### 任务1：FAQ答案的显示和隐藏

#### 1.1 show()和hide()方法

show()和hide()方法是最常用的显示与隐藏方法，它们相当于CSS中的display属性取不同的值。它们的基本语法是：

$(selector).show(speed,callback);

$(selector).hide(speed,callback);

$(selector)表示选中的元素，speed表示动画速度，callback是回调函数，在显示或隐藏结束时调用。参数取值：

|  |  |
| --- | --- |
| **参数** | **描述** |
| speed | 可选。规定元素从隐藏到完全可见的速度。默认为"0"。  可取的值：   * 毫秒（比如1500） * "slow"（为600毫秒） * "normal"（为400毫秒） * "fast"（为200毫秒）   在设置速度的情况下，元素从隐藏到完全可见的过程中，会逐渐地改变其高度、宽度、外边距、内边距和透明度。如果没有设置速度，元素会直接显示。 |
| callback | 可选。show()方法执行完之后，要执行的方法。  除非设置了speed参数，否则不能设置该参数。 |

jQuery专门提供了toggle()方法来交替显示元素。基本语法如下：

$(selector).toggle(speed,callback);

#### 1.2 fadeIn()和fadeout()方法

这两个方法实现显示或隐藏的淡入淡出效果，即只改变元素的透明度，不改变高度、宽度等内容。

fadeIn()方法实现淡入效果，其基本语法如下：

$(selector).fadeIn(speed,callback);

它与show()方法的参数设置一样，speed设置淡入效果的时间。

fadeOut()方法实现淡出效果，其基本语法如下：

$(selector).fadeOut(speed,callback);

fadeToggle()方法交替进行fadeIn()和fadeOut()，如果元素是淡出的，那么fadeToggle()将淡入该元素，如果之前是淡入的，fadeToggle()将淡出该元素。其基本语法如下：

$(selector).fadeToggle(speed,callback);

fadeTo()方法把所有匹配元素的不透明度以渐进方式调整到指定的不透明度，并在动画完成后可选地触发一个回调函数，其基本语法如下：

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

其中，speed为必需参数，给出透明度变化的时间。opacity为必需参数，设置将要调整到的透明度，值域为0到1之间。callback为可选参数，是回调函数，在fadeTo()方法结束后调用。

#### 1.3 slideUp()和slideDown()方法

slideDown()方法用来实现向下滑动动画效果，控制元素高度在指定时间内从display:none延伸至完整高度。其基本语法为：

$(selector).slideDown(speed,callback);

slideUp()方法用于实现向上滑动动画效果，控制元素高度在指定时间内从下到上缩短至display:none。其基本语法为：

$(selector).slideUp(speed,callback);

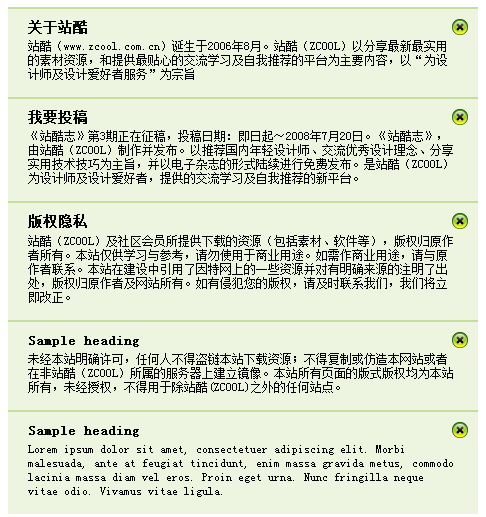
slideToggle方法用来实现交替显示向上向下滑动动画效果，如果之前是下滑显示，则slideToggle()可向上滑动它们，反之slideToggle()可向上滑动它们。其基本语法为：

$(selector).slideToggle(speed,callback);

#### 1.4任务实现

#### 1.5 练一练

单击图标，通过透明度的变化以淡出形式来隐藏某条内容，如图所示。



#### 1.6 课后作业

在图1上面单击“了解学校”图标，向下滑动展开内容介绍，同时“了解学校”右边的箭头变为向上箭头，如图2所示。再次单击“了解学校”图标，向上收缩隐藏内容，箭头变为向下箭头。

****

图1 内容隐藏时

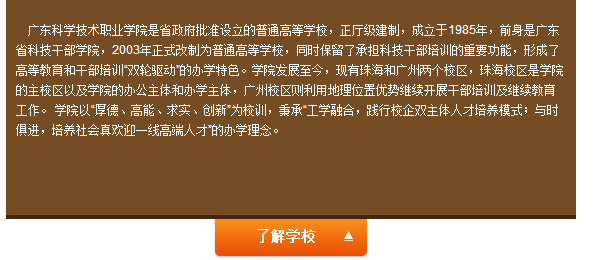
****

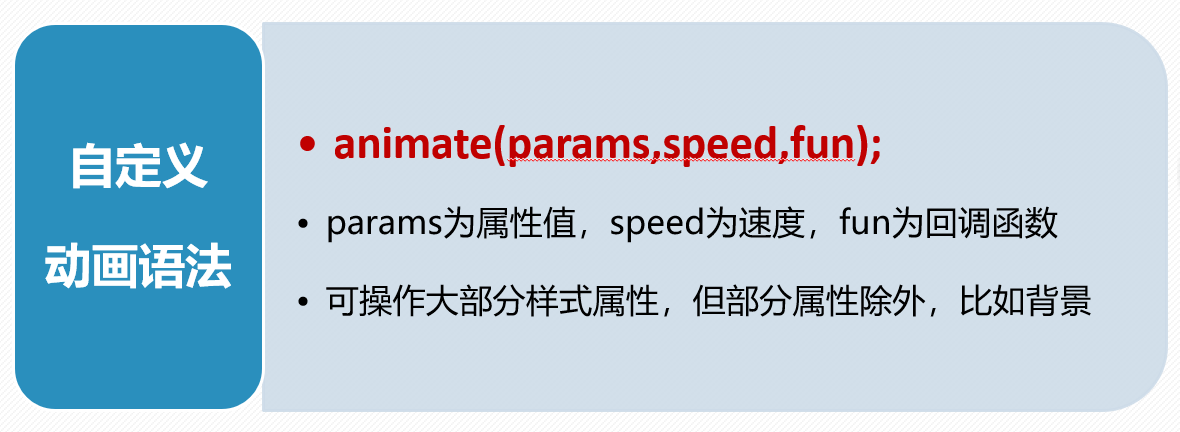
图2 内容显示时

### 任务1：图片横向移动效果

#### 1.1自定义动画方法animate()

animate()方法的基本语法如下：

$(selector).animate({params},speed,callback);



##### 1.简单动画

单击“开始动画”,把<div>元素移至距左边250像素的位置，即left等于250px。

$("#btn1").click(function(){

$(".divStyle").animate({left:'250px’}, 2000);});

##### 2.累加动画

每单击一次“累加动画”按钮，把<div>元素移至距左边100像素的位置。

$(“#btn2”).click(function(){

$(“.divStyle”).animate({left:”+=100px”},”fast”);})

##### 3. 同时执行多个动画

每单击一次“执行多个动画”，<div>元素往左移动100px、高度增加50px。

$("#btn3").click(function(){

$(“.divStyle”).animate({left:”+=100px”,height:”+=50px”},”fast”);})

##### 4. 顺序执行多个动画

每单击一次“动画队列”，<div>元素先往左移动100px，后高度增加50px。

$(“#btn4”).click(function(){

$(“.divStyle”).animate({left:”+=100px”},”fast”)

.animate({height:”+=50px”},”

**动画顺序执行 称为“动画队列”**

#### 1.2动画回调函数

$("button").click(function(){

$("div").animate({left:'250px',top:'300px'})

.animate({opacity:'0.5'})

.animate({width:'150px'},'slow')

.css("border","5px solid orange");

});

如果想在动画执行完毕后，为div添加边框，此时该代码达不到预期效果。出现这个问题的原因是css()方法并不是动画方法，它不会被加入到动画队列进行排队，而是插队立即执行。如果要实现我们预期的效果，就必须使用回调函数让非动画方法实现排队。

$("button").click(function(){

$("div").animate({left:'250px',top:'300px'})

.animate({opacity:'0.5'})

.animate({width:'150px'},'slow', function(){

$("div").css("border","5px solid orange");

});

});

#### 1.3停止元素的动画

网页中有时需要停止匹配元素正在进行的动画，这时要使用停止元素的动画方法stop()，它的语法结构为：

stop([clearQueue],[gotoEnd]);

参数clearQueue 和 gotoEnd 都为可选参数，为布尔值，即true或false，默认值都是false。clearQueue 代表是否要清空未执行完的动画队列，gotoEnd代表是否直接将正在执行的那个动画跳转到末状态，而非动画队列中最后一个动画的末状态。

* stop(false,false):停止当前动画，并从目前的动画状态开始动画队列中的下一个动画
* stop(true,false): 停止所有动画，保持当前状态，瞬间停止
* stop(false,true): 停止当前动画，跳到当前动画的末状态，然后进入队列中的下一个动画
* stop(true,true): 停止所有动画，跳到当前动画的末状态

#### 1.4 判断元素是否处于动画状态

if(! $(elememt).is(":animated") ){ //判断元素是否处于动画状态

//如果当前没有进行动画，则添加新动画

}

#### 1.5延迟动画

delay()方法用于将队列中的函数或动画延时执行。它既可以推迟动画队列中函数的执行，也可以用于自定义队列。语法结构如下：

delay(duration, [queueName])

#### 1.6任务实现

#### 1.7 练一练

1.根据所给素材实现图片横向轮播效果。



#### 1.8 课后作业

根据所给素材实现图片纵向轮播效果。。