بسم الله الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

JqueryUI - Widget Factory

مدرس : مهندس افشین رفوآ

JqueryUI - Widget Factory

قبلا تنها راه برای نوشتن کنترل های سفارشی (custom controls) در jQuery گسترش فضانام $.fn بود. این کار برای ویجت های ساده موثر بود. تصور کنید که ویجت های دارای وضعیت بیشتری ساخته اید که خیلی سریع سنگین می شود. برای کمک در فرایند ساخت ویجت ها، Widget Factory در jQuery UI معرفی شد که بسیاری از boilerplate ها را حذف می کند که به ویژه با تنظیم یک ویجت در ارتباط هستند.

یک jQueryUI Widget Factory تابعی است که نام یک رشته و یک آبجکت را به عنوان برهان ها، خود گرفته و یک jQuery plugin و یک Class ایجاد می کند تا قابلیت آن را در هم بپیچد.

ترکیب:

در زیر ترکیب مربوط به متد jQueryUI Widget Factory را مشاهده می کنید:

```
jQuery.widget( name [, base ], prototype )
```

name: رشته ای است که حاوی یک namespace و widget name از ویجت برای ایجاد می باشد.

base: ویجت پایه برای دریافت فرم و باید یک سازنده باشد که می تواند با لغت کلیدی new نمونه گذاری شود. پیش فرض برای jQuery.Widget.

prototype: یک آبجکت برای اینکه به عنوان یک نمونه ی اولیه برای ویجت برای دریافت فرم، استفاده می شود. برای نمونه jQuery UI دارای یک پلاگین "mouse" است که روی آن بقیه ی تعاملات پلاگین ها بر اساس

این مورد می باشند، که برای به دست آوردن draggable, droppable و غیره می باشد. تمام مواردی که از پلاگین های ماوس دریافت می کنند: (jQuery.widget("ui.draggable", $.ui.mouse, {...}؛ اگر شما این آرگیومنت را تامین نکنید، ویجت مستقیما از jQuery.Widget "base widget," دریافت می کند. (به تفاوت بین widget با w کوچک و Widget با W دقت داشته باشید.)

Base Widget

Base widget ویجتی است که به وسیله ی widget factory استفاده می شود.

گزینه ها:

جدول زیر گزینه های مختلفی را ارائه می دهد که با Base widget استفاده می شوند.

Option	Description
disabledhide	این گزینه اگر روی true تنظیم شده باشد، ویجت را غیرفعال می سازد. مقدار پیش فرض آن false است.
hide	این گزینه چگونگی محرک سازی پنهان کردن عنصر را تعیین می کند. مقدار پیش فرض آن null است.
show	این گزینه چگونگی محرک سازی نمایش عنصر را تعیین می کند. مقدار پیش فرض آن null است.

متدها:

جدول زیر متدهای مختلفی را ارائه می دهد که با base widget مورد استفاده قرار می گیرند:

Action	Description
_create()	این متد سازنده ی widget است. هیچ پارامتری وجود ندارد، اما this.elementand this.options تقریبا تنظیم شده اند.
_delay(fn [, delay])	این متد تابع ارائه شده را پس از تاخیر تعیین شده، تحریک می کند. timeout ID را برای استفاده با clearTimeout() بازمی گرداند.
_destroy()	متد عمومی destroy() که همه ی داده های مشترک، رویدادها و غیره را پاک می کند و به این واگذار می کند – متد destroy() برای custom, widget-specific, cleanup.
_focusable(element)	این متد عنصر را برای به کار گرفتن گروه theui-state-focu تنظیم می کند. گرداننده های رویداد به طور خودکار پاک می شوند.
_getCreateEventData()	همه ی ویجت ها رویداد creat را آغاز می کنند. به طور پیش فرض هیچ داده ای در رویداد ارائه نمی شود. اما این متد می تواند یک آبجکتی را بازگرداند که به عنوان داده ی رویداد creat منتقل خواهد شد.
_getCreateOptions()	این متد به ویجت اجازه می دهد تا یک متد custom برای گزینه های تعریف شده در دوره ی نمونه گذاری، تعیین کند. گزینه های ارائه شده توسط یوزر، گزینه های مربوط به این متد را رد می کنند که گزینه های پیش فرض را نیز رد می کنند.

این متد یا با استفاده از متدهای داخلی انیمیشن یا با استفاده از افکت های custom، یک عنصر را فورا پنهان می کند. گزینه ی hide را برای مقادیر ممکن برای گزینه بررسی کنید.	_hide(element, option [, callback])
این متد عنصری را تنظیم می کند برای به کار گرفتن گروه ui-state-hover روی hover. این گرداننده های رویداد به طور خودکار تا ازبین رفتن تمیز می شوند.	_hoverable(element)
هر زمانی که plugin بدون هیچ آرگیومنتی یا تنها با یک گزینه ی hash فرا خوانده می شود، ویجت مقدار دهی می شود. این شامل زمانی است که ویجت ایجاد می شود.	_init()
این متد گرداننده ی رویداد را از عناصر تعیین شده رها می کند.	_off(element, eventName)
گرداننده های رویداد را در عناصر مخصوصی محدود می کند. انتقال از طریق انتخابگرهای داخل نام های مربوط به رویداد پشتیبانی می شود، به عنوان مثال "click .foo".	_on([suppressDisabledCheck] [, element], handlers)
این متد از متد _setOptions() برای هر گزینه ی مجزایی، فرا خوانده می شود. وضعیت ویجت باید براساس تغییرات آپدیت شود.	_setOption(key, value)
زمانی که متد option() فراخوانده می شود، این متد نیز فرا خوانده می شود، بدون توجه به فرمی که متد option() از آن فرا خوانده شد.	_setOptions(options)

با استفاده از متدهای انیمیشن داخلی یا با استفاده از افکت های custom، یک عنصر را فورا نشان می دهد. گزینه show را برای مقادیر ممکن گزینه بررسی کنید.	_show(element, option [, callback])
این متد، یک متد هم نام را از ویجت parent، با آرگیومنت های تعیین شده، تحریک می کند. به ویژه ()call.	_super([arg] [, ...])
متد هم نام را از ویجت parent با ردیفی از آرگیومنت ها، تحریک می کند.	_superApply(arguments)
این متد یک رویداد و کال بک مربوط به آن را آغاز می کند. گزینه ای با نامی به همان مقدار برای تایپ، به عنوان یک کال بک تحریک می شود.	_trigger(type [, event] [, data])
این متد قابلیت ویجت را به طور کل حذف می کند. این امر عنصر را به وضعیت pre-init خود باز خواهد گرداند.	destroy()
این متد ویجت را غیرفعال می کند.	disable()
این متد ویجت را فعال می کند.	enable()
این متد مقدار کنونی مربوط به optionName مشخص شده را می گیرد.	option(optionName)
این متد آبجکتی حاوی جفت های key/value را دریافت می کند که گزینه های hash کنونی مربوط به ویجت را نشان می دهند.	option()
این متد مقدار مربوط به گزینه ی widget که با optionName تعیین شده در ارتباط است، تنظیم می کند.	option(optionName, value)

option(options)	این متد یک یا چند گزینه را برای ویجت تنظیم می کند.
widget()	این گزینه یک آبجکت jQuery را بازمی گرداند که حاوی عنصر اصلی یا دیگر عناصر تولید شده مرتبط می باشد.

Events

Event Method	Description
create(event, ui)	این رویداد با ایجاد یک ویجت آغاز می شود.

jQueryUI widget factory یک روش آبجکت محور برای مدیریت طول دوره ی (lifecycle) یک ویجت، ارائه می دهد. این فعالیت ها عبارتند از:

ایجاد و یا از بین بردن ویجت، به عنوان مثال:

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

تغییر گزینه های ویجت، به عنوان مثال:

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

فراخوانی های super در ویجت های زیر مجموعه، برای مثال:

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

اطلاعیه های رویداد، برای مثال:

```
$( "#elem" ).bind( "progressbarchange", function() {

alert( "The value has changed!" );

});
```

مثال:

اکنون اجازه بدهید در مثال زیر یک custom widget ایجاد کنیم. ما یک دکمه ی widget ایجاد خواهیم کرد.

چگونگی ایجاد گزینه ها، متدها و رویدادها را در یک ویجت در مثال های زیر مشاهده خواهیم کرد:

اجازه بدهید یک custom widget ساده ایجاد کنیم:

```
<!doctype html>
<head>
  <meta charset="utf-8">
  <title>jQuery UI Widget - Default functionality</title>
  <link rel="stylesheet" href="//code.jquery.com/ui/1.10.4/themes/smoothness/jquery-ui.css">
  <script src="http://code.jquery.com/jquery-1.10.2.js"></script>
  <script src="http://code.jquery.com/ui/1.10.4/jquery-ui.js"></script>
  <script>
    $(function () {
      $.widget("iP.myButton", {
        _create: function () {
          this._button = $("<button>");
          this._button.text("My first Widget Button");
          this._button.width(this.options.width)
          this._button.css("background-color", this.options.color);
          this._button.css("position", "absolute");
          this._button.css("left", "100px");
          $(this.element).append(this._button);
        },
      });
      $("#button1").myButton();
    });
  </script>
</head>
<body>
  <div id="button1"></div>
</body>
</html>
```

اکنون اجازه بدهید کد بالا را در یک فایل HTML به نام widgetfactoryexample.htm ذخیره کرده و در یک مرور گر استاندارد که javascript را پشتیبانی می کند، باز کنیم. شما باید خروجی زیر را مشاهده کنید:

افزودن گزینه ها به custom widget:

در مثال قبل از تابع _create برای ایجاد یک custom control استفاده کردیم. اما یوزرها به طور کل تمایل دارند که با تنظیم و اصلاح گزینه ها، کنترل را سفارشی سازند. ما می توانیم یک آبجکت options تعریف کنیم که مقادیر پیش فرض را برای همه ی گزینه هایی که شما تعریف کردید، تعریف می کند. تابع _setOption برای این هدف استفاده می شود. این تابع برای گزینه های مجزایی که یوزرها تنظیم کرده اند، فرا خوانده می شود. در مورد زیر دکمه های width و background-color را تنظیم می کنیم.

```
<!doctype html>
<head>
  <meta charset="utf-8">
  <title>jQuery UI Widget - Default functionality</title>
  <link rel="stylesheet" href="//code.jquery.com/ui/1.10.4/themes/smoothness/jquery-ui.css">
  <script src="http://code.jquery.com/jquery-1.10.2.js"></script>
  <script src="http://code.jquery.com/ui/1.10.4/jquery-ui.js"></script>
  <script>
    $(function () {
      $.widget("iP.myButton", {
        _create: function () {
          this._button = $("<button>");
          this._button.text("My first Widget Button");
          this._button.width(this.options.width)
          this._button.css("background-color", this.options.color);
          this._button.css("position", "absolute");
          this._button.css("left", "100px");
          $(this.element).append(this._button);
        },
        _setOption: function (key, value) {
          switch (key) {
            case "width":
              this._button.width(value);
              break;
            case "color":
              this._button.css("background-color", value);
              break;
          }
        },
      });
      $("#button2").myButton();
      $("#button2").myButton("option", { width: 100, color: "#cedc98" });
    });
  </script>
</head>
<body>
```

```
  <div id="button2"></div>
</body>
</html>
```

اکنون اجازه بدهید کد بالا را در یک فایل HTML به نام widgetfactoryexample.htm ذخیره کرده و در یک مرور گر استاندارد که javascript را پشتیبانی می کند، باز کنیم. شما باید خروجی زیر را مشاهده کنید:

افزودن متدها به custom widget:

در مثال زیر متدهایی را اضافه خواهیم کرد که یوزر می تواند مورد استفاده قرار دهد و برای ساخت در یک چارچوب بسیار ساده هستند. ما یک متد Move خواهیم نوشت که دکمه را در یک فاصله ی افقی مشخص، تغییر می دهد. برای انجام این کار لازم است موقعیت و پراپرتی های سمت چپ را در تابع _create تعیین کنیم:

```
this._button.css("position", "absolute");

this._button.css("left", "100px");
```

در ادامه ی آن، اکنون یوزر می تواند متد شما را به روش معمول jQuery UI فرا بخواند:

```
this._button.css("position", "absolute");

this._button.css("left", "100px");

$("button3").myButton("move", 200);

<!doctype html>
<head>
  <meta charset="utf-8">
  <title>jQuery UI Widget - Default functionality</title>
  <link rel="stylesheet" href="//code.jquery.com/ui/1.10.4/themes/smoothness/jquery-ui.css">
  <script src="http://code.jquery.com/jquery-1.10.2.js"></script>
  <script src="http://code.jquery.com/ui/1.10.4/jquery-ui.js"></script>
  <script>
    $(function () {
      $.widget("iP.myButton", {
        _create: function () {
          this._button = $("<button>");
          this._button.text("My first Widget Button");
          this._button.width(this.options.width)
          this._button.css("background-color", this.options.color);
          this._button.css("position", "absolute");
          this._button.css("left", "100px");
```

```
        $(this.element).append(this._button);
      },

      move: function (dx) {
        var x = dx + parseInt(this._button.css("left"));
        this._button.css("left", x);
        if (x > 400) { this._trigger("outbounds", {}, { position: x }); }
      }
    });
    $("#button3").myButton();
    $("#button3").myButton("move", 200);
  });
 </script>
</head>
<body>
 <div id="button3"></div>
</body>
</html>
```

اکنون اجازه بدهید کد بالا را در یک فایل HTML به نام widgetfactoryexample.htm ذخیره کرده و در یک مرور گر استاندارد که javascript را پشتیبانی می کند، باز کنیم. شما باید خروجی زیر را مشاهده کنید:

افزودن رویدادها به Custom Widget:

در این مثال چگونگی ایجاد یک رویداد را توضیح خواهیم داد. برای ایجاد یک رویداد تمام کاری که باید انجام دهید، استفاده از متد _trigger است. اولین پارامتر نام رویداد و دومین پارامتر آبجکتی است که می خواهید انتقال دهید و سومین پارامتر نیز هر آبجکت custom event که می خواهید انتقال دهید، می باشد.

در اینجا در حال اجرای رویدادی هستیم که هنگامی اتفاق می افتد که دکمه بیشتر از x=400 جابه جا می شود. تمام آنچه باید انجام دهید، افزودن به تابع جابه جایی می باشد:

```
if(x<400){ this._trigger("outbounds",{}, {position:x}); }
```

در این مورد رویداد outbound نامیده می شود و یک آبجکت رویداد خالی با یک آبجکت custom event منتقل می شود که به سادگی موقعیت را به عنوان تنها پراپرتی خود ارائه می دهد.

کل تابع move عبارت است از:

```
move: function(dx) {

var x = dx+parseInt(this._button.css("left"));
```

```
this._button.css("left", x);

if(x<400){ this._trigger("outbounds",{},

                {position:x}); }

}
```

یوزر می تواند عملکرد رسیدگی به رویداد را به سادگی و با تعریف یک گزینه ی هم نام، تنظیم کند.

```
$("button4").myButton("option",

  {width: 100,

color: "red",

   outbounds:function(e,ui){

      alert(ui.position);}

});

<!doctype html>
<head>
  <meta charset="utf-8">
  <title>jQuery UI Widget - Default functionality</title>
  <link rel="stylesheet" href="//code.jquery.com/ui/1.10.4/themes/smoothness/jquery-ui.css">
  <script src="http://code.jquery.com/jquery-1.10.2.js"></script>
  <script src="http://code.jquery.com/ui/1.10.4/jquery-ui.js"></script>
  <script>
    $(function () {
      $.widget("iP.myButton", {
        _create: function () {
          this._button = $("<button>");
          this._button.text("My first Widget Button");
          this._button.width(this.options.width)
          this._button.css("background-color", this.options.color);
          this._button.css("position", "absolute");
          this._button.css("left", "100px");
          $(this.element).append(this._button);
        },
        move: function (dx) {
          var x = dx + parseInt(this._button.css("left"));
          this._button.css("left", x);
          if (x > 400) { this._trigger("outbounds", {}, { position: x }); }
        }
      });
      $("#button4").myButton();
      $("#button4").on("mybuttonoutbounds", function (e, ui) {
        alert("out");
      });
      $("#button4").myButton("move", 500);
    });
  </script>
</head>
<body>
```

```
  <div id="button4"></div>
</body>
</html>
```

اکنون اجازه بدهید کد بالا را در یک فایل HTML به نام widgetfactoryexample.htm ذخیره کرده و در یک مرور گر استاندارد که javascript را پشتیبانی می کند، باز کنیم، یک کادر هشدار باز می شود.