Web design and development II (CW2)

Class 10

Managing images with CSS

Full width images

For a container's images to be all full width, all there is to do is to create a class with the property ***max-width: 100%***. The image will then use 100% of the horizontal space available.

It is also possible to use the property ***object-fit*** with the same values as for background-size (contain, cover, etc.), but it is not supported by every browsers yet.

Limiting images width when using relative measurement units

To avoid degradation of quality to images that would be enlarged beyond their original size, it is possible to limit the site of images display. It is possible to use a width and a max width.

Cent an image horizontally like it is done for containers

Al there is to do is to define your image as a block (like a div) using ***display: block***, so ***margin: auto*** works on it.

Rounded corners

Just like for containers, ***border-radius*** will do the job. To get a round image, simply use ***border-radius: 50%***.

Images transparency

We have used ***rgba*** to get transparency using colors, but property opacity works as well. Opacity defines 0 as transparent and 1 as opaque. All decimals can be used to adjust the transparency level. Since it is not supported on all browsers, you may have to use alternate solutions such as webkit.

HTML:

```
<div>
        <img class="photo" src="image.jpg" />
</div>
```

CSS:

```
div {
        width: 500px;
}

img.photo {
        display: block;
        width: 100%;
        max-width: 500px;
        margin: auto;
        border: 10px solid red;
        border-radius: 40px;
        opacity: .5;
}
```

Filter effects

Many CSS filters allow to modify images aspect. Please note these filters aren't supported by every browsers.

Blur:

filter: blur(4px);

Luminosity:

.filter: brightness(250%);

Contrast:
filter: contrast(180%);

Grayscale:
filter: grayscale(100%);

Tint:
filter: hue-rotate(180deg);

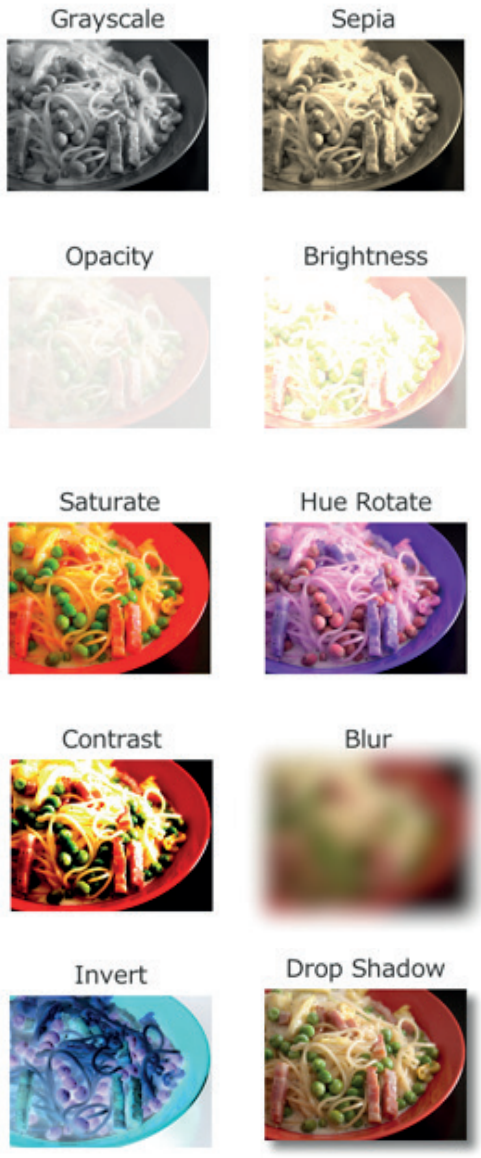
Invert (negative):
filter: invert(100%);

Opacity:
filter: opacity(50%);
ALSO: opacity: .5;

Saturation:
filter: saturate(7);

Sepia:
filter: sepia(100%);

Drop shadow:
filter:drop-shadow(8px 8px 10px #dddddd);



Transform functions

Many CSS functions allow to modify images aspect. Please note these filters aren't supported by every browsers.

translate()

Moves the element from its original position.
transform: translate(50px, 100px);

rotate() [rotateX(), rotateY(), rotateZ()]

Perform a clockwise or counterclockwise rotation base on the elemnts anchor point (center).
transform: rotate(20deg);

scale()

Scales the element. In the following example, the element will be two times higer and three times wider than its original dimensions.
transform: scale(2, 3);

skewX()

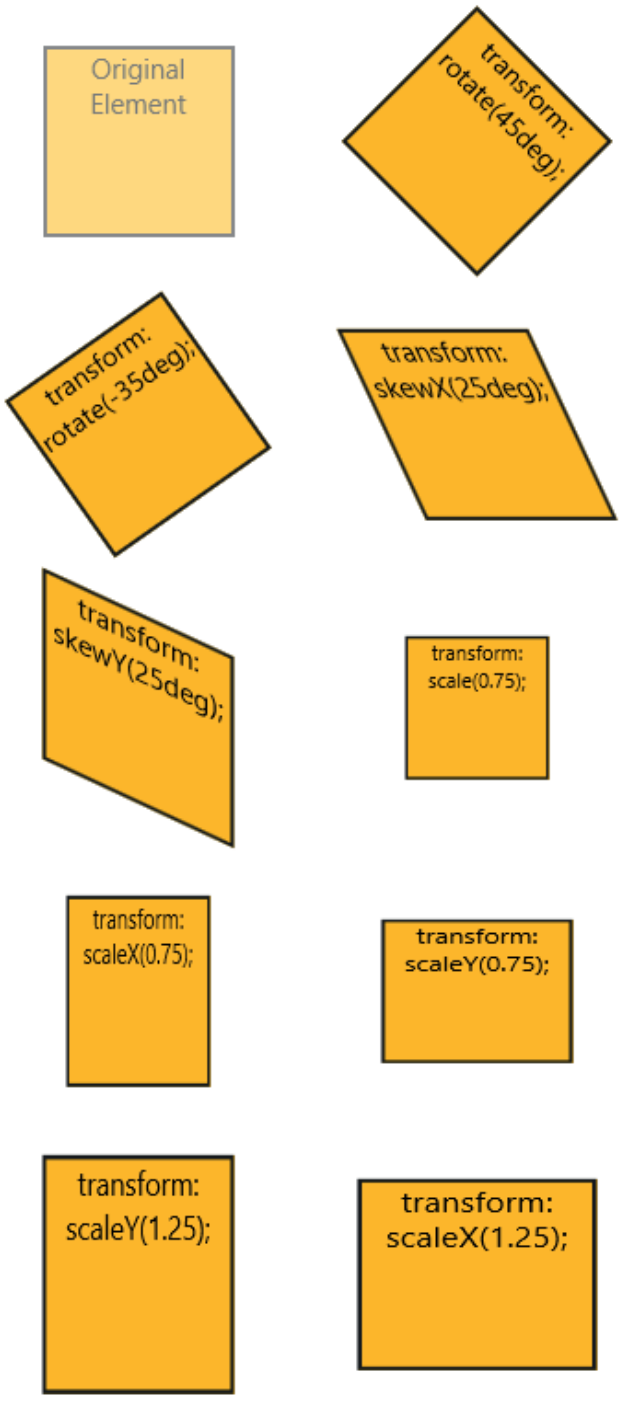
Skews the element. The first value represents the horizontal axis (x) and the second the vertical axis (y).
skew(20deg, 10deg);

skewX() / skewY()

Skews the element on one axis only.
transform: skewY(20deg);

matrix()

Shorthand performing all transformations using the following order:
matrix(scaleX(),skewY(),skewX(),scaleY(),translateX(),translateY())
transform: matrix(1, -0.3, 0, 1, 0, 0);



Shadows

Text shadow

Creating shadows on text is pretty easy and does not request using filters. Simply use the property ***text-shadow***.

```
text-shadow: 2px 2px 5px #dddddd;
```

The two first values are the horizontal and vertical offset, the third value is the blur level, and the last value is the shadow's color.

It is also possible to create multiple shadows:

```
text-shadow: 0 0 3px rouge, 0 0 5px blue;
```

Containers shadow

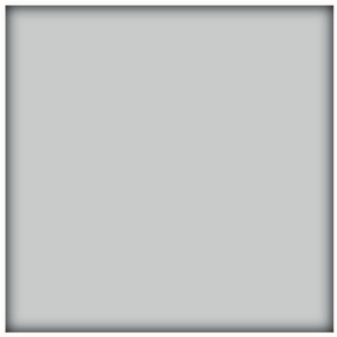
Just like for text, container has a property for shadows: ***box-shadow***:

```
box-shadow: 10px 10px 5px grey;
```

Internal shadows

It is possible to create an inner shadow effect using the inset value withing the box-shadow property:

```
box-shadow: inset 0 0 10px #000000;
```



Gradients

Many types of gradients are possible in CSS.

Linear gradients

Linear gradients make it possible to create a gradient from one color to anther in a desired direction.

The create such a gradient in a container, use the property ***background-image***, then use the value ***linear-gradient(color1,color2)***.

Gradient's orientation

The default orientation of gradient is top to bottom, but you can customize this easily indicating ***to right***, ***to left***, ***to bottom*** and ***to top***.

Diagonal gradients

Again, simply indicate the direction wanted.

Example :
to bottom right

You can also precisely define the angle using degrees.

Example :
45deg

Multiple colors gradients

Complex gradient can also be produced. All that's needed to do is to specify the colors to be used in the wanted order.

To produce transparency, simply use ***rgba()***.

```
div {
        background-image: linear-gradient(red, yellow);
}
```

```
div {
        background-image: linear-gradient(to right, red, yellow);
}
```

```
div {
        background-image: linear-gradient(to bottom right, red, yellow);
}
```

```
div {
        background-image: linear-gradient(-45deg, red, yellow);
}
```

```
div {
        background-image: linear-gradient(red, green, yellow);
}
```

Radial gradients

Radial gradient produces a rounded gradient going from the center toward the exterior of its container.

To do so, use the value ***radial-gradient*** in the property ***background-image***.

```
div {
background-image: radial-gradient(red, green, yellow);
}
```

Define the form

To define the for of the radial gradient, define it using ***circle*** or ***ellipse***) before specifying colors.

```
div {
background-image: radial-gradient(circle, red, yellow);
}
```

Defining gradients span

It is also possible to specify the span size of the different colors used with a percentage (which also works with linear gradients).

```
div {
background-image: radial-gradient(circle red 5%, green 15%, yellow 55%);
}
```