**Deferred对象学习**

**Deferred对象（1.5.0版本开始使用）：**就是jquery的回调函数解决方案。就是延迟到未来某个点在执行。

一．传统Ajax的链式写法：使用低于1.5.0返回XHR对象，高于1.5.0返回deferred对象。

$.ajax(

url:“”；

success:function(){},//成功的回调函数

error:function(){},//失败的回调函数

);

新的Ajax的链式写法：

$ajax(“text.html”);

.done(function(){}）；//success函数

.fail(function(){});//error函数

1. 指定同一操作的多个回调函数

Deferred函数的一大好处，就是他允许你自由的添加多个回到函数。他们按顺序执行

$ajax(“text.html”);

.done(function(){});

.fail(function(){});

.done(function(){});//第二个回调函数

1. 为多个操作指定统一回调函数

$.when($ajax(“tex1t.html”),$ajax(“text2.html”));

.done(function(){});

.fail(function(){});

$.when();提供一种方法来执行一个或者多个对象的回调函数，延迟对象通常表示异步对象。只接受Deferred对象。

四．普通操作的回调函数接口

第一种写法：//////////////////////////////////////////

var dtd = $.Deferred(); // 新建一个Deferred对象

　　var wait = function(dtd){

　　　　var tasks = function(){

　　　　　　alert("执行完毕！");

　　　　　　dtd.resolve(); // 改变Deferred对象的执行状态

　　　　};  
  
　　　　setTimeout(tasks,5000);

**return dtd.promise(); // 返回promise对象**

　　};

**var d = wait(dtd); // 新建一个d对象，改为对这个对象进行操作**

　　$.when(d)

　　 .done(function(){ alert("哈哈，成功了！"); })

.fail(function(){ alert("出错啦！"); });

**d.resolve(); // 此时，这个语句是无效的**

var wait = function(dtd){

**var dtd = $.Deferred(); //在函数内部，新建一个Deferred对象**

　　　　var tasks = function(){

　　　　　　alert("执行完毕！");

　　　　　　dtd.resolve(); // 改变Deferred对象的执行状态

　　　　};  
  
　　　　setTimeout(tasks,5000);

　　　　return dtd.promise(); // 返回promise对象

　　};

**$.when(wait())**

　　.done(function(){ alert("哈哈，成功了！"); })

　　.fail(function(){ alert("出错啦！"); });

第二种写法：///////////////////////////////////////

　　　　var wait = function(dtd){

　　　　var tasks = function(){

　　　　　　alert("执行完毕！");

　　　　　　dtd.resolve(); // 改变Deferred对象的执行状态

　　　　};

　　　　setTimeout(tasks,5000);

　　　　return dtd.promise();

　　};

　　$.Deferred(wait)

　　.done(function(){ alert("哈哈，成功了！"); })

.fail(function(){ alert("出错啦！"); });

第三种方法：/////////////////////////////////////

var dtd = $.Deferred(); // 生成Deferred对象

　　var wait = function(dtd){

　　　　var tasks = function(){

　　　　　　alert("执行完毕！");

　　　　　　dtd.resolve(); // 改变Deferred对象的执行状态

　　　　};

　　　　setTimeout(tasks,5000);

　　};

**dtd.promise(wait);//在wait对象上面部署Deferred接口**

　　wait.done(function(){ alert("哈哈，成功了！"); })

　　.fail(function(){ alert("出错啦！"); });

　　wait(dtd);

Deferred对象三种执行状态---未完成，已完成，已失败

Deferred.resolve()-----已完成

Deferred.reject()------以失败

Deferred.promise()-----在原来的deferred对象上面返回另一个deferred对象，后者只开放或改变执行状态无关的方法（比如：done和fail方法），屏蔽与改变执行状态无关的方法（比如reject和resolve方法），从而是执行状态不能改变。

Jquery规定。$Deferred（）可以接受一个函数名作为参数，$.Deferred所生成的deferred对象将作为这个 默认参数。

$------------是指一个标示符，让浏览器知道这个是jquery，在jquery中可以使用

var jq = jQuery.noConflict();就可以用jq代替jQuery和$；

$(function(){});等同于$(document.ready(function(){}));--------是告诉浏览器Dom结构完成之后要加载这个方法，不同于onload，onload是内容加载完之后。