CSS- More Tags

Margin

- The CSS margin properties are used to create space around elements, outside of any defined borders.
- CSS has properties for specifying the margin for each side of an element:
- •margin-top
- •margin-right
- •margin-bottom
- •margin-left

- All the margin properties can have the following values:
- •auto - the browser calculates the margin
- •length - specifies a margin in px
- •% - specifies a margin in % of the width of the containing element
- •inherit - specifies that the margin should be inherited from the parent element

Example 1

```
<style>
div {
    border: 1px solid black;
    margin-top: 200px;
    margin-bottom: 100px;
    margin-right: 10px;
    margin-left: 200px;
    background-color: lightblue;
}
</style>
```

```
<head>
<style>
ul {
    list-style-type<html>
: none;
    margin: 0;
    padding: 0;
}

li a {
    display: block;
    width: 60px;
    background-color: #dddddd;
}
</style>
</head>
```

```
<body>

<ul>
  <li><a href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>
</ul>

<p>A background color is added to the links to show the link area.</p>
<p>Notice that the whole link area is clickable, not just the text.</p>

</body>
</html>
```

CSS Transforms

- CSS transforms allow you to translate, rotate, scale, and skew elements.
- A transformation is an effect that lets an element change shape, size and position.
- CSS supports 2D and 3D transformations.
- following 2D transformation methods:
- •translate()
- •rotate()
- •scale()
- •skewX()
- •skewY()
- •matrix()

The translate() Method

- The translate() method moves an element from its current position (according to the parameters given for the X-axis and the Y-axis).
- div {

```
    -ms-transform: translate(50px, 100px); /* IE 9 */
    -webkit-transform: translate(50px, 100px); /* Safari */
    transform: translate(50px, 100px);
}
```

The rotate() Method

- The rotate() method rotates an element clockwise or counter-clockwise according to a given degree.
- The following example rotates the element clockwise with 20 degrees:
- Example
- div {
- -ms-transform: rotate(20deg); /* IE 9 */
- -webkit-transform: rotate(20deg); /* Safari */
- transform: rotate(20deg);
- }

The scale() Method

- The scale() method increases or decreases the size of an element (according to the parameters given for the width and height).

- The following example increases the element to be two times of its original width, and three times of its original height:

- Example

```
div {
    -ms-transform: scale(2, 3); /* IE 9 */
    -webkit-transform: scale(2, 3); /* Safari */
    transform: scale(2, 3);
}
```

The skewX() Method

- The skewX() method skews an element along the X-axis by the given angle.
- The following example skews the element 20 degrees along the X-axis:
- Example
- div {
- -ms-transform: skewX(20deg); /* IE 9 */
- -webkit-transform: skewX(20deg); /* Safari */
- transform: skewX(20deg);
- }

The skewY() Method

- The skewY() method skews an element along the Y-axis by the given angle.

- The following example skews the element 20 degrees along the Y-axis:

- Example

```
div {
    -ms-transform: skewY(20deg); /* IE 9 */
    -webkit-transform: skewY(20deg); /* Safari */
    transform: skewY(20deg);
}
```

The skew() Method

- The skew() method skews an element along the X and Y-axis by the given angles.

- The following example skews the element 20 degrees along the X-axis, and 10 degrees along the Y-axis:

- Example

```
div {
    -ms-transform: skew(20deg, 10deg); /* IE 9 */
    -webkit-transform: skew(20deg, 10deg); /* Safari */
    transform: skew(20deg, 10deg);
}
```

The matrix() Method

- The matrix() method combines all the 2D transform methods into one.
- The matrix() method take six parameters, containing mathematic functions, which allows you to rotate, scale, move (translate), and skew elements.
- The parameters are as follow:
  matrix(scaleX(),skewY(),skewX(),scaleY(),translateX(),translateY())
- Example
- div {
- -ms-transform: matrix(1, -0.3, 0, 1, 0, 0); /* IE 9 */
- -webkit-transform: matrix(1, -0.3, 0, 1, 0, 0); /* Safari */
- transform: matrix(1, -0.3, 0, 1, 0, 0);
- }

CSS Transitions

- CSS transitions allows you to change property values smoothly (from one value to another), over a given duration.
- 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
-
```
div {
    width: 100px;
    height: 100px;
    background: red;
    -webkit-transition: width 2s; /* Safari */
    transition:  background 2s;

    transition-delay:4s;
}
```
-
```
div:hover {
    background:blue;
}
```

Remember these 4 properties to work with transitions

- transition-property
- transitions-duration
- transition-delay
- transition-timing-function

```
div {
    -webkit-transition: width 2s, height 4s; /* Safari */
    transition: width 2s, height 4s;
}
```

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

- The transition-delay property specifies a delay (in seconds) for the transition effect.
- The following example has a 1 second delay before starting:
- Example
- div {
- -webkit-transition-delay: 1s; /* Safari */
- transition-delay: 1s;
- }

Transition + Transformation

- The following example also adds a transformation to the transition effect:
- **Example**
- div {
  -webkit-transition: width 2s, height 2s, -webkit-transform 2s; /* Safari */
  transition: width 2s, height 2s, transform 2s;
  }

CSS Animations

- An animation lets an element gradually change from one style to another.
- You can change as many CSS properties you want, as many times you want.
- Basic difference between transitions and animations
  - Transitions require a trigger to start. Ex: on hover
  - Animations do not require one. When the page loads they will automatically start
  - Transitions can only change from one state to another
  - Animations can change from no of intermediate states to final state.
- We write animations under @keyframes
- This rule specifies what styles the element will have at certain times.
- Next step is to bind an animations to an element

Remember these properties to work with animations

- animation-name
- animation-duration
- animation-delay
- animation-iteration-count
- animation-direction
- animation-timing-function
- animation-fill-mode
- animation

- When you specify @keyframes, you can write in 2 ways

```
@keyframes example {
  from {background-color: red;}
  to {background-color: yellow;}
}


div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}
```

Here example is name of animation.
"from" and "to" are keywords(which represents 0% (start) and 100% (complete)).

```
@keyframes example {
  0%   {background-color: red;}
  25%  {background-color: yellow;}
  50%  {background-color: blue;}
  100% {background-color: green;}
}
div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}
```

Here example is name of animation
By using percent, you can add as many style changes as you like.

- animation-duration property defines how long time an animation should take to complete. Default value is 0s
- animation-delay property specifies a delay for the start of an animation. Negative values are also allowed. If using negative values, the animation will start as if it had already been playing for *N* seconds.
- animation-iteration-count property specifies the number of times an animation should run.
- animation-direction property specifies whether an animation should be played forwards, backwards or in alternate cycles. It has values normal, reverse,alternate, alternate-reverse
- animation-timing-function property specifies the speed curve of the animation. Values are ease,ease-in,ease-out,linear
- animation-fill-mode property specifies a style for the target element when the animation is not playing. Values are none,forwards,backwards,both