**第二章 翻转课堂技术概述**

**2.1 HTML5 Canvas技术概述**

**2.1.1 Canvas简介**

<Canvas>是HTML5中新出现的一个元素。Canvas是网页上的一块位图区域，这个元素可以被JavaScript用来绘制图形。Canvas可以实时渲染屏幕上的像素，它通过调用API，每一帧都可以完全重画屏幕上的位图区域。Canvas元素可以通过网页上的DOM获取，但是Canvas上绘制的图形元素不能够通过DOM获取。原因在于Canvas的实时模式，Canvas在图形绘制完成之后，不会保留绘制对象的集合。

Canvas的概念最早由苹果公司提出，用于在Mac OS X WebKit中创建控制板部件（dashboard widget）。在Canvas之前，浏览器中只能使用Falsh和SVG插件、VML以及一些奇怪的JavaScript技巧来绘制图形。Canvas提供了一套二维绘图API，使得在浏览器中绘图变得更加方便。苹果起初想要为Canvas规范申请知识产权，但最终苹果公司还是按照W3C的免版税专利权许可条款公开了其专利。

Canvas API允许程序员把各种图形、文字和图片直接绘制到浏览器上。程序员也可以选择使用不同的颜色、旋转图形、使用各种各样的线条来绘制图形。Canvas 2D只是一个用来在屏幕上显示图像的API，但是通过JavaScript，Canvas可以实现鼠标和键盘的监听、时间间隔、对象、声音和数学函数等功能。在此基础之上，Canvas就可以用来制作动画和游戏。

Canvas使用非常简单，只需要向HTML5中添加Canvas元素即可：<canvas></canvas>。

**2.1.2 Canvas API**

Canvas元素并没有提供很多API，只有width和height两个属性，分别用来表示Canvas元素的宽度和高度，默认情况下的值为300和150。Canvas方法中最重要的是getContext()，可以用这个方法获取Canvas绘图环境对象，然后通过这个绘图环境对象在Canvas上绘制图形。

实际上，在JavaScript中，很少会用到Canvas元素本身。偶偶会用来获取Canvas元素的宽度和高度，大部分情况下还是用来获取Canvas绘图环境对象的引用，通过这个绘图环境对象提供的强大API来绘制图像。

**2.1.2 Canvas 绘图**

Canvas主要就是用来绘制图形。Canvas的2d绘图环境提供了一套很强大的API来支持绘图，所以在Canvas上绘制线条、矩形、圆形等几何图形十分简单、方便。

Canvas具有默认的坐标系统，默认情况下，左上角为原点，x轴向右增长，y轴向下增长。也可以通过各种平移、缩放或者旋转等操作来改变默认的坐标系。

在Canvas上绘制图像时，浏览器会按照以下的步骤来操作：

1. 将图形或图像绘制到一个无限大的透明位图中，在绘制时遵从当前的填充模式、描边模式以及线条模式。
2. 将图形或图像的阴影绘制到另外一幅位图中，在绘制时使用当前绘图环境的阴影设定。
3. 将阴影中每一个像素的alpha分量乘以绘图环境的globalAlpha属性值。
4. 将绘有阴影的位图与经过剪切区域剪切过的Canvas进行图像合成。在操作时使用当前的合成模式参数。
5. 将图形或图像的每一个像素的颜色分量，乘以绘图环境对象的globalAlpha属性值。
6. 将绘有图形或图像的位图，合成到当前经过剪切区域剪切过的Canvas位图之上，在操作时使用当期操作符。

浏览器起初会将图形绘制到一张无限大的位图上，当然，无限大的位图是不存在的，浏览器只是假设无限大的。接下来就是处理阴影，如果没有启用阴影的话，那就直接跳过。最后，浏览器根据当前合成设定与剪切区域，将图形或位图与Canvas元素进行图像合成。

（参考《HTML5 Canvas核心技术 图形、动画与游戏开发》 2.2）

**2.1.3 Canvas 文本**

几乎所有的HTML元素都支持文本，一些元素只是简单地显示文本，而另一些则支持复杂的编辑操作。Canvas所支持的文本操作非常简单，仅仅可以用来显示文本，给文本描边和填充，而不支持文本选择、文本编辑、文本复制和粘贴功能。因为Canvas本身不是一个文本编辑器，所以Canvas绘图环境对象也只有3个属性（表2-1）和3个方法（表2-2）与文本有关：

|  |  |
| --- | --- |
| 属性 | 描述 |
| font | 设置文本的字体。 |
| textAlign | 决定文本水平对齐方式。可以取值为start、canter、end、left、right，默认start |
| textBaseline | 决定文本垂直对齐方式。可以取值为top、button、middle、alphabetic、idealgraphic、hanging，默认alphabetic |

表2-1 CanvasRenderingContext2D与文本有关的属性

|  |  |
| --- | --- |
| 方法 | 描述 |
| strokeText(text,x,y) | 描边显示文本。 |
| fillText(text,x,y) | 填充显示文本。 |
| measureText(text) | 获取文本的宽度。 |

表2-2 CanvasRenderingContext2D与文本有关的方法

**2.1.2 Canvas 图片**

图片与绘图相似，都是Canvas所支持的重要的功能。开发人员可以把一副图片的全部或者部分内容画到Canvas上。同时，也可以对图片进行放大和缩小。图像并不需要完全覆盖整个Canvas界面，可以被画到任意一块指定的区域。图像可以被画到离屏Canvas中，通过这种方式，就可以对图片进行一些更高级的处理，比如实时图像查看器。

图像的绘制通过drawImage()方法实现。在绘制时，不仅可以指定绘制的位置，还可以指定绘制的大小。不仅仅可以把图像绘制到Canvas中，也可以把一个Canvas绘制到另外一个Canvas中。

**2.1.2 Canvas 鼠标事件**

HTML应用程序是以事件来驱动的，几乎所有的基于Canvas的应用程序都会处理鼠标事件。只需要在Canvas上注册事件监听器，并编写用于响应这些事件的处理代码，就可以在鼠标点击的时候处理响应的事件。

在Canvas中监听鼠标事件非常简单，只需要在Canvas中添加一个事件监听器，当事件发生的时候，浏览器就会调用这个监听器了。具体有以下两种实现方式：

1. canvas.onmousedown = function(e){//do something}
2. canvas.addEventListener(‘moudedown’,function(e){//do...});

上面两种实现方式效果相同，都是在鼠标按下的时候作出响应。但是第一种方式比较简单一些，而如果要对同一个事件注册多个监听器的话，就不得不使用第二种方式了。除了mousedown之外，还有mousemove、mouseup等监听。

浏览器本身也会对某个鼠标事件作出响应，但是如果你重写了鼠标事件的响应，大多数情况下某个鼠标事件处理完之后，就不会希望浏览器再去作出响应。这时，就可以使用preventDefault()方法，它可以阻止浏览器对该事件作出默认的反应。