CSS 3

Created by :

Sangeeta Joshi

C S S 3

- CSS3 contains several new important features to enhance your designs,
- With CSS3 and HTML5, one can now create extremely modern and very stylish web designers, loaded with effects and animations.
- CSS3 selectors are relatively new and are not supported in older browser, especially the older Internet Explorer versions.

- all modern browsers supports css3 and they will continue to improve this technology.
- CSS3 and HTML5 will completely change the way websites are designed.

CSS 3 Layers

- Stacking Elements in Layers :Using z-index Property Usually HTML pages are considered two-dimensional, because text, images and other elements are arranged on page without overlapping.
- However, in addition to their horizontal and vertical positions, boxes can be stacked along the z-axis
- i.e. one on top of other by using CSS z-index property.
- This property specifies the stack level of a box whose position value is one of absolute, fixed, or relative.

CSS 3 Layers

- The z-axis position of each layer is expressed as an integer representing the stacking order for rendering
- An element with a larger z-index overlaps an element with a lower one.
- A z-index property can help you to create more complex webpage layouts.

BOX Model

All HTML elements can be considered as boxes.
In CSS, the term "box model" is used when talking about design and layout.
The CSS box model is essentially a box that wraps around every HTML element.
It consists of: margins, borders, padding, and the actual content.

BOX Model

All HTML elements can be considered as boxes.
In CSS, the term "box model" is used when talking about design and layout.
The CSS box model is essentially a box that wraps around every HTML element.
It consists of: margins, borders, padding, and the actual content.

BOX Model

Explanation of the different parts:

Content - The content of the box, where text and images appear

Padding - Clears an area around the content. The padding is transparent

Border - A border that goes around the padding and content

Margin - Clears an area outside the border. The margin is transparent

The box model allows us to add a border around elements, and to define space between elements.

BOX Model

- div {
-     width: 300px;
-     border: 25px solid green;
-     padding: 25px;
-     margin: 25px;
- }

Psudo Classes & Psudo Elements

- A CSS *pseudo-class* is a keyword added to a selector that specifies a special state of the selected element(s).

  :*hover can be used to apply a style when the user hovers over a button.*

  Pseudo-classes : to apply a style to an element not only in relation to the content of the document tree, but also *in relation to external factors like:*

  the history of the navigator (:visited, for example),

  the status of its content (like :*checked* on certain form elements),

  the position of the mouse (like :*hover*)

  In contrast to pseudo-classes, *pseudo-elements* can be used to style

  a specific part of an element.

pseudo-classes

A pseudo-class starts with a colon (:). Its syntax :

- selector:pseudo-class { property: value; } Anchor Pseudo-classes

Anchor Pseudo classes

Using anchor pseudo-classes links can be displayed in different ways:

- These pseudo-classes let you style unvisited links differently from visited ones.
- The most common styling technique is to remove underlines from visited links.

Pseudo-classes (Relational)

:empty – targets elements that don’t have any children or any text, for example an empty element such as <p></p>

:not() - Removes elements from an existing matched set that match the selector inside the parameter of :not(). all divs except those with a class of "music" = div:not(.music)

Ex:

all divs except those with a class of "music" =

:root – this one targets the root element of a document.

Pseudo-classes (Text related)

::first-letter - Selects the first letter of the text in the element. *Typical use: dropcaps.*

::first-line - Selects the first line of text in the element. *Typical use: setting the first sentence in small-caps as a typographical eye-catcher / lead-in*.

pseudo-classes

- *:first-of-type* – this targets the first of a specific type of element within a parent, and is the opposite of :last-of-type.
- *:first-child* – targets the first child element in a parent, regardless of its type. It is the opposite of :last-child.

pseudo-classes (n$_{th}$ child)

- ***:only-child*** – in case you have an element in the document tree that is the only child of its parent, it can be targeted by this pseudo-class.

- ***:nth-child(n)*** – it takes advantage of numeric (n) values and targets child elements in relation to their position within the parent. (For example, a list of blog comments would probably look more appealing with alternating background colors – this can be done using this pseudo-class.)

Attribute Selectors

- Attribute selectors are a special kind of selector that will match elements based on their attributes and attribute values.
- Their generic syntax consists of square brackets ([]) containing an attribute name followed by optional condition to match value of attribute.
- Attribute selectors can be divided into two categories :

  -Presence and value attribute selectors and –

  -Substring value attribute selectors.

Presence and value attribute selectors

- These attribute selectors try to match an exact attribute value:
- [attr] : This selector will select all elements with the attribute attr, whatever its value.
- [attr=val] : This selector will select all elements with the attribute attr, but only if its value is val.
- [attr~=val]: This selector will select all elements with the attribute attr, but only if the value val is one of a space-separated list of values contained in attr's value,

Substring value attribute selectors

- Attribute selectors in this class are also known as "RegExp-like selectors", because they offer flexible matching in a similar fashion to regular expression
- [attr|=val] : This selector will select all elements with the attribute attr for which the value is exactly val or starts with val-
- [attr^=val] : This selector will select all elements with the attribute attr for which the value starts with val.
- [attr$=val] : This selector will select all elements with the attribute attr for which the value ends with val.
- [attr*=val] : This selector will select all elements with the attribute attr for which the value contains the string

Combinators

combinators allow you to combine multiple selectors together .The four types are:

- *The descendant selector* — (space) — allows you to select an element nested somewhere inside another element (not necessarily a direct descendant; it could be a grandchild)
- The child selector — > — allows you to select an element that is an immediate child of another element.

Combinators

- *The adjacent sibling selector* — + — allows to select an element that is an immediate sibling of another element (Adjacent : (i.e. right next to it, at the same level in the hierarchy).

- *The general sibling selector* — ~ — allows to select any elements that are siblings of another element (i.e. at the same level in the hierarchy, but not necessarily right next to it).

```
section p {
  color: blue;
}
section > p {
  background-color: yellow;
}
h2 + p {
  text-transform: uppercase;
}
h2 ~ p {
  border: 1px dashed black;
}
```

Combinators

- section p selects all the <p> elements — both the first two that are direct children of the <section> element, and the second two that are grandchildren of the <section>
- section > p selects only the first two <p> elements, which are direct children of the <section> element (but not the second two, which are not direct children).
- h2 + p selects only <p> elements that come directly after <h2> elements on the same hierarchy level — in this case the first and third paragraphs.
- h2 ~ p selects any <p> elements on the same hierarchy level as (and coming after) <h2> elements — in this case all the paragraphs.

Assignment

Styling football results :

- try hand at adding attribute selectors to some rules to style a simple football results listing: There are three things to try to do here:

1. add a UK, German, and Spanish flag icon respectively to the lhs of list items. fill in appropriate attribute selectors so that the teams are given their correct country flags, matched by language.
2. make teams that are set to be promoted bold, and teams that are in danger of being relegated italic and gray. Fill in appropriate attribute selectors to match these styles to the correct teams, matched by the ***pro and rel*** strings that appear in the data-perf attribute values.

Assignment

- Rules 7–8 make teams that are set to be promoted bold, and teams that are in danger of being relegated italic and gray. Fill in appropriate attribute selectors to match these styles to the correct teams, matched by the pro and rel strings that appear in the data-perf attribute values.

Other CSS Statements:At-rules

At-rules : used to convey metadata, conditional information, or other descriptive information.

Examples include:

@charset and @import (metadata)

@media or @document (conditional information, also called nested statements.)

@font-face (descriptive information)

Specific syntax example:

@import 'custom.css';

This at-rule imports another CSS file into the current CSS.

Other CSS Statements: nested st

Nested statements are a specific subset of at-rule:

-a nested block of CSS rules that will only be applied to the document if a specific condition is matched:

@media at-rule content is applied only if the device which runs the browser matches the expressed condition;

@supports at-rule content is applied only if the browser actually supports the tested feature;

the @document at-rule content is applied only if the current page matches some conditions.

Media Query

- CSS Media Queries : a feature in CSS3 which used to specify when *certain CSS rules should be applied.*
- This allows you to apply a special CSS for mobile, or adjust a layout for print.

Media Query

- We can use as many media queries as we would like in a CSS file.
- we may use the *and* operator to require multiple queries to be true,
- we use the comma *(,)* as the or operator to separate groups of multiple queries.
- The *not* keyword can be used to alter the logic as well.

Media Query

```
@media (min-width: 801px) {
  body {
    margin: 0 auto;
    width: 800px;
  }
}
```

:only applies the nested rule when the page's width exceeds 800 pixels.

- / normal style

```
#header-image {
   background-repeat: no-repeat;
   background-image:url('image.gif');
}

// show a larger image when you're on a big screen
@media screen and (min-width: 1200px) {
   #header-image {
      background-image:url('large-image.gif');
   }
}

// remove header image when printing.
@media print {
   #header-image {
      display: none;
   }
}
```

Transitions

- CSS3 Transitions are a presentational effect
- They allow property changes in CSS values, to occur *smoothly over a specified duration* rather than happening instantaneously as is the normal behaviour.
- Transition effects can be applied to a wide variety of CSS properties, including background-color, width, height, opacity, and many more.

Transitions

- a basic example:
- The boxes initially have an orange background,
- color set to change to green on :hover.
- For the first box no transition is specified, so the change occurs instantaneously.
- For the second box a transition with a duration of 5 seconds is specified, as such the change occurs smoothly over the given duration and automatically reverses when the mouse is moved off the element.

How to Use CSS3 Transitions?

To create a transition effect, you must specify two things:

- the CSS property you want to add an effect to
- the duration of the effect

Note: If the duration part is not specified, the transition will have no effect, because the default value is 0.

How to Use CSS3 Transitions?

The transition-timing-function property specifies the speed curve of the transition effect.

The transition-timing-function property can have the following values:

- ease - specifies a transition effect with a slow start, then fast, then end slowly (this is default)
- linear - specifies a transition effect with the same speed from start to end
- ease-in - specifies a transition effect with a slow start

How to Use CSS3 Transitions?

The transition-timing-function property values continued :

ease-out - specifies a transition effect with a slow end

ease-in-out - specifies a transition effect with a slow start and end

cubic-bezier(n,n,n,n) - lets you define your own values in a cubic-bezier function

Step 1: transition-property

specify which CSS property (or properties) the transition effect will be applied to :

The Syntax:

*transition-property: none | all | [<IDENT>] [, <IDENT>]**

Examples:

transition-property: all;

transition-property: none;

transition-property: background-color;

transition-property: background-color, height, width;

Step 2: transition-duration

which determines how long take to complete their transition.

Examples:

transition-duration: 2s;

transition-duration: 4000ms;

transition-duration: 4000ms, 8000ms;

Step 3: transition-timing-function

- *Used to specify how the pace of transition changes over its duration.*
- *two ways:*
  1. *by keywords (ease, linear, ease-in, ease-out or ease-in-out),*
  2. *by defining a custom timing function (by specifying four coordinates to define a cubic bezier curve).*

The Syntax:

<timing-function> = cubic-bezier(<number>, <number>, <number>, <number>)

Examples:

transition-timing-function: ease;

transition-timing-function: ease, linear;

transition-timing-function: cubic-bezier(0.6, 0.1, 0.15, 0.8);

Step 4: transition-delay

The Syntax:

*transition-delay: <time> [, <time>]**

Examples:

```
transition-delay: 5s;
transition-delay: 4000ms, 8000ms;
transition-delay: -5s;
```

The transition shorthand property

- The Syntax:

```
transition: <transition> [, <transition>]*
<transition> = <transition-property> <transition-duration>
<transition-timing-function> <transition-delay>
```

Examples:

```
transition: background-color 3s linear 1s;
transition: 4s ease-in-out;
transition: 5s;
```

How to Use CSS3 Transitions?

Specify the Speed Curve of the Transition

Example

- #div1 {transition-timing-function: linear;}
- #div2 {transition-timing-function: ease;}
- #div3 {transition-timing-function: ease-in;}
- #div4 {transition-timing-function: ease-out;}
- #div5 {transition-timing-function: ease-in-out;}

Border Radii

- The *border-radius* property allows you to easily curve the corners of an element.
- Previously, this was a rather tricky thing to do, involving the use of images and extra markup.
- To apply a border radius to an element, simply select the element in your CSS code and apply the *border-radius* property
- Then, give the property an absolute or percentage value. The higher the value, the more curvy your corners will appear.

CSS gradients

- CSS gradients are new types of <image> added in the CSS3 Image Module.
- Using CSS gradients : smooth transitions between two or more specified colors.
- This lets you avoid using images for these effects, thereby reducing download time and bandwidth usage.
- In addition, because the gradient is generated by the browser, objects with gradients look better when zoomed, and you can adjust your layout with much more flexibility.
- Browsers support two types of gradients: *linear,* defined with *the linear-gradient()* function, and *radial*, defined with *radial-gradient().*

Gradients

- you can declare that background to be a gradient.
- Using gradients declared in CSS, rather using an actual image file, is better for control and performance.
- Gradients are typically one color that fades into another, but in CSS you can control every aspect of how that happens, from the direction to the colors

Gradients

- CSS3 defines two types of gradients:
- Linear Gradients (goes down/up/left/right/diagonally)
- Radial Gradients (defined by their center)

Linear gradients

- set a starting point and a direction (specified as an angle) along which gradient effect is applied
- Color stops : Color stops are the colors you want browser to render smooth transitions among,

Assignment on CSS3

- Create a Rainbow Effect :

2D

- Demos : F:\Demos\CSS_Demos\CSS3\2d_3dTransform
- Assignments :

Transform

- Transform explainantion
- https://codepen.io/team/css-tricks/pen/ebb6b5a5cec86aa04168f03e26c7501c :Pen by different

3D Transform

- 3D transform includes Z-axis transformation of the HTML elements.
- translate3d(<translation-value>, <translation-value>, <length>) : it defines a 3D translation. It takes three parameters x, y and z values. The z value specifies the translation in the Z-axis.
- translateZ(<length>) : To define the translation only in the Z-direction, use this transform function. It works similar to translateX() and translateY().
- scale3d(<number>, <number>, <number>) : This function does the scaling in all the three dimensions. It takes three parameters as sx, sy and sz. Each value defines scaling in the respective direction.

3D Transform

- scaleZ(<number>) : defines scaling only in one direction ie Z-direction. scaleX() and scaleY() functions also work similar to scaleZ() but in their respective directions.
- rotate3d(<number>, <number>, <number>, <angle>) : It rotates a single HTML element by the specified angle provided in the last parameter in the [tx, ty, tz] vector specified by the first three numbers.
- rotateX(<angle>), rotateY(<angle>) and rotateZ(<angle>) take only a single angle value to rotate in the respective axis.

Perspective

- main part of 3D Transform using CSS is the *perspective.*
- To activate a 3D space to make 3D transformation, you need to active it. This activation can be done in two ways as follows:
- transform: perspective(500px);
- or
- perspective: 500px;
- The functional notation is used to activate a single element
- whereas the second notation is used to apply perspective to multiple elements at a same time.

Perspective

- The value of perspective determines the intensity of the 3-D effect.
- Think of it as a distance from the viewer to the object.
- The greater the value, the further the distance, so the less intense the visual effect.
- perspective: 2000; yields a subtle 3-D effect, as if we were viewing an object from far away.
- perspective: 100; produces a tremendous 3-D effect, like a tiny insect viewing a massive object.

Perspective

- By default, the vanishing point for a 3-D space is positioned at its centre.
- You can change the position of the vanishing point with perspective-origin property.

- Step 1 :
- https://codepen.io/sangeetaj/pen/WOLpQE

- Step 2
- https://codepen.io/sangeetaj/pen/yXGMzx

Animation