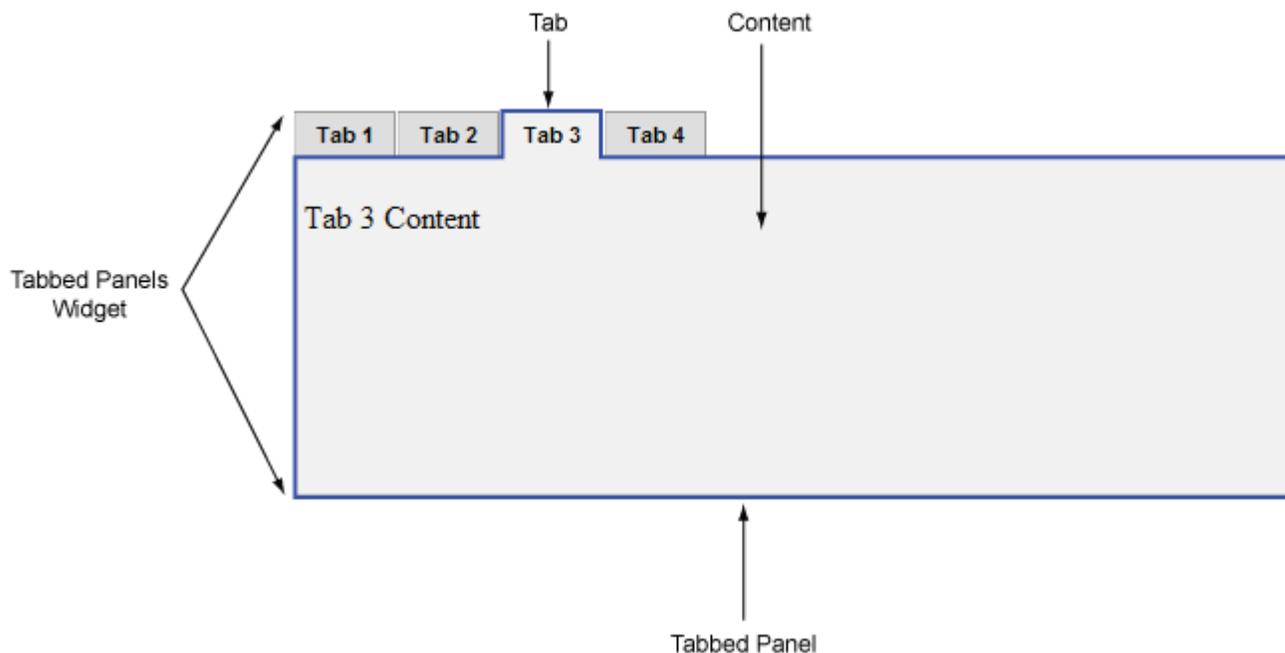


## Using Spry Widgets: Tabbed Panels 2 Overview

### Working with the Tabbed Panels 2 widget

#### Tabbed Panels 2 widget overview and structure

A Tabbed Panels 2 widget is a set of panels that can store content in a compact space. Site visitors hide or reveal the content stored in the Tabbed Panels 2 by clicking the tab of the panel they want to access. The panels of the widget open accordingly as the visitor clicks different tabs. In a Tabbed Panels 2 widget, only one content panel is open at a given time. The following example shows a Tabbed Panels 2 widget, with the third panel open:



The Tabbed Panels 2 widget requires script tags in the head of the document and after the widget's HTML code. The script tags in the head of the document defines all of the JavaScript functions related to the Tabbed Panel 2 widget. The script tag after the Tabbed Panel 2 widget code creates a JavaScript object that makes the Tabbed Panel 2 interactive. Following is a sample of HTML code for a Tabbed Panels 2 widget:

```
<head>
...
<script type="text/javascript" src="SpryAssets/SpryDOMUtils.js"></script>
<script type="text/javascript" src="SpryAssets/SpryDOMEffects.js"></script>
<script type="text/javascript" src="SpryAssets/SpryWidget.js"></script>
<script type="text/javascript" src="SpryAssets/SpryPanelSet.js"></script>
<script type="text/javascript" src="SpryAssets/SpryPanelSelector.js"></script>
<script type="text/javascript" src="SpryAssets/SpryFadingPanels.js"></script>
<script type="text/javascript" src="SpryAssets/SpryTabbedPanels2.js"></script>
<script type="text/javascript" src="SpryAssets/plugins/SpryFadingPanelsPlugin.js"></script>
<script type="text/javascript" src="SpryAssets/plugins/SpryTabbedPanelsKeyNavigationPlugin.js"></script>

<link href="SpryAssets/css/SpryTabbedPanels2.css" rel="stylesheet" type="text/css" />
...
</head>

<body>

<div id="TabbedPanels2">
  <h2>Tab 1</h2>
  <p>Tab 1 Content</p>
  <h2>Tab 2</h2>
  <p>Tab 2 Content</p>
  <h2>Tab 3</h2>
  <p>Tab 3 Content</p>
  <h2>Tab 4</h2>
</div>
```

```

    <p>Tab 4 Content</p>
  </div>

  <script type="text/javascript">
    var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2");
  </script>
</body>

```

In the code, the "new" JavaScript operator initializes the Tabbed Panel 2 widget object, and transforms the div content with the ID of TabbedPanels2 from static HTML code into an interactive page element. The `Spry.Widget.TabbedPanels2` method is a constructor in the Spry framework that creates Tabbed Panels 2 objects, and the information necessary to initialize the object is contained in the `SpryTabbedPanels2.js` JavaScript library that you linked to in the head of the document.

The widget can be applied on a variety of repeating content. It uses CSS selectors to find elements inside a specified container that will make up the tab buttons and panels. By default, the widget finds the h1 to h6 heading elements and use their content as source for the tab buttons. Everything in between the heading elements is used as content for the panels.

#### VARIATION ON THE MARKUP USED FOR THE TABBED PANELS 2 WIDGET

In the preceding example, heading tags and paragraphs were used to display several items with a repeating structure. Source markup:

```

Container <div>
  Tab <h2>
  Panel <p>
  ...

```

Alternative source markup with custom CSS selectors for tabs:

```

Container <div id="newsSection" class="section">
  Tab <div class="articleHeader">News</div>
  Panel <div class="article"><p>Paragraph 1 of the news section</p>
  ...

<script type="text/javascript">
  var TabbedPanels2 = new Spry.Widget.TabbedPanels2("newsSection", { tabSelector: ".articleH
</script>

```

#### WIDGET DYNAMIC GENERATED MARKUP

The Tabbed Panels 2 widget creates its markup at runtime replacing the original markup with a nested structure and uses the initial HTML to fill the tabs and panels content:

```

<div class="TabbedPanels">
  <div class="TabbedPanelsTabGroup">
    <div class="TabbedPanelsTab TabbedPanelsTabFirst PanelSelectorButtonUnselected">
      <div class="TabbedPanelsTab PanelSelectorButtonUnselected">
        ...
      <div class="TabbedPanelsTab TabbedPanelsTabSelected">
        ...
      <div class="TabbedPanelsTab TabbedPanelsTabLast PanelSelectorButtonUnselected">
    <div class="TabbedPanelsContentGroup">
      <div class="TabbedPanelsContent TabbedPanelsContentVisible">
      <div class="TabbedPanelsContent PanelHidden">
      <div class="TabbedPanelsContent PanelHidden">
      <div class="TabbedPanelsContent PanelHidden">

```

Each of the elements in the Tabbed Panels widget contains a CSS class. These classes control the style of the widget and exist in the accompanying `SpryTabbedPanels2.css` file.

You can change the appearance of any given part of the widget by editing the CSS rule that corresponds to the class names assigned to it in the HTML code. For example, to change the background color of the tabs, edit the `TabbedPanelsTab` rule in the `SpryTabbedPanels2.css` file. Keep in mind that changing the CSS code in the

SpryTabbedPanels2.css file will affect all tabbed panels that are linked to that file. Alternately, you can override the widget default styles by redefining widget classes in a CSS file, also linked from your page.

For example, to make the tab font bold create a TabbedPanels2\_cssoverride.css file and link to it after the SpryTabbedPanels2.css file:

```
<link href="SpryAssets/css/SpryTabbedPanels2.css" rel="stylesheet" type="text/css" />
<link href="css/TabbedPanels2_cssoverride.css" rel="stylesheet" type="text/css" />
```

Then set the font-weight CSS property to bold for the TabbedPanelsTab CSS rule in TabbedPanels2\_cssoverride.css:

```
.TabbedPanelsTab {
    font-weight: bold;
}
```

In addition to the classes shown in the HTML code, the Tabbed Panels 2 widget includes certain default behaviors that are attached to the widget. These behaviors are a built-in part of the Spry framework, and are in the SpryTabbedPanels2.js JavaScript library file. The Tabbed Panels 2 library includes behaviors related to hovering, tab clicking (to open panels), panel focus, and keyboard navigation.

You can change the look of the Tabbed Panels as it relates to these behaviors by editing the appropriate classes in the SpryTabbedPanels2.css file. If you want to remove a given behavior, you can delete the CSS rules that correspond to that behavior.

Note: While you can change the look of the Tabbed Panels as it relates to a certain behavior, you cannot alter the built-in behaviors themselves. For example, Spry still places a TabbedPanelsContentVisible class on the currently open panel, even if no properties are set for the TabbedPanelsContentVisible class in the SpryTabbedPanels2.css file.

### CSS code for the Tabbed Panels widget

---

The SpryTabbedPanels2.css file contains the rules that style the Tabbed Panels 2 widget. You can edit these rules to style the tabbed panels' look and feel. The names of the rules in the CSS file correspond directly to the names of the classes specified in the Tabbed Panels 2 widget's HTML code.

Note: You can replace style-related class names in the SpryTabbedPanels2.css file and HTML code with class names of your own. Doing so does not affect the functionality of the widget, as CSS code is not required for widget functionality.

The default functionality for the behaviors classes at the end of the style sheet are defined in the SpryTabbedPanels2.js JavaScript library file. You can change the default functionality by adding properties and values to the behavior rules in the style sheet.

The following is the CSS code for the SpryTabbedPanels2.css file. The first half of the file contains styling rules for horizontal tabbed panels. The second half of the file contains styling rules for vertical tabbed panels.

```
/* Horizontal Tabbed Panels */
.TabbedPanels {
}

.TabbedPanelsTabGroup {
    position: relative;
    top: 1px;
    overflow: hidden;
    _width: 100%; /* IE Hack to force proper layout when preceded by a paragraph. (hasLayout)
    padding: 0;
    margin: 0;
}

.TabbedPanelsTab {
    float: left;
    padding: 4px 10px;
    margin: 0px 1px 0px 0px;
    background-color: #DDD;
    list-style: none;
    border-left: solid 1px #CCC;
    border-bottom: solid 1px #999;
    border-top: solid 1px #999;
    border-right: solid 1px #999;
```

```
-moz-user-select: none;
-khtml-user-select: none;
cursor: pointer;
}

.TabbedPanelsTabHover {
  background-color: #CCC;
}

.TabbedPanelsTabSelected {
  background-color: #EEE;
  border-bottom: 1px solid #EEE;
}

.TabbedPanelsTab a {
  color: black;
  text-decoration: none;
}

.TabbedPanelsContentGroup {
  border-left: solid 1px #CCC;
  border-bottom: solid 1px #CCC;
  border-top: solid 1px #999;
  border-right: solid 1px #999;
  background-color: #EEE;
  padding: 0px;
}

.TabbedPanelsContent {
  overflow: hidden;
  _width: 100%; /* IE Hack to force proper layout when preceded by a paragraph. (hasLayout
padding: 4px 12px;
}

.TabbedPanelsContentVisible {
}

/* Tabbed Bottom Panels */
.BTabbedPanels .TabbedPanelsTabGroup {
  top: -1px;
}

.BTabbedPanels .TabbedPanelsTab {
  border-top: solid 1px #CCC;
  border-bottom: solid 1px #CCC;
}

.BTabbedPanels .TabbedPanelsTabSelected {
  border-top: 1px solid #EEE;
}

/* Vertical Tabbed Panels, applied for left and right tabs position */
.VTabbedPanels {
  overflow: hidden;
  zoom: 1;
}

.VTabbedPanels .TabbedPanelsTabGroup {
  float: left;
  width: 10em;
  height: 20em;
  background-color: #EEE;
  position: relative;
  border-top: solid 1px #999;
  border-right: solid 1px #999;
  border-left: solid 1px #CCC;
  border-bottom: solid 1px #CCC;
}

.VTabbedPanels .TabbedPanelsTab {
  float: none;
}
```

```

margin: 0px;
border-top: none;
border-left: none;
border-right: none;
}

.VTabbedPanels .TabbedPanelsTabSelected {
background-color: #EEE;
border-bottom: solid 1px #999;
}

.VTabbedPanels .TabbedPanelsContentGroup {
clear: none;
float: left;
padding: 0px;
width: 30em;
height: 20em;
}

```

The `SpryTabbedPanels2.css` file contains extensive comments, explaining the code and the purpose for certain rules. For further information, see the comments in the file.

### Insert the Tabbed Panels 2 widget

1. Locate the `SpryTabbedPanels2.js` file and add it to your web site. You can find the `SpryTabbedPanels2.js` file in the `includes/` directory, located in the Spry directory that you downloaded from Adobe Labs.

For example, create a folder called *SpryAssets* in the root folder of your web site, and move the `SpryTabbedPanels2.js` file to it.

2. Locate the `SpryTabbedPanels2.css` file and add it to your web site. You might choose to add it to the main directory, a `SpryAssets` directory, or to a CSS directory if you have one.
3. Open the web page to add the Tabbed Panels 2 widget, and add the required JavaScript files to the page by inserting the following script tags in the page's head tag:

```

<script type="text/javascript" src="SpryAssets/SpryDOMUtils.js"></script>
<script type="text/javascript" src="SpryAssets/SpryDOMEffects.js"></script>
<script type="text/javascript" src="SpryAssets/SpryWidget.js"></script>
<script type="text/javascript" src="SpryAssets/SpryPanelSet.js"></script>
<script type="text/javascript" src="SpryAssets/SpryPanelSelector.js"></script>
<script type="text/javascript" src="SpryAssets/SpryFadingPanels.js"></script>
<script type="text/javascript" src="SpryAssets/SpryTabbedPanels2.js"></script>

```

Make sure that the file path to the `SpryTabbedPanels2.js` file is correct. This path varies depending on where you've placed the file in your web site.

4. Link the `SpryTabbedPanels2.css` file to your web page by inserting the following link tag in the page's head tag:

```

<link href="SpryAssets/SpryTabbedPanels2.css" rel="stylesheet" type="text/css" />

```

Make sure that the file path to the `SpryTabbedPanels2.css` file is correct. This path varies depending on where you've placed the file in your web site.

5. To add the Tabbed Panels 2 widget to your web page, insert the following div tag where you want the widget to appear on the page:

```

<div id="TabbedPanels2">
</div>

```

6. To add tabs to the tabbed panel, insert a `h2` tag for each tab you want, as follows:

```

<div id="TabbedPanels2">
  <h2>Tab 1</h2>
  <h2>Tab 2</h2>

```

```

    <h2>Tab 3</h2>
    <h2>Tab 4</h2>
  </div>

```

The preceding code adds four tabs to the widget. You can add unlimited tabs.

- To add a content area (or panel) for each of the tabs, insert a div container tag after each h3 tag, as follows:

```

<div id="TabbedPanels2">
  <h2>Tab 1</h2>
  <p>Tab 1 Content</p>
  <h2>Tab 2</h2>
  <p>Tab 2 Content</p>
  <h2>Tab 3</h2>
  <p>Tab 3 Content</p>
  <h2>Tab 4</h2>
  <p>Tab 4 Content</p>
</div>

```

- To initialize the Spry tabbed panels object, insert the following script block after the HTML code for the Tabbed Panels widget:

```

<div id="TabbedPanels2">
  .
  .
  .
</div>
<script type="text/javascript">
  var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2");
</script>

```

The new JavaScript operator initializes the Tabbed Panels widget object, and transforms the div content with the ID of TabbedPanels2 from static HTML code into an interactive tabbed panels object. The `Spry.Widget.TabbedPanels2` method is a constructor in the Spry framework that creates tabbed panels objects. The information necessary to initialize the object is contained in the `SpryTabbedPanels2.js` JavaScript library that you linked to at the beginning of this procedure.

Make sure that the ID of the tabbed panels' div tag matches the ID parameter you specified in the `Spry.Widgets.TabbedPanels2` method. Make sure that the JavaScript call comes after the HTML code for the widget.

- Save the page.

The complete code looks as follows:

```

<head>
  ...
  <script type="text/javascript" src="SpryAssets/SpryDOMUtils.js"></script>
  <script type="text/javascript" src="SpryAssets/SpryDOMEffects.js"></script>
  <script type="text/javascript" src="SpryAssets/SpryWidget.js"></script>
  <script type="text/javascript" src="SpryAssets/SpryPanelSet.js"></script>
  <script type="text/javascript" src="SpryAssets/SpryPanelSelector.js"></script>
  <script type="text/javascript" src="SpryAssets/SpryFadingPanels.js"></script>
  <script type="text/javascript" src="SpryAssets/SpryTabbedPanels2.js"></script>

  <link href="SpryAssets/SpryTabbedPanels2.css" rel="stylesheet" type="text/css" />
  ...
</head>

<body>

  <div id="TabbedPanels2">
    <h2>Tab 1</h2>
    <p>Tab 1 Content</p>
    <h2>Tab 2</h2>
    <p>Tab 2 Content</p>
    <h2>Tab 3</h2>
    <p>Tab 3 Content</p>
    <h2>Tab 4</h2>
  </div>

```

```
<p>Tab 4 Content</p>
</div>

<script type="text/javascript">
    var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2");
</script>
</body>
```

---

### Delete a panel from a Tabbed Panels widget

Delete the desired h2 tag and the following paragraph from the code. Do not forget to delete the closing `</h2>` and `</p>` tags when you delete the code.

---

### Enable keyboard navigation

Making widgets accessible for keyboard navigation is an important part of every widget. Keyboard navigation lets the user control the widget with arrow keys.

The keyboard navigation behavior is implemented in the Tabbed Panels Key Navigation plug-in defined in the `plugins/TabbedPanels2/SpryTabbedPanelsKeyNavigationPlugin.js` file.

To enable keyboard navigation for the widget just link to the plug-in JavaScript file in your page, at the end of the list of included JavaScript files:

```
<script type="text/javascript" src="SpryAssets/plugins/SpryTabbedPanelsKeyNavigationPlugin.js">
```

The plug-in automatically adds the keyboard navigation behavior to all the widgets in the page.

You can disable the keyboard navigation for a specific Tabbed Panels 2 widget instance by using an options in its JavaScript constructor:

```
<script type="text/javascript">
    var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2", { enableKeyboardNavigation: false });
</script>
```

---

### Specify the panel activation event

By default a new tabbed panel is shown when the user clicks on a tab button.

To specify another DOM event set the `event` option in the widget constructor:

```
<script type="text/javascript">
    var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2", { event: "mouseover" });
</script>
```

---

### Enable the slide show behavior

The Tabbed Panels 2 widget can be made to show the panels using a slide show behavior. When the slide show behavior is enabled, all panels are displayed sequentially for a certain configurable amount of time.

To enable the slide show behavior set the `autoPlay` option to `true`:

```
<script type="text/javascript">
    var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2", { autoPlay: true });
</script>
```

By default, the slide show stops when the user clicks on a tab, but this can be changed setting the `stopOnUserAction` option to `false`:

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

```
var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2", { autoPlay: true, stop
</script>
```

### Enable the fading panels effect for the slide show behavior

The fading panels effect for the slide show behavior is implemented in the Tabbed Panels Fading Panels plug-in defined in the plugins/TabbedPanels2/SpryFadingPanelsPlugin.js file. To make the fade effect available to the tabbed panels widget instances from the page you first have to link to the plug-in JavaScript file in your page, at the end of the list of included JavaScript files:

```
<script type="text/javascript" src="SpryAssets/plugins/SpryFadingPanelsPlugin.js"></script>
```

To enable the fading panels effect for a specific widget instance you should add the plug-in to the widgets' instance list of plugins:

```
<script type="text/javascript">
  var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2", { plugIns: [Spry.Widge
</script>
```

The fade effect can be customized using options in the widget constructor. Available options are:

Option	Description	Default value
displayInterval	The amount of time a panel is displayed (milliseconds).	5000
minDuration	The fade out effect duration of the disappearing panel (milliseconds).	500
maxDuration	The fade in effect duration of the appearing panel (milliseconds).	300
stoppedMinDuration	The fade out effect duration of the disappearing panel when the user clicks on a tab (milliseconds).	100
stoppedMaxDuration	The fade in effect duration of the appearing panel when the user clicks on a tab (milliseconds).	200

The fade effect should be much faster when the user clicks on a tab.

Example of configuration for the fading panels effect:

```
<script type="text/javascript">
  var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2", {
    displayInterval: 5000,
    minDuration: 300,
    maxDuration: 500,
    stoppedMinDuration: 100,
    stoppedMaxDuration: 200,
    plugIns: [Spry.Widget.TabbedPanels2.FadingPanelsPlugin]
  });
</script>
```

### Change the orientation of the tabbed panels

By default, the tabbed panels buttons appear horizontally, above the panels, but you can set them to appear below, on the left or the right side of the panels.

To change the tabs position, set the tabsPosition option to one of the following values: "top", "left", "right", "bottom" in the widget JavaScript constructor, as follows:

```
<script type="text/javascript">
  var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2", { tabsPosition: "left"
</script>
```

## Set default open tab

You can set a panel to be open when the page containing the Tabbed Panels 2 widget loads in a browser.

Set the defaultTab option in the constructor as follows:

```
<script type="text/javascript">
  var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2", { defaultTab: 2 });
</script>
```

Note: The Tabbed Panels 2 widget uses a zero-based counting system, so the first tab has index 0. Setting the defaultTab option value to 2 opens the third tabbed panel.

## Other properties

The Tabbed Panels 2 widget has a few properties you can set to control the basic functionality of the widget:

Option	Type	Possible values	Default value	Description
injectionType	String	"replace", "inside"	"replace"	Specifies how the dynamic generated markup is inserted in the document. A value of "replace" means that the widget removes the existing markup and injects it's own. Other option is "inside", which injects the widget markup inside the widget container.
hideHeader	Boolean	true, false	true	Hide the header elements used to create the tabs.
tabSelector	String	A valid CSS selector	"h1,h2,h3,h4,h5,h6"	CSS selector used to find the delimiter elements for tabs
contentDelimiterSelector	String	A valid CSS selector	null	CSS selector used to find the delimiter elements for the panels. If not specified, the tabSelector property is used.
widgetClass	String	A valid CSS class name	"TabbedPanels"	Class name applied on the widget container.
tabClass	String	A valid CSS class name	"TabbedPanelsTab"	Class name used for the tab elements.
contentClass	String	A valid CSS class name	"TabbedPanelsContent"	Class name used for the content panel elements.
tabHoverClass	String	A valid CSS class name	"TabbedPanelsTabHover"	Class name used for the hovered tab element.
tabSelectedClass	String	A valid CSS class name	"TabbedPanelsTabSelected"	Class name used for the selected tab element.
panelVisibleClass	String	A valid CSS class name	"TabbedPanelsContentVisible"	Class name used for the visible content panel element.

Option	Type	Possible values	Default value	Description
groupClassStr	String	A valid CSS class name	"Group"	Suffix added to the tab and panel content group elements class names.
firstClassStr	String	A valid CSS class name	"First"	Suffix added to the class name of the first panel element.
firstClassStr	String	A valid CSS class name	"First"	Suffix added to the class name of the first tab element.
lastClassStr	String	A valid CSS class name	"Last"	Suffix added to the class name of the last tab element.
classToRemove	String	A valid CSS class name	""	Class name to be removed from the widget container after the widget markup is generated and inserted into the document.
classToAdd	String	A valid CSS class name	""	Class name to be add to the widget container after the widget markup is generated and inserted into the document.
plugIns	Array	An array of widget plugins.	[]	Spry or user widget plugins. See the Plugins section.
sliceMap	Object	A SliceMap	{}	A hash of class names and corresponding slice types. See the Slicing section.

Any properties can be changed by overwriting the default within the constructor.

For instance:

```
<script type="text/javascript">
  var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2", {
    tabSelector: "#TabbedPanels2 ul li",
    contentDelimiterSelector: "#TabbedPanels2 div div",
    hideHeader: false
  });
</script>
```

All of the properties in this constructor are overwriting the defaults found in the 'defaultConfig' object within [SpryTabbedPanels2.js](#).

## Methods

Syntax is:

```
onclick="widgetname.method()"
```

where widgetname is the var name used in the constructor.

The Tabbed Panels 2 has the following methods:

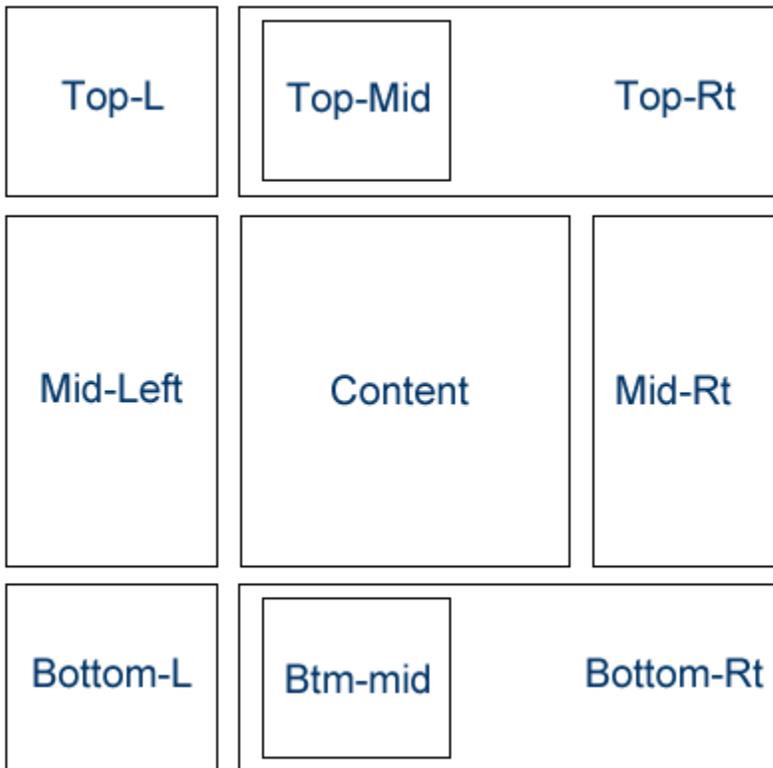
- `getTabs()`
- `getContentPanels()`
- `getTabbedPanelCount()`
- `getTabIndex(tabElement)`

- `getCurrentTabIndex()`
- `showPanel(indexOrID)`
- `showPreviousPanel()`
- `showNextPanel()`
- `showFirstPanel()`
- `showLastPanel()`

## Slicing

Spry UI components are highly skinnable. This is achieved by injecting markup with levels of slicing controlled by the developer.

Slicing is a technique for styling page elements. It takes a page element and dynamically injects a set of `<div>`s around it to provide hooks for CSS styles.



These `<div>`s have established class names applied to them. Using these classes, developers can make scalable elements, add rounded corners or apply other styles.

There are a few levels of slicing: 2, 3-slicing and 9 slicing. The image above shows a 9-sliced element. Start with one and 8 are added. Note that the top and bottom-mid are within the right div. This layout simplifies alignment and stretching.

3-slicing will only add 2 `<div>`s. In the above, it would be just the middle row.

2-slicing will wrap the content in 2 divs.

## SLICEMAP

Slicing is done via the SliceMap. The SliceMap is a javascript object that lists the elements to be sliced. The elements are referenced by their class name. By default, nothing is sliced. And note that not every element is sliceable. The Tabbed Panels 2 widget supports slicing of the widget container element, the tabs group and panels group elements, the tab and panel elements.

Example of a SliceMap specifying slices for all the sliceable widget components:

```
<script type="text/javascript">
  var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2", {
    sliceMap: {
      TabbedPanels: "9slice",
```

```

        TabbedPanelsTabGroup: "3slice",
        TabbedPanelsContentGroup: "3slice",
        TabbedPanelsTab: "9slice",
        TabbedPanelsContent: "9slice"
    });
</script>

```

This means that any element that has the class="TabbedPanels" will get 9-sliced. Same for class="TabbedPanelsTabGroup" with 3-slice.

Slicing options available:

- "2slice" - Wraps the content element with 2 <div>s inside the top-level widget container.
- "3slice" - Wraps the content element with 3 <div>s inside the top-level widget container.
- "9slice" - Injects the content element with 9 <div>s inside the top-level widget container.

These <div>s are dynamically generated and only show up when you view the generated (DOM) source.

## Plugins

Spry uses a plugin model for customizing almost all pieces of the widget. A plugin is a piece of javascript that overwrites or extends existing functionality.

For instance, if you want to use a custom transition function, this can be written as a plugin and included in the constructor (using the 'plugins' array in the defaultConfig object).

Almost any functionality of the widget/component can be overwritten with a plugin. The [fading](#) or [keyboard navigation](#) are examples of plugins.

## Open panels programmatically

Use JavaScript functions to programmatically open specific panels. For example, you might have a button on your page that opens a particular tabbed panel when the user clicks the button.

Remember, Spry uses a zero-based counting system, so 0 (zero) indicates the first, left most tabbed panel. If the tabbed panel has an ID, you can also use the ID to refer to panels.

Use the following functions to open specific tabbed panels:

```

<button onclick="TabbedPanels2.showPanel(0)" >open first panel</button>
<button onclick="TabbedPanels2.showPanel('tabID')">open panel</button>
<script type="text/javascript">
    var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2");
</script>

```

## Customize the Tabbed Panels widget

The SpryTabbedPanels2.css file provides the default styling for the Tabbed Panels 2 widget. You can, however, easily customize the widget by changing the appropriate CSS rule. The CSS rules in the SpryTabbedPanels2.css file use the same class names as the related elements in the accordion's HTML code, so it's easy for you to know which CSS rules correspond to the different sections of the Tabbed Panels 2 widget. Additionally, the SpryTabbedPanels2.css file contains class names for behaviors that are related to the widget (for example, hovering and clicking behaviors).

The SpryTabbedPanels2.css file should already be included in your website before you start customizing.

Note: Internet Explorer up to and including version 6 does not support sibling and child contextual selectors (for example, .TabbedPanels + .TabbedPanels or .TabbedPanels > .TabbedPanels).

### STYLE A TABBED PANELS 2 WIDGET (GENERAL INSTRUCTIONS)

Set properties for the entire Tabbed Panels 2 widget container, or set properties for the components of the widget individually.

1. Open the SpryTabbedPanels2.css file.
2. Locate the CSS rule for the part of the tabbed panel to change. For example, to change the background color of the tabbed panels' tabs, edit the TabbedPanelsTab rule in the SpryTabbedPanels2.css file.
3. Make your changes to the CSS rule and save the file.

You can replace style-related class names in the SpryTabbedPanels2.css file and HTML code with class names of your own. Doing so does not affect the functionality of the widget.

The SpryTabbedPanels2.css file contains extensive comments, explaining the code and the purpose for certain rules. For further information, see the comments in the file.

### STYLE TABBED PANELS 2 WIDGET TEXT

Set properties for the entire Tabbed Panels 2 widget container, or set properties for the components of the widget individually.

Use the following table to locate the appropriate CSS rule, and then add your own text-styling properties and values.

Text to change	Relevant CSS rule	Example of properties and values to add
Text in the entire widget	.TabbedPanels	font: Arial; font-size:medium;
Text in panel tabs only	.TabbedPanelsTabGroup or .TabbedPanelsTab	font: Arial; font-size:medium;
Text in content panels only	.TabbedPanelsContentGroup or .TabbedPanelsContent	font: Arial; font-size:medium;

### CHANGE TABBED PANELS 2 WIDGET BACKGROUND COLORS

Use the following table to locate the appropriate CSS rule, and then add or change background-color properties and values.

Color to change	Relevant CSS rule	Example of property and value to add or change
Background color of panel tabs	.TabbedPanelsTabGroup or .TabbedPanelsTab	background-color: #DDD; (This is the default value.)
Background color of content panels	.TabbedPanelsContentGroup or .TabbedPanelsContent	background-color: #EEE; (This is the default value.)
Background color of selected tab	.TabbedPanelsTabSelected	background-color: #EEE; (This is the default value.)
Background color of panel tabs when the mouse pointer moves over them	.TabbedPanelsTabHover	background-color: #CCC; (This is the default value.)

### CONSTRAIN THE WIDTH OF TABBED PANELS 2

By default, the Tabbed Panels 2 widget expands to fill available space. To constrain the width of a Tabbed Panels 2 widget, set a width property for the accordion container.

1. Locate the TabbedPanels 2 CSS rule in the SpryTabbedPanels2.css file. This rule defines properties for the main container element of the Tabbed Panels 2 widget.
2. Add a width property and value to the rule, for example width: 300px;.

### CHANGE TABBED PANELS 2 HEIGHT

By default, the height of tabbed panels expands according to content. To set a specific height for panels, add a height property to the TabbedPanelsContent rule.

1. Locate the TabbedPanelsContent CSS rule in the SpryTabbedPanels2.css file.
2. Add a height property and value to the rule, for example height: 300px;.

### CHANGE TABBED PANELS 2 BEHAVIORS

The Tabbed Panels 2 widget includes a few predefined behaviors. These behaviors consist of changing CSS classes when a particular event occurs. For example, when a mouse pointer hovers over a panel tab, Spry applies the TabbedPanelsTabHover class to the widget. (This behavior is similar to a: hover for links.) The behaviors are blank by default; to change them, add properties and values to the rules.

1. Open the SpryTabbedPanels2.css file and locate the CSS rule for the tabbed panels behavior to change. The

Tabbed Panels 2 widget includes four built-in rules for behaviors.

Behavior	Description
.TabbedPanelsTabHover	Activates when hovering over the panel tab
.TabbedPanelsTabFocused	Activates when a tab has keyboard focus
.TabbedPanelsTabSelected	Activates on currently selected tab
.TabbedPanelsContentVisible	Activates on content area of currently selected tab

For examples, see the Tabbed Panels 2 sample file located in the samples directory of the Spry directory that you downloaded from Adobe Labs.

2. Add CSS rules for any of the behaviors you want to enable.

Note: You can replace the behavior-related class names with class names of your own. to override the default class with your class names, send the new values as arguments to the Tabbed Panels 2 constructor:

```
<script type="text/javascript">
  var TabbedPanels2 = new Spry.Widget.TabbedPanels2("TabbedPanels2", {
    tabHoverClass: "customHover",
    panelVisibleClass: "customOpen",
    tabSelectedClass: "customSelected",
    tabFocusedClass: "customFocused"
  });
</script>
```

Copyright © 2010. Adobe Systems Incorporated. All rights reserved.

[back to top](#)