← jQuery UI 小部件方法调用 jQuery UI 如何使用部件库 →

jQuery UI 为什么使用部件库（Widget Factory）

编写 jQuery 插件与向 `jQuery.prototype`（通常显示为 `$.fn`）添加方法一样简单，且需要遵循一些简单的规则，比如返回 `this`。所以为什么会存在部件库（Widget Factory）？

在本章节中，我们将讲解部件库（Widget Factory）的好处，并了解何时使用它，以及为什么要使用它。

无状态 vs. 有状态插件

大多数 jQuery 插件是无状态的，它们执行一些动作即完成了它们的任务。例如，如果您使用 `.text( "hello" )` 设置元素的文本，没有安装阶段，结果都是一样的。对于这种类型的插件，它只是扩展了 jQuery 的原型。

然而，一些插件是有状态的，它们有全生命周期、维持状态以及对变化的反应。这些插件需要大量专门的代码来初始化和状态管理（有时是销毁）。这就导致出现了用于创建有状态插件的模板。更糟糕的是，每个插件的作者按照不同的方式进行管理插件的生命周期和状态，这就导致了不同的插件有不同的 API 样式。部件库（Widget Factory）旨在解决这些问题，它移除了模板，并为插件创建了一个一致的 API。

一致的 API

部件库（Widget Factory）定义了如何创建和销毁小部件，获取和设置选项，调用方法，以及监听小部件触发的事件。通过使用部件库（Widget Factory）来创建有状态的插件，会自动符合定义的标准，让新用户更容易使用您的插件。另外，部件库（Widget Factory）还能实现定义接口的功能。如果您对部件库（Widget Factory）提供的 API 还不熟悉，请查看 如何使用部件库（Widget Factory）。

在初始化时设置选项

当您创建一个接受选项的插件时，您应该为尽可能多的选项定义 defaults。然后在初始化时，把用户提供的选项与 defaults 进行合并。您也可以暴露 defaults，这样用户就可以更改默认值。在 jQuery 插件中，一个常用的模式如下所示：

```
$.fn.plugin = function( options ) {
    options = $.extend( {}, $.fn.plugin.defaults, options );
    // Plugin logic goes here.
};

$.fn.plugin.defaults = {
    param1: "foo",
    param2: "bar",
    param3: "baz"
};
```

部件库（Widget Factory）也提供了这个功能，并在这上面做了改进。使用部件库（Widget Factory）之后，它将如下所示。

```
$.widget( "ns.plugin", {

```

```
    // Default options.
    options: {
        param1: "foo",
        param2: "bar",
        param3: "baz"
    },

    _create: function() {
        // Options are already merged and stored in this.options
        // Plugin logic goes here.
    }

});
```

jQuery UI 小部件方法调用 jQuery UI 如何使用部件库

点我分享笔记