# Jquery概述

## 什么是Jquery？

Query 是一个 JavaScript 库。极大地简化了 JavaScript 编程。

## 如何使用Jquery

在HTML页面使用 <script> 标签引用它

<script src="https://cdn.staticfile.org/jquery/1.10.2/jquery.min.js"> </script>

# jQuery 语法

## 1.基本语法

jQuery 语法是通过选取 HTML 元素，并对选取的元素执行某些操作.

1. 所有 jQuery 函数位于一个 document ready 函数中。

$(document).ready(function(){ // 开始写 jQuery 代码... });

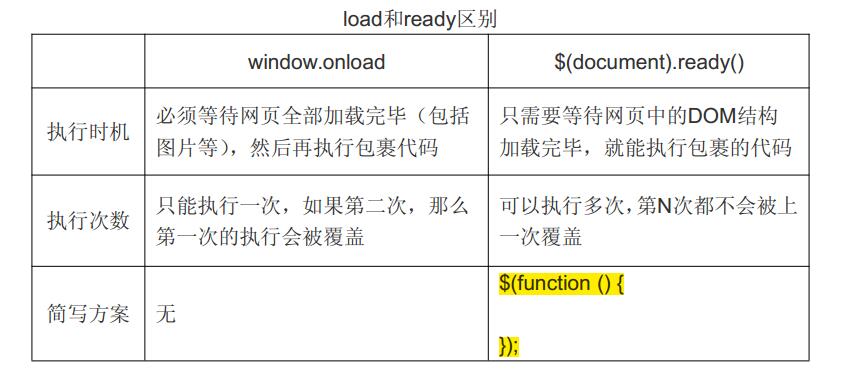
这是为了防止文档在完全加载（就绪）之前运行 jQuery 代码，即在 DOM 加载完成后才可以对 DOM 进行操作。如果在文档没有完全加载之前就运行函数，操作可能失败。

简洁写法（与以上写法效果相同）:

$(function(){ // 开始写 jQuery 代码... });

## jQuery 入口函数与 JavaScript 入口函数的区别

1. jQuery 的入口函数是在 html 所有标签(DOM)都加载之后，就会去执行。
2. JavaScript 的 window.onload 事件是等到所有内容，包括外部图片之类的文件加载完后，才会执行。



# jQuery 选择器

## Id选择器

## Class选择器

## 元素(标签)选择器

## 属性选择器

## 层级选择器

## 伪类选择器

# Jquery中的事件

# Jquery效果

1. 隐藏和显示

# Jquery对DOM操作

## 获取内容和属性

### 获得内容 - text()、html() 以及 val()

### 获取属性 - attr()

## 添加元素

### 添加元素-append()、prepend() 、after()、before（）

## 删除元素

### 删除元素-remove()、empty()

## 获取并设置 CSS 类

### jquery操作 CSS-addClass()、removeClass()、css()

## css() 方法

# jquery遍历

# jquery中Aajx

## 什么是ajax

AJAX 是与服务器交换数据的技术，它在不重载全部页面的情况下，实现了对部分网页的更新。

## load() 方法

load() 方法从服务器加载数据，并把返回的数据放入被选元素中。是简单但强大的 AJAX 方法。

**语法:**

$(selector).load(URL,data,callback);

必需的 *URL* 参数规定您希望加载的 URL。

可选的 *data* 参数规定与请求一同发送的查询字符串键/值对集合。

可选的 *callback* 参数是 load() 方法完成后所执行的函数名称

## AJAX get() 和 post() 方法

两种在客户端和服务器端进行请求-响应的常用方法是：GET 和 POST。

* *GET* - 从指定的资源请求数据
* *POST* - 向指定的资源提交要处理的数据

GET 基本上用于从服务器获得（取回）数据。注释：GET 方法可能返回缓存数据。

POST 也可用于从服务器获取数据。不过，POST 方法不会缓存数据，并且常用于连同请求一起发送数据。

### jQuery $.get() 方法

$.get() 方法通过 HTTP GET 请求从服务器上请求数据。

**语法：**

$.get(URL,callback);

或

$.get( URL [, data ] [, callback ] [, dataType ] )

* **URL**：发送请求的 URL字符串。
* **data**：可选的，发送给服务器的字符串或 key/value 键值对。
* **callback**：可选的，请求成功后执行的回调函数。
* **dataType**：可选的，从服务器返回的数据类型。默认：智能猜测（可以是xml, json, script, 或 html）。

### jQuery $.post() 方法

$.post() 方法通过 HTTP POST 请求向服务器提交数据。

**语法:**

$.post(URL,callback);

或

$.post( URL [, data ] [, callback ] [, dataType ] )

* **URL**：发送请求的 URL字符串。
* **data**：可选的，发送给服务器的字符串或 key/value 键值对。
* **callback**：可选的，请求成功后执行的回调函数。
* **dataType**：可选的，从服务器返回的数据类型。默认：智能猜测（可以是xml, json, script, 或 html）

### jQuery ajax() 方法

## 定义和用法

ajax() 方法用于执行 AJAX（异步 HTTP）请求。

所有的 jQuery AJAX 方法都使用 ajax() 方法。该方法通常用于其他方法不能完成的请求。

## 语法

$.ajax(*{name:value, name:value, ... }*)

该参数规定 AJAX 请求的一个或多个名称/值对。

下面的表格中列出了可能的名称/值：

|  |  |
| --- | --- |
| **名称** | **值/描述** |
| async | 布尔值，表示请求是否异步处理。默认是 true。 |
| beforeSend(xhr) | 发送请求前运行的函数。 |
| cache | 布尔值，表示浏览器是否缓存被请求页面。默认是 true。 |
| complete(xhr,status) | 请求完成时运行的函数（在请求成功或失败之后均调用，  即在 success 和 error 函数之后）。 |
| contentType | 发送数据到服务器时所使用的内容类型。  默认是："application/x-www-form-urlencoded"。 |
| context | 为所有 AJAX 相关的回调函数规定 "this" 值。 |
| data | 规定要发送到服务器的数据。 |
| dataFilter(data,type) | 用于处理 XMLHttpRequest 原始响应数据的函数。 |
| dataType | 预期的服务器响应的数据类型。 |
| error(xhr,status,error) | 如果请求失败要运行的函数。 |
| global | 布尔值，规定是否为请求触发全局 AJAX 事件处理程序。默认是 true。 |
| ifModified | 布尔值，规定是否仅在最后一次请求以来响应发生改变时才请求成功。默认是 false。 |
| jsonp | 在一个 jsonp 中重写回调函数的字符串。 |
| jsonpCallback | 在一个 jsonp 中规定回调函数的名称。 |
| password | 规定在 HTTP 访问认证请求中使用的密码。 |
| processData | 布尔值，规定通过请求发送的数据是否转换为查询字符串。默认是 true。 |
| scriptCharset | 规定请求的字符集。 |
| success(result,status,xhr) | 当请求成功时运行的函数。 |
| timeout | 设置本地的请求超时时间（以毫秒计）。 |
| traditional | 布尔值，规定是否使用参数序列化的传统样式。 |
| type | 规定请求的类型（GET 或 POST）。 |
| url | 规定发送请求的 URL。默认是当前页面。 |
| username | 规定在 HTTP 访问认证请求中使用的用户名。 |
| xhr | 用于创建 XMLHttpRequest 对象的函数。 |

# Jquery $.extned();

相关帮助：

1. <https://www.runoob.com/jquery/misc-extend.html>
2. <https://api.jquery.com/jQuery.extend/>

$.extend(target, objec);将一个或多个对象的内容合并到目标对象。

注意事项：

1.如果只为$.extend()指定了一个参数，则意味着参数target被省略。此时，target就是jQuery对象本身。通过这种方式，我们可以为全局对象jQuery添加新的函数。

2.如果多个对象具有相同的属性，则后者会覆盖前者的属性值。

# Jquery中添加全局函数

## 全局函数相关博客:

<https://www.cnblogs.com/weblv/p/5249841.html>

<https://www.cnblogs.com/wang715100018066/p/5997770.html>

<https://blog.csdn.net/younghaiqing/article/details/84107435>

## 添加全局函数方式

1. 添加一个新的全局函数

$.foo = function() {

   console.log("全局函数");

  }

*//调用*

$.foo();

1. 使用extend()

$.extend({

   add: function() {

    console.log("add--->全局函数")

   }

  });

  $.add();

1. 使用命名空间

$.mySlider = {

   slider: function() {

    console.log("slider");

   }

  }

  $.mySlider.slider();

## 添加插件的方式

写法1

(function($) {

   $.fn.extend({

    test1: function() {

     console.log("test");

    }

   })

  })(jQuery);

*//调用*

  $("body").test1();

写法二

(function($) {

   $.fn.test2 = function() {

    console.log("test2");

   }

  })(jQuery);

*//调用*

  $($).test2();

写法三

$(function($) {

   $.fn.test3 = function() {

    console.log("test3");

   }

  }(jQuery));

*//调用*

  $($).test3();

写法四

; $(function ($, window, document, undefined) {}());

;(function ( $, window, document, undefined ){

*//函数体内具体代码*

})(jQuery, window,document);

写法四相关博客解释

<https://www.cnblogs.com/yuqingfamily/p/5774994.html>

<https://www.cnblogs.com/baiJr/p/6135979.html>

<https://blog.csdn.net/qq814965130/article/details/79420977?utm_source=blogxgwz7>

<http://www.bcty365.com/content-47-805-1.html>

# Jquery对Ajax的封装

1.需要了解javaScript的对象

2.需要了解javaScript的回调函数

3.需要了解$.extend(target, objec);

4.需要jquery的支持(Ajax使用了jquery的)

 var postAjax = function(url,postData,successFunction,options){

   var returnFlag=false;

   var defaultOptions={

    type:"post",

    async:true,*//请求是否异步处理,默认是 true*

    cache:true,*//浏览器是否缓存被请求页面,默认是 true*

    errorFunction:errorFunction,

    successArguments: "",

    errorArguments: "",

    contentType:"application/json; charset=utf-8",*/\*传入数据类型\*/*

    dataType:"json",*/\* 返回的数据类型\*///https://www.cnblogs.com/likui-bookHouse/p/8399827.html*

    processData:true

   };

*//合并属性*

   var currentOptions = $.extend(defaultOptions, options);*//https://www.runoob.com/jquery/misc-extend.html*

   $.ajax({

    type:currentOptions.type,*//规定请求的类型（GET 或 POST）*

    url:getContextPath()+url,*//发送请求的 URL。默认是当前页面*

    data:postData,*//发送到服务器的数据*

    async:currentOptions.async,

    contentType:currentOptions.contentType,*//发送数据到服务器时所使用的内容类型。默认是："application/x-www-form-urlencoded"*

    dataTyp:currentOptions.dataType,*//预期的服务器响应的数据类型*

    processData:currentOptions.processData,

    beforeSend:function(){

    },

    success:function(resultData){

     if(resultData&&isSuccess(resultData.resultCode)){

      successFunction(resultData);

      returnFlag = true;

     else{

      defaultOptions.errorFunction(resultData);

      returnFlag = false;

     }

    },

    complete:function(XMLHttpRequest, status){

    },

    error:function(result){

    },

   });

   return returnFlag;

  }

function errorFunction(resultData) {

showMessage(resultData);

}

function showMessage(resultData) {

alert(resultData.resultMessage);

}

*//调用*

var data = {"id": "1"};

postAjax("a.html", JSON.stringify(data), function(data) {

}, {errorFunction(data) {

    alert("wangtao");

 },

 type: "get",

 sync: true,

 dataType: "html",

 contentType: "application/x-www-form-urlencoded; charset=UTF-8"

  });