HTML5& CSS3

Presentation by Uplatz

Contact Us: https://training.uplatz.com/

Email: info@uplatz.com

Phone:+44 7836 212635

Uplatz

Table Of Contents:

- Background Color
- Background Gradients
- Background Image
- Background Shorthand
- Background Position
- Background Size
- Multiple Background Image

Uplatz

Backgrounds:

➢ With CSS you can set colors, gradients, and images as the background of an element.

➢ It is possible to specify various combinations of images, colors, and gradients, and adjust the size, positioning, and repetition (among others) of these.

Background Color:

➢ The background-color property sets the background color of an element using a color value or through keywords, such as transparent, inherit or initial.

➢ transparent, specifies that the background color should be transparent.

➢ This is default.inherit, inherits this property from its parent element.

Uplatz

initial, sets this property to its default value.

➤ This can be applied to all elements, and ::first-letter/::first-line pseudo-elements.

➤ Colors in CSS can be specified by different methods.

Color names

CSS

div {

 background-color: red; /* red */

}

HTML

<div>This will have a red background</div>

➤ The example used above is one of several ways that CSS has to represent a single color.

Hex color codes:

➤ Hex code is used to denote RGB components of a color in base-16 hexadecimal notation.

➤ #ff0000, for example, is bright red, where the red component of the color is 256 bits (ff) and the corresponding green and blue portions of

➤ the color is 0 (00).

➤ If both values in each of the three RGB pairings (R, G, and B) are the same, then the color code can be shortened into three characters (the first digit of each pairing).

➤ #ff0000 can be shortened to #f00, and #ffffff can be

➤ shortened to #fff.

Uplatz

Hex notation is case-insensitive.

body {
 background-color: #de1205; /* red */
}
.main {
 background-color: #00f; /* blue */
}
RGB / RGBa:

➤ Another way to declare a color is to use RGB or RGBa.

➤ RGB stands for Red, Green and Blue, and requires of three separate values between 0 and 255, put between brackets, that correspond with the decimal color values for respectively red, green and blue.

➤ RGBa allows you to add an additional alpha parameter between 0.0 and 1.0 to define opacity.

header {

background-color: rgb(0, 0, 0); /* black */

}

footer {

background-color: rgba(0, 0, 0, 0.5); /* black with 50% opacity */

}

HSL / HSLa:

➤ Another way to declare a color is to use HSL or HSLa and is similar to RGB and RGBa.

➤ HSL stands for hue, saturation, and lightness, and is also often called HLS:

- Hue is a degree on the color wheel (from 0 to 360).
- Saturation is a percentage between 0% and 100%.
- Lightness is also a percentage between 0% and 100%.
- HSLa allows you to add an additional alpha parameter between 0.0 and 1.0 to define opacity.

li a {
background-color: hsl(120, 100%, 50%); /* green */
}
#p1 {
background-color: hsla(120, 100%, 50%, .3); /* green with 30% opacity */
}
Interaction with background-image

➢ The following statements are all equivalent:

```
body {
 background: red;
 background-image:
url(partiallytransparentimage.png);
}
body {
 background-color: red;
 background-image:
url(partiallytransparentimage.png);
}
body {
 background-image:
url(partiallytransparentimage.png);
 background-color: red
```

Uplatz

```
}
body {
 background: red url(partiallytransparentimage.png);
}
```

➤ They will all lead to the red color being shown underneath the image, where the parts of the image are transparent, or the image is not showing (perhaps as a result of background-repeat).

➤ Note that the following is not equivalent:

```
body {
 background-image:
url(partiallytransparentimage.png);
 background: red;
}
```

Background Gradients:

➤ Gradients are new image types, added in CSS3.

➤ As an image, gradients are set with the background-image property, or the background shorthand.

➤ There are two types of gradient functions, linear and radial.

➤ Each type has a non-repeating variant and a repeating

variant:

➤ linear-gradient()

➤ repeating-linear-gradient()

➤ radial-gradient()

➤ repeating-radial-gradient()

➤ linear-gradient()

A linear-gradient has the following syntax

background: linear-gradient(<direction>?, <color-stop-1>, <color-stop-2>, ...);

Value   Meaning

<direction>   Could be an argument like to top, to bottom, to right or to left; or an angle as 0deg,

90deg... . The angle starts from to top and rotates clockwise. Can be specified in deg, grad, rad,

or turn. If omitted, the gradient flows from top to bottom

<color-stop-list>   List of colors, optionally followed each one by a percentage or length to display it at. For example, yellow 10%, rgba(0,0,0,.5) 40px, #fff 100%...

➤ For example, this creates a linear gradient that starts from the right and transitions from red to blue

.linear-gradient {

background: linear-gradient(to left, red, blue); /* you can also use 270deg */

}

➤ You can create a diagonal gradient by declaring both a horizontal and vertical starting position.

.diagonal-linear-gradient {

background: linear-gradient(to left top, red, yellow 10%);

}

It is possible to specify any number of color stops in a gradient by separating them with commas. The following examples will create a gradient with 8 color stops

```css
.linear-gradient-rainbow {
 background: linear-gradient(to left, red, orange, yellow, green, blue, indigo, violet)
}
radial-gradient()
.radial-gradient-simple {
 background: radial-gradient(red, blue);
}
.radial-gradient {
 background: radial-gradient(circle farthest-corner at top left, red, blue);
}
```

Value	Meaning
circle	Shape of gradient. Values are circle or ellipse, default is ellipse.

farthest-corner Keywords describing how big the ending shape must be. Values are closest-side, farthestside, closest-corner, farthest-corner

top left Sets the position of the gradient center, in the same way as background-position.

Repeating gradients:

➤ Repeating gradient functions take the same arguments as the above examples, but tile the gradient across the background of the element.

.bullseye {

 background: repeating-radial-gradient(red, red 10%, white 10%, white 20%);

}

.warning {

 background: repeating-linear-gradient(-45deg, yellow, yellow 10%, black 10%, black 20%);

}

Uplatz

Value	Meaning
-45deg	Angle unit. The angle starts from to top and rotates clockwise. Can be specified in deg, grad, rad, or turn.
to left	Direction of gradient, default is to bottom. Syntax: to [y-axis(top OR bottom)] [x-axis(left OR right)] ie to top right
yellow 10%	Color, optionally followed by a percentage or length to display it at. Repeated two or more times.

➢ Note that HEX, RGB, RGBa, HSL, and HSLa color codes may be used instead of color names. Color names were used for the sake of illustration.

➢ Also note that the radial-gradient syntax is much more complex than linear-gradient, and a simplified version is shown here

Uplatz

Background Image

➤ The background-image property is used to specify a background image to be applied to all matched elements.

➤ By default, this image is tiled to cover the entire element, excluding margin.

.myClass {

background-image: url('/path/to/image.jpg');

}

➤ To use multiple images as background-image, define comma separated url()

.myClass {

background-image: url('/path/to/image.jpg'),

url('/path/to/image2.jpg');

}

Uplatz

➤ The images will stack according to their order with the first declared image on top of the others and so on.

Value   **Result**

url('/path/to/image.jpg')   Specify background image's path(s) or an image resource specified with data URI schema (apostrophes can be omitted), separate multiples by comma

none   No background image

initial   Default value

inherit   Inherit parent's value

More CSS for Background Image:

➤ This following attributes are very useful and almost essential too.

background-size: xpx ypx | x% y%;

Uplatz

background-repeat: no-repeat | repeat | repeat-x | repeat-y;

background-position: left offset (px/%) right offset (px/%) | center center | left top | right bottom;

 Background Shorthand:

➤ The background property can be used to set one or more background related properties:

Value	Description	CSS Ver.
background-image	Background image to use	1+
background-color	Background color to apply	1+
background-position	Background image's position	1+
background-size	Background image's size	3+
background-repeat	How to repeat background image	

background-origin How the background is positioned (ignored when background-attachment is fixed)

3+

background-clip How the background is painted relative to the content-box, border-box, or the padding-box   3+

background-attachment How the background image behaves, whether it scrolls along with its containing block or has a fixed position within the viewport   1+

initial   Sets the property to value to default   3+

inherit   Inherits property value from parent   2+

➢ The order of the values does not matter and every value is optional

Syntax

The syntax of the background shorthand declaration is:

background: [<background-image>] [<background-color>] [<background-position>]/[<background-size>] [<background-repeat>] [<background-origin>] [<background-clip>] [<background-attachment>] [<initial|inherit>];

Examples

background: red;

➢ Simply setting a background-color with the redvalue.

background: border-box red;

➢ Setting a background-clip to border-box and a background-color to red.

background: no-repeat center url("somepng.jpg");

- Sets a background-repeat to no-repeat, background-origin to center and a background-image to an image.

background: url('pattern.png') green;

- In this example, the background-color of the element would be set to green with pattern.png, if it is available,overlayed on the colour, repeating as often as necessary to fill the element.
- If pattern.png includes any transparency then the green colour will be visible behind it.

background: #000000 url("picture.png") top left / 600px auto no-repeat;

- In this example we have a black background with an image 'picture.png' on top, the image does not repeat in either axis and is positioned in the top left corner.

- The / after the position is to be able to include the size of thebackground image which in this case is set as 600px width and auto for the height**.**
- This example could work well with a feature image that can fade into a solid colour.

NOTE: Use of the shorthand background property resets all previously set background property values,

even if a value is not given.

- If you wish only to modify a background property value previously set, use a longhand property instead.

Background Size:

General overview:

- The background-size property enables one to control the scaling of the background-image.

- It takes up to two values, which determine the scale/size of the resulting image in vertical and and horizontal direction.
- If the property is missing, its deemed auto in both width and height.
- auto will keep the image's aspect ratio, if it can be determined.
- The height is optional and can be considered auto.
- Therefore, on a 256 px × 256 px image, all the following background-size settings would yield an image with height and width of 50 px:
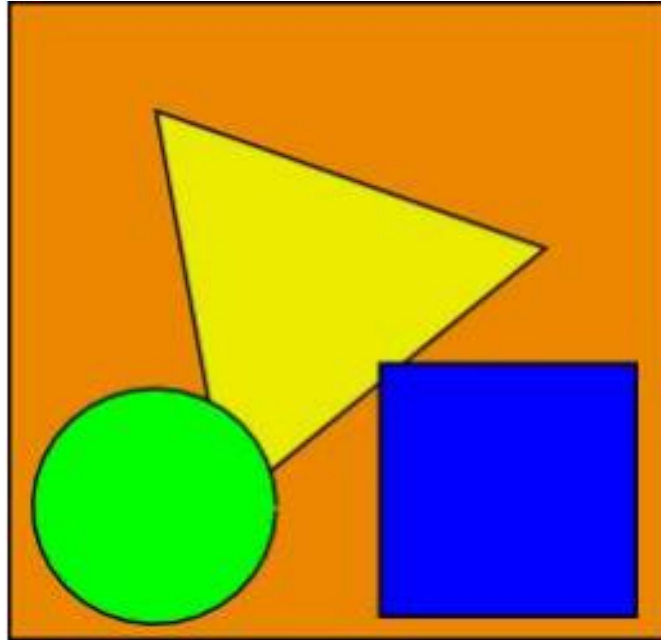
background-size: 50px;

background-size: 50px auto; /* same as above */

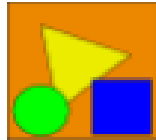**background-size: auto 50px;**

background-size: 50px 50px;

➢ So if we started with the following picture (which has the mentioned size of 256 px × 256 px),

➢ we'll end up with a 50 px × 50 px on the user's screen, contained in the background of our element:

Uplatz

➤ we'll end up with a 50 px × 50 px on the user's screen, contained in the background of our element:

➤ One can also use percentage values to scale the image with respect of the element.

➤ The following example would yield a 200 px × 133 px drawn image:

#withbackground {
 background-image: url(to/some/background.png);
 background-size: 100% 66%;
width: 200px;
 height: 200px;
padding: 0;
 margin: 0; }

Uplatz
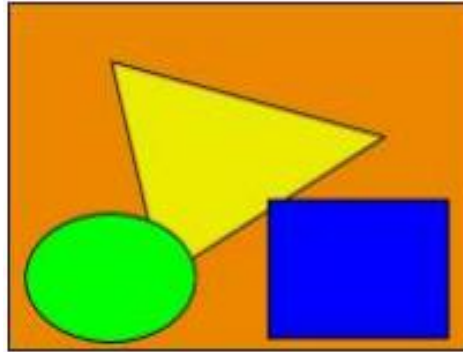
Keeping the aspect ratio:

➢ The last example in the previos section lost its original aspect ratio.

➢ The circle got into an ellipse, the square into a rectangle, the triangle into another triangle.

➢ The length or percentage approach isn't flexible enough to keep the aspect ratio at all times.

➢ auto doesn't help, since you might not know which dimension of your element will be larger.

➤ However, to cover certain areas with animage (and correct aspect ratio) completely or to contain an image with correct aspect ratio completely in a

background area, the values, contain and cover provide the additional functionality.

➤ Lets say that this is your screen, and the gray area is outside of your visible screen.

Contain:

➤ Scale the image, while preserving its intrinsic aspect ratio (if any), to the largest size such that both its

width and its height can fit inside the background positioning area.

➤ This makes sure that the background image is always completely contained in the background positioning area, however, there could be some empty space filled with your background-color in this case
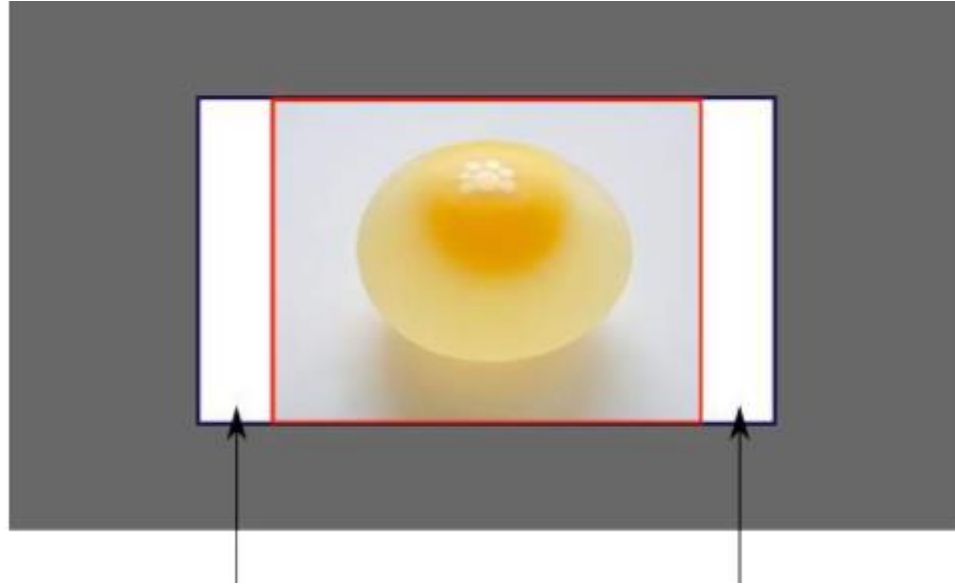
cover

➤ Scale the image, while preserving its intrinsic aspect ratio (if any), to the smallest size such that both its width and its height can completely cover the background positioning area.

➤ This makes sure that the background image is covering everything.

➢ There will be no visible background-color,however depending on the screen's ratio a great part of your image could be cut off:

Demonstration with actual code

div > div {

 background-image: url(http://i.stack.imgur.com/r5CAq.jpg);

 background-repeat: no-repeat;

```css
  background-position: center center;
  background-color: #ccc;
  border: 1px solid;
  width: 20em;
  height: 10em;
}
div.contain {
  background-size: contain;
}
div.cover {
  background-size: cover;
}
/***************************************************
  Additional styles for the explanation boxes
***********************************************************/
```

```css
div > div {
 margin: 0 1ex 1ex 0;
 float: left;
}
div + div {
 clear: both;
 border-top: 1px dashed silver;
 padding-top:1ex;
}
div > div::after {
 background-color: #000;
 color: #fefefe;
 margin: 1ex;
 padding: 1ex;
 opacity: 0.8;
```

```
display: block;
width: 10ex;
font-size: 0.7em;
content: attr(class);
}
<div>
 <div class="contain"></div>
 <p>Note the grey background. The image does not
cover the whole region, but it's fully
<em>contained</em>.
 </p>
</div>
<div>
 <div class="cover"></div>
```

<p>Note the ducks/geese at the bottom of the image. Most of the water is cut, as well as a part

of the sky. You don't see the complete image anymore, but neither do you see any background color;

the image covers all of the <code><div></code>.</p>

</div>

Background Position

➢ The background-position property is used to specify the starting position for a background image or gradient

.myClass {

background-image: url('path/to/image.jpg');

background-position: 50% 50%;

}

Uplatz

➢ The position is set using an X and Y co-ordinate and be set using any of the units used within CSS.

   Unit   **Description**

value% value%   A percentage for the horizontal offset is relative to (width of background positioning area - width of background image).

A percentage for the vertical offset is relative to (height of background positioning area - height of background image)

The size of the image is the size given by background-size.

valuepx valuepx

➢ Offsets background image by a length given in pixels relative to the top left of the background positioning area

Longhand Background Position Properties:

➤ In addition to the shorthand property above, one can also use the longhand background properties backgroundposition-x and background-position-y.

➤ These allow you to control the x or y positions separately.

The background-origin property:

➤ The background-origin property specifies where the background image is positioned.

Note: If the background-attachment property is set to fixed, this property has no effect.

➤ Default value: padding-box

➤ Possible values:

padding-box - The position is relative to the padding box

border-box - The position is relative to the border box
content-box - The position is relative to the content
box
initial
inherit
CSS
.example {
 width: 300px;
 border: 20px solid black;
 padding: 50px;
 background:
url(https://static.pexels.com/photos/6440/magazines-
desk-work-workspace-medium.jpg);
 background-repeat: no-repeat;
}

```
.example1 {}
.example2 { background-origin: border-box; }
.example3 { background-origin: content-box; }
```

HTML

```
<p>No background-origin (padding-box is default):</p>
div class="example example1">
 <h2>Lorem Ipsum Dolor</h2>
 <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
 <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.</p>
```

```html
</div>
<p>background-origin: border-box:</p>
<div class="example example2">
 <h2>Lorem Ipsum Dolor</h2>
 <p>Lorem ipsum dolor sit amet, consectetuer
adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat
volutpat.</p>
 <p>Ut wisi enim ad minim veniam, quis nostrud exerci
tation ullamcorper suscipit lobortis nisl ut
aliquip ex ea commodo consequat.</p>
</div>
<p>background-origin: content-box:</p>
<div class="example example3">
```

<h2>Lorem Ipsum Dolor</h2>

<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>

<p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut

aliquip ex ea commodo consequat.</p>

</div>

Multiple Background Image:

➤ In CSS3, we can stack multiple background in the same element.

#mydiv {

background-image: url(img_1.png), /* top image */

url(img_2.png), /* middle image */
url(img_3.png); /* bottom image */
background-position: right bottom,
left top,
right top;
background-repeat: no-repeat,
repeat,
no-repeat;
}

- Images will be stacked atop one another with the first background on top and the last background in the back.
- img_1 will be on top, the img_2 and img_3 is on bottom

➢ We can also use background shorthand property for this:

#mydiv {
 background: url(img_1.png) right bottom no-repeat,
 url(img_2.png) left top repeat,
 url(img_3.png) right top no-repeat;
}

➢ We can also stack images and gradients:

#mydiv {
 background: url(image.png) right bottom no-repeat,
 linear-gradient(to bottom, #fff 0%,#000 100%);
}

Summary:

- Background Color
- Background Gradients
- Background Image
- Background Shorthand
- Background Position
- Background Size
- Multiple Background Image

Uplatz

Thank You………

If you have any quries please write to   info@uplatz.com".

Uplatz