CSS3 - Transitions and Animations

The new CSS version 3 will provide some interesting effects for web development.

There are quite a few transforms available. These functions can be used with almost any element in order to affect on it's visual representation. These functions vary from rotate, skew to generated gradients, box reflection and masking.

It is important to remember that even the visual looks will change, the structure in itself stays intact with no modifications.

CSS3 is structured in modules. This way the modelus can be used separately and fully independently of each others. For the web design, this is a good approach.

The W3C description on modular approach:

"Rather than attempting to shove dozens of updates into a single monolithic specification, it will be much easier and more efficient to be able to update individual pieces of the specification. Modules will enable CSS to be updated in a more timely and precise fashion, thus allowing for a more flexible and timely evolution of the specification as a whole."

The modularity helps the browse manufacturers to decide on the implementation in parts, so they do not need to wait until the end of the specification life-cycle, thus also being able to deploy ready features to the users faster. What comes to the W3C specification work on CSS3, the work may continue for years for this functionality to become Candidate Recommendations.

The specification will contain a lot of not yet implemented functionality, and of course what has been implemented already are the ones that can actually be used. And also then there maybe differences in browsers.

Using CSS3 in web applications and web pages should be thought from the criticality point of view. Branding, usability and layout related items may be well said to be related to the critical group. Most of the layout controllers are very much needed.

Many times CSS3 is being mentioned with the HTML5. However it is good to remember that HTML5 is not CSS3 and CSS3 is not HTML5 – but they do complement each others nicely. HTML5 is meant for content and CSS3 on the other hand is designed to affect on what is displayed to the user. The interaction is provided via JavaScript. With e.g. ability of shift elements around there is some functionality that was originally only possible by JavaScript, as e.g. jQuery was required. But even there are full animations and transitions, CSS3 keeps itself in the limits of visual display of an object. The :hover is an exception, it has been throughout the CSS implementation that it has offered CSS to act on user activity. Still CSS3 cannot execute itself even an interaction is there. Durations and delayed start times can be used but still one needs JavaScript to compose pages that truly are user interactive.

In this article will now go and concentrate to check through more closely on effects using CSS3 transitions.

[b]CSS3 TRANSITIONS[/b]

Transitions in CSS are something that can be used to smooth our value changes in stylesheets when triggered by user interactions like, property changes in hovering, clicking and focusing. These are important interactions between the user and the application to detect.

In principle, transitions are only simplistic rules within the stylesheet, purpose of which is to offer richer experience for the user on certain events in design.

Again, the W3C explains the CSS transitions as follows:

"CSS Transitions allow property changes in CSS values to occur smoothly over a specified duration."

With the smoothing is meant that it will animate the changing of a CSS value when the user clicks on it with mouse or touches in mobile devices (if supported) or changes the focus or active state or any changes to the element - where the browser supports these parameters. This also includes changes on element's class attribute.

Example on "hover";

This example shows how the "hover" works, and it will be added to the background color with link.This example functions so that when hovered over, the background color of the link will change. Transition is used to smooth out the change. This has been something familiar with Flash and JavaScript, but is now possible without them, using only pure CSS3:

```
<a href="#" class="test">Transition example</a>
```

```
a.test {
padding: 5px 10px;
```

```
background: #63C;
}
a.test:hover {
background: #690;
}
```

The hover affects with changing the color as seen.

The transition effect will be added next to the change of the background color. The idea is to get animation of a smooth out for the background color change. The animation effect will be the result (works on Webkit.-based browsers).

```
a.test {
padding: 5px 10px;
background: #63C;
-webkit-transition-property: background;
-webkit-transition-duration: 0.4s;
-webkit-transition-timing-function: ease;
}
a.test:hover {
background: #690;
}
```

The transition takes place in notice the three parts. We are using the following properties to further change

• transition-property: background in this example.

• transition-duration: latst for 0.3 seconds

• transition-timing-function: How fast the transition happens over time (ease)

There are timing functions to allow adjusting the speed of the transition on specific period of time. This has six possible choices:

The timing function value allows the speed of the transition

to change over time by defining one of six possibilities:

- ease,
- linear
- ease-in
- ease-out
- ease-in-out
- cubic-bezier (custom timing curve possible)

Timed animations, property changes; e.g. hovering can be controlled to be happening over a certain period of time and relate this to the transformation timing, e.g. easing out effect.

Further example:

div.someclass {
-webkit-transition: opacity 0.4s linear;

}

For example, the code above makes the browser to perform a change over period of 2 seconds, when the opacity changes. This appears a non-immediate change for the user.

These transitions do support most of CSS properties, including transformations, e.g. scale and rotate, generated gradients and masks. These can be affected by the position and size properties.

These combinations provide great possibilities of creating cool UI stuff, so to speak.

The CSS animations cover these new visual effects by allowing the developer to define an animation and specify keyframes to alter CSS properties, any of them.

As an example, here is an animation consisting of three phases. It changes the object from translucent to opaque and at the same time, increases the size with ratio 1.2x. It then collapses to have the original size and opacity. In all this repeats 10 times and creates a pulsate effect.

@-webkit-keyframes pulse {

```
  from {
    -webkit-transform: scale(1.0);
    opacity: 0.75;
```

```
  }
  50% {
    -webkit-transform: scale(1.2);
    opacity: 1.0;
  }
  to {
    -webkit-transform: scale(1.0);
    opacity: 0.75;
  }
}
```

```
img.pulse { opacity: 0.75; }
img.pulse:hover {
  -webkit-animation-name: pulse;
  -webkit-animation-duration: 0.5s;
  -webkit-animation-iteration-count: 10;
}
```

The animation in the example is assigned to an object, then provided a duration and it is also possible to define how many times to repeat. The animation will execute while hovered over.

It is quite easy to figure out by this, that even complex effects can be achieved with a pretty few lines of definition code.

Delaying the transition:

The delaying can be executed at the very moment when the triggering happens on the screen. If we for example want to delay the background transition away from the hovering, say one second later we would be using the following code:

```
transition-delay property.
a.test {
padding: 5px 10px;
background: #63C;
-webkit-transition-property: background;
-webkit-transition-duration: 0.4s;
-webkit-transition-timing-function: ease;
-webkit-transition-delay: 1s;
}
a.test:hover {
background: #690;
```

}

Shorthand transitions

There are non-delayed declarations using the transitions shorthand property, which will mean the same reault.

Example:

```
a.test {
padding: 5px 10px;
background: #63C;
-webkit-transition: background 0.4s ease;
}
a.test:hover {
background: #690;
}
```

Shorthand transition with a delay

To add the one second delay back to the shorthand version

If we wanted to add back in the half-second delay to the shorthand

version of the transition, we can do that by placing the

duration value at the end of the rule, like this:

```
a.test {
padding: 5px 10px;
background: #63C;
-webkit-transition: background 0.4s ease 0.5s;
}
a.test:hover {
background: #690;
}
```

Ensuring the best experience on a browser

We need to add prefixes to the declaration part, with -moz-, -o- prefices so, that the non-prefixed property will be last in the stack to ensure to support the final implementation after it has become finished. The background color will change with this change in the most recent versions of browsers.

```
a.test {
padding: 5px 10px;
background: #63C;
-webkit-transition: background 0.4s ease;
-moz-transition: background 0.4s ease;
-o-transition: background 0.4s ease;
transition: background 0.4s ease;
}
a.test:hover {
background: #690;
}
```

TRANSITIONING STATES

The hover is not the only state for an element. Most of the time the transitions is wanted to happen on each of those without the need of duplication. Such pseudo-classes are the “focus” and “active” for a link. Instead of having to add the transition property stack to each of those declarations, the transition instructions are attached to the normal state, thus they will be declared on ly one-time.

Here is an example, to add the same background trigger to the “focus” state. By this we enable executing the transition from either hovering over or focusing the link.

```
a.test {
padding: 5px 10px;
background: #63C;
-webkit-transition: background 0.4s ease;
-moz-transition: background 0.4s ease;
-o-transition: background 0.4s ease;
transition: background 0.4s ease;
}
a.test:hover,
a.test:focus {
background: #690;
}
```

Transitions having multiple properties

There are cases that when the background color is changed, we would also like to change the text color of the link, then also transition that alike.

This can be achieved by combining many transitions together, separated by a comma. Any of these can have varying duration and timing functions.

```
a.test {
padding: 5px 10px;
background: #63C;
-webkit-transition: background .4s ease, »
color 0.4s linear;
-moz-transition: background .4s ease, »
color 0.4s linear;
-o-transition: background .4s ease, color 0.4s linear;
transition: background .4s ease, color 0.4s linear;
}
a.test:hover,
a.test:focus {
color: #030;
background: #690;
}
```

Transitioning all properties eglible

Another method to do what was done up there is by using the “all” value. This will transition all available properties.

This implemented in the previous example used:

a.test {

padding: 5px 10px;

background: #63C;

-webkit-transition: **all** 0.4s ease;

-moz-transition: **all** 0.4s ease;

-o-transition: **all** 0.4s ease;

transition: **all** 0.4s ease;

}

a.test:hover,

a.test:focus {

```
color: #030;

background: #690;

}
```

By doing this we can get all the changes happening on :hover, :focus, or:active events without having to list each property you'd like to transition:

Properties supporting transitioning

These include opacity, position and font-size, the full chart is available from W3C.

The role of CSS3 is not to replace JavaScript

CSS3 has seen to interfere with the original rule of good design for the web, which is that the HTML is considered to be for the content, whereas CSS would be for the display and in between, the JavaScript working as the glue for the interactivity.

Now, when the elements are allowed to be changed around, the CSS3 thus has taken over for some of that what JavaScript has been all about. Earlier, e.g. fade in and fade out, smooth move around or rescaling, some jQuery would have been needed.

There are opinions that the CSS3 has kind of expanded beyond the borders of CSS, and not only anymore being to deal with the visual display. This is not exactly so.

If we add to the category both animations and transitions, the CSS3 still only deals with the way it would be with the visual interpretation of an object. As an exception, the “:hover” has allowed CSS to capture user actions, the CSS3 is unable to execute itself regarding interaction.

Even with durations and delayed start times, JavaScript is however required for creating interactive web pages.

The role of the CSS3 is to ease complicated interactions with JavaScript, code implementation becomes faster. Without the need to use jQuery or similar ways, the elements can become visually reactive using the Javascript just to launch a CSS animation.

There is extensions in JavaScript to handle and launch CSS animations, also to listen to objects for animation begin and end.