فصل اول : کلیات jquery

Jqueryچیست ؟

Jquery یک کتابخانه بسیار مفید برای جاوااسکریپت است .

Jqueryبسیار ساده و کارآمد است و مشکل جاوااسکریپت را برای تطابق با بروزرهای مختلف برطرف نموده .

یادگیری jquery بسیار آسان است .

در Jquery کد جاوااسکریپت از فایل html جدا شده و بنابراین کنترل کد ها و بهینه سازی آنها بسیار ساده تر خواهد شد .

Jquery توابعی برای کار با ایجاکس فراهم نموده و در این زمینه نیز کار را بسیار ساده کرده است .

در jquery میتوان از خصوصیت فراخوانی زنجیره ای متد ها استفاده نمود و این باعث میشود چندین کد فقط در یک سطر قرار گیرد و در نتیجه کد بسیار مختصر گردد.

پیش نیازها :

برای دانستن jquery باید با زبانهای html ، CSS و جاوااسکریپت آشنایی داشته باشید .

چگونه از زبان jquery استفاده کنیم ؟

شما تنها نیاز دارید که آخرین نسخه jquery را از سایت http://jquery.com بصورت رایگان دانلود کنید . این فایل یک فایل جاوااسکریپت با پسوند js است که باید آن را در فایل html خود در قسمت <head> فراخوانی نمایید .

```
<script type="text/javascript"  src="jquery.js"></script>
```

بعد میتوانید کد های جیکوئری مورد نظر خود را در یک تگ جاوااسکریپت جداگانه بنویسید .به عنوان مثال :

```
<script type="text/javascript">
$(document).ready(function(){
  $("p").click(function(){
  $(this).hide();
  });
});
</script>
```

به عنوان مثال قطعه کد زیر هنگامی که روی عبارت "Click Me " کلیک شود این تگ پنهان (hide) میشود .

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
```

```
  $("p").click(function(){
  $(this).hide();
  });
});
</script>
</head>
<body>
<p> Click Me <p>
</body>

</html>
```

همان طور که در کد بالا میبینید تمام کدهای jquery درون یک تابع به نام $ نوشته شده و قابل استفاده هستند .

به جای حرف $ میتوان از نام کامل آن jQuery استفاده کرد .

در خط اول کد (خط پنج قطعه کد بالا) تابعی را هنگامی که برنامه ما آماده اجراشدن است صدا میزنیم (هنگامی که رویداد ready از شی document صدا زده شده است) .

در این تابع میگوییم (خط دوم کد) : هنگامی که روی همه تگهای "p" کلیک شد تابعی صدا زده شود که کارش این است که این تگ (تگی که رویش کلیک شده) را پنهان (hide) نماید .

به همین سادگی . حال اگر صد تا تگ p هم در قسمت body فایل خود بنویسیم . هنگامی که روی هرکدام کلیک میشود این تابع اجرا میشود .

اگر نگران ساختار مبهم jquery هستید باید به شما اطمینان دهم که بزودی در طی روند این آموزش به این ساختار عادت کرده و از آن لذت خواهید برد .

نکته : شما همچنین میتوانید کدهای خود را در یک فایل js جداگانه بنویسید و سپس آن را **پس از** فراخوانی کتابخانه jquery.js ، فراخوانی نمایید :

<script type="text/javascript" src="jquery.js"></script>

<script type="text/javascript" src="mycode.js"></script>

برنامه ide مناسب برای jquery :

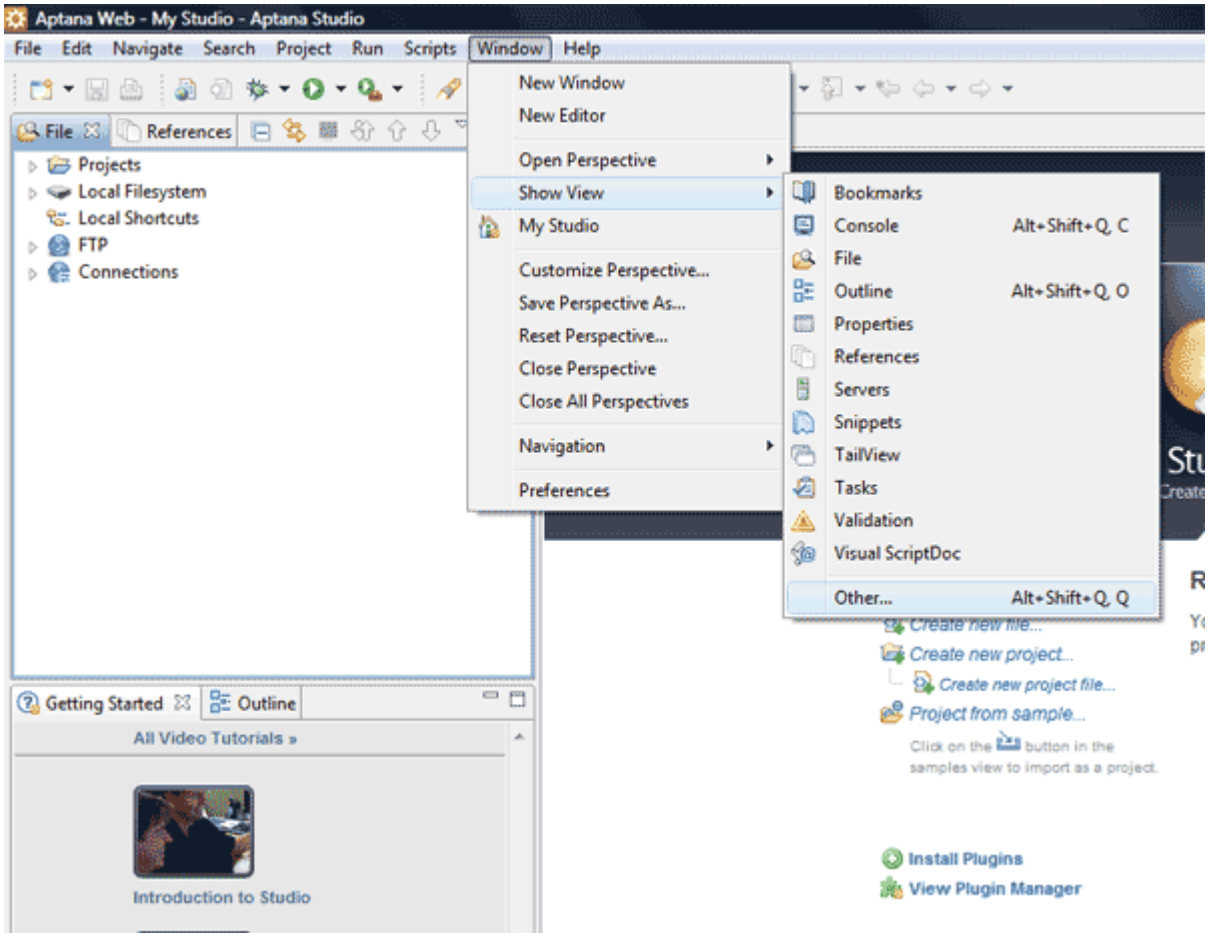
شما میتوانید کدهای jquery را در تمام برنامه های طراحی وب مانند Dreamweaver Zend Studion , Eclips, بنویسید اما معروفترین محیط برای برنامه نویسی jquery در حال حاضر نرم افزار open source به نام AptanaStudio است که برای کد نویسی جاوااسکریپت نوشته شده اما کتابخانه jquery هم به آن اضافه شده که البته در این زمان که بنده مشغول نوشتن این آموزش هستم باید نرم افزار را جداگانه دانلودنمایید و کتابخانه راهنمای jquery را جداگانه تهیه و بر روی آن نصب کنید :

نرم افزار Aptana را از سایت http://www.aptana.com دانلود نمایید .

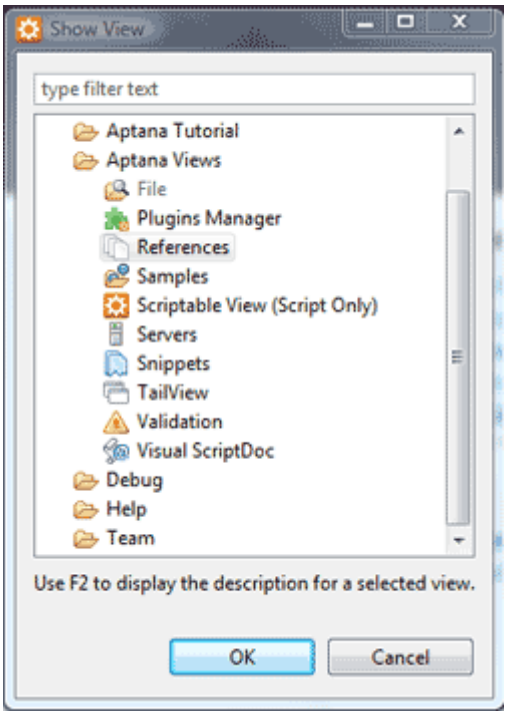
کتابخانه jquery را که برای Aptana تهیه شده از سایت http://www.bitstorm.org/edwin/jquery دریافت نمایید و یا دریافت مستقیم از همین وبلاگ :
لینک دریافت : http://blog.monavarian.ir/wp-content/aptana_jquery-111_sdoc.zip

وارد نر افزار Aptana شوید :

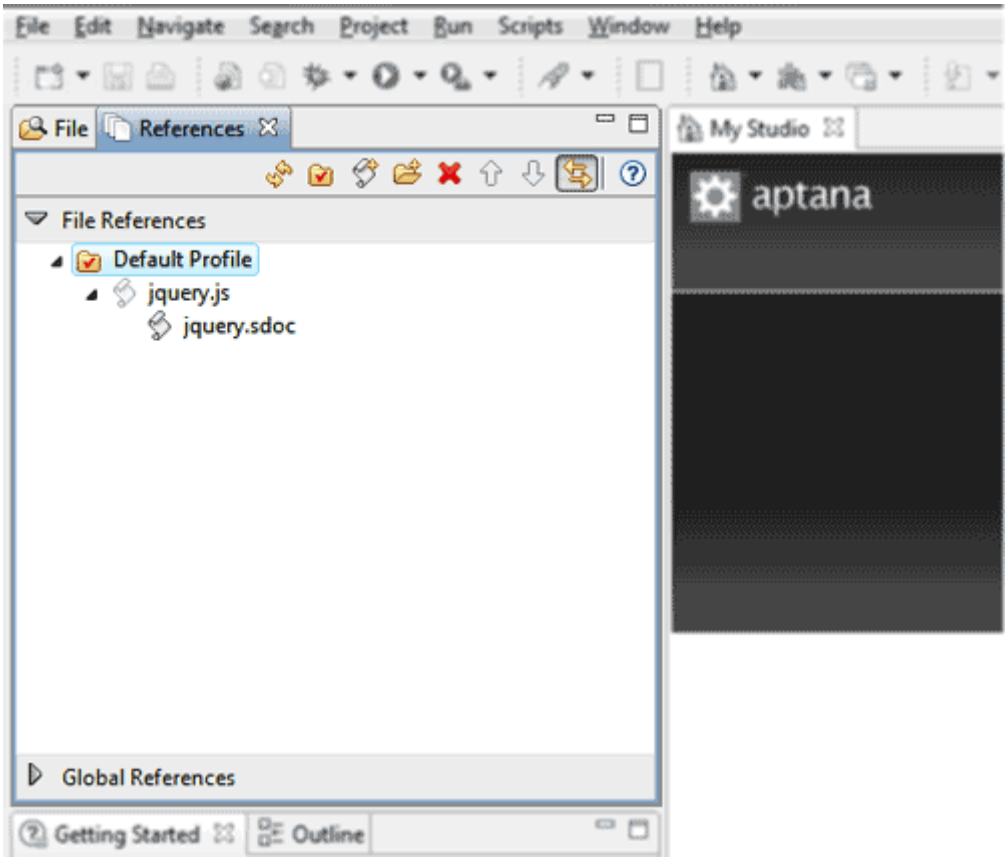
به قسمت Windows -> Show view -> Other همانند شکل زیر وارد شوید :



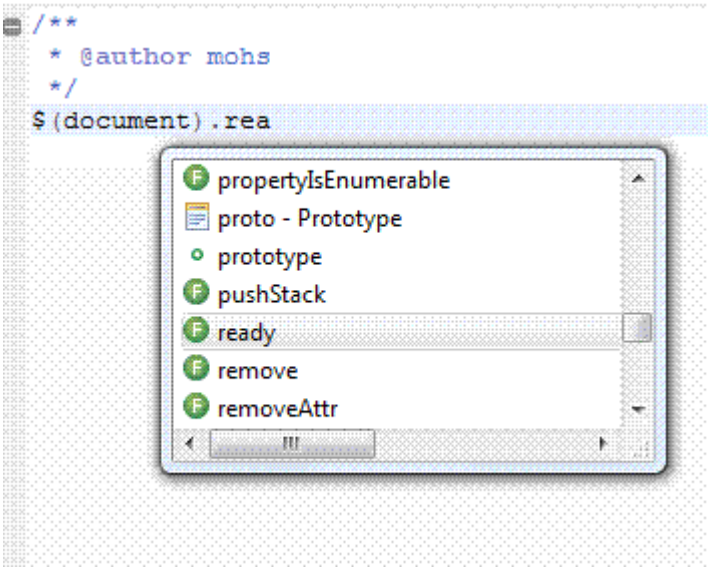
پنجره زیر نمایش میابد . مانند شکل در قسمت Aptana views روی Prefrences کلیک کنید :



در پنجره preferences روی دکمه Add کلیک کنید . به مسیر فایل کتابخانه jqury (شامل دو فایل js و sdoc) بروید و آن را به پنجره اضافه کنید :



همان طور که در تصویر فوق میبینید . این کتابخانه به Aptana اضافه شده است . حال در صورتی که یک فایل html یا js باز کنید . میتوانید درون آن هنگام نوشتن کدهای jquery از پنجره راهنما استفاده نمایید :



در صورتی که این پنجره باز نشد برای باز کردن آن میتوانید از کلید های ترکیبی Ctrl+Space استفاده نمایید .

فصل دوم : شروع آموزش

ساختار کلی دستورات :

در jquery شما تعدادی تگ html را انتخاب کرده (selector) و بر روی آنها عملیاتی (actions) انجام میدهید .

```
$(selector).action();
```

در کد فوق :

- $ همان شی jQuery است .
- Selector نام یک المنت یا مشخصات یک سری المنت برا انتخاب شدن است .
- action() هم عملی است که روی المنت های انتخابی انجام میشود .

به عنوان مثال :

$(this).hide() : عنصر جاری را مخفی میکند

$("p").hide() : عمل پنهان شدن را برای تمام تگ های "p"موجود در صفحه اعمال میکند

$("p.test").hide() : تمام تگ های p که کلاسشان "test" تعریف شده را پنهان میکند.

$("#test").hide() : تگی که آیدی آن "test" است را مخفی میکند.

نکته : همان طور که مشاهده فرمودید قسمت selector برای انتخاب مجموعه ای از عناصر از ساختار CSS تبعیت میکند . در این مورد بعدا مفصل تر توضیح داده خواهد شد .

$(document).ready :

در jquery سعی میشود تمام کد ها در رویداد **$(document).ready** اجرا شود .

```
$(document).ready(function(){

--- jQuery functions ----

});
```

این رویداد هنگامی اتفاق می افتد که صفحه جاری کاملا لود شده باشد و آماده اجرا باشد . اجرای دستورات در خارج از رویداد document.ready میتواند مشکلاتی را بوجود آورد مثلا :

- پنهای کردن (hide) عنصری که هنوز لود نشده باعث بروز خطا یا اجرا نشدن کد میشود.

- تغییر سایز یک عکسی که هنوز به طور کامل لود نشده نیز میتواند باعث ایجاد مشکل در نمایش سایت گردد.

: Jquery Selectors

توسط Selector ها شما میتوانید یک عنصر html یا تعدادی از عناصر html با یک خصوصیت مشترک را انتخاب نمایید .

برای انتخاب تگ ها میتوان از نام آنها ، خصوصیات تعریف شده آنها (مانند id یا class یا href و ...) و یا محتوی داخلی آنها استفاده کرد .

- برای انتخاب تگهای هم نام ، نام تگ را در قسمت Selecotrs مینویسیم :

$("p") : انتخاب تمام پاراگرافها

$("a") : انتخاب تمام لینک ها

- برای انتخاب تگی با یک آیدی خاص آیدی را به همراه # مینویسیم :

$("#test")

کد بالا تگی با آیدی test را انتخاب میکند .

- برای انتخاب تگی با یک خصوصیت (Attribute) آن خصوصیت و مقدارش را بین [] مینویسیم :

$("[href='http://blog.monavarian.ir'") :

کد بالا تمام المنتهایی را که دارای خصوصیت href با مقدار http://blog.monavarian.ir هست را انتخاب میکند.

نکته : میتوان به جای مساوی در کد بالا از =! و یا $= استفاده کرد :

$("[href!='#']") : انتخاب تمام المنتها که href آنها مخالف عبارت #باشد

$("[href$='.jpg']") : انتخاب تمام عناصر که عبارت داخل href آنها به jpg. ختم شود . (تمام لینک هایی که ب تصویر لینک شده اند)

- میتوانید از ترکیب این selector ها استفاده نمایید :

$("p.test") : تمام تگ های p که کلاسشان test است .

$("div#main") : تگ div که آیدی آن main است .

$("div#main .note") : تمام تگ هایی با کلاس note که داخل تگ div با آیدی main هستند .
دقت نمایید که در این مثال note. با یک فاصله از div#main درج شده .

نکته : به دو انتخاب زیر دقت نمایید :

$("p.test")

$("p .test")

ظاهرا این دو کد شبیه هم هستند (در دومی یک فاصله space بین p و test. وجود دارد) . اما اولی تگها p با کلاس test را انتخاب میکند . در حالی که دومی تمام تگهای با کلاس test که درون تگهای p قرار دارند انتخاب خواهد کرد .

- برای انتخابهای خاص یک سری نماد اضافه نیز تعریف شده است :

$("ul li:first") : اولین li از تگ های ul را انتخاب مینماید .

$("tr:even") : تمام سطر های زوج جدول .

$("tr:odd") : تمام سطر های فرد جدول .

$("input:not(:empty)") : تمام input هایی که خالی نیستند

برای دیدن مرجع Selector ها به بخش پیوست ها مراجعه نمایید .

jQuery Event Functions :

Events چیست ؟ Event یا رویداد واقعه هایی است که در یک صفحه وب اتفاق می افتد . مثلا رویداد click هنگامی که کلیکی روی عنصری زده میشود این رویداد از آن عنصر اتفاق می افتد . یا رویداد ready از عنصر document هنگامی که صفحه آماده اجرا میشود اتفاق می افتد .

چند تا از رویداد ها در jquery :

- mouseover: هنگامی که موس روی شی قرار میگیرد
- mouseout : هنگامی که موس از روی شی کنار میرود .
- Dblclick : هنگام دبل کلیک کردن روی شی اتفاق می افتد .
- Focus : هنگام فوکوس کردن روی یک شی اتفاق می افتد .

- Blur : هنگامی که شی از حالت فوکوس خارج میشود اتفاق می افتد .

jQuery Event Functions : توابعی هستند که هنگامی که یک رویداد در کد html اتفاق می افتد بطور اتوماتیک اجرا میشوند .

مثلا میتوان در jquery گفت هنگامی که روی یک selector کلیک شد تابعی اجرا شود . به عنوان مثال :

```
$("button").click(function(){
 $("p").hide();
 });
```

کد فوق بر روی رویداد کلیک تمام تگ های button تابعی اجرا میکند . این تابع تمام پاراگراف های موجود در صفحه را hide میکند . در واقع هنگام اجرای صفحه html .اگر روی هر دکمه ای که در صفحه موجود است کلیک شود ، تمام تگ های p موجود در صفحه پنهای میشود.

نکته: رویداد ها را به دو صورت در کد میتوان استفاده کرد :

- حالت اول وقتی که میخواهیم برای رویدادی تابعی بنویسیم :

```
$(selector).event(function(){ ... some code ... })
```

- حالت دوم وقتی است که میخواهیم یک رویداد را صدا بزنیم :

```
$(selector).event()
```

در این حالت رویداد مربوط به selector را صدا میزنیم .

برای دیدن مرجع رویداد ها به پیوست مراجعه نمایید

jQuery Name Conflicts :

jquery از علامت اختصاری $ به جای jQuery استفاده میکند. بعضی کتابخانه های جاوااسکریپت دیگر نیز از این نام استفاده میکنند . اگر شما در صفحه خود از کتابخانه دیگری نیز استفاده میکنید که آن کتابخانه نیز از نام $ استفاده میکند . شما با کا بر انه jQuery به مشکل بر خواهید خورد .

jQuery برای حل این مشکل تابعی دارد به نام noConflict() که توسط آن میتوانید نام جدید برای صدا زدن توابع jQuery ایجاد نمایید :

```
var jq=jQuery.noConflict()
```

دستور فوق باعث میشود تا شما نام jq را جایگزین $ نمایید .

برای اینکه jquery به بهترین حالت ممکن کار خود را انجام دهد شما باید :

- تمام کد های jquery را درون event handler ها قرار دهید .
- تمام event handler ها (توابع رویداد ها) را درون رویداد document.ready قرار دهید .
- کدهای خود را درون یک فایل js جداگانه قرار دهید و آن را درون صفحه خود فراخوانی کنید.
- اگر تابع هم نام $ دارید توسط تابع noConflict نام تابع jquery را تعویض نمایید.

یک مثال ساده :

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $('a').click(function(){
      alert('clicked');
  })
});
</script>
</head>
<body>
      <a href="http://blog.monavarian.ir" > Blog.Monavarian.ir </a>
</body>
</html>
```

در مثال بالا :

۱. ابتدا فایل کتابخانه jquery یعنی فایل jquery.js را صدا زدیم .

۲. در سطر بعدی در یک تگ script گفتیم که هنگامی که رویداد ready از شی document صدا زده شد این کار ها را انجام بده :
- تابعی روی رویداد کلیک تگ های a موجود در صفحه بنویس (این تابع فقط عبارت Clicked را بصورت پاپ آپ نمایش میدهد)

۳. در بدنه صفحه یک تگ a نوشتیم .

۴. با ذخیره فایل بالا به نام test1.html و اجرای آن در browser ، لینکی را میبینیم که با کلیک روی آن عبارت clicked بصورت popup نمایش داده می یابد .

نکته : در مثال فوق هنگامی که روی لینک کلیک میکنیم . بعد از نمایش پیغام clicked ، لینک اجرا شده و به صفحه blog.monavarian.ir منتقل خواهیم شد . اگر میخواهید تگ a رویداد پیشفرض خود را نادیده بگیرد میتوانید از تابع preventDefault() استفاده نمایید . در این حالت تکه کد jquery ما به شکل زیر تغییر میکند :

```
$(document).ready(function(){
  $('a').click(function(event){
      alert('clicked');
      event.preventDefault();
  })
});
```

فصل سوم :

jQuery Effects :

افکت ها توابعی هستند که بر روی نمایش عناصر صفحه عملیاتی انجام میدهند .

افکت های موجود : Hide, Show, Toggle, Slide, Fade , Animate

Hide & Show :

توسط دو تابع hide() و show() میتوانید عناصر صفحه را پنهان یا آشکار نمایید :

```
$("#btnhide").click(function(){
$("p").hide();
});
$("#btnshow").click(function(){
$("p").show();
});
```

در کد فوق هنگامی که روی عنصری در صفحه با آیدی btnhide کلیک شود پاراگرافهای درون صفحه (تگهای p) پنهای میشوند و با کلیک روی عنصر با آیدی btnshow کلیک شود پاراگرافها نمایش میبابند .

این دو تابع دو پارامتر speed و callback نیز می پذیرند :

$(selector).hide(speed,callback)

$(selector).show(speed,callback)

پارامتر speed میتوانید مقدارهای 'normal' , 'fast' , 'slow' و همچنین مقداری به میلی ثانیه بپذیرد .

پارامتر callback تابعی است که هنگامی که عمل show یا hide به طور کامل انجام شد صدا زده میشود .

```
$("button").click(function(){
$("p").hide(1000);
});
```

در مثال فوق با کلیک روی button پاراگراف p ظرف مدت ۱۰۰۰ میلی ثانیه پنهان میشود .

```
$("button").click(function(){
$("p").hide('slow',function(){ alert('hide is finished'); } );
});
```

در مثال فوق با کلیک روی button پاراگرافهای p بصورت آهسته پنهان میشوند و پس از پنهان شدن پیغام 'hide is finished' بر روی صفحه نمایش می یابد .

Toggle :

این افکت وضعیت عنصر مربوطه را از hide به show و بلعکس تبدیل میکند . یعنی اگر عنصر پنهان باشد با اجرای این افکت عنصر پیدا میشود و اگر پیدا باشد پنهان میشود .

```
$("#line").click(function(){
$("#lineinfo").toggle();
});
```

در کد فوق با کلیک روی عنصری با آیدی line مشخصات آن عنصر که در عنصر دیگری با آیدی lineinfo قرار دارد نمایش می یابد و با کلیک مجدد پنهان میشود .

این افکت هم همانند دو افکت قبلی دارای دو پارامتر speed و Callback می ب شد .

Slide :

$(selector).slideDown(speed,callback)

این دستور selector را با سرعت speed به سمت پایین باز میکند .

$(selector).slideUp(speed,callback)

این دستور selector را با سرعت speed به سمت بالا جمع میکند (hide)

$(selector).slideToggle(speed,callback)

این دستور نیز selector را بین دو وضعیت باز و بسته تعویض مینماید .

پارامتر های این توابع هم مانند افکت های قبلی تنظیم میشوند .

Fade :

این افکت ها نیز برای نمایش و پنهان کردن عنصر استفاده میشود با این تفاوت که نوع پنهان کردن و نمایش دادن عناصر در این دستوران متفاوت است و با تغییر شفافیت (opacity) انجام میشود.

$(selector).fadeIn(speed,callback)

این تابع عنصر selector را در صفحه نمایان میکند .

$(selector).fadeOut(speed,callback)

این تابع عنصر selector را در صفحه مخفی میکند .

$(selector).fadeTo(speed,opacity,callback)

این تابع پارامتری به عنوان opacity میپذیرد که میزان شفافیت را بین دو عدد ۰ و ۱ تعیین میکند که عدد ۱ یعنی کاملا آشکار و عدد ۰ یعنی پنهان و عنصر را بع اندازه عدد شفافیت ، شفاف میکند .

مثلا :

```
$("button").click(function(){
$("div").fadeTo("slow",0.25);
});
```

در کد فوق با کلیک روی button عنصر div درون صفحه به آرامی شفاف میشود تا به 25 درصد شفافیت برسد .

: Custom Animation

```
$(selector).animate({params},[duration],[easing],[callback])
```

پارامتر params پارامترهایی را میپذیرد که قرار است همزمان با هم بر روی selector اعمال شوند .

مثلا :

```
animate({width:"70%",opacity:0.4,marginLeft:"0.6in",fontSize:"3em"});
```

تمام پارامتر ها درون {} نوشته میشوند و با کاما ',' از هم جدا میشوند . هر پارامتر با عنوان و سپس ":" و بعد مقدار آن مشخص میشود .

پارامتر duration مانند پارامتر speed در افکت های قبلی عملی میکند که میتواند مقادیر 'slow' , 'fast' , normal و یا عددی به میلی ثانیه را در بر بگیرد .

```
<script type="text/javascript">
$(document).ready(function(){
$("#start").click(function(){
$("#box").animate({left:"100px"},"slow");
$("#box").animate({fontSize:"3em"},"slow");
});
});
</script>
```

در قطعه کد فوق وقتی روی عنصر با آیدی start کلیک میشود عنصر box# در صفحه اول ۱۰۰ پیکسل به سمت چپ رفته و سپس نوشته های داخلش به اندازه 3em خواهد شد .

میتوان خطهای ۴ و ۵ را در یک خط نوشت :

```
$("#box").animate({left:"100px", fontSize:"3em"},"slow");
```

که البته در این حالت هر دو مقدار همزمان با هم به box# اعمال خواهد شد.

نکته : تابع stop() هم برای متوقف کردن افکت در حال اجرا استفاده میشود :

```
$(selector).stop();
```

نکته مهم : خصوصیت فراخوانی زنجیره ای : میتوان متدها را بجای نوشتن در خطهای جداگانه ، پشت سر هم به یک عنصر اعمال کرد مثلا :

```
$('#box').fadeIn(1000);

$('#box').animate({left:"100px"},'slow');

$('#box').fadeOut(1000);
```

بجای کدهای فوق که همه بر روی یک عنصر اعمال میشوند میتوان نوشت :

```
$('#box').fadeIn(1000).animate({left:"100px"},'slow').fadeOut(1000);
```

سوال : چرا تابع Callback باید استفاده شود ؟

گاهی اوقات ما نیاز داریم عملی را دقیقا بعد از اعمال کامل یک افکت اجرا کنیم .

به مثال زیر توجه نمایید :

```
$("button").click(function(){
$("p").hide(1000);
alert("The paragraph is now hidden");
});
```

در این مثال ما میخواهید هنگامی که پاراگراف به طور کامل پنهان شد عبارت " the paragraph is now hidden " نمایش یابد . اما در عمل این اتفاق نمی افتد زیرا در حین انجام عمل hide خط بعدی فرمان صدا زده خواهد شد و این امر باعث تداخل خواهد شد .

اما به این مثال دقت نمایید :

```
$("button").click(function(){
$("p").hide(1000,my_alert);
});


function my_alert()
{
alert("The paragraph is now hidden");
}
```

در این مثال از تابع my_alert به عنوان تابع Callback استفاده شده . در این حالت دقیقا بعد از اتمام افکت hide تابع my_alert صدا زده میشود .

مثال : در مثال زیر قصد داریم تعدادی لینک در صفحه قرار دهیم که با رفتن موس روی آنها توضیح مختصر با افکت مناسب نمایش یابد :

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
       $('a').mouseover(function(){
            $(this).next('div.info').slideDown('slow');
       })
      $('a').mouseout(function(){
      $(this).next('div.info').slideUp('slow');
      });

});
</script>
<style>
      ul {
            list-style:none;
            margin:0;
            padding:0;
      }
      ul li {
            padding:3px;
            background:#EEE;
            border:1px solid #000000;
            width:200px;
            margin:2px;
      }

      .info {
            padding:5px;
            background:#FFF;
            color:#000;
            display:none;
      }
</style>
</head>
<body>
<ul>
      <li>
      <a href="http://blog.monavarian.ir" > Blog.Monavarian.ir </a>
              <div class="info">mohsen monavarian Weblog</div>
      </li>
      <li>
      <a href="http://share.wanted.ir" >wanted.ir </a>
        <div class="info">The First shared TextBoxes in Iran</div>

      </li>
      <li>
      <a href="http://www.res2ran.com" >Res2Ran.com </a>
        <div class="info">Shiraz restuarants And FastFoods Guide , Food
instructions</div>
```

```
      </li>
</ul>
</body>
</html>
```

همانطور که در کد بالا مشاهده میکنید در قسمت کد jquery دو تابع برای رویدادهای mouseover و mouseout تگهای a (لینکهای صفحه) نوشته شده که تابع اولی باعث نمایش تگ div.info با افکت slideDown میشود و تابع دومی نیز با افکت slideUp تگ را مخفی می کند .

شما میتوانید به جای slideDown و slideUp از افکت های دیگری که در این فصل بیان شد استفاده نمایید .

نکته : تابع next :

```
$(selector1).next(selector2);
```

این تابع به دنبال عناصر انتخابی مطابق با selecotr2 میگردد که پس از عنصر selector1 در صفحه موجود باشد . مثلا در کد بالا دنبال اولین عنصر div با کلاس info بعد از عنصر جاری (که همان تگ a انتخاب شده است) می گردد .

فصل چهارم :

Jquery و تغییر محتوای عناصر :

در jquery شما میتوانید به راحتی محتوای تگ ها و عناصر موجود در صفحه را تغییر نمایید .

برای اینکار چندین تابع در jquery وجود دارد :

```
$(selector).html(content);
```

با این دستور محتوی content را داخل عناصر Selectors قرار میدهد .

به عنوان مثال :

```
$("p#head").html(" Welcome ");
```

این قطعه کد در تگ p با آیدی head عبارت Welcome را مینویسد .

نکته : content میتوان متن ساده یا html باشد :

```
$('p').html("<h1> hello </h1>");
```

توابع دیگری نیز برای تغییر محتوا عناصر وجود دارند :

```
$(selector).append(content)
```

این تابع عبارت content را به **انتهای** محتوای Selector اضافه میکند .

```
$(selector).prepend(content)
```

این تابع عبارت content را به **ابتدای** محتوای Selector اضافه میکند .

```
$(selector).after(content)
```

این تابع عبارت content را **بعد** از تمام عناصر نظیر Selector درج میکند .

```
$(selector).before(content)
```

این تابع عبارت content را **قبل** از تمام عناصر نظیر Selector درج میکند .

نکته : دستور ()html به تنهای محتوی عنصر را برمیگرداند :

```
alert($('#box').html());
```

تابع دیگری هم برای خواندن محتوی وجود دارد به نام ()text که همانند تابع ()html عمل میکند با این تفاوت که این تابع تگ های html را حذف میکند و فقط نوشته را برمیگرداند در حالی که ()html تمام محتوی را با تگهایش برمیگرداند .

به مثال زیر توجه نمایید :

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $("button").click(function(){
alert($('#test').text());
  alert($('#test').html());
});
</script>
</head>

<body>
<p id="test"> <b> paragraph </b> </p>
<button>Click me</button>
</body>
</html>
```

پس از اجرای دستور فوق و کلیک رو دکمه اولین پنجره مقدار "paragraph" را نمایش میدهد و دومین پنجره عبارت " paragraph " نمایش داده میشود .

تغییر خصوصیات (Attribute) شی :

توسط تابع attr() میتوان به خصوصیات یک تگ html دسترسی پیدا کرد .

$(selector).attr(attribute);

خصوصیت attribute از عنصر انتخابی را بر میگرداند .

$(selector).attr(attribute,value);

به خصوصیت attribute از عناصر انتخابی مقدار value را نسبت میدهد .

مثلا :

$('a').attr('href');

مقدار خصوصیت href از اولین تگ a موجود در صفحه را بر میگرداند .

$(this).attr('id','newName');

مقدار id عنصر جاری را به newName تغییر نام میدهد.

دسترسی به value عناصر فرم ها :

$(selector).val();

$(selector).val(newValue);

توسط تابع val() میتوانیم به محتوا (value) عناصر موجود در فرم ها دسترسی و مقدارش را تغییر دهیم .

به مثال های زیر توجه نمایید :

```
$('input#name').val();
```

مقدار value عنصر input با آیدی name را بر میگرداند .

```
$('select.foo option:selected').val();
```

هنگامی که پنجره dropdown باز است مقدار انتخابی را برمیگرداند .

```
$('select.foo').val('two');
```

مقدار انتخاب شده عنصر select با کلاس foo را روی مقدار two تنظیم میکند.

```
$('input:checkbox:checked').val();
```

مقدار چک باکس انتخاب شده را بر میگرداند .

```
$('input:radio[name=bar]:checked').val();
```

مقدار انتخاب شده از دکمه های رادیویی در گروه با نام bar را بر میگرداند.

مثال :

```
<html>
<head>
  <script src="jquery.js"></script>
  <script>
  $(document).ready(function(){
    $("input:button").click(function () {
      var text = $(this).val();
      $("input:text").val(text);
    });
 })
</script>
</head>
<body>
      <div>
    <input type="button" value="First" />
    <input type="button" value="Secont" />
    <input type="button" value="third" />
  </div>
  <input type="text" value="click a button" />
</body>
</html>
```

پس از اجرا با کلیک روی هر دکمه مقدار value آن دکمه ها در تکست باکس نمایش می یابد .

فصل پنجم :

تغییر استایل در جی کوئری :

توسط jquery میتوانید مستقیما استایل یک شی را در صفحه تغییر دهید . برای تغییر استایل چند تابع مهم وجود دارد :

- $(selector).css(name,value)
- $(selector).css({properties})
- $(selector).css(name)
- $(selector).height(value)
- $(selector).width(value)

توسط تابع css به سه روش فوق میتوان مقدار استایل عناصر را تغییر داد .

$(selector).css(name,value)

در این دستور مقدار value را برای خصوصیت name در تمام عناصر html مط بق با selector تنظیم میکند.

```
$("p").css("background-color","yellow");
```

عبارت بالا رنگ پیش زمینه همه پاراگراف های موجود در صفحه را به زرد تبدیل میکند.

$(selector).css({properties}) :

این دستور برای تنظیم چندین خصوصیت css برای عناصر انتخابی می با د . به عنوان مثال :

```
$("p").css({"background-color":"yellow","font-size":"200%"});
```

عبارت فوق رنگ پیش زمینه پاراگرافها را زرد و اندازه نوشته آنها را دوبرابر میکند. همانطور که میبینید هر خوصوصیت با یک کاما "," از خصوصیت بعدی جدا میشود و همه خصوصیت ها بین { و } قرار گرفته اند.

$(selector).css(name) :

این دستور مقدار خصوصیت name از عنصر selector را برمیگرداند . مثلا :

```
$(this).css("color");
```

عبارت فوق مقدار رنگ نوشته عنصر جاری را برمیگرداند .

- $(selector).height(value)
- $(selector).width(value)

دو تابع width و height به ترتیب برای تغییر اندازه عرض و ارتفاع selector ها استفاده میشود :

```
$("#boxinfo").height("200px");
```

عبارت فوق مقدار ارتفاق عنصری با آیدی boxinfo در صفحه را ۲۰۰ پیکسل میکند.

```
$(this).width("200px").height("200px");
```

عبارت فوق اندازه عرض و ارتفاع عنصر جاری را برابر ۲۰۰پیکسل میکند.

نکته : دو تابع width و height در صورتی که بدون پارامتر استفاده شوند به ترتیب مقدار عرض و ارتفاع عنصر انتخابی را بر میگردانند.

افزودن و حذف کلاس :

$(selector).addClass(classname);

$(selector).removeClass(classname);

توسط دو دستور فوق میتوان یک کلاس css را به عناصر انتخابی اضافه نمود و یا اگر عناصری دارای کلاسی هستند آن کلاس را حذف کرد .

مثال :

```
<html>
<head>
<script src="jquery.js" type="text/javascript" ></script>
<script type="text/javascript">
$(document).ready(function(){
  $('button').mouseover(function(){
      $('p').addClass('info');
  })
  $('button').mouseout(function(){
      $('p').removeClass('info');
  })

})
</script>
<title>Example 1</title>
<style>
      .info {
            background:#000;
            color:#FFF;
      }

</style>
</head>

<body>
      <button>move mouse to add or remove </button>
      <p>paragraph 1</p>
      <p class="info"> paragraph 2 whit class info</p>
</body>
```

در کد فوق پس از اجرا با بردن موس روی دکمه کلاس info به پاراگراف ها اعمال میشود و با برن موس به خارج از دکمه کلاس info حذف میشود.

نکته : توسط دستور $(selector).toggleClass(content) میتوان به عنصری کلاسی را اضافه کرد و در صورتی که عنصر از این کلاس استفاده میکند آن را حذف کرد . (تغییر وضعیت بین اضافه و حذف کلاس content برای عنصر selector) .

hasClass(classname) :

مشخص میکند که آیا عنصر دارای کلاس classname هست یا خیر .

```
if ( $('#mydiv').hasClass('info') )
        alert('mydiv has class info');
```

position() :

$(selector).position()

این تابع فاصله selector را از بالا و چپ صفحه مشخص میکند . برای دسترسی به فاصله بالا از top. و برای دسترسی به فاصله چپ از left. استفاده میشود :

```
var pos = $('#mydiv').position();
var left = pos.left;
var top = pos.top;
```

فصل ششم :

jQuery و ایجاکس :

Jquery کتابخانه ای بسیار کارآمد برای کار با تکنولوژی Ajax در خود دارد که در این فصل به معرفی آن خواهیم پرداخت .

قبل از شروع باید بدانیم Ajax چیست ؟

Ajax مختصر شده عبارت Asynchronous JavaScript and XML میباشد . این تکنولوژی اولین بار برای استفاده از داده های Xml در جاوا اسکریپت ایجاد شد.

Ajax یک زبان برنامه نویسی نیست . بلکه تکنولوژی برای ارتباط با سرور از طریق جاوااسکریپت و ایجاد صفحات داینامیک میباشد .

هسته اصلی ایجاکس شیی به نام XMLHttpRequest می ب شد .

بطور مختصر : ایجاکس ارتباط غیر مستقیم و تبادل اطلاعات با وب سرور است بطوریکه همه چیز در بک گراند اتفاق بیافتد و نتیجه فقط در قسمتی از صفحه نمایش یابد بطوریکه کل صفحه نیاز به بازنگری (رفرش) نداشته باشد .

ایجاکس در jquery :

در jquery توابعی برای کار با ایجاکس وجود دارد که کار را با این تکنولوژی بسیار آسان نموده است . توسط این توابع میتوان تبادل اطلاعات را با سرور بصورت XML , HTML , TXT و JASON و با استفاده از دو متد GET و POST انجام داد .

و شما میتوانید اطلاعات نتیجه که از سرور بدست می آید را در عنصر انتخابی (selector) به نمایش در آورید .

در ادامه به بررسی این توابع خواهیم پرداخت :

تابع load :

```
$(selector).load(url,data,callback)
```

این تابع فایل موجود در آدرس url را خوانده و پس از اجرای کامل نتیجه را در عنصر انتخابی (Selector) نمایش میدهد .

یک مثال ساده :

فرض کنید فایلی با نام loadtxt.txt در پوشه files سرور دارید و میخواهید با کلیک روی دکمه ای آن را در صفحه به نمایش در آورید :

```
<html>
<head>
```

```
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $("button").click(function(){
    $('#result').load('files/loadtxt.txt');
  });
});
</script>
</head>
<body>
<button>Click to Load Data</button>
<div id="result"></div>
</body>

</html>
```

پس از اجرای این برنامه با کلیک روی دکمه محتوی موجود در فایل تکست در تگ div نم یش می یابد .

تابع load دو آرگومان دیگر دارد . آرگومان data : زمانی که شما میخواهید یک اسکریپ زبان سرور را (فایلهای php یا asp یا aspx و ...) از سرور بخوانید و باید دیتایی را برای پردازش به آن اسکریپت بفرستید توسط این آرگومان این کار را انجام میدهید .

آرگومان Callback نیز تابعی است که هنگامی که عملیات لود به طور کامل انجام شد اجرا خواهد شد .

مثال :

فرض کنید شما در پوشه files در سرور فایلی به نام loadname.php دارید که محتوای آن بصورت زیر است :

```
<?php

$name = $_REQUEST['name'];

echo 'Welcome '.$name;

?>
```

کار این برنامه این است که نامی را به عنوان داده دریافت میکند و پیام خوش آمد را نمایش میدهد .

حال میخواهید برنامه ای بنویسید که کاربری در جعبه Text یک نام وارد نماید و پس از کلیک روی دکمه . نام به فایل فوق فرستاده شود و نتیجه نمایش داده شود و پس از اتمام کار پیغامی ن ایش دهد :

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $("button").click(function(){
```

```
      var inputname = $('input#name').val();
    $('#result').load('files/loadname.php',{name:inputname},finish_load);
  });
});

function finish_load()
{
      alert('load finish');
}
</script>
</head>
<body>
Please Enter Your Name :
<input type="text" id="name" />
<button>Click to Load Data</button>
<dir id="result"></dir>
</body>

</html>
```

همان طور که در کد فوق میبینید پس از کلیک روی دکمه button ابتدا توسط تابع val مقدار موجود در textbox را خوانده و در متغیر inputnum قرار میدهیم . سپس با استفاده از دستور load این مقدار را به فایل loadname.php می فرستیم و در پایان هنگامیکه پاسخ از سرور داده شد و پیام خوش آمد نمایش یافت تابع finish_load اجرا میشود و عبارت "load finish" بر روی صفحه نمایش داده خواهد شد .

نکته ۱ : تابع val() : این تابع مقدار یک عنصر فرم را برمیگرداند و در صورتی که مقدار داشته باشد آن مقدار را به خصوصیت value عنصر انتخابی selector نصبت میدهد.

```
$(selector).val(content);
```

نکته ۲ : در صورتی که چند داده را میخواهید به عنوان data به سرور ارسال نمایید آنها را داخل { و } قرار داده و هر داده را با کاما "," از داده بعدی جدا نمایید :

```
{var1:value1,var2:value2,var3:value3}
```

تابع $.ajax(option) :

سطح پایین ترین تابع کار با ایجاکس در جی کوئری تابع ajax می ب شد . برای استفاده از این تابع باید تمام پارامتر های ارسال را توسط آرگومان option به سرور ارسال نماییم .

مثال ساده :

```
$.ajax({
  url: 'ajax/test.html',
  success: function(data) {
    $('.result').html(data);
    alert('Load was performed.');
  }
});
```

در این مثال تابع ajax فایل test.html را از سرور فراخوانی کرده و نتیجه را در عنصری با کلاس result نمایش میدهد .

کار با این تابع به نسبت سایر توابعی که معرفی خواهد شد دشوار تر میباشد . بنا براین پیشنهاد میشود از این تابع فقط در مواقع نیاز استفاده شود و به جای آن از توابع ساده شده استفاده گردد .

تابع : $.get و $.post :

$.post(url,data,callback,type);

$.get(url,data,callback,type);

این دو تابع همان طور که از نامشان پیداست با دو متد مختلف post و get اسکریپت url را از سرور فراخوانی نموده و نتیجه را توسط تابع callback برمیگردانند.

- url : آدرس اسکریپتی که سمت سرور باید اجرا شود
- Data : داده هایی که به سرور فرستاده میشود : {name:value,name:value,...}
- Callback : تابعی است که هنگامی که داده ها از سرور به طور کامل بارگزاری شد صدا زده میشود .
- Type : نوع داده های خروجی از سرور را مشخص میکند و میتواند مقادیر (html,xml,json,jasonp,script,text) باشد

تنها تفاوت این دو تابع در نوع ارسال داده به سرور است که اولی از متد post استفاده مینماید و دومی از متد get . بنابراین در صورتی که میخواهید فقط فایلی را از سرور بازخوانی نمایید و قصد ارسال داده به سرور را ندارید، بهتر است از تابع get استفاده نمایید . (و یا تابع load) .

(اگر در مورد این دو متد آشنایی ندارید به آموزش زبانهای برنامه نویسی تحت سرور مانند php یا asp مراجعه نمایید).

مثال :

مثالی که در بالا برای تابع ajax نوشته شد را با تابع get بصورت زیر میتوان نوشت :

```
$.get('ajax/test.html', function(data) {
  $('.result').html(data);
  alert('Load was performed.');
});
```

و یا مثالی دیگر : در این مثال میخواهیم همان مثالی را که در ابتدای فصل برای تابع load نوشتیم با تابع post بنویسیم . در این صورت قسمت کد ما به شکل زیر تغییر خواهد کرد :

```
$(document).ready(function(){
  $("button").click(function(){
     var inputname = $('input#name').val();
    $.post('files/loadname.php',{name:inputname},function(Data){
                        $('#result').html(Data);
                  finish_load();
```

```
                });
  });
});
```

همان طور که در مثال میبینید . تابع callback با مقدار data پس از فراخوانی کامل loadname.php صدا زده میشود و در این تابع مقدار data که همان نتیجه اجرای اسکریپت loadname.php است توسط دستور html(data) در عنصر با آیدی result نمای م یابد .

تابع getScript :

$.getScript(url,callback)

تابع getScript ، برای فراخوانی و اجرای یک فایل جاوااسکریپت (js) به کار برده میشود .

این تابع ساده شده تابع ajax با فرمت زیر می باشد :

```
$.ajax({
  url: url,
  dataType: 'script',
  success: success
});
```

مثال :

```
$.getScript("test.js");
```

این مثال باعث فراخوانی و اجرای فایل test.js میشود .

```
$.getScript("http://blog.monavarian.ir/test.js", function(){
   alert("Script loaded and executed.");
 });
```

در مثال فوق فایل test.js در آدرس blog.monavarian.ir فراخوانی میشود و در صورتی که فراخوانی با موفقیت انجام شود پیغام مناسب نمایش می یابد .

نکته : میتوانید از این تابع هنگامی استفاده نمایید که میخواهید متغیر های سراسری را در یک فایل جداگانه قرار دهید و بعد در برنامه ها از آن استفاده نمایید .

تابع getJSON :

```
$.getJSON(url,data,callback);
```

این تابع برای فراخوانی فایل با فرمت Jason از سرور می باشد .

توی پرانتز (مختصری درباره Jason) :
فایلهای Jason ، فایلهای داده ای هستند که داده ها را با فرمت خاصی ذخیره میکنند . مانند فایلهای xml
فایلهای Jason نیز دارای یک فرمت خاص برای ذخیره اطلاعات می باشند .
ساختار هر شی jason به صورت روبرو می باشد :
{string : value , string:value , ... }
Value ها میتوانند شامل مقادیر null , false , true , array , number , object , string باشند .

ساختار Array در jason نیز به فرم زیر می باشد :

[item1,item2,item3,...]

هر item نیز خود میتواند شامل مقادیر معرفی شده در بالا باشد .

برای کسب اطلاعات بیشتر به سایت مرجع Jason به آدرس http://www.json.org مراجعه نمایید .

یک مثال کل با post : در این مثال قصد داریم فرمی را توسط توابع jquery بررسی کرده و سپس توسط تابع post به سرور فرستاده و نتیجه را نمایش دهیم :
برای مثال میخواهیم یک فرم ساده ورود ایجاد نماییم . فایل اسکریپت forms.php بررسی میکند که نام کاربری و کلمه عبور داده شده در لیست کاربران موجود است . در صورتی که موجود باشد مقدار ۱ و در صورتی که موجود نباشد مقدار صفر را بر میگرداند . محتوی این فایل به صورت زیر است :

```
<?php
$users = array (
'admin' => 'demo',
'test' => 'test',
'user' => 'pass',
);

$username = trim($_POST['username']);
$pass = trim($_POST['password']);

if ( array_search($pass,$users) == $username )
  echo '1';
else
  echo '0';
?>
```

این فایل را در پوشه files ذخیره میکنیم .
در فایل اصلی html :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<script src="jquery.js" type="text/javascript" ></script>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script type="text/javascript">
$(document).ready(function(){

  $('input').focus(function(){
      $('input').removeClass('select');
      $(this).addClass('select');
  });

  $('form').submit(function(event){
      event.preventDefault();
      var user = $('input#username').val();
      var pass = $('input#pass').val();
      if (!user)
      {
            $('.result').html('please Enter username');
            $('input#username').focus();
      }
      else if(!pass)
      {
            $('.result').html('please Enter password');
```

```
            $('input#pass').focus();
      }
      else
      {
         $.post('files/forms.php',{username:user,password:pass},function(data){
            if (data == '1')
            {
                  $('.result').html('Login Success <br/> welcome '+user);
                  $('form').hide();

            }
            else if (data == '0')
               $('.result').html('username or password is not correct')
            else
              $('.result').html('can not connect to server')

         })
      }
      return false;
  })
})
</script>
<title>Form Validation</title>
<style>
.result {
      color:#FF0000;
}
.select {
      background:#fff;
      border:2px solid #0000ff;
 }
</style>
</head>

<body>
<div class="result" > </div>
<form action="#" >
      <input type="text" name="username" id="username" />
      <input  type="password" name="pass" id="pass" />
      <input  type="submit" value="Login"/>
</form>

</body>
</html>
```

همان طور که در کد بالا مشاهده می کنید . در ابتدا هنگام submit فرم بررسی میکند و در صورت خالی بودن فیلدهای username و یا pass پیغام مناسب درج میکند و سپس نشانگر را به فیلد خالی هدایت می کند (focus) .
در صورتی که username و pass وارد شده باشد . این مقادیر را به فایل forms.php میفرستد و در نتیجه بررسی میکند : در صورتی که نتیجه فرستاده شده از سرور ۱ باشد پیغام خوش آمد نمایش میدهد و فرم را مخفی (hide) می نماید در غیر این صورت پیغام مناسب را درج می نماید.

فصل هفتم :

پیمایش عناصر html : (Tree Traversal) :

همانطور که میدانید هر صفحه html از تعدادی تگهای تو در تو تشکیل شده است . هر تگ فرزند تگ بیرونی و پدر تگ های درونی اش می باشد . همچنین به تگ ها ی کنار هم در یک سطح نیز تگ های همسایه گفته میشود .

برای دسترسی به عناصر صفحه علاوه بر انتخاب خصوصیت مشترکی از آنها میتوان از تعاریف مربوط به پیمایش درختی آنها استفاده نمود .

به عنوان مثال در کد روبرو :

```
<html>
  <head>
      <title> test page </title>
  </head>
  <body >
     <div>
          <ul>
              <li> one </li>
              <li> two </li>
               <li> three </li>
          </ul>
    </div>
  </body>
</html>
```

در مثال بالا :

- همه تگ ها فرزند تگ html هستند
- تگ head پدر تگ title و در نتیجه تگ title فرزند تگ head می ب شد .
- تگ ها li همسایه یا هم نژاد (برادر یا خواهر) یکدیگر هستند و همگی فرزند ul هستند و همچنین در سطح بالاتر فرزند تگ div و body و html .
- تگ ul نیز فرزند div و body و html است .
- تگ body همسایه تگ head و فرزند html است .

با توجه به تعاریف فوق . توابعی در jQuery برای پیمایش درختی عناصر html ایجاد شده که در این فصل آنها را شرح خواهیم داد :

$(selector).children([selector]) :
$(selector).parent([selector]);

تابع children() فرزندهای عنصر انتخابی را بر میگرداند . به عنوان مثال :

```
$('ul').children().css('background','Red');
```

دستور بالا باعث میشود رنگ پیش زمینه تمام تگهای درون تگ ul (فرزند های ul یا همان تگهای li) قرمز شود .

همچنین شما میتوانید از آرگومان selector استفاده نمایید تا فرزند های خاصی از تگها را انتخاب نمایید .

به مثال زیر توجه نمایید :

```
<div>
    <span>Hello</span>
    <p class="selected">Hello Again</p>
    <div class="selected">And Again</div>

    <p>And One Last Time</p>
</div>
```

اگر در این مثال کد زیر را اجرا کنیم :

```
$("div").children(".selected").css("color", "blue");
```

در این مثال فرزندهایی از تگهای div انتخاب میکند که دارای کلاس selected باشد بنابر این نتیجه دستور فوق :

Hello
Hello Again
And Again
And One Last Time

تابع parent() پدر عناصر انتخابی را بر میگرداند (اولین پدر) .

```
$('li').parent().addClass('listExpander');
```

در این مثال به اولین والد عناصر li در صفحه کلاس listExpander را اعمال میکند .
همانند تابع children() این تابع نیز میتواند مقدار selector را قبول کند :

```
<div class="one">
   <div class="two">
         <span class="text" > click </span>
  </div>
<div>
```

در مثال فوق :

```
$('span').parent() :
```

به div با کلاس two اشاره میکند . در صورتی که

```
$('span').parent('.one') :
```

به div با کلاس one اشاره میکند .

نکته : دقت نمایید که تابع parent() فقط یک عنصر را به عنوان پدر بر میگرداند . اگر بخواهید همه والدهای یک عنصر را برگردانید میتوانید از تابع parents() استفاده کنید :

```
$('span').parents() :
```

در این مثال تمام والد های span (یعنی هم div با کلاس two و هم div با کلاس one) را بر میگرداند .

تابع : parentsUntil() :

```
$(selector).parentsUntil([parent_selector]);
```

(کلمه until به م نی "تا اینکه " می باشد و همانطور که از معنی این عبارت مشخص است تابع فوق تمامی والدهای عنصر selector را تا رسیدن به عنصر parent_selector بر میگرداند .
به عنوان مثال اگر در مثال اول این فصل کد زیر را بنویسیم :

```
$('li').parentsUntil('body').addClass('test');
```

به تمام تگ های والد li تا تگ body (تگهای ul و div) کلاس test را اضافه میکند .

تابع next() و prev() :

$(selector).next([selector]);
$(selector).prev([selector]);

این دو تابع هم همانطور که از نامشان پیداست برای بدست آوردن عنصر بعدی (next) و عنصر قبلی (prev) استفاده میشوند :

```
<ul>
   <li>list item 1</li>
   <li>list item 2</li>
   <li class="third-item">list item 3</li>
   <li>list item 4</li>
   <li class="test">list item 5</li>
</ul>
```

در مثال فوق با اجرای کد :

```
$('li.third-item').next().css('color', 'red');
```

کد فوق باعث میشود عبارت **list item 4** به رنگ قرمز نمایش داده شود .

```
$('li.third-item').next().css('color', 'blue');
```
این کد عبارت list item 2 را به رنگ آبی نمایش میدهد .

```
$('li.third-item').next('.test').css('background-color', 'blue');
```
این کد عبارت list item 5 را با پیش زمینه آبی نمایش میدهد .

```
$('li.third-item').next('.test').prev().css('background-color', 'Red');
```
این کد عبارت list item 4 را با پیش زمینه قرمز نمایش میدهد .

نکته : توابع زیر نیز برای پیمایش بهتر عناصر قبلی و بعدی در jquery ایجاد شده اند :

$(selector).nextAll([nextselector])

$(selector).prevAll([prevselector])

تمام عناصر بعد next و قبل prev از selector را بر میگرداند (مط بق با nextselector یا prevselector)

$(selector).nextUntil([nextselector])

$(selector).prevUntil([prevselector])

تمام عناصر بعد next و قبل prev از selector را تا عنصر nextselector و prevselector بر میگرداند .

تابع siblings() :

$(selector).siblings([sibling_selector])

این تابع هم نژاد های selector را بر میگرداند . و در صورتی که sibling_selector مشخص شده باشد مقادیری که با این عناصر مطابق باشند بر می گرداند .

```
<!DOCTYPE html>
<html>
<head>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
<body >
   <ul>
        <li > one </li>
         <li class="test" > two </li>
         <li class="test2" > three </li>
   </ul>

<script>
      $("li.test").siblings().css("background", "yellow");
</script>
</body>
</html>
```

در مثال بالا به غیر از خود li با کلاس test ، رنگ پیش زمینه زرد به سایر li ها اضافه میشود .

در صورتیکه کد بالا را به شکل زیر تغییر دهیم فقط li هایی انتخاب میشوند که دارای کلاس test2 باشند :

```
$('li.test').siblings('.test2').css('background','yellow');
```

تابع find() :

$(selector).find(find_selector)

تابه find در فرزندهای selector به دنبال find_selector میگردد و آن عنصر را بر میگرداند .

این تابع شبیه تابع children() عمل میکند و تنها تفاوت آن این است که تابع children فقط یک سطح فرزند را جستجو میکند . همچنین در تابع find(selector) پارامتر selector الزامی است (میتوان برای انتخاب همه از عبارت '*' استفاده کرد) :

```
$(selector).find('*')
```

مثال :

```
<ul class="level-1">
  <li class="item-i">I</li>
  <li class="item-ii">II
    <ul class="level-2">
      <li class="item-a">A</li>
      <li class="item-b">B
        <ul class="level-3">
          <li class="item-1">1</li>
          <li class="item-2">2</li>
          <li class="item-3">3</li>
        </ul>
      </li>
      <li class="item-c">C</li>
    </ul>
  </li>
  <li class="item-iii">III</li>
</ul>
```

اگر در مثال فوق کد زیر را بنویسیم :

```
$('li.item-ii').find('li').css('background-color', 'red');
```

این کد رنگ پیش زمینه عبارت های A , B , 1 ,2,3, C را قرمز میکند . اما همین عمل با تابع children تنها بر روی عبارات A,B,C اعمال میشود .

تابع closest() :

$(selector).closest(closest_selector)

این تابع هم مانند parents() عمل میکند . اما تفاوت هایی بین این دو تابع وجود دارد :

- تابه closest از خود عنصر انتخابی پیمایش را شروع میکند . در صورتی که تابع parents از اولین والد شروع به پیمایش میکند .
- تابع closest عملیات پیمایش را تا رسیدن به عنصر انتخابی (closest_selector) انجام میدهد در صورتی که در تابع parents ابتدا عملیات تا انتها (root) انجام میشود . سپس همه عناصر انتخاب شده در یک متغیر temp ریخته شده و سپس آنهایی که مطابق با شرط (عنصر selector) می باشد را از بین آنها انتخاب میکند . بنابراین سرعت اجرای تابع closest بیشتر است .
- تابع closest میتواند صفر یا یک عنصر بر گرداند در صورتی که تابع parents صفر یا یک یا چند عنصر بر میگرداند .

در مثال بالا :

```
$('li.item-a').closest('ul').css('background-color','red')
```

باعث میشود رنگ پیش زمینه ul با کلاس level-2 قرمز شود .

مرجع فصل : http://api.jquery.com/category/traversing/tree-traversal

فصل هشتم

افزونه ها (پلاگین ها Plugins) :

برخی طراحان وب علاوه بر کتابخانه توابع jquery برای مرتفع ساختن نیازهای خاص خود و دیگران ، اقدام به نوشتن متدها و توابعی بر روی jquery کرده اند و این توابع را در قالب فایلهای جاوااسکریپت (js) مجزا ارائه نموده اند .
به عنوان مثال سایت http://apycom.com/menus/1-white-smoke.html پلاگینی ارائه میدهد برای ساخت منو با jquery .
هر پلاگین ممکن است شامل یک یا چند فایل جاوااسکریپت و فایلهای مرتبط (مانند تصویر یا استایل یا ..) باشد . همچنین برای استفاده هر پلاگین باید به آموزش نصب آن پلاگین مراجعه نمود . اما یک نکته را همیشه باید رعایت کرد و آن این است که فایل جاوااسکریپت پلاگین همیشه باید بعد از فایل jquery.js فرا خوانی شود .
برای دریافت پلاگین های jquery به مرجع آن در آدرس http://plugins.jquery.com مراجعه نمایید .

ایجاد یک پلاگین برای jquery :

برای ایجاد پلاگین ساده ترین راه این است که متدهای مورد نظر را توسط شی query.fn ایجاد کرده و آن ها را در قالب یک فایل js ذخیره نمود . همچنین برای تعریف توابع آنها را میتوان توسط شی jQuery ایجاد نمود.
یک مثال ساده :
در این مثال پلاگینی ساده برای کنترل خطا ایجاد شده :

```
jQuery.fn.debug = function() {
  return this.each(function(){
    alert(this);
  });
};
jQuery.log = function(message) {
  if(window.console) {
     console.debug(message);
  } else {
     alert(message);
  }
};
```

کد فوق را با نام jquery.debug.js ذخیره میکنیم .
در کد بر نامه بعد از فایل jquery.js این فایل را فراخوانی می کنیم :

```
<script type="text/javascript"  src="jquery.js"></script>

<script type="text/javascript"  src="query.debug.js"></script>
```

حال میتوانیم از این توابع استفاده نماییم :

```
$("div p").debug();
```

و یا در این عبارت :

```
try {
  // do some error prone stuff
} catch(exception) {
  $.log(exception);
}
```

چند نکته مهم برای نوشتن پلاگین مد نظر داشته باشید :

- نام فایل پلاگین را با فرمت jquery.pluginName.js نامگذاری نمایید مانند : jquery.debug.js
- تمام متد ها در jquery.fn نوشته شود و تمام توابع در jquery (مانند مثال فوق)
- در داخل متد ها متغیر this به شی jquery (همان عناصر انتخابی که تابع برایشان اجرا شده) اشاره میکند . به عنوان مثال :

```
$('p').debug();
```

در داخل تابع debug متغیر this به "p" اشاره می کند.

- تمام متد ها و توابع باید با علامت سیمی کالن ";" تمام شوند . در غیر اینصورت هنگام فشرده سازی کد ها با مشکل مواجه میشوند .
- مقدار بازگشتی متدهای نوشته شده باید از جنس شی jquery باشند مگر بطور صریح مشخص شده باشند .
- در صورتی که میخواهید متد شما برای تمام عناصر انتخابی اجرا گردد از رویداد **this.each** برای نوشتن رویداد برای عناصر انتخابی استفاده کنید
- همیشه برای سربارگزاری تابع از jquery.fn به جای $.fn استفاده نمایید . زیرا ممکن است برخی از کاربران توسط تابع ()noConflict از نام مستعار دیگری بجای $ استفاده نموده باشند و در این صورت نمیتوانند از کد شما استفاده کنند . البته میتوانید در داخل توابع تان از $ استفاده نمایید. (در مورد تابع noConflict در فصل اول توضیح داده شده است) .
- راه دیگر بجای استفاده از تابع jQuery به جای $. قرار دادن همه کدهای پلاگین در یک بلاک به فرمت زیر است

```
(function($) { .... })(jQuery);
```

این کار چند فایده دارد . اول اینکه کدها محافظت میشوند و در صورتی که تابعی هم نام با توابع استفاده شده در پلاگین داشته باشید به مشکل بر نمیخورید و همچنین فایده دیگر این کار این است که میتوانید توابع و متغیر ها را از دسترس سایر برنامه ها پنهان کنید و در نتیجه از تغییرات احتمالی آنها جلوگیری نمایید . پس توصیه میشود کدهای پلاگین خود را در این بلاک تعریف کنید .
مثلا :

```
(function($) {

  var   x,y,z;

})(jQuery);
```

در این مثال متغیر های x ,y ,z متغیرهای سراسری نیستند و فقط داخل همین بلاک قابل استفاده اند .

همین مطالب فوق برای نوشتن یک پلاگین ساده کفایت میکند . شما میتوانید با استفاده از همین قوانین اقدام به نوشتن پلاگین و استفاده از آن در وبسایت خود نمایید .

حال بهتر است پله پله آموزش ایجاد پلاگین را پیش ببریم :

تفاوت متد (**Method**) و تابع (**function**) در jQuery :

به مثال بالا دقت نمایید . در مثال بالا یک متد و یک تابع ایجاد شده است . نحوه ایجاد توابع در jQuey به فرمت زیر می باشد :

```
jQuery.funcName = function(a,b,...) { ... } ;
```

و تعریف متد به صورت زیر می باشد :

```
jQuery.fn.MethodName= function(options,...){ ...  return this; } ;
```

برای صدا زدن توابه از عبارت $.funcName(a,b,...) استفاده مینماییم در صورتی که برای صدا زدن متد ها از عبارت $(selector).methodName(options) .
همان طور که قبلا بیان شد متد ها حتما باید مقدار برگشتی از جنس شی jQuery داشته باشند . این بدلیل خصوصیت فراخوانی زنجیره ای متد هاست (هر متد شی نتیجه را بر میگرداند و متد بعدی روی نتیجه عملیات جدیدی انجام میدهد و مجددا شی را بر میگرداند و این عمل بصورت زنجیره وار تا پایان زنجیره ادامه پیدا میکند) در صورتی که در توابع امکان فراخوانی زنجیره ای وجود ندارد و میتواند مقدار برگشتی نداشته باشد .

استفاده از رویداد this.each :

اگر میخواهید متد شما برای تمام عناصر انتخابی (selectors) اجرا گردد از روش زیر استفاده نمایید .

```
jQuery.fn.newMethod = function(){
    return this.each(function(){
        // کدهای شما ...
    });
};
```

همان طور که میبینید از عبارت return this.each استفاده شده که در این حالت پس از اجرای تابع رویداد each ، شی جاری را نیز بر می گرداند .

نوشتن چندین تابع در یک پلاگین :

فرض کنید شما قصد نوشتن چندین متد را دارید :

```
jQuery.logError = function() { ... };
jQuery.logWarning = function() { ... };
jQuery.logDebug = function() { ... };
```

شما میتوانید این توابع را در یک شی قرار دهید :

```
jQuery.log = {
  error : function() { ... },
  warning : function() { ... },
  debug : function() { ... }
};
```

تعیین پارامتر برای متد ها : (پارامتر options) :

شما میتوانید برای متد ها پارامتر های ثابت و یا options بنویسید . پارامتر options در jquery کمک میکند تا بجای نوشتن تعداد زیادی پارامتر برای تابع آنها را با فرمت options بنویسیم و هر کدام را که خواستیم در زمان صدا زدن متد ، تغییر دهیم . ساختار options به صورت روبرو می باشد :

{ parametr1 : value , parametr2:value , ... }

حال در تعریف کد :

```
$.fn.newMethod = function(options) {

  var opt = {
            name : 'default' ,
            size : 5 ,
            flag : true
           };

}
```

همان طور که در مثال میبینید متغیری برای تعریف پیش فرض های options به نام opt ایجاد نمودیم . حال باید راهی پیدا شود که این متغیر را به options نسبت دهیم . تابعی در jquery به نام extend وجود دارد که کار توسعه کدها برای jquery را انجام میدهد . با این تابع در ایجاد پلاگین کارهای زیادتری داریم اما در اینجا برای توسعه مقادیر پیش فرض options استفاده میشود :

```
var setting = $.extend(opt,options);
```

در حقیقت تکه کد بالا را میتوان بطور کامل بصورت زیر نوشت :

```
$.fn.newMethod = function(options) {

  var opt = {
            name : 'default' ,
            size : 5 ,
            flag : true
```

```
        };
var setting = $.extend(opt,options);
}
```

jquery.extend :

همانطور که در مثال بالا از extend برای تعمیم متغیرهای options استفاده کردیم . میتوانیم از این تابع برای تعمیم دادن متد ها و یا حتی تعریف selector های جدید استفاده کرد .
در تعریف پلاگین ها برای اینکه چند متد هم شکل را تعریف نماییم میتوانیم از دستور زیر استفاده کنیم :

```
jQuery.fn.extend({
  check       : function() { ... },
  uncheck     : function() { ... },
  toggleCheck : function() { ... }
});
```

همچنین از تابع extend میتوان برای تعمیم سایر اشیاء jquery مانند selector ها استفاده کرد :

```
jQuery.extend(jQuery.expr[":"], {
  text     : "a.type=='text'",
  radio    : "a.type=='radio'",
  checkbox : "a.type=='checkbox'"
});
```

با توجه به قوانین فوق مثال ساده از ساختار یک پلاگین به صورت زیر می باشد :

```
(function($) {

   $.fn.myPlugin = function(settings) {
     var config = {var1: 'val1',var2 : 'val2'};

     if (settings) $.extend(config, settings);

     this.each(function() {
       // element-specific code here
     });

     return this;

   };

 })(jQuery);
```

و برای صدا زدن این پلاگین :

```
$('p').myPlugin();
$('p').myPlugin({var1 : 'string'});
$('p').myPlugin({var2 : 'values',var1 : 'values'});
```

به م الهای زیر توجه نمایید :

مثال ۱ : پلاگین ساده :

```
jQuery.fn.firstPlugin = function () {
return this.each (function () {
alert (this.id);
});
};
```

فراخوانی تابع :

```
$('#test').firstPlugin();
```

مثال ۲ : (با پارامتر) :
پلاگین زیر دو عدد میگیرد و عملیاتی را با آن دو عدد انجام میدهد و نتیجه را بصورت یک text بر میگرداند .
(عملیات پیش فرض جمع + است) :

```
(function($){

$.fn.calc = function (number1, number2, options) {
var myoptions = $.extend ( {operation: "+",label: "The result is"}, options);
return this.each(function(){
   var msg = myoptions.label + " (" + myoptions.operation + ") : ";
   var result =  eval( number1+' '+ myoptions.operation+' '+ number2 );

   return $(this).html(msg + result);
});
};

})(jQuery);
```

همان طور که در کد مشاهده میکنید برای دسترسی به مقادیر options از options.var استفاده میشود.
ولی سایر پارامتر های تابع در تابع مستقیم قابل دسترس می باشد .
توسط مثال های زیر این پلاگین را تست و نتیجه را مشاهده می نماییم :

```
$('#test').calc(4,2);
// result :       The result is (+) : 6

$('#test').calc(4,2,{operation:'*'});
// result :       The result is (*) : 8

$('#test').calc(4,2,{operation:'/',label:'calculate operation '});
// result :       calculate operation (/) : 2
```

نکته : میتوانیم مقدار پیش فرض options را بصورت استاتیک تعریف نماییم تا مستقیما بتوان آنها را تغییر داد در ای صورت کد بالا به فرم زیر تبدیل خواهد شد :

```
(function($){

$.fn.calc = function (number1, number2, options) {
var myoptions = $.extend ( $.fn.calc.defaultvars, options);
var t = $(this);
return this.each(function(){
   var msg = myoptions.label + " (" + myoptions.operation + ") : ";
   var result =  eval( number1+' '+ myoptions.operation+' '+ number2 );
   return $(this).html(msg + result);
});

};

$.fn.calc.defaultvars = {
      operation: "+",
      label: "The result is"
      };

   })(jQuery);
```

در این حالت اگر بخواهیم در برنامه یک بار مقدار پیش فرض را تغییر داده و در تمام کدهای بعدی از مقدار جدید استفاده نماییم میتوانیم مانند مثال زیر عمل کنیم :

```
$.fn.calc.defaultvars.operation = '*';
```

اگر بعد از تعریف عبارت بالا کد زیر را بنویسیم :

```
$('#test').calc(4,2);
// result :        The result is (*) : 8
```

(**پیشنهاد :** برای یادگیری بهتر پیشنهاد میشود کدهای پلاگین موجود در سایت http://plugins.jquery.com را دریافت و سورس ها را مشاهده نمایید . این میتواند به شما در ارائه ساختار مناسب برای نوشتن پلاگین یاری رساند)

فصل نهم
برخی توابع مفید jquery

در این فصل برخی از توابع موجود در کتابخانه جی کوئری را معرفی خواهیم کرد .

length : این متغیر برای هر شی jquery تعریف شده و اندازه آن شی را بر میگرداند .در صورتی که شی string باشد طول رشته را بر میگرداند و در صورتی که یک selector را با این تابع صدا بزنیم تعداد عناصر انتخابی را نشان میدهد .
بنابر این توسط این تابع میتوانیم تشخصی دهیم که آیا شی موجود است یا خیر :

```
If ($('.info').length) { ... }
```

jQuery.browser : این شی برای بررسی بروزر کاربر بکار میرود و از بروزر های زیر پشتیبانی می کند :

- webkit (as of jQuery 1.4)
- safari (deprecated)
- opera
- msie
- mozilla

مثلا :

```
If ($.browser.msie)
  alert('you are using internet explorer);
elseif ($.browser.mozilla)
  alert('you are using mozilla firefox browser ');
```

این شی متغیری به نام version دارد که ورژن بروزر کاربر را ارائه میدهد :

```
$.browser.version
```

مثلا :

```
If ($.browser.opera)
  alert('you are using opera version : ' + $.browser.version );
```

jQuery.globalEval(code) :

این تابع یک code جاوااسکریپت را بصورت سراسری اجرا میکند :

```
function test(){
    jQuery.globalEval("var newVar = true;")
}
```

حال اگر تابع test() را صدا بزنیم . متغیر newVar با مقدار true ایجاد میکند و میتوان در تمام بر نامه از آن استفاده کرد (متغیر سراسری) .

jQuery.inArray(value , array) :

این تابع مقدار value را در آرایه array جستجو کرده و ایندکس اولین خانه از آرایه را که مقدار value با آن مساوی است بر میگرداند .

```
var index = $.inArray(2,[1,2,3,4,5]);
```

نتیجه : index=1

jQuery.isFunction(obj) :

این تابع مشخص میکند آیا پارامتر ارسال شده obj از نوع تابع می باشد یا خیر . در صورت تابع بودن Obj مقدار این تابع true میشود و در غیر این صورت false

jQuery.isArray(obj) :

این تابع نیز در صورتی که پارامتر ارسالی (obj) از جنس آرایه باشد ، مقدار true و در صورتی که نباشد false بر میگرداند .

پیوست ها

پیوست ۱: **jquery Selectors**

Selector	مثال	شرح : (انتخاب میکند ...
*	$("*")	همه عناصر
#id	$("#lastname")	عنصری با آیدی lastname
.class	$(".intro")	همه عناصر که دارای کلاس intro هستند
element	$("p")	همه تگهای <p> موجود در صفحه
.class.class	$(".intro.demo")	تمام عناصر که دارای کلاسهای intro و demo هستند
:first	$("p:first")	اولین پاراگراف <p> در صفحه
:last	$("p:last")	آخرین <p> صفحه
:even	$("tr:even")	همه سطر های زوج جدول
:odd	$("tr:odd")	همه سطر های فرد جدول
:eq(index)	$("ul li:eq(3)")	چهارمین عنصر لیست (اولین عنصر 0 می ب شد)
:gt(no)	$("ul li:gt(3)")	تمام عناصر لیست بعد از عنصر چهارم
:lt(no)	$("ul li:lt(3)")	تمام عناصر لیست قبل از عنصر چهارم
:not(selector)	$("input:not(:empty)")	تمام تگ های input که خالی نیستند
:header	$(":header")	تمام تگهای هدر (...<h1><h2><h3>)
:animated		تمام عناصر متحرک صفحه
:contains(text)	$(":contains('hello')")	همه عناصری صفحه که دارای محتوی hello هستند
:empty	$(":empty")	تمام عناصر صفحه که فرزند ندارند (درونشان تگی تعریف نشده)
:hidden	$("p:hidden")	تمام پاراگرافهای مخفی درون صفحه
:visible	$("table:visible")	تمام جدولهای نمایان در صفحه
s1,s2,s3	$("th,td,.intro")	(انتخاب چند selector) تمام تگ های th و تمام تگ های td و

		تمام تگ های دارای کلاس intro
[attribute]	$("[href]")	تمام عناصر که دارای خصوصیت href هستند
[attribute=value]	$("[href='#']")	تمام عناصر صفحه که خصوصیت href آنها برابر # باشد
[attribute!=value]	$("[href!='#']")	تمام عناصر صفحه که خصوصیت href آنها مخالف # باشد
[attribute$=value]	$("[href$='.jpg']")	تمام عناصر که خصوصیت href آنها به .jpg ختم شود .
:input	$(":input")	تمام عناصر فرم صفحه
:text	$(":text")	تمام input های text
:password	$(":password")	تمام input ها که نوع password هستند
:radio	$(":radio")	تمام دکمه های رادیویی صفحه
:checkbox	$(":checkbox")	تمام چک باکس های صفحه
:submit	$(":submit")	تمام دکمه های submit صفحه
:reset	$(":reset")	تمام دکمه های reset صفحه
:button	$(":button")	تمام input های از نوع button
:image	$(":image")	تمام input های از نوع image (image button)
:file	$(":file")	تمام input های از نوع file
:enabled	$(":enabled")	تمام input های فعال
:disabled	$(":disabled")	تمام input های غیر فعال
:selected	$(":selected")	تمام عناصر انتخاب شده از یک جعبه select
:checked	$(":checked")	تمام چک باکس های انتخاب شده

پیوست ۲ : jquery Event Functions
رویدادهای jQuery :

تابع اجرا می شود وقتی	توابع رویداد
رویداد ready از شیdocument (هنگامی که صفحه آماده اجرا میباشد)	$(document).ready(function)
از عناصر انتخابی خارج (blur) می شد یم	$(selector).blur(function)
محتوی عناصر انتخابی تغییر میکند	$(selector).change(function)
روی عناصر انتخابی کلیک میشود	$(selector).click(function)
روی عناصر انتخاب دبل کلیک میشود	$(selector).dblclick(function)
خطایی روی عناصر انتخاب اتفاق می افتد	$(selector).error(function)
روی عناصر انتخاب فوکوس میشود (بر عکس blur)	$(selector).focus(function)
روی عناصر انتخاب کلیدی فشار داده میشود	$(selector).keydown(function)
روی عناصر انتخابی کلید زده میشود	$(selector).keypress(function)
کلید فشار داده شده روی عناصر انتخابی رها میشود	$(selector).keyup(function)
عناصر انتخاب لود (بارگزاری) می شود	$(selector).load(function)
روی عناصر انتخابی کلیک چپ موس زده شود (قبل از رها شدن کلیک)	$(selector).mousedown(function)
موس وارد ناحیه عناصر انتخاب شود	$(selector).mouseenter(function)
موس از ناحیه عناصر انتخابی خارج شود	$(selector).mouseleave(function)
نشانگر موس روی عناصر انتخابی حرکت کند	$(selector).mousemove(function)
نشانگر موس از روی عناصر انتخابی کنار رود	$(selector).mouseout(function)
نشانگر موس روی عناصر انتخابی برود	$(selector).mouseover(function)
کلیک موس رها شود	$(selector).mouseup(function)
وقتی عناصر انتخابی تغییر اندازه دهد	$(selector).resize(function)
وقتی روی عناصر انتخابی scroll شود	$(selector).scroll(function)

$(selector).select(function)	وقتی عناصر انتخابی ، انتخاب select شود
$(selector).submit(function)	وقتی فرمی submit شود .
$(selector).unload(function)	هنگامی که عنصری حذف (unoad) میشود . مثل بستن صفحه وب

jQuery Trigger Functions :

گاهی لازم است مستقیما توسط کد رویدادی را صدا بزنیم (تابع های مربوط به رویداد صدا زده میشود) . به این عمل trigger گفته میشود .

مثال : در مثال زیر رویداد کلیک از دکمه با آیدی demo صدا زده شده است . بنابر این تمام توابعی که به این رویداد وصل شده اند اجرا خواهند شد .

$("button#demo").click()

توابع	باعث اجرای :
$(selector).blur()	رویداد blur از عناصر انتخابی (selectors)
$(selector).change()	رویداد change از عناصر انتخابی
$(selector).click()	رویداد click از عناصر انتخابی
$(selector).dblclick()	رویداد dblclick از عناصر انتخابی
$(selector).error()	رویداد خطا از عناصر انتخابی
$(selector).focus()	رویداد فوکوس از عناصر انتخابی
$(selector).keydown()	رویداد keydown از عناصر انتخابی
$(selector).keypress()	رویداد keypress از عناصر انتخابی
$(selector).keyup()	رویداد keyup از عناصر انتخابی
$(selector).select()	رویداد select از عناصر انتخابی
$(selector).submit()	رویداد submit از عناصر انتخابی

برای ارسال نظرات ، پیشنهادات خود و همچنین برای رفع اشکال و بحث و تبادلی نظر در مورد این آموزش به وبلاگ http://blog.monavarian.ir و به بخش اموزش jquery مراجعه نمایید .

محسن منوریان http://blog.monavarian.ir
اسفند ۸۸

منابع :

http://blog.monavarian.ir
http://w3schools.com
http://jquery.com
http://www.learningjquery.com
http://snook.ca/archives/javascript/jquery_plugin
http://www.queness.com/post/112/a-really-simple-jquery-plugin-tutorial
http://www.myinkblog.com/2009/08/10/learn-how-to-create-your-own-jquery-plugin