sohtanaka.com

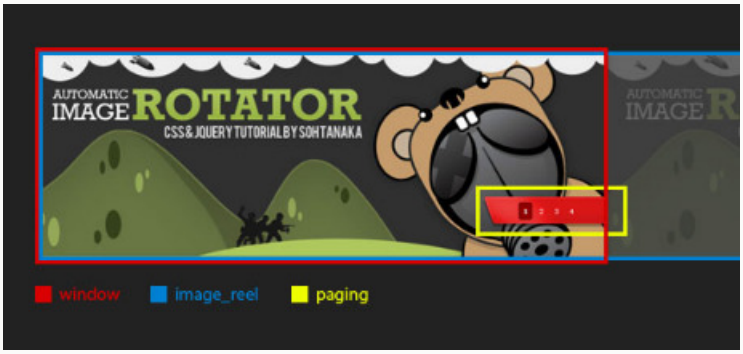
Tutorials

Tags: Intermediate, Widgets

With the release of the iPad and its lack of support for flash, it has stirred up a lot of debates regarding the future of flash. With this in mind, I believe it is wise to build simple widgets like the image slider using HTML/CSS/Javascript, and leave more interactive applications for flash if needed. The html based image slider will have its benefits with SEO and will also degrade gracefully for those w/out js.

The Wireframe HTML

Start with having a wrapping container div called `main_view`, and two sections nested inside called `image_reel` and `paging`. The `image_reel` will contain the sliding images, and `paging` contains the paging controls. Take a look at the image below for a visual.




```
<div class="main_view">
 <div class="window">
 <div class="image_reel">
 <a href="#"><img src="reel_1.jpg" alt="" /></a>
 <a href="#"><img src="reel_2.jpg" alt="" /></a>
```

```
 <a href="#"><img src="reel_3.jpg" alt="" /></a>
 <a href="#"><img src="reel_4.jpg" alt="" /></a>
 </div>
 </div>
 <div class="paging">
 <a href="#" rel="1">1</a>
 <a href="#" rel="2">2</a>
 <a href="#" rel="3">3</a>
 <a href="#" rel="4">4</a>
 </div>
</div>
```

Styling CSS

Take a look at the comments below for an explanation of the styles.

```
/*--Main Container--*/
.main_view {
 float: left;
 position: relative;
}
/*--Window/Masking Styles--*/
.window {
 height:286px; width: 790px;
 overflow: hidden; /*--Hides anything outside of the set width/height--
*/
 position: relative;
}
.image_reel {
 position: absolute;
 top: 0; left: 0;
}
.image_reel img {float: left;}

/*--Paging Styles--*/
.paging {
 position: absolute;
 bottom: 40px; right: -7px;
 width: 178px; height:47px;
 z-index: 100; /*--Assures the paging stays on the top layer--*/
 text-align: center;
 line-height: 40px;
 background: url(paging_bg2.png) no-repeat;
 display: none; /*--Hidden by default, will be later shown with
```

```
jQuery--*/
}
.paging a {
 padding: 5px;
 text-decoration: none;
 color: #fff;
}
.paging a.active {
 font-weight: bold;
 background: #920000;
 border: 1px solid #610000;
 -moz-border-radius: 3px;
 -khtml-border-radius: 3px;
 -webkit-border-radius: 3px;
}
.paging a:hover {font-weight: bold;}
```

Step 3. Setting up jQuery

For those who are not familiar with jQuery, do check out their site first and get an overview of how it works. I've shared a few tricks that I have picked up along the way, you can check those out as well.

Initial Step – Call the jQuery file

You can choose to download the file from the jQuery site, or you can use this one hosted on Google.

```
<script type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.4.1/
jquery.min.js"></script>
```

Directly after the line where you called your jQuery, start a new <script> tag and start your code by using the $(document).ready event. This allows your jQuery code to run the instant the DOM is ready to be manipulated. The code you will be writing in the next few steps will all take place within.

```
$(document).ready(function() {
 //Code goes here
});
```

Step 4. Bringing it to Life – jQuery

The following script contains comments explaining which jQuery actions are being performed.

Setting up the Image Slider

Start by showing the paging and activating the first link. Then we will calculate and adjust the width of the `image_reel` according to how many slides there are.

```
//Show the paging and activate its first link
$(".paging").show();
$(".paging a:first").addClass("active");

//Get size of the image, how many images there are, then determin the
size of the image reel.
var imageWidth = $(".window").width();
var imageSum = $(".image_reel img").size();
var imageReelWidth = imageWidth * imageSum;

//Adjust the image reel to its new size
$(".image_reel").css({'width' : imageReelWidth});
```

Setting up the Slider Function and Timer

We first create the function for the slide event by itself (`rotate`). Then create another function (`rotateSwitch`) that will rotate and repeat that slide event (`rotate`).

```
//Paging and Slider Function
rotate = function(){
 var triggerID = $active.attr("rel") - 1; //Get number of times to
slide
 var image_reelPosition = triggerID * imageWidth; //Determines the
distance the image reel needs to slide

 $(".paging a").removeClass('active'); //Remove all active class
 $active.addClass('active'); //Add active class (the $active is
declared in the rotateSwitch function)

 //Slider Animation
 $(".image_reel").animate({
 left: -image_reelPosition
 }, 500 );

};

//Rotation and Timing Event
rotateSwitch = function(){
 play = setInterval(function(){ //Set timer - this will repeat itself
every 7 seconds
 $active = $('.paging a.active').next(); //Move to the next paging
```

```
 if ( $active.length === 0) { //If paging reaches the end...
 $active = $('.paging a:first'); //go back to first
 }
 rotate(); //Trigger the paging and slider function
 }, 7000); //Timer speed in milliseconds (7 seconds)
};

rotateSwitch(); //Run function on launch
```

Take a look at this tutorial for an explanation of how the timer (`setInterval`) works.

Hover and Click Events

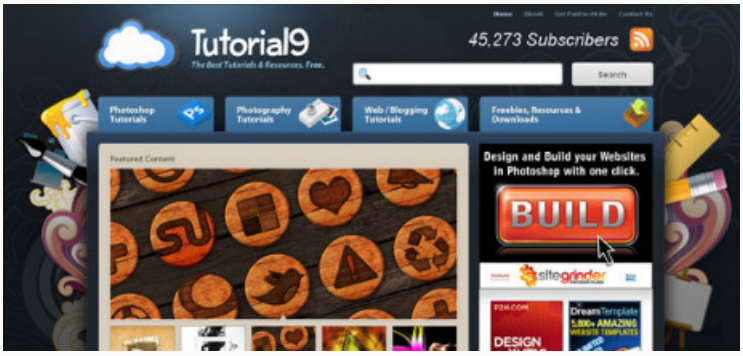
In case the user wants to view the slide for a longer period of time, we will allow the slider to stop when it is hovered. Another thing to consider is we should reset the timer each time the paging is clicked. This will prevent unexpected slide switches and allow for a smoother experience.

```
//On Hover
$(".image_reel a").hover(function() {
 clearInterval(play); //Stop the rotation
}, function() {
 rotateSwitch(); //Resume rotation timer
});

//On Click
$(".paging a").click(function() {
 $active = $(this); //Activate the clicked paging
 //Reset Timer
 clearInterval(play); //Stop the rotation
 rotate(); //Trigger rotation immediately
 rotateSwitch(); // Resume rotation timer
 return false; //Prevent browser jump to link anchor
});
```

Inspiration

Below are some sites that use similar techniques, check them out for inspiration!



Related Articles