Mobile Design Pattern: High Performance Widgets: CSS Sprites

Introduction

This is the first of the series of articles **High Performance Widgets**. The objective of these articles is to show tips and tricks that improve your WRT widgets or Mobile Websites performance, and also to make your widget optimized for a better user interaction.

The Back-end of your widgets is only 20% of the loading process, so if you want to make it faster, you have to focus on the Front-end. The **XTHML** documents spend between **10** to **20%** to load, the rest is divided among Style Sheets, JavaScript and Images.

How do CSS Sprites works

CSS sprites replaces the old method of slicing and dicing in a more flexible way, using nothing more than creative thinking and CSS.

The technique consists in making image-replacements using a CSS mask. Behind the mask we'll use a **master image**, which is a combination of all images needed and it's different states. You can combine all in master image or separated by specific parts (like Navigation/ Buttons/ or whatever you wish).

You can use CSS sprites in any XHTML element that accepts Backgrounds, such as any **display:block** element.

This **Master Image** contains all the images of the example used in this article.

The master image

In this image we have the buttons, the icons, the logo, and the navigation. Notice that all the items that have human interaction (in this case the navigation) appear in two different forms:

- The normal: Before the interaction.

- The active: After the interaction. In this case, when the selected screen is displayed.

The XHTML

Is important for every good CSS trick, to have a clean block of code.

Let's see the example bellow:

```
<div id="header">
    <h1 id="logo">Catch Up</h1>

     <ul id="menu">
        <li id="btn-url">
            <a href="#" class="active">Urls</a>
        </li>
        <li id="btn-read">
            <a href="#">Read</a>
        </li>
    </ul>
</div>
```

This one represents the logo and the navigation. Notice that there's a link with the class **active**. This is because the Div **#url** , which is relate to **#btn-url**, is the first screen displayed, so it's active.

This code will serve as the basis for our example. Light-weight, simple markup and easy to understand. Anyway, at the bottom of this article you'll find the whole code (XHTML/CSS/JS) used in the interface, for mores CSS sprites examples.

(C) Copyright Nokia 2014. All rights reserved.

Applying the CSS

Master image as background

Now, it's time to style our structure. First, we have to set the **master image** as the background of all elements that will use it.

```
#logo,
#menu #btn-read a,
#menu #btn-url a,
#menu #btn-read .active,
#menu #btn-url .active,
{
   background: url(../images/set.png) no-repeat;
}
```

That line of CSS code says that the **logo**, the **navigation links** and their **active** states, have the same background, **set.png**

Positioning the elements

Now we'll style them individually, changing element positions and setting the widths and heights to create the CSS mask effect.

Like in a common image replacement, we use the **display:block** for the links, so we can set their width and height properties and the **text-indent** to make the link's text disappear.

```
          #logo{
  text-indent: -999px;
  width: 170px;
  height: 50px;
  float: left;
  margin: 20px 0 0 20px;
  }

          #menu{
  width: 140px;
  height: 55px;
  float: left;
  margin: 20px 0 0 30px;
  }


#menu #btn-url a{
  display: block;
  width: 64px;
  height: 53px;
  float: left;
  overflow: hidden;
  text-indent: -999px;
  }

#menu #btn-read a{
  display: block;
  width: 66px;
  height: 53px;
  float: left;
  overflow: hidden;
  text-indent: -999px;
  margin-left: 5px;
               }
```


(C) Copyright Nokia 2014. All rights reserved.

Positioning backgrounds

Now the mask is created, let's position the backgrounds, making the correct elements meet the correct background.

```
         #logo{
                background-position: -0px -0px ;
              }

#menu #btn-read a{
                background-position: -236px -0px ;
              }


#menu #btn-url a{
                background-position: -170px -0px;
             }
```

The active

Now we're going to format the navigation links in the 'active' state.

When you click the navigation buttons the JavaScript in this example, says that this class *active* will be added to the link, changing his style.

Gladly in our case, we just have to change the background position.

```
#menu #btn-read a{
                background-position: -236px -54px ;
              }


#menu #btn-url a{
                background-position: -170px -54px;
             }
```

Advantages

CSS sprites are far more flexible than Image Mapping.

Using CSS sprites on this case we save 10 HTTP Requests, which reduces a lot the widget loading time.

Other advantage is reducing the download size of the file. Though most people assume than the master image is heavier than all images sliced individually, in fact, **the master image** is a lot smaller.

In this example the master image size was 14kb and the size of all slices together where 44kb. That implicates that the CSS sprites made a reduction of 3.14 times the image size.

That happens because the combination of images eliminates the need to load individual elements for each one, such as color tables or formatting info.

Archives

This is the .zip file, with the XHTML, CSS and JavaScript used in this article.

This examples contains a base interface for you to guide your self.

Example.zip

(C) Copyright Nokia 2014. All rights reserved.

See also

- High performance Widgets: Combine your JavaScripts and CSS in external Files
- High performance Widgets: Optimize your JavaScript

(C) Copyright Nokia 2014. All rights reserved.