**ANIMATIONS USING CSS3**

This document aims at giving a basic insight into CSS3 animations and transitions. For our daily project based animation activities, we make use of JQuery animate function. This document aims at understanding how CSS3 animations and transitions can be used to create similar animations like the ones we do using JQuery.

**What is CSS3?**

Before we jump into CSS3 animations, I want to give a brief idea as to what CSS3 is exactly and what are its merits over JQuery animations. CSS3 is the latest version in CSS specifications. It succeeds CSS2.1 and CSS2. Some main added features in CSS3 include multiple backgrounds, rounded corners, animations, additional selectors etc.

**How are CSS3 animations better than JQuery animations?**

1. In a head to head comparison, CSS3 animations will almost always be faster than JQuery animations. The main reason being that JQuery is an external plugin whereas CSS3 is handles natively by the browser.
2. JQuery has to modify the properties of the DOM element using timers and loops whereas CSS3 is a part of the browser engine hence it relies mainly on the hardware support than some external plugin.
3. JavaScript is an interpreted language and the JS engine of the browser has to parse and execute every instruction during run-time. On the other hand, browsers can implement CSS transitions natively. This code will be compiled to machine language.

**How are CSS3 animations not better than JQuery animations?**

The main concern while using CSS3 animations is cross browser compatibility. There are several instances wherein if the end user is using an older browser, the animations may not perform as desired. But for our purpose, with the appropriate prefixes, these animations should work just fine since we mainly rely on the animations to work well on an iPad.

**CSS3 animations**

There are two different types of animations in CSS3. One is called an **Animation** and the other is called a **Transition.**

1. **Transition**: A transition is an animation which occurs with respect to change in state. A transition is an animation which is performed between two distinct stated i.e. a start state and an end state.
2. **Animations**: An animation runs by default. We don’t have to wait for a particular state for an animation to work.

**Some Functions to remember while using transitions and animations:**

1. **Property**: This will define the property which we need to animate. For e.g. width, height, color etc.
2. **Duration**: This will define the duration for which the animation will run or execute. Duration tells us the total time for which we will be able to run the animation. It can be defined in seconds or milliseconds for e.g. 1s, 5s, 500ms etc.
3. **Timing function**: Timing function will define the style in the animation will appear for e.g. linear, ease etc.
4. **Name**: When we are using CSS3 animation, we need to define a name for the respective animation. We use the animation-name function to define the name of the animation.
5. **Iteration count**: The iteration-count property tells us the number of times the animation will iterate
6. **Animation fill mode**: This defines a style for the element when the animation is not playing i.e. when it is complete. For our purposes, we will be using animation-fill-mode as forwards as we in most projects, we want the animation final state to persist. The forwards fill mode will keep the final state of the element after the animation is completed.

**Implementing animations:**

We implement animations in CSS3 using keyframes. The `@keyframe` rule specifies the actual animation code. Consider the following example:

I want to animate a `<div>` box to move from its current position to 100px below. We use the following rules:

**.div {**

**-webkit-animation-duration: 3s;**

**-webkit-animation-name: myAnimation;**

**-webkit-animation-timing-function: linear;**

**-webkit-animation-iteration-count: infinite;**

**-webkit-animation-fill-mode-count: forwards;**

**}**

As shown above, we have defined the animation name as “myAnimation”. We define the keyframe rule as follows:

**@-webkit-keyframes myAnimation {**

**from {width: 0px;}**

**to {width: 330px;}**

**}**

The above rule states that the div box will should animate from 0px width to 330px width over the duration of the animation. Further we can fine tune this animation even more precisely by defining percentages as follows:

**@-webkit-keyframes myAnimation {**

**10% {width: 0px;}**

**50% {width: 165px;}**

**100% {width: 330px;}**

**}**

We can thus define any sort of animation using CSS3 directly which will completely wipe out dependency on JQuery!

**Some quick tips:**

* Instead of defining all the properties, the animation and transition shorthand are a handy way to quickly define all the properties in one line. For consider the following example:

I want to animate a `<div>` box to move from its current position to 100px below. We use the following rules:

**.div {**

**-webkit-animation-duration: 3s;**

**-webkit-animation-name: myAnimation;**

**-webkit-animation-timing-function: linear;**

**-webkit-animation-iteration-count: infinite;**

**-webkit-animation-fill-mode-count: forwards;**

**}**

Now here, using the shorthand, we can write this as follows:

**.div {**

**-webkit-animation: myAnimation 3s linear forwards infinite;**

**}**

We can now say that we know the fundamentals of CSS3 animations. I conclude this document by stating CSS3 is a vast ocean and there are several other learnings to be discovered from this ocean!

**GOOD LUCK AND HAPPY CODING!**