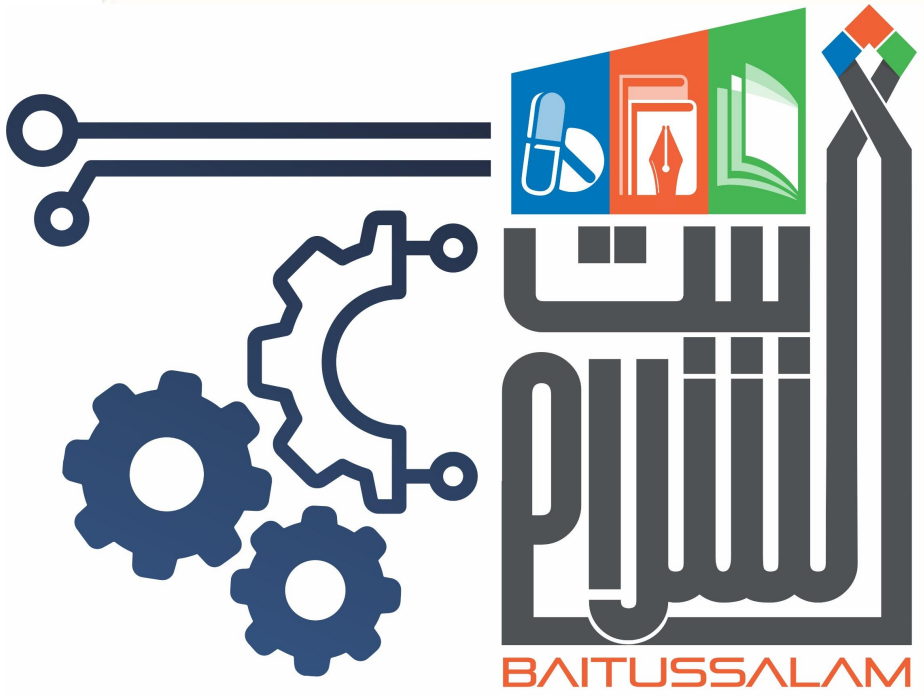


BAITUSSALAM
—TECH PARK—

BAITUSSALAM
—TECH PARK—

PSDC-150

WEB development with HTML & CSS

Working with HTML tags and attributes

BAITUSSALAM
—TECH PARK—

Continued…

Block Level Elements

Some elements will always appear to start on a new line in the browser window. These are known as **block level** elements.

Examples of block elements are
<h1>, <p>, <ul>, and <li>.


```
<h1>Hiroshi Sugimoto</h1>
<p>The dates for the ORIGIN OF ART exhibition are as
   follows:</p>
<ul>
  <li>Science: 21 Nov - 20 Feb 2010/11</li>
  <li>Architecture: 6 Mar - 15 May 2011</li>
  <li>History: 29 May - 21 Aug 2011</li>
  <li>Religion: 28 Aug - 6 Nov 2011</li>
</ul>
```

Hiroshi Sugimoto

The dates for the ORIGIN OF ART exhibition are as follows:

- Science: 21 Nov - 20 Feb 2010/11
- Architecture: 6 Mar - 15 May 2011
- History: 29 May - 21 Aug 2011
- Religion: 28 Aug - 6 Nov 2011

Inline Elements

Some elements will always appear to continue on the same line as their neighbouring elements. These are known as **inline elements**.

Examples of inline elements are
<a>, <b>, , and <img>.


```
Timed to a single revolution of the planet around
the sun at a 23.4 degrees tilt that plays out the
rhythm of the seasons, this <em>Origins of Art</em>
cycle is organized around four themes: <b>science,
architecture, history</b> and <b>religion</b>.
```

Timed to a single revolution of the planet around the sun at a 23.4 degrees tilt that plays out the rhythm of the seasons, this *Origins of Art* cycle is organized around four themes: **science, architecture, history** and **religion**.

Grouping Text & Elements

Sometimes we need to group certain elements which are related & they can be either block elements or inline elements for that HTML provides us multiple options.

tag

The element allows you to group a set of elements together in one block-level box.

For example, you might create a element to contain all of the elements for the header of your site (the logo and the navigation), or you might create a element to contain comments from visitors.

In a browser, the contents of the element will start on a new line, but other than this it will make no difference to the presentation of the page.

Using an id or class attribute on the element, however, means that you can create CSS style rules to indicate how much space the element should occupy on the screen and change the appearance of all the elements contained within it.

It can also make it easier to follow your code if you have used elements to hold each section of the page.

tag

The element acts like
an inline equivalent of the
element. It is used to either:

1. Contain a section of text where there is no other suitable element to differentiate it from its surrounding text
2. Contain a number of inline elements

The most common reason why people use elements is so that they can control the appearance of the content of these elements using CSS.

You will usually see that a class or id attribute is used with elements:

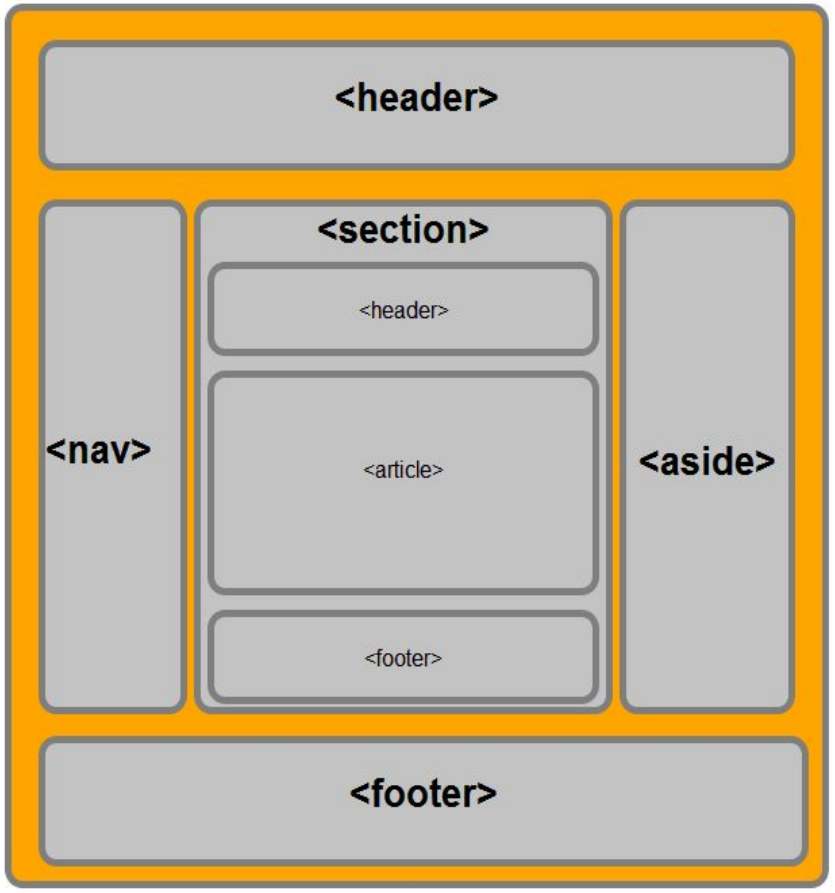
- To explain the purpose of this element
- So that CSS styles can be applied to elements that have specific values for these attributes

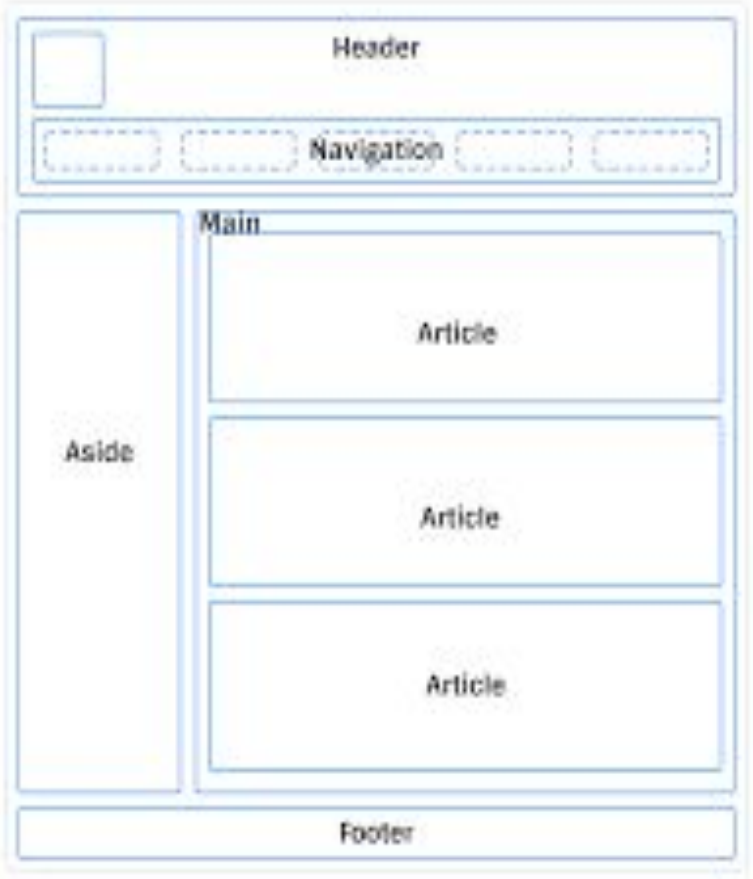
Common block level elements

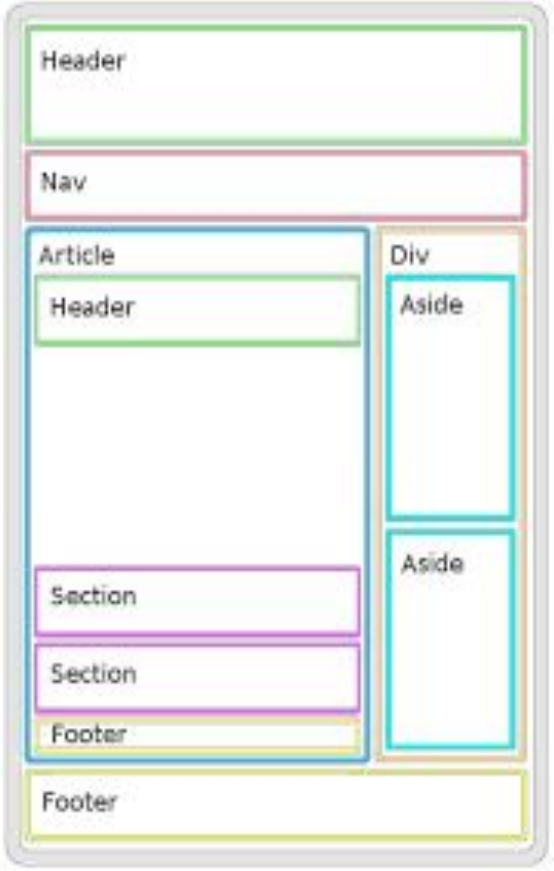
Let’s discuss some block level elements that are commonly used for structuring the web page

<header>
<footer>
<section>
<article>
<main>
<nav>
<aside>

Some layout examples







Introduction to CSS

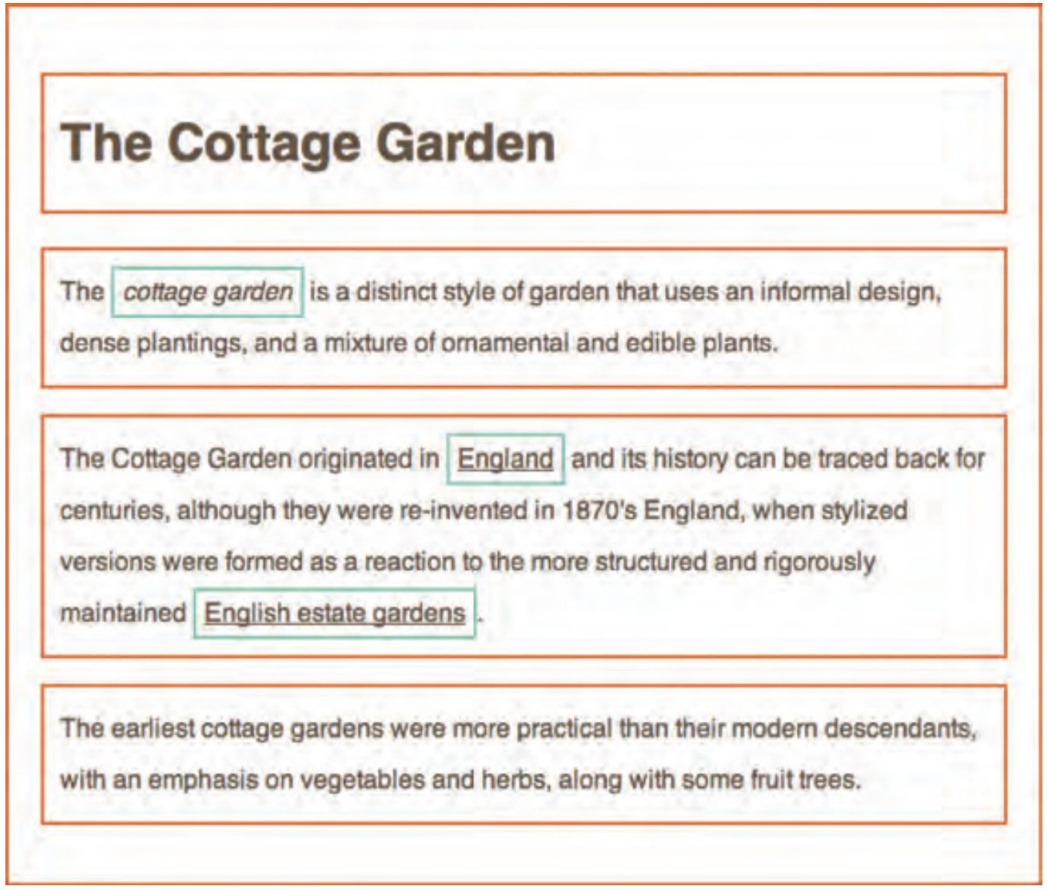
BAITUSSALAM
—TECH PARK—

CSS (cascade styling sheet)

CSS describes how HTML elements are to be displayed on screen, paper, or in other media

CSS saves a lot of work. It can control the layout of multiple web pages all at once

External stylesheets are stored in CSS files

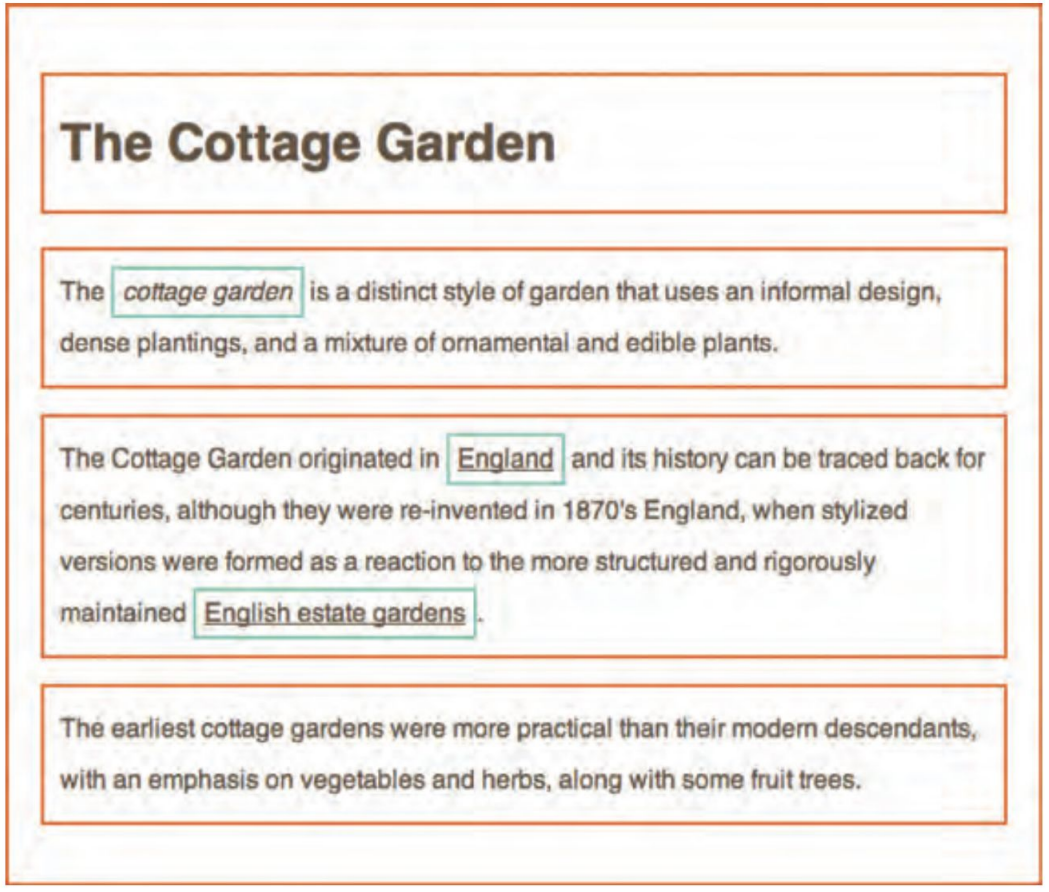


Styling with CSS

Using CSS we can style:

- **Text**
  - Typeface, Size, Color
  - Italics, bold, uppercase, lowercase, small-caps)
- **Boxes**
  - Width and height,
  - Borders (color, width, and style)
  - Background color and images
  - Position in the browser window
- **Specific**

  There are also specific ways in which you can style certain elements such as lists, tables, and forms.

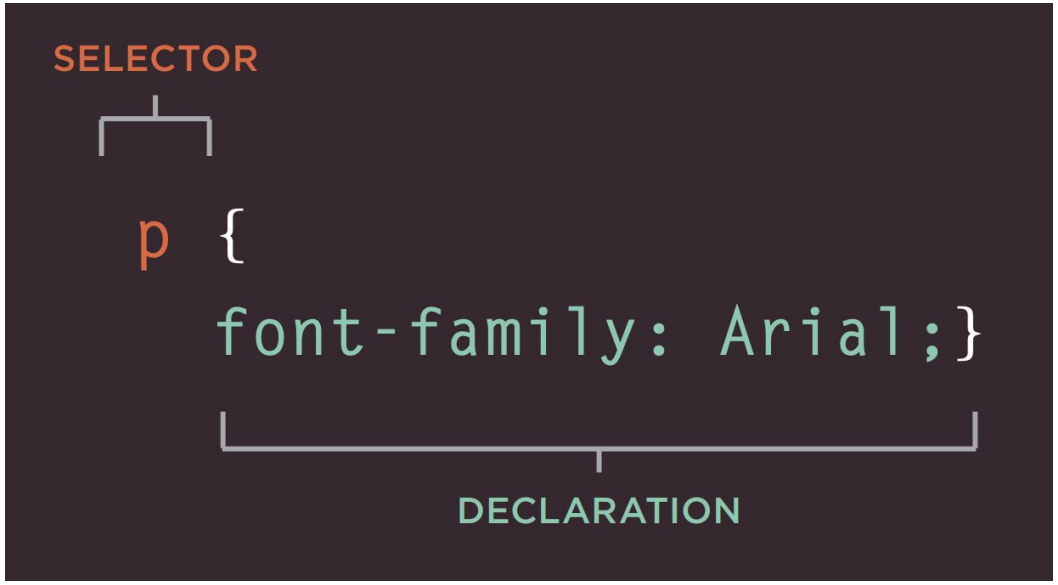


How CSS works

CSS works by associating rules with HTML elements. These rules govern how the content of specified elements should be displayed. A CSS rule contains two parts: a selector and a declaration.

Selectors indicate which element the rule applies to. The same rule can apply to more than one element if you separate the element names with commas.

Declarations indicate how the elements referred to in the selector should be styled. Declarations are split into two parts (a property and a value), and are separated by a colon.

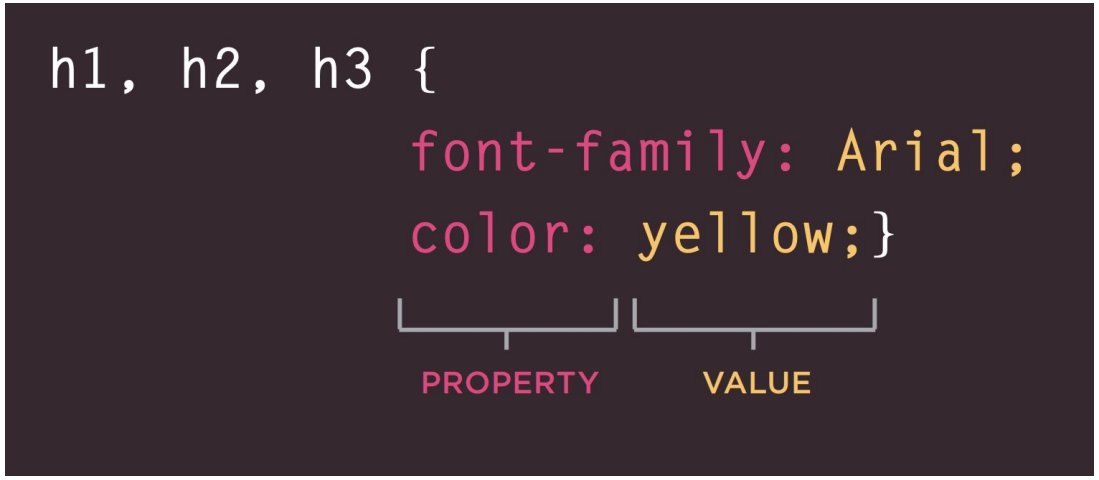


How CSS works

CSS declarations sit inside curly brackets and each is made up of two parts: a property and a value, separated by a colon. You can specify several properties in one declaration, each separated by a semi-colon.

Properties indicate the aspects of the element you want to change. For example, color, font, width, height and border.

Values specify the settings you want to use for the chosen properties. For example, if you want to specify a color property then the value is the color you want the text in these elements to be.



Internal CSS

You can also include CSS rules within an HTML page by placing them inside a **<style>** element, which usually sits inside the **<head>** element of the page. The **<style>** element should use the type attribute to indicate that the styles are specified in CSS. The value should be text/css.

When building a site with more than one page, you should use an external CSS style sheet. This:

- Allows all pages to use the same style rules (rather than repeating them in each page).
- Keeps the content separate from how the page looks.
- Means you can change the styles used across all pages by altering just one file (rather than each individual page).

```
<head>
  <title>Using Internal CSS</title>
  <style type="text/css">
    body {
        font-family: arial;
        background-color: rgb(185,179,175);}
    h1 {
        color: rgb(255,255,255);}
  </style>
</head>
<body>
  <h1>Potatoes</h1>
  <p>There are dozens of different potato
     varieties. They are usually described as
     early, second early and maincrop.</p>
</body>
```

External CSS

When we have a lot of stylings defined we can separate these styles on multiple separate files and include them in HTML file.

Similarly third party stylesheets can also be included in our HTML files.

The <link> element can be used in an HTML document to tell the browser where to find the CSS file used to style the page and it lives inside the <head> element.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Using External CSS</title>
    <link href="css/styles.css" type="text/css"
      rel="stylesheet" />
  </head>
  <body>
    <h1>Potatoes</h1>
    <p>There are dozens of different potato
       varieties. They are usually described as
       early, second early and maincrop.</p>
  </body>
</html>
```

External CSS

To define external stylesheet links link tag must have 3 attributes:

href specifies the path to the CSS file (which is often placed in a folder called css or styles)

type attribute specifies the type of document being linked to. The value should be text/css.

rel specifies the relationship between the HTML page and the file it is linked to. The value should be stylesheet when linking to a CSS file.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Using External CSS</title>
    <link href="css/styles.css" type="text/css"
      rel="stylesheet" />
  </head>
  <body>
    <h1>Potatoes</h1>
    <p>There are dozens of different potato
       varieties. They are usually described as
       early, second early and maincrop.</p>
  </body>
</html>
```

CSS Selectors

There are different types of selectors in CSS which allows us to target different HTML elements.

For example universal selector, class selector, id selector, child selector.

CSS selectors are case sensitive, so they must match element names and attribute values exactly.

Refer pg.238 of HTML & CSS book by Ducket.

CSS rules precedence

Last Rule
If in case, there are more than one classes defined with same name, then last one would take precedence over.

For example:

```
.btn {
        color: blue;
}
.btn {
        color: green;
}
```

In above example, second btn rule will take precedence.

CSS rules precedence

Specificity
The more specific rule is defined, the more precedence it will take.

For example:

```
p .text {
        color: blue;
}
.text {
        color: red;
}
```

```
<p class=”text”>my random text</p>
```

In above example, text will be shown as blue since p.text will take precedence.

```
* {
        font-size: 14px;
}
h1 {
        font-size: 24px;
}
```

```
<h1>my random heading</h1>
```

In above example, h1 will have 24px of font-size.

CSS rules precedence

Important
Adding “!important” after any property value would make it take more precedence.
For example:

```
.text {
        color: red !important;
}
.text {
        color: green;
}
```

```
<p class=”text”>my random text</p>
```

In above example, text will be shown as red since in first .text rule declaration, color property is defined as !important.

Inheritance

If you specify the font-family or color properties on the <body> element, they will apply to most child elements. This is because the value of the font-family property is inherited by child elements. It saves you from having to apply these properties to as many elements (and results in simpler style sheets).

You can compare this with the background-color or border properties; they are not inherited by child elements. If these were inherited by all child elements then the page could look quite messy.

You can force a lot of properties to inherit values from their parent elements by using inherit for the value of the properties. In this example, the element with a class called page inherits the padding size from the CSS rule that applies to the <body> element.

```
<div class="page">
  <h1>Potatoes</h1>
  <p>There are dozens of different potato
     varieties.</p>
  <p>They are usually described as early, second
     early and maincrop potatoes.</p>
</div>
```

```
body {
  font-family: Arial, Verdana, sans-serif;
  color: #665544;
  padding: 10px;}
.page {
  border: 1px solid #665544;
  background-color: #efefef;
  padding: inherit;}
```

Summary

CSS treats each HTML element as if it appears inside its own box and uses rules to indicate how that element should look.

Rules are made up of selectors (that specify the elements the rule applies to) and declarations (that indicate what these elements should look like).

Different types of selectors allow you to target your rules at different elements.

Declarations are made up of two parts: the properties of the element that you want to change, and the values of those properties. For example, the font-family property sets the choice of font, and the value arial specifies Arial as the preferred typeface.

CSS rules usually appear in a separate document, although they may appear within an HTML page.

Text

The properties that allow you to control the appearance of text.

Those that directly affect the font and its appearance (including the typeface, whether it is regular, bold or italic, and the size of the text)

Those that would have the same effect on text no matter what font you were using (including the color of text and the spacing between words and letters)

Typeface Terminology

SERIF

Serif fonts have extra details on the ends of the main strokes of the letters. These details are known as serifs.

In print, serif fonts were traditionally used for long passages of text because they were considered easier to read.

SANS-SERIF

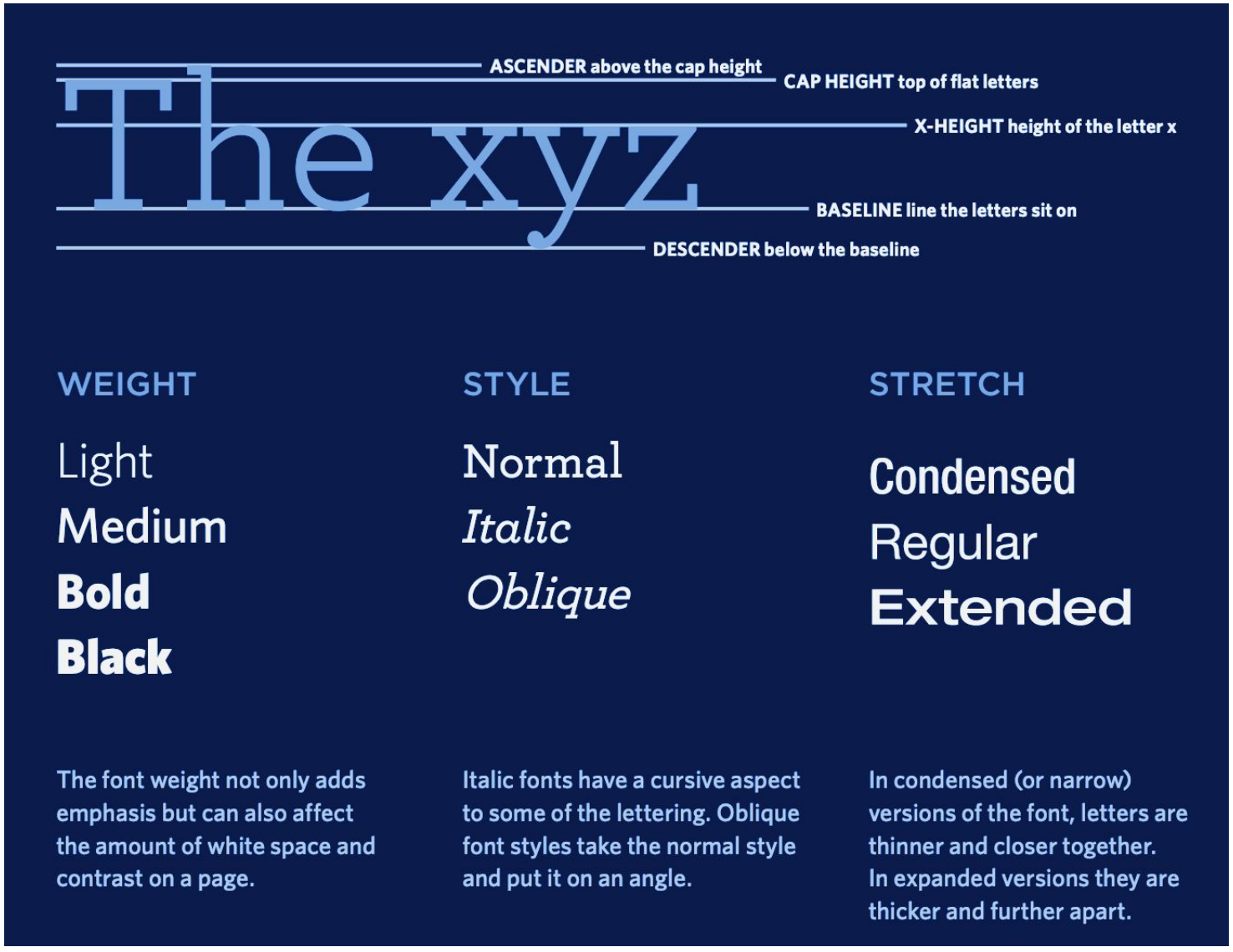
Sans-serif fonts have straight ends to letters, and therefore have a much cleaner design.

Screens have a lower resolution than print. So, if the text is small, sans-serif fonts can be clearer to read.

MONOSPACE

Every letter in a monospace (or fixed-width) font is the same width. (Non-monospace fonts have different widths.)

Monospace fonts are commonly used for code because they align nicely, making the text easier to follow.


ASCENDER above the cap height
CAP HEIGHT top of flat letters
X-HEIGHT height of the letter x
The xyz
BASELINE line the letters sit on
DESCENDER below the baseline
WEIGHT
Light
Medium
Bold
Black
STYLE
Normal
Italic
Oblique
STRETCH
Condensed
Regular
Extended
The font weight not only adds emphasis but can also affect the amount of white space and contrast on a page.
Italic fonts have a cursive aspect to some of the lettering. Oblique font styles take the normal style and put it on an angle.
In condensed (or narrow) versions of the font, letters are thinner and closer together. In expanded versions they are thicker and further apart.

Text properties

- font-family
- @font-face
- font-style
- font-size
- font-weight
- text-transform
- text-decoration
- line-height
- letter-spacing
- text-align
- text-shadow
- text-indent

```
@font-face {
  font-family: 'ChunkFiveRegular';
  src: url('fonts/chunkfive.eot');}
h1, h2 {
  font-family: ChunkFiveRegular, Georgia, serif;}
```

Boxes

Control the dimensions of your boxes

Create borders around boxes

Set margins and padding for boxes

Show and hide boxes

Boxes

Dimensions
width, min-width, max-width
height, min-height, max-height

Overflowing content
overflow: hidden/scroll

Borders
style, width, color, radius, shadow

Padding
padding: 10px 5px 3px 1px;

Margin
margin: 10px 5px 3px 1px;

Boxes

Display
The display property allows you to turn an inline element into a block-level element or vice versa, and can also be used to hide an element from the page. The values this property can take are:

inline
This causes a block-level element to act like an inline element.

block
This causes an inline element to act like a block-level element.

inline-block
This causes a block-level element to flow like an inline element, while retaining other features of a block-level element.

none
This hides an element from the page. In this case, the element acts as though it is not on the page at all (although a user could still see the content of the box if they used the view source option in their browser).

Boxes

Hiding boxes can be done by using **visibility** property

The **visibility** property allows you to hide boxes from users but It leaves a space where the element would have been.

hidden
This hides the element.

visible
This shows the element.

If the visibility of an element is set to hidden, a blank space will appear in its place.

If you do not want a blank space to appear, then you should use the display property with a value of none instead.

The End

BAITUSSALAM
—TECH PARK—