jQuery UI 定制 jQuery UI 主题

jQuery UI 工作原理

jQuery UI 包含了许多维持状态的小部件（Widget），因此，它与典型的 jQuery 插件使用模式略有不同。其安装方式与大部分 jQuery 插件的安装方式类似，jQuery UI 的小部件是基于 部件库（Widget Factory）创建的，小部件库提供了通用的 API。所以，只要您学会使用其中一个，您就知道如何使用其他的小部件（Widget）。本教程将通过 进度条（progressbar）小部件代码实例介绍常见的功能。

安装

为了跟踪部件的状态，我们首先介绍一下小部件的全生命周期。当小部件安装时，生命周期开始。我们只需要在一个或多个元素上调用插件，即安装了小部件。

```
$( "#elem" ).progressbar();
```

这将会初始化 jQuery 对象中的每个元素，在本例中，元素 id 为 "elem"。因为我们调用无参数的 `.progressbar()` 方法，小部件则会按照它的默认选项进行初始化。我们可以在安装时传递一组选项，这样既可重写默认选项。

```
$( "#elem" ).progressbar({ value: 20 });
```

安装时传递的选项数目多少可根据我们的需要而定。任何我们未传递的选项则都使用它们的默认值。
选项是小部件状态的组成部分，所以我们也可以在安装后再进行设置选项。我们将在后续的 `option` 方法中介绍这部分内容。

方法

既然小部件已经初始化，我们就可以查询它的状态，或者在小部件上执行动作。所有初始化后的动作都以方法调用的形式进行。为了在小部件上调用一个方法，我们可以向 jQuery 插件传递方法的名称。例如，为了在进度条（progressbar）小部件上调用 `value` 方法，我们应该使用：

```
$( "#elem" ).progressbar( "value" );
```

如果方法接受参数，我们可以在方法名后传递参数。例如，为了传递参数 `40` 给 `value` 方法，我们可以使用：

```
$( "#elem" ).progressbar( "value", 40 );
```

就像 jQuery 中的其他方法一样，大部分的小部件方法为链接返回 jQuery 对象。

```
$( "#elem" )
    .progressbar( "value", 90 )
    .addClass( "almost-done" );
```

公共的方法

每个小部件都有它自己的一套基于小部件所提供功能的方法。然而，有一些方法是所有小部件都共同具有的。

option

正如我们前面所提到的，我们可以在初始化之后通过 `option` 方法改变选项。例如，我们可以通过调用 `option` 方法改变 progressbar（进度条）的 value 为 30。

```
$( "#elem" ).progressbar( "option", "value", 30 );
```

请注意，这与之前我们调用 `value` 方法的实例有所不同。在本实例中，我们调用 `option` 方法，改变 value 选项为 30。

我们也可以为某个选项获取当前的值。

```
$( "#elem" ).progressbar( "option", "value" );
```

另外，我们可以通过给 `option` 方法传递一个对象，一次更新多个选项。

```
$( "#elem" ).progressbar( "option", {
    value: 100,
    disabled: true
});
```

您也许注意到 `option` 方法有着与 jQuery 代码中取值器和设置器相同的标志，就像 `.css()` 和 `.attr()`。唯一的不同就是您必须传递字符串 "option" 作为第一个参数。

disable

`disable` 方法禁用小部件。在进度条（progressbar）实例中，这会改变样式让进度条显示为禁用状态。

```
$( "#elem" ).progressbar( "disable" );
```

调用 `disable` 方法等同于设置 `disabled` 选项为 `true`。

enable

`enable` 方法是 `disable` 方法的对立面。

```
$( "#elem" ).progressbar( "enable" );
```

调用 `enable` 方法等同于设置 `disabled` 选项为 `false`。

destroy

如果您不再需要小部件，那么可以销毁它，返回到最初的标记。这意味着小部件生命周期的终止。

```
$( "#elem" ).progressbar( "destroy" );
```

一旦您销毁了一个小部件，您就不能在该部件上调用任何方法，除非您再次初始化这个小部件。如果您要移除元素，可以直接通过 `.remove()`，也可以通过 `.html()` 或 `.empty()` 来实现，小部件会自动销毁。

widget

一些小部件生成包装器元素，或与原始元素断开连接的元素。在下面的实例中，`widget` 将返回生成的元素。在进度条（progressbar）实例中，没有生成的包装器，`widget` 方法返回原始的元素。

```
$( "#elem" ).progressbar( "widget" );
```

事件

所有的小部件都有跟他们各种行为相关的事件，用于在状态改变时通知您。对于大多数的小部件，当事件被触发时，名称以小部件名称为前缀。例如，我们可以绑定进度条（ ）的 change 事件，一旦值发生变化时就触发。

```
$( "#elem" ).bind( "progressbarchange", function() {
    alert( "The value has changed!" );
});
```

每个事件都有一个相对应的回调，作为选项进行呈现。我们可以使用进度条（progressbar）的 `change` 回调，这等同于绑定 `progressbarchange` 事件。

```
$( "#elem" ).progressbar({
    change: function() {
        alert( "The value has changed!" );
    }
});
```

公共的事件

大多数事件是针对特定的小部件，所有的小部件都有一个公共的 `create` 事件。该事件在小部件被创建时即被触发。

← jQuery UI 定制 jQuery UI 主题 →