Chapter 6

HTML5 Drawing APIs

A cool feature of HTML5 is that you have the option to dynamically render 2D shapes and bitmap images on the fly, as it now has its own native drawing API. This is a huge move for HTML, which remained fairly static and limited in this area since its creation. Now those days are over, and a new realm of possibilities is here to deal with graphics created within the HTML page itself. It is now possible to build shapes, graphs, animations, and even games without needing to rely on any external plug-in (like Flash). This brings great value when it comes to developing web pages and applications, but even more so when it comes to compatibility with mobile devices.

The drawing API is part of the new HTML5 canvas element, and it provides an API for 2D drawing. More recently, the possibility of 3D drawing via HTML5’s WebGL support seems to be on its way. At the time of this writing, few browsers are compatible with 3D drawing, but it’s very likely that it will become part of the HTML5 options in the near future.

All major browsers that support HTML5 are compatible with the canvas element, and are therefore compatible with the 2D drawing API. However, when it comes to Internet Explorer, HTML5 compatibility will not come until version 9.0. Browser and mobile device compatibility for the canvas element and the 2D API is shown in Table 6-1.

Table 6-1. Browser and Mobile Device Compatibility with the Canvas Element and 2D API

<table>
<thead>
<tr>
<th>Firefox</th>
<th>IE</th>
<th>Chrome</th>
<th>Safari</th>
<th>Opera</th>
<th>iPhone</th>
<th>Android</th>
<th>BlackBerry</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.0+</td>
<td>9</td>
<td>3.0+</td>
<td>3.0+</td>
<td>10.0+</td>
<td>1.0+</td>
<td>1.0+</td>
<td>OS 6.0</td>
</tr>
</tbody>
</table>
Solution 6-1: How to draw with HTML5 using the canvas element’s drawing API

The canvas element is a rectangular area that you can add to your HTML5 page, and it offers a wide range of graphic possibilities, as you can control every pixel through its 2D drawing API. By itself, the canvas element has no drawing abilities; everything you will create in it will be drawn programmatically using the JavaScript language.

In this solution, we will define a canvas element and make it ready for use so that you can take advantage of its drawing API.

What’s involved

Setting the basis and background to be able to draw in HTML5 is pretty straightforward. You just have to define a canvas element, `<canvas></canvas>`. The canvas tag is really simple, and has only three attributes: width, height in pixels, and an ID to identify which canvas you’re drawing in.

```html<canvas id="canvasID" width="300" height="200"></canvas>
```

You can position your canvas in your HTML5 page and apply CSS to it as you would do with any other tag. Your `<canvas>` is initially empty—a plain area—unless you put a border or background color on it through CSS. However, it won’t appear on the page until you draw something inside it. A canvas is simply a plain rectangle which will constitute the environment where you will be able to draw graphics, make animations, and so on.

To use your canvas and draw in it, you’ll need to use JavaScript. If you have already drawn programmatically with languages like ActionScript 3 or Java, you will find a lot of similarities here.

Once the canvas is created, the first thing you need is to gain access to it. You can do that just like with any other element of the Document Object Model (DOM):

```javascript
var myCanvas=document.getElementById("canvasID");
```

You can then access the canvas rendering context, which provides the access to the drawing API and its methods. To retrieve the context object, the canvas element uses the DOM method `getContext()`, which has only one parameter: the type of context. At the time of this writing, the 2D context is the only one