes

All the form tags extend the `UIBean` class. This base class provides a set of common attributes, that can be grouped in to three categories: `templated-related`, `javascript-related`, and `general` attributes. The individual attributes are documented on each tag's reference page.

In addition to the common attributes, a special attribute exists for all form element tags: `form` (`${parameters.form}`). The `form` property represents the attributes used to render the form tag, such as the form's id. In a template, the form's ID can be found by calling `${parameters.form.id}`.

## Template-Related Attributes

```
{snippet:id=templateRelatedAttributes|javadoc=true|url=org.apache.struts2.components.UIBean}
```

## Javascript-Related Attributes

```
{snippet:id=javascriptRelatedAttributes|javadoc=true|url=org.apache.struts2.components.UIBean}
```

## Tooltip Related Attributes

```
{snippet:id=tooltipAttributes|javadoc=true|url=org.apache.struts2.components.UIBean}
```

## General Attributes

```
{snippet:id=generalAttributes|javadoc=true|url=org.apache.struts2.components.UIBean}
```

When some attributes don't apply

Some tag attributes may not be utilized by all, or any, of the templates. For example, the form tag supports the `tabindex` attribute, but none of the themes render the `tabindex`.

## Value/Name Relationship

In many of the tags (except for the form tag) there is a unique relationship between the `name` and `value` attributes. The `name` attribute provides the name for the tag, which in turn is used as the control attribute when the form is submitted. The value submitted is bound to the `name`. In most cases, the `name` maps to a simple `JavaBean` property, such as `postalCode`. On a submit, the value would be set to the property by calling the `setPostalCode` mutator.

Likewise, a form control could be populated by calling a `JavaBean` accessor, like `getPostalCode`. In the expression language, we can refer to the `JavaBean` property by name. An expression like `"${postalCode}"` would in turn call `getPostalCode`.

```
xmlUsing Expressions to populate a form for editing<@s.form action="updateAddress"> <@s.textfield label="Postal Code" name="postalCode" value="{postalCode}"/> ... </@s.form>
```

However, since the tags imply a relationship between the `name` and `value`, the `value` attribute is optional. If a `value` is not specified, by default, the `JavaBean` accessor is used instead.

```
xmlPopulating a form for editing, the easy way<@s.form action="updateAddress"> <@s.textfield label="Postal Code" name="postalCode"/> ... </@s.form>
```

While most attributes are exposed to the underlying templates as the same key as the attribute (`$(parameters.label)`), the `value` attribute is not. Instead, it can be accessed via the `nameValue` key (`$(parameters.nameValue)`). The `nameValue` key indicates that the value may have been generated from the `name` attribute rather than explicitly defined in the `value` attribute.

## ID Name Assignment

All form tags automatically assign an ID to the control, but the ID can be overridden if needed.

<b>Forms</b>	The default ID is the action name. For example, "updateAddress".
<b>Controls</b>	The default ID is the form's name concatenated with the tag name. For example, "updateAddress_postalCode".

## Form labelposition propagation

When `labelposition` attribute was defined for `<s:form>` tag it will be propagated to all form elements, but if form element defines its own `labelposition` it will take precedence over `<s:form>`'s attribute. Since 2.3.17.

## Required Attribute

The `required` attribute on many UI tags defaults to true only if you have client-side validation enabled, and a validator is associated with that particular field.

## Tooltip

```
{snippet:id=tooltipdescription|javadoc=true|url=org.apache.struts2.components.UIBean}{snippet:id=tooltipexample|lang=xml|javadoc=true|url=org.apache.struts2.components.UIBean}
```

Next: [UI Tag Reference](#)