Introduction to CSS3

CSS3 is the new version of CSS. Prior we had CSS version 2s.

CSS3 is modular. It comes up with different set of modules with each module having its independent path.

CSS3 comes up with improved capabilities and new features.

Like HTML5, CSS3 also has varied support across browsers. The support for the newer properties is increasing as browsers are increasingly adding support to the newer features

CSS New features

Following are the new CSS3 Features

- Box Sizing
- Multiple Backgrounds
- Border Radius
- Box Shadow
- Text Shadow
- Gradients
- transition
- Fonts
- Transforms
- Flexible boxes

Box-sizing

The width of the html element comprises of padding and border width.

Example: A div element width is actually addition of content area + padding + border width. This sometimes creates confusion as we have to add padding and border width to the content to get the final width. CSS3 comes up with box-sizing property. This property has 3 values : content-box padding-box and border-box ,

- The default value is content-box means that the width of an element includes its border width and padding value.
- padding-box value means that only padding will be included in the width of the div.
- border-box means that only border and padding both will be included when calculating the width of the div.

This property eliminates the task of adding border width and padding every time go get the div width .

Multiple backgrounds

With CSS3 we can apply multiple backgrounds to an element. In CSS2 we could only get one background image. We can also specify the position for the images. This way we can display different images at different positions too! We can also set whether the images need to be repeated or not using the background-repeat property. Below is the example

```
h3 {

 background-image: url(bgimg1.png),url(bgimg2.png);

background-position: top right, center left;

background-repeat: no-repeat, no-repeat;

}
```

border-radius

Prior to CSS3, It was very difficult to get rounded borders. But CSS3 comes up with a new border-radius property to set the radius around the element. We can also set the radius of all the four sides. Below is the example,

```
h3 {
border-top-left-radius: 11px;
 border-top-right-radius: 12px;
border-bottom-right-radius: 15px;
border-bottom-left-radius: 25px;
}
```

In case we want to set all the sides with the same radius, then we just need to use the border-radius property.

h3{ Border-radius:10px; }

Alternatively we can use a shortcut for all the four side as border-radius:

<top left> <top right> <bottom right> <bottom left>

See the example below

h3{ Border-radius: 11px 12px 15px 25px ; }

box-shadow

The box-shadow property is used to create shadow around an element.

This is the format: box-shadow:

<inset> <x-offset> <y-offset> <blur-radius> <spread-radius> <color>

inset: we don't need to specify it if we want the default shadow going outside. If we need to get the shadow inside, then we can use this property.

x-offset and y-offset: set the offset x and y values.

blur-radius: sets the blur for the shadow

spread-radius : sets the spread radius for the shadow

color :sets the color of the shadow

```
h3{
box-shadow: 2px 4px 3px #000FF;
}
```

CSS3 also gives us flexibility to specify multiple shadows! Just separate using commas as shown below

```
h3 {
box-shadow: inset 2px31px 2px green, 1px 1px 2px #000FF;
 }
```

text-shadow

The text-shadow property is very similar to box-shadow property. The difference is that it is used to create shadow around the text. See the format below

text-shadow: <offset-x> <offset-y> <blur-radius> <color>

offset-x,offset-y,blur-radius and color are similar to box-shadow property. Below is the example

```
 h3{
box-shadow: 2px 4px 3px blue;
}
```

gradient(linear)

CSS3 gradients, display smooth transitions between two or more specified colors either radially or linearly.

There are 2 types of gradients.

1)Linear gradients

2)Radial gradients

Below is syntax for linear gradient linear-gradient:

(<angle> to <side-or-corner>, <color-stops>)

angle: The angle for the gradient. 0 degrees angle means gradient begins from bottom to top, 90 degree means left to right,180 means top to bottom and so on. Let's see the example of linear gradient. This sets gradient on a div towards right

```
div{

background:linear-gradient(to right, red , green)

}
```

To set gradient from top to bottom do this

```
div{

 linear-gradient(to bottom right, red ,green)

 }
```

gradient(radial)

Radial gradient creates a gradient that extends from the center of the element, extending outward in a circular or elliptical shape.

Below is syntax for linear gradient

```
radial-gradient(<shape> <size> at <position>, <color-stop>s)
```

shape: shape has default value of ellipse. It can also be set to circle.

size: This defines the size of the gradient. It can take the following values closest-side, farthest-side, closest-corner, farthest-corner

position: defaults to center. It's the position of center of the origination of gradient

color-stops: can have multiple color stops

Below is the example code

```
div {

background: radial-gradient(circle at top, red, yellow, green);

}
```

Animation using transition

Transition allows transition between two states of a specified element. It let's an element gradually change from one style to another. For transition we need to specify the duration for effect and the css property we want to add for effect .

transition property: The properties that will change during transition.

transition-duration: The duration of effect.Note that multiple values correspond to multiple values of transition property.

transition-delay: delay in transition

transition-timing-function: specifies the speed curve for the transition. The values are ease|linear|ease-in| ease-out, ease-in-out, cubic-bezier() and step-end

Below is the example

```
div {
width: 100px;
height: 100px;
background: green;
 transition-property: width, height, background;
 transition-duration: 1s 2s 3s;
transition-timing-function: linear;
transition-delay: 1s;
 }


div:hover {
 width: 200px;
height:200px;
background:green;
}
```

Transition shorthand

We have seen earlier, how we can specify the properties using multiple values. We can do it using the shorthand notation as below

```
div {
width: 100px;
height: 100px;
background: blue;
transition: width 3s, height 2s background 4s
 }

div:hover {
width: 200px;
 height: 200px;
background: green
}
```

CSS3 fonts

Using the new @font-face rule we can define a name for our custom font and then point to the font file, usually kept at the server. This gives us flexibility to use custom fonts for our web application. Below is how we can use this property. The src property specifies the url of the custom font kept at server.

```
@font-face {
font-family: 'customfont';
src: url('customfont.eot');
font-style: normal;
font-weight: normal;
}
```

Now to use it in our css use it as below. Note that in case the custom font is not found we fall back to Arial font. If Arial not found, then

```
sans-serif. h1 {

font-family: 'customfont',  Arial, sans-serif;

 }
```

transform:translate

With the transform property in CSS3, we can translate, rotate, scale, and skew elements.

Let's start with translate property Translate property moves the element from the actual position to the coordinates specified.

Below is the syntax. Note the x and y value can be in relative length or absolute length values

```
transform: translateX(<x value>);
```

For moving along y axis:

```
transform: translateY(<y value>);
```

For both we can use:

```
transform: translate(<x value>, <y value>);
```

Example:

```
.target-element {  transform: translate(10px, 40px); }
```

transform:rotate

Rotate property, let's us rotate an element clockwise around its origin by the specified angle. Below is the syntax:

```
transform: rotate(<angle>);
```

Example

```
transform: rotate(30deg);
```

transform:scale

Scale property is used to scale the element.

Example. scale(1.5) scales the element 1.5 times the actual size. Below is the syntax:

transform: scale(<scale-x><scale-y>);

Example

target-element{ transform: scale(1.5,1.3); }

Note: If we want to scale it uniformly, we can just use the scale with one value

target-element{ transform: scale(1.5); }

transform:skew

Skew propert, skews an element around the x or y axis by the angle specified Below is the syntax:

transform: skew(<x angle><y angle>);

Example

target-element{ transform: skew(45 deg,35 deg); }

Note: If we want can also use skewX and skewY property for x and y angles separately.

target-element{ transform: skewX(30 deg); }

CSS3 Pseudo Classes

CSS3 has new Pseudo classes added. Pseudo classes are listed below,

:nth-child(N) This class matches the elements on the basis of their positions within a parent element's list of child elements. It starts from the beginning.

Example : nth:child(2) will match the second element from the beginning.

:nth-last-child(N) This class matches the elements on the basis of their positions within a parent element's list of child elements. It starts from the end.

Example: nth:child(2) will match the second element from the ending.

:nth-of-type(N) This class matches elements on the basis of their positions within a parent element's list of child elements of the same type from beginning

:nth-last-of-type(N) This class matches elements on the basis of their positions within a parent element's list of child elements of the same type from the last.

:only-child This class matches an element if it's the only child element of its parent

:last-child This class matches an element that's the last child element of its parent element

:first-of-type This class matches the first child element of the specified element

:last-of-type This class matches the last child element of the specific element type

:only-of-type This class matches an element that's the only child element of its type

:enabled This class matches user interface elements that are enabled.

:disabled matches disabled user interface elements.

:checked Pseudo-class This class matches elements that are checked.

Example : checkboxes or radio buttons

:not(S) This class matches elements that aren't matched by the specified selector

:root This class matches the element that's the root element of a document

:empty This class matches elements that have no children

:target This class matches the element that's the target of a URL fragment

CSS3 flexbox

Css3 flexbox property creates flexible boxes that are stretchable, squeezable and capable of changing visual order within the document. Flexbox is a very new feature and the browsers support is limited but is increasing in fast pace. As of now, March 2015,Old versions of IE (9) and Opera (12) don't support flexbox.

We can set the container to flexbox. Setting container to flexbox the children follow the rules defined in the parent container. The children follow the rules of flexbox instead of block, inline or inline-block rules.

Consider the CSS code below

```
<style>
        flex-type {
        display: flex;
         flex-flow : row ;
        align-items : center ;
        justify-content : center ;
        border:1px solid green; }

        .flex-type > * {
        min-width: 100px;
         min-height: 100px; }
        header,article,footer,nav        { border:1px  solid blue; }
</style>
<div class="flextype">
        <header>asdf</header>
        <article>asdf</article>
        <aside>asdf</aside>
        <footer>asdf     <nav>asdf</nav>          </footer>
</div>
```

The flex-flow property controls the axis orientation. This property can be set to column or rows. The axis is vertical if the property is set to column. To move the items to the beginning or end set the align items to flex-end or flex-start.

Controlling the containers size: We can also control whether the containers will be of fixed size or will grow or shrink with the space available. To achieve this we can set the flex-grow, flex-shrink and flex-basis property.