

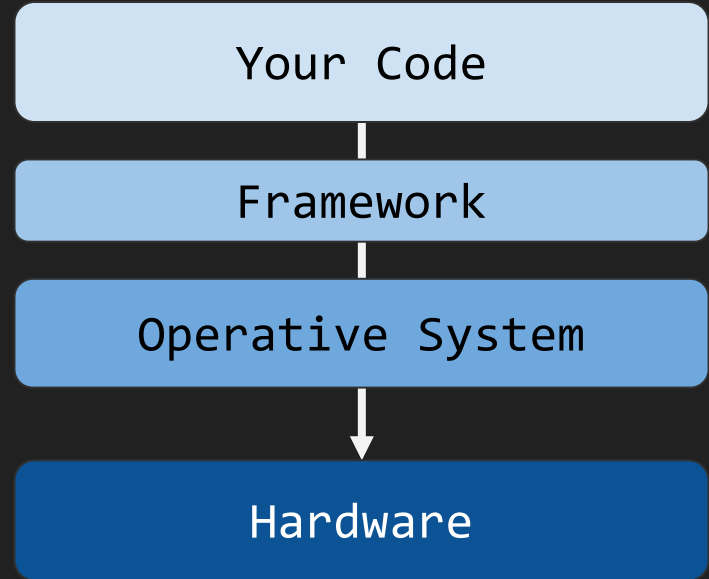


# Introduction

Problem - Programming languages do not include a library to create User Interfaces.

Solution - Frameworks

Ready to use - Web provides a very rich and simple framework to create applications



# Goals

Introduction to web technologies:

- **HTML** to create the document structure and content
- **CSS** to control its visual aspect
- **Javascript** for interactivity



# Deploy

What do we need to start:

- a good web-browser (Chrome or Firefox)
- a good text editor like Editplus (win), VSCode (cross platform), textWrangler (osx), vim (unix) or sublime text (cross platform)

# How can I test my code

- **Simple**

Just open the `index.html` from the template in your text editor and in your browser. When you do any change to the code, check it in the browser by pressing F5 (refresh site)

- **To open the developer tools press:**

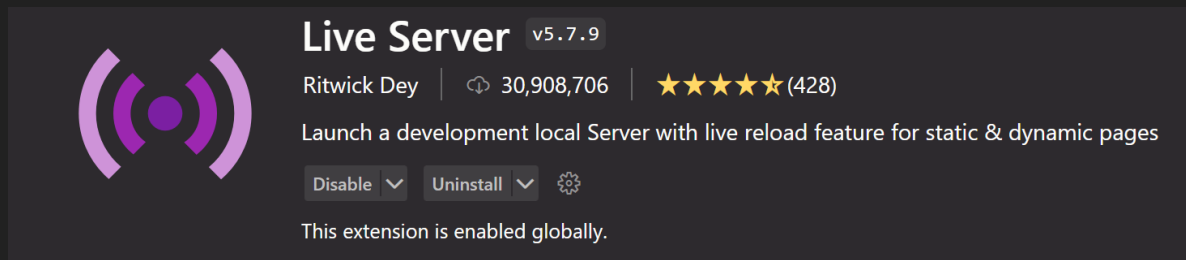
Windows: Control + Shift + I or `F12`

OSX: Command + Opt + I

Other tools are online editors like [scratchpad](#) or [htmledit](#)

- **Vscode extension**

Live server



The image shows the Live Server extension interface in VS Code. It features a purple icon of a signal tower on the left. To the right, the text 'Live Server' is displayed with a version tag 'v5.7.9'. Below this, the author 'Ritwick Dey' is listed, followed by a download count of '30,908,706' and a star rating of '★★★★★ (428)'. A description states: 'Launch a development local Server with live reload feature for static & dynamic pages'. At the bottom, there are buttons for 'Disable' and 'Uninstall', both with dropdown arrows, and a gear icon for settings. A status message at the very bottom says 'This extension is enabled globally.'

**Live Server** v5.7.9

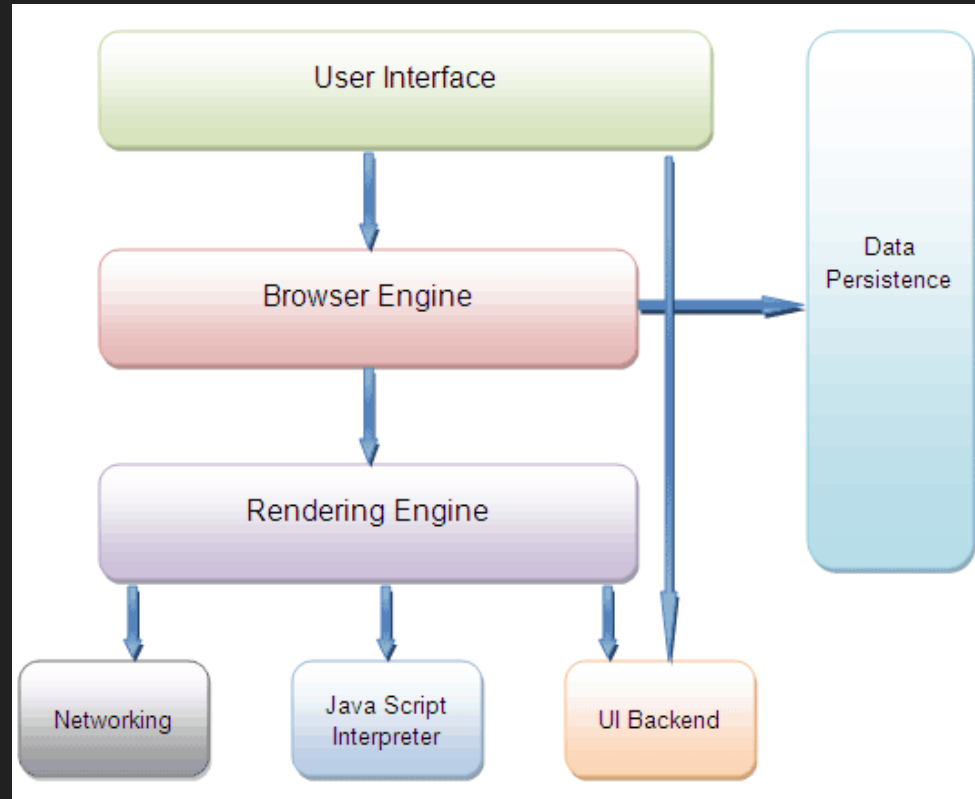
Ritwick Dey | 30,908,706 | ★★★★★ (428)

Launch a development local Server with live reload feature for static & dynamic pages

Disable Uninstall ⚙️

This extension is enabled globally.

# Inside a browser



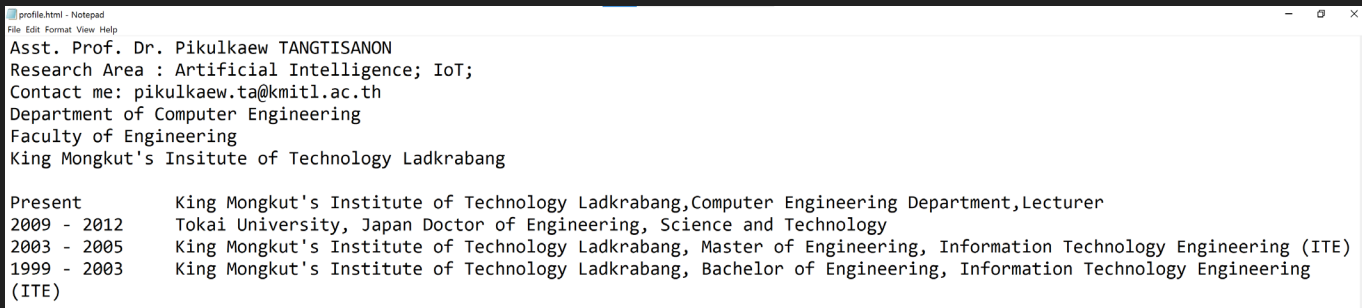
# Technologies

- HTML
- CSS
- Javascript



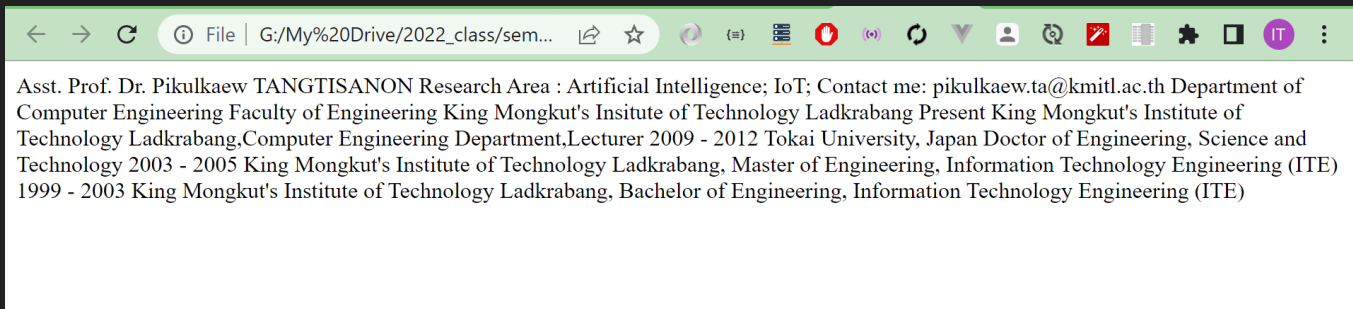
# Browsers as a renderer

- Open notepad => Create text file to introduce yourself (no code just plain text file.html)
- Open it in chrome



```
profile.html - Notepad
File Edit Format View Help
Asst. Prof. Dr. Pikulkaew TANGTISANON
Research Area : Artificial Intelligence; IoT;
Contact me: pikulkaew.ta@kmitl.ac.th
Department of Computer Engineering
Faculty of Engineering
King Mongkut's Insitute of Technology Ladkrabang

Present      King Mongkut's Institute of Technology Ladkrabang,Computer Engineering Department,Lecturer
2009 - 2012   Tokai University, Japan Doctor of Engineering, Science and Technology
2003 - 2005   King Mongkut's Institute of Technology Ladkrabang, Master of Engineering, Information Technology Engineering (ITE)
1999 - 2003   King Mongkut's Institute of Technology Ladkrabang, Bachelor of Engineering, Information Technology Engineering (ITE)
```





# HTML

HTML = Hyper Text Markup Language.

- Use to define the structure of a document or a website.
- HTML is Not Case Sensitive, but lowercase is recommended.
- HTML is **NOT** a programming language, it's a markup language.
- Series of nested tags (it is a subset of [XML](#))

```
<title>This is a title</title>
```

```
<html>
  <head>
  </head>
  <body>
  <div>

    <p>Hi</p>

  </div>
</body>
</html>
```





# HTML: syntax example

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>
<br>

</body>
</html>
```

# HTML: syntax example

Diagram illustrating HTML syntax components with annotations:

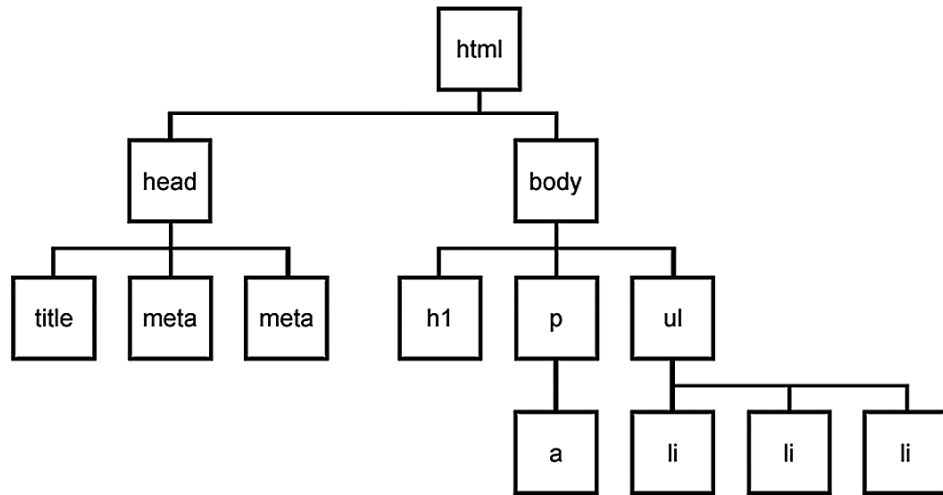
```
<div id="main">  
  <!-- this is a comment -->  
  This is text without a tag.  
  <button class="mini">press me</button>  
    
</div>
```

Annotations:

- Tag name: points to `<div`
- attributes: points to `id="main"`
- comment: points to `<!-- this is a comment -->`
- text tag: points to `This is text without a tag.`
- self-closing tag: points to ``

# DOM is a tree

Every node can only have one parent, and every node can have several children, so the structure looks like a tree.

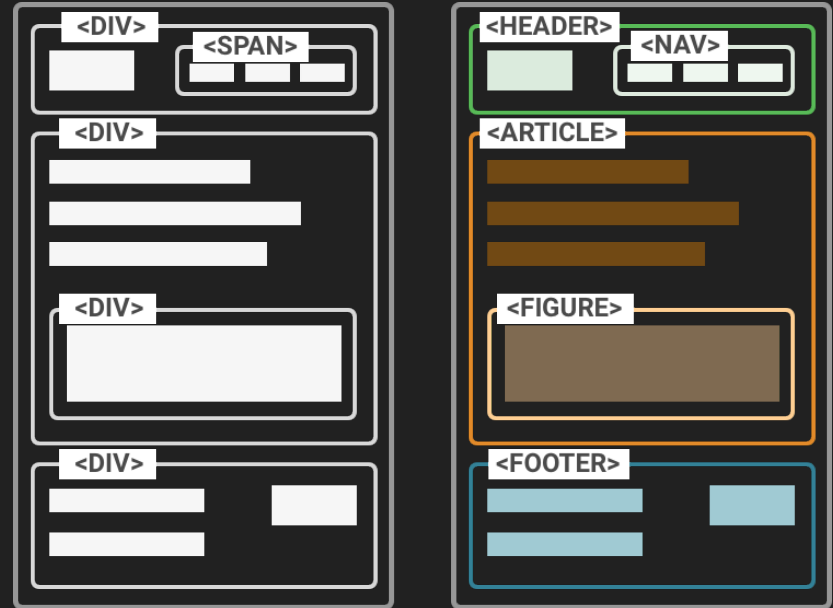




# HTML: main tags

Although there are lots of tags in the HTML specification, 99% of the webs use a subset of HTML tags with less that 10 tags, the most important are:

- `<div>`
- `<img>`
- `<a href="https://www.kmitl.ac.th"></a>`
- `<p>`
- `<h1>, ... , <h6>`
- `<img>`
- `<style>`
- `<script>`
- `<span>`





# HTML: other interesting tags

There are some tags that could be useful sometimes:

- `<button>`
- `<audio>`
- `<video>`
- `<canvas>`
- `<iframe>`



# HTML: tagging correctly

Try to avoid doing this:

```
<div>  
Title  
  
Here is some content  
Here is more content  
</div>
```

Do this instead

```
<div>  
  <h1>Title</h1>  
  <p>Here is content.</p>  
  <p>Here is more content</p>  
</div>
```



# Attributes

- **id**: tells a unique identifier for this tag
- **class**: tells a generic identifier for this tag

```
<div id="profile-picture" class="mini-image">...</div>
```

Attribute	Example
id	<table <b>id</b> ="table01"
class	<p <b>class</b> ="normal">
style	<p <b>style</b> ="font-size:16px">
data-	<div <b>data-id</b> ="500">
onclick	<input <b>onclick</b> ="myFunction()">
onmouseover	<a <b>onmouseover</b> ="this.setAttribute('style','color:red')">

# Image

- GIF, ICO, JPEG, PNG, SVG

- Absolute URL

src=[https://www.kmitl.ac.th/images/img\\_student.jpg](https://www.kmitl.ac.th/images/img_student.jpg)

- Relative URL

src="images/img\_student.gif"

- The width and height Attributes



- Favicon

<link rel="icon" type="image/x-icon" href="/images/favicon.ico">







# Table

```
<table>
  <tr>
    <td>Emil</td>
    <td>Tobias</td>
    <td>Linus</td>
  </tr>
</table>
```

- Border

```
table, th, td {
  border: 1px solid black;
},
```

```
table, th, td {
  border: 1px solid black;
  border-collapse: collapse;
}
```



# List

- Unordered Lists : `<ul> <li></li></ul>`
- Ordered List : `<ol><li></li></ol>`
- Description Lists : `<dl><dt></dt><dd></dd></dl>`

## **An unordered HTML list**

- Coffee
- Tea
- Milk

## **An ordered HTML list**

1. Coffee
2. Tea
3. Milk

## **A Description List**

Coffee  
- black hot drink

Milk  
- white cold drink



# Attribute : Style

`<tagname style="property:value;">`

- Background Color

`<h1 style="background-color:blue;">This is a heading</h1>`

- Text Color

`<h1 style="color:blue;">This is a heading</h1>`

- Fonts

`<h1 style="font-family:verdana;">This is a heading</h1>`

- Text Size

`<h1 style="font-size:300%;">This is a heading</h1>`

- Text Alignment

`<h1 style="text-align:center;">Centered Heading</h1>`



# Form

```
<form action="/action_page.php">  
  <label for="fname">First name:</label><br>  
  <input type="text" id="fname" name="fname" value="John"><br>  
  <label for="lname">Last name:</label><br>  
  <input type="text" id="lname" name="lname" value="Doe"><br><br>  
  <input type="submit" value="Submit">  
</form>
```

First name:

Last name:



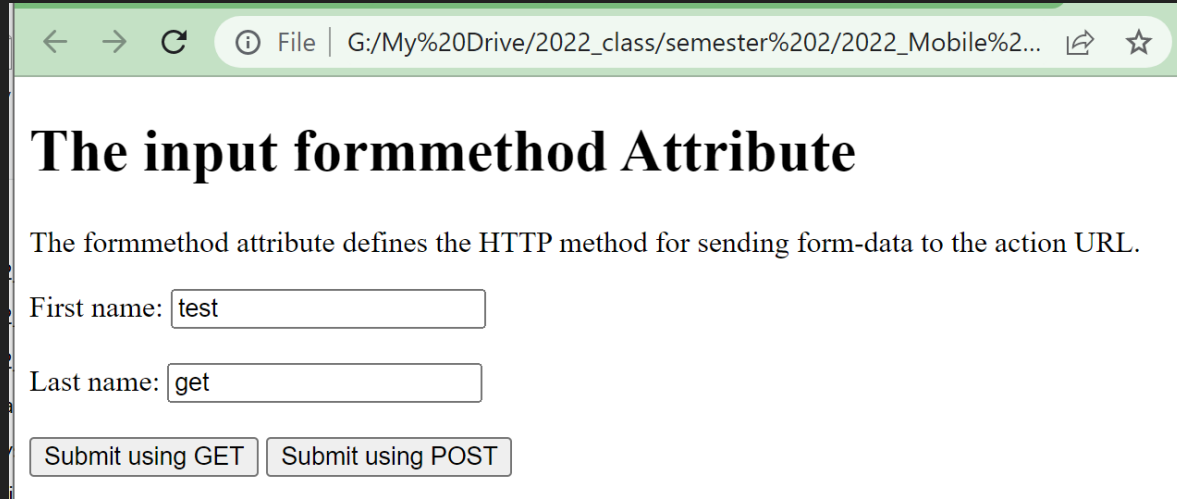
# Form Attribute

- `_blank`
- `_self => default`
- `_parent`
- `_top`
- `framename`

```
<form action="/action_page.php" target="_blank">
```

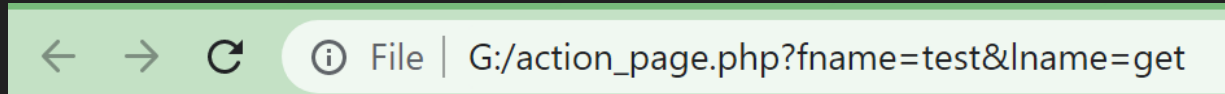
# Form Method

- get => default
- post



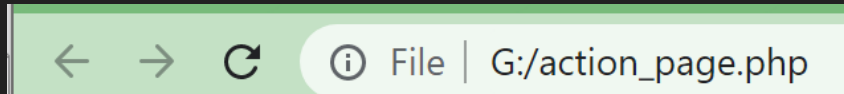
The screenshot shows a web browser window with the address bar displaying 'G:/My%20Drive/2022\_class/semester%202/2022\_Mobile%2...'. The page title is 'The input formmethod Attribute'. The content explains that the formmethod attribute defines the HTTP method for sending form-data to the action URL. Below the text, there is a form with two input fields: 'First name:' with the value 'test' and 'Last name:' with the value 'get'. At the bottom, there are two buttons: 'Submit using GET' and 'Submit using POST'.

```
<form action="/action_page.php" method="get" target="_blank">
```



The screenshot shows a web browser window with the address bar displaying 'G:/action\_page.php?fname=test&lname=get'. The browser has navigation buttons (back, forward, refresh) and a status bar at the bottom.

```
<input type="submit" formmethod="post" value="Submit using POST">
```



The screenshot shows a web browser window with the address bar displaying 'G:/action\_page.php'. The browser has navigation buttons (back, forward, refresh) and a status bar at the bottom.



# HTML 5

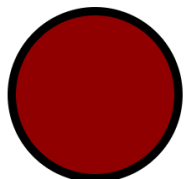
- HTML5 is the newest version of HTML
- new elements, attributes, new video format
- New Semantic Elements
  - <section>, <header>, <footer>  
<nav>, <mark>, <figure>,  
<aside> <figcaption>, <data>,  
<time>, <output>, <progress>,  
<meter> and <main>.
- Inline SVG => SVG tag

HTML

Result

LIVE

```
<svg width="400" height="200">  
  <circle cx="150" cy="100" r="50"  
  style="fill:#900000; stroke:#000000;  
  stroke-width:5;" />  
</svg>
```



Resources

1x 0.5x 0.25x

Rerun



# HTML 5

- Form Features => number, range, form validator
- WebM Video Format => <audio> and <video> tags
- Placeholder Attribute

Enter email :





# HTML 5

- Server-sent Events
- Local Web Storage

## Getting server updates

The server time is: Fri, 17 Feb 2023 17:33:27 +0000  
The server time is: Fri, 17 Feb 2023 17:33:31 +0000  
The server time is: Fri, 17 Feb 2023 17:33:34 +0000  
The server time is: Fri, 17 Feb 2023 17:33:38 +0000  
The server time is: Fri, 17 Feb 2023 17:33:41 +0000  
The server time is: Fri, 17 Feb 2023 17:33:44 +0000  
The server time is: Fri, 17 Feb 2023 17:33:48 +0000  
The server time is: Fri, 17 Feb 2023 17:33:51 +0000  
The server time is: Fri, 17 Feb 2023 17:33:54 +0000

```
<!DOCTYPE html>
<html>
<body>

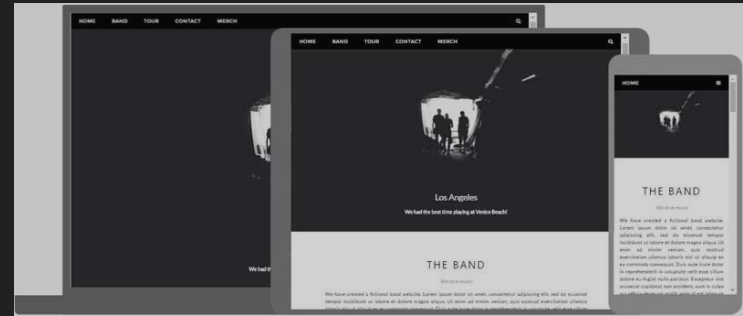
<h1>Getting server updates</h1>
<div id="result"></div>

<script>
if(typeof(EventSource) !== "undefined") {
  var source = new EventSource("demo_sse.php");
  source.onmessage = function(event) {
    document.getElementById("result").innerHTML += event.data + "
<br>";
  };
} else {
  document.getElementById("result").innerHTML = "Sorry, your browser
does not support server-sent events...";
}
</script>

</body>
</html>
```

# What is Responsive web design

- Responsive web design is about creating web pages that look good on all devices!
- A responsive web design will automatically adjust for different screen sizes and viewports.
- Viewport



```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

# Responsive Image

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>

<h2>Responsive Image</h2>
<p>When the CSS width property is set in a percentage value, the image will scale up and down when
resizing the browser window. Resize the browser window to see the effect.</p>



</body>
</html>


```

# Example

<https://mobile2006.000webhostapp.com/image.html>

# Choose image depends on size of screen

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>

<h2>Show Different Images Depending on Browser Width</h2>
<p>Resize the browser width and the image will change at 600px and 1500px.</p>

<picture>
  <source srcset="img_smallflower.jpg" media="(max-width: 600px)">
  <source srcset="img_flowers.jpg" media="(max-width: 1500px)">
  <source srcset="flowers.jpg">
  
</picture>

</body>
</html>
```

<https://mobile2006.000webhostapp.com/image3.html>

### Show Different Images Depending on Browser Width

Resize the browser width and the image will change at 600px and 1500px.



### Show Different Images Depending on Browser Width

Resize the browser width and the image will change at 600px and 1500px.



← → ↺ mobile2006.000webh... ☆ 🛡️ 📱 IT ⋮

### Show Different Images Depending on Browser Width

Resize the browser width and the image will change at 600px and 1500px.

A close-up photograph of a single white hibiscus flower. The flower has five large, white petals and a prominent, bright red center. The background is dark and out of focus.



# Responsive text size => vw

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>

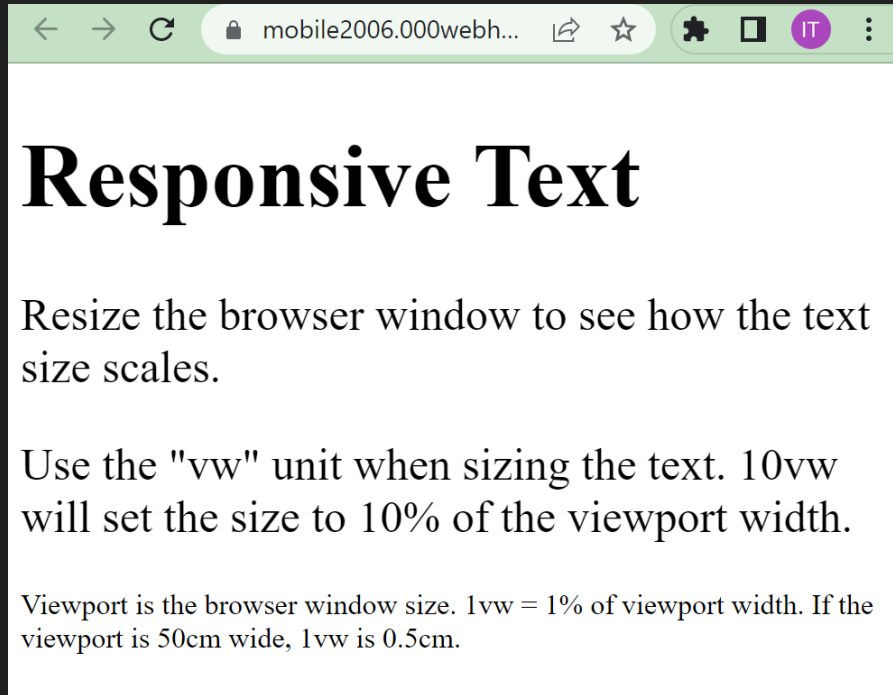
<h1 style="font-size:10vw;">Responsive Text</h1>

<p style="font-size:5vw;">Resize the browser window to see how the text size scales.</p>

<p style="font-size:5vw;">Use the "vw" unit when sizing the text. 10vw will set the size to 10% of the viewport width.</p>

<p>Viewport is the browser window size. 1vw = 1% of viewport width. If the viewport is 50cm wide, 1vw is 0.5cm.</p>

</body>
</html>
```





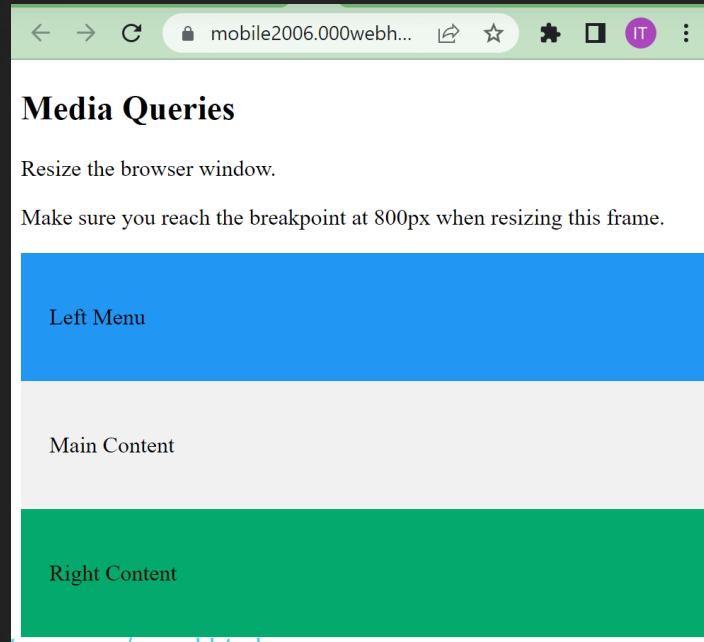
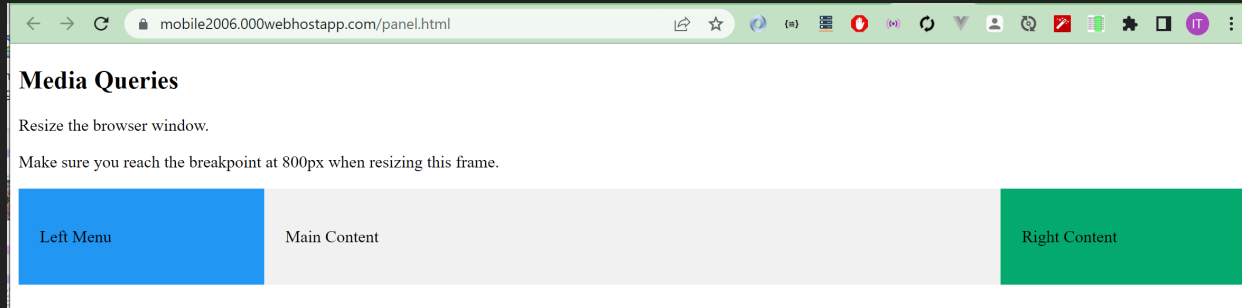


# Media Queries

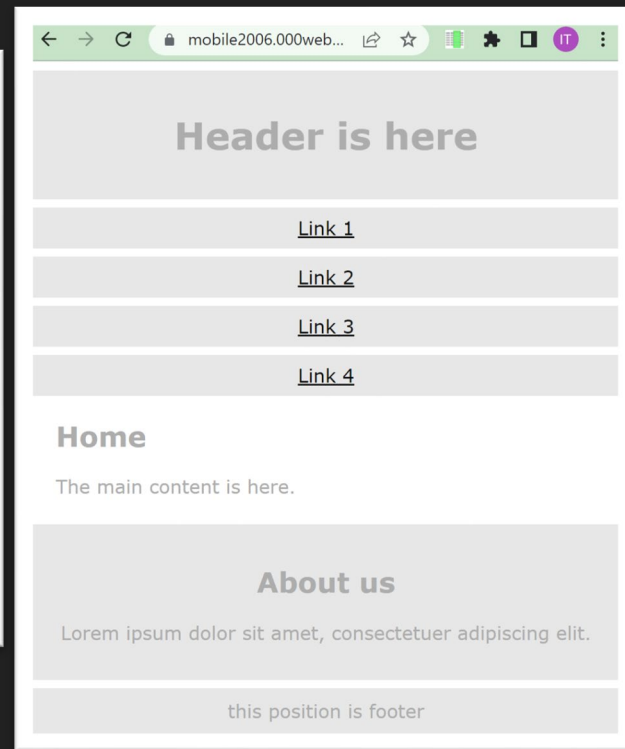
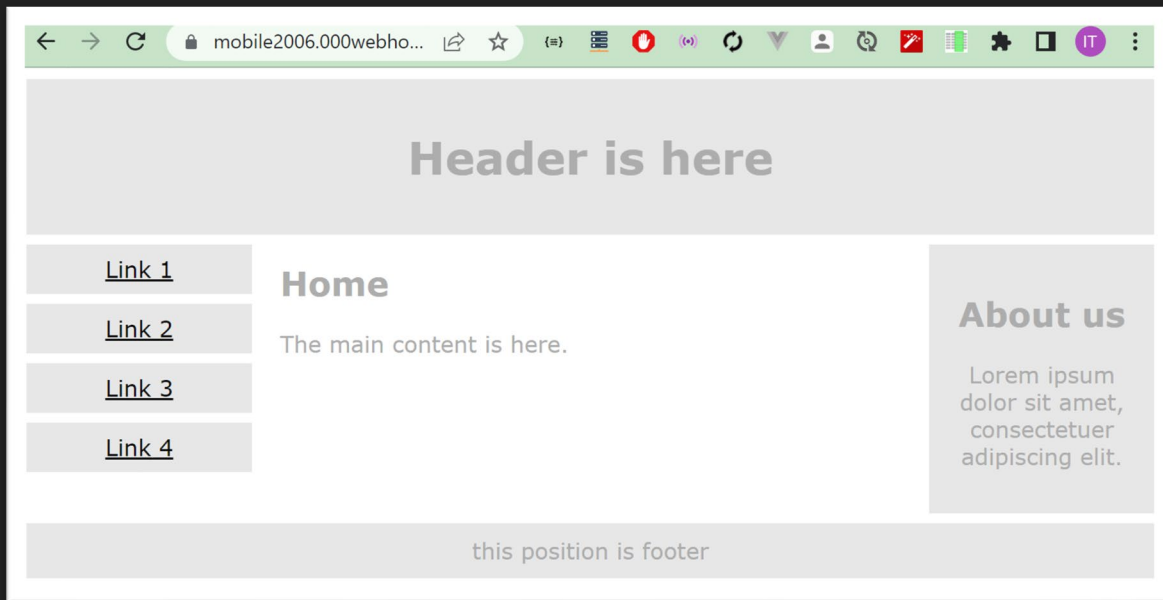
```
<style>
.left, .right {
  float: left;
  width: 20%; /* The width is 20%, by default */
}

.main {
  float: left;
  width: 60%; /* The width is 60%, by default */
}

/* Use a media query to add a breakpoint at 800px: */
@media screen and (max-width: 800px) {
  .left, .main, .right {
    width: 100%; /* The width is 100%, when the viewport is 800px or smaller */
  }
}
</style>
```



# Responsive Web page



```

<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width,
initial-scale=1.0">
<style>
* {
  box-sizing: border-box;
}
.menu {
  float: left;
  width: 20%;
  text-align: center;
}
.menu a {
  background-color: #e5e5e5;
  padding: 8px;
  margin-top: 7px;
  display: block;
  width: 100%;
  color: black;
}
.main {
  float: left;
  width: 60%;
  padding: 0 20px;
}
.right {
  background-color: #e5e5e5;
  float: left;
  width: 20%;
  padding: 15px;
  margin-top: 7px;
  text-align: center;
}

@media only screen and (max-width: 620px) {
  /* For mobile phones: */
  .menu, .main, .right {
    width: 100%;
  }
}
</style>

```

```

</head>
<body style="font-family:Verdana;color:#aaaaaa;">

<div style="background-color:#e5e5e5;padding:15px;text-align:center;">
  <h1>Hello World</h1>
</div>

<div style="overflow:auto">
  <div class="menu">
    <a href="#">Link 1</a>
    <a href="#">Link 2</a>
    <a href="#">Link 3</a>
    <a href="#">Link 4</a>
  </div>

  <div class="main">
    <h2>Lorum Ipsum</h2>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy
nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  </div>

  <div class="right">
    <h2>About</h2>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
  </div>
</div>

<div style="background-color:#e5e5e5;text-align:center;padding:10px;margin-
top:7px;">© copyright w3schools.com</div>

</body>
</html>

```

# Framework

<https://www.w3schools.com/w3css/default.asp>

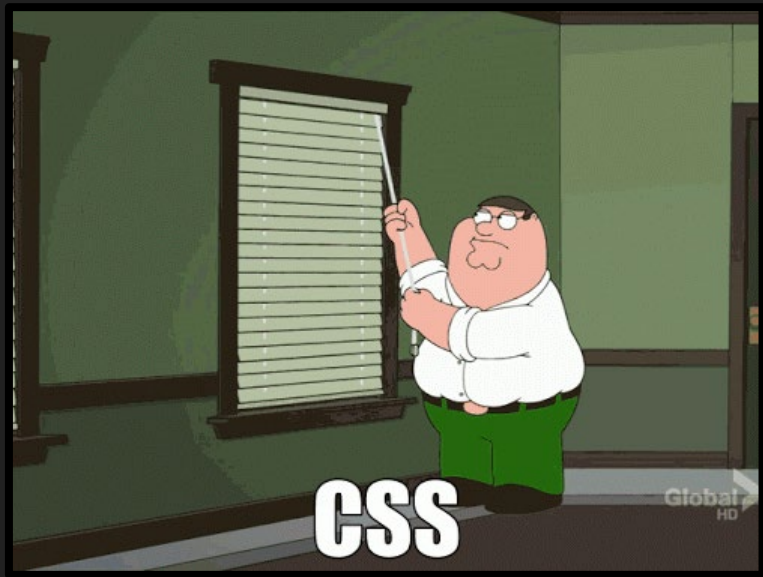
```
<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
```

<https://getbootstrap.com/>

```
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
```

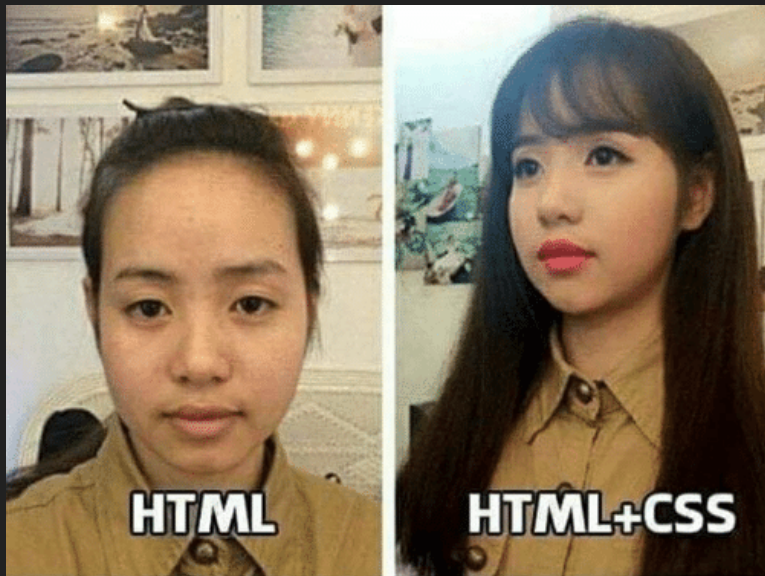
# Technologies

- HTML
- CSS
- Javascript





# HTML + CSS



Jordy Gabriel



#Jdam



# HTML Vs CSS

```
<I><FONT face="Arial" color="green"><H3> Test HTML vs CSS</H3></FONT></I>
```

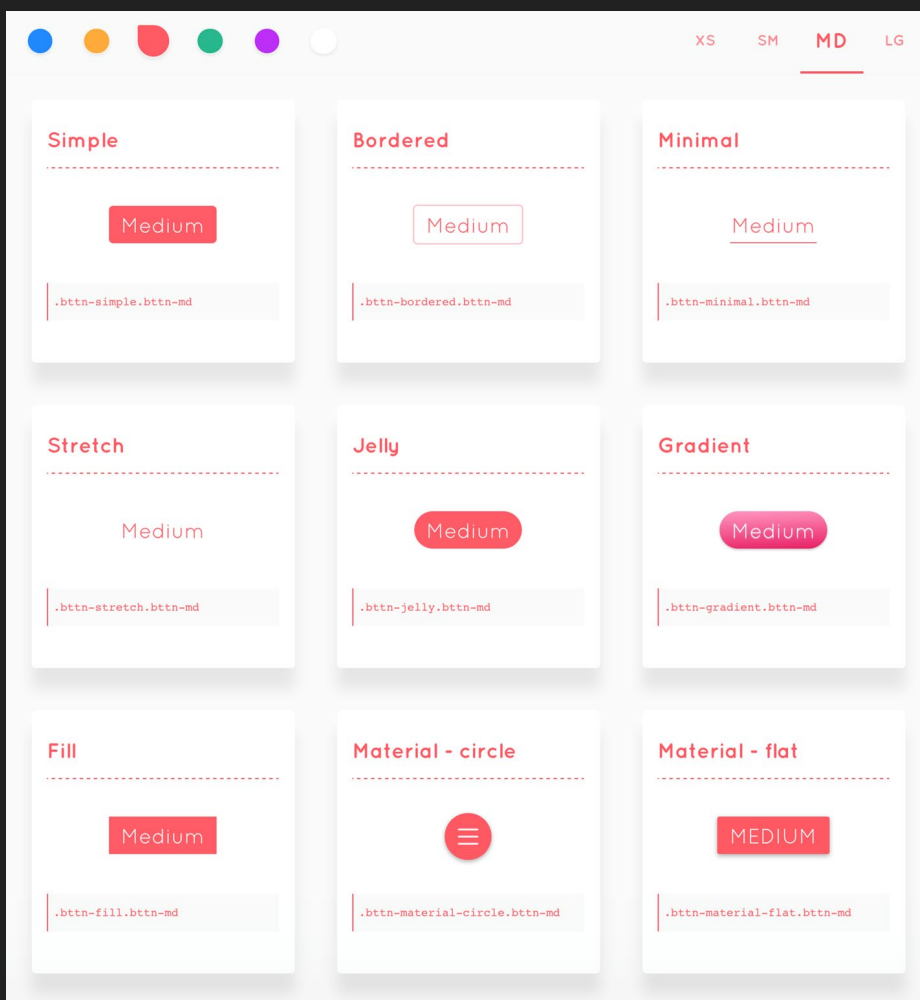
*Test HTML vs CSS*

**H3 {font-family : Arial; font-style : italic; color : green}**



# CSS

- **Colors**: content, background, borders
- **Margins**: interior margin, exterior margin
- **Position**: where to put it
- **Sizes**: width, height
- **Behaviour**: changes on mouse over





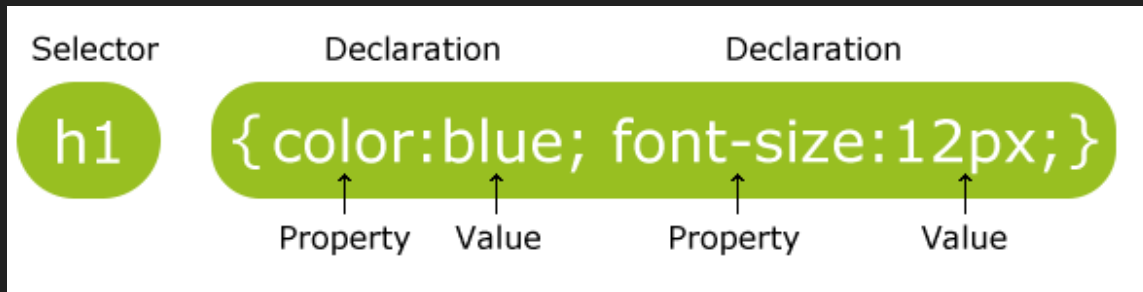
## CSS example

```
* {  
    color: blue; /*a comment */  
    margin: 10px;  
    font: 14px Tahoma;  
}
```



# CSS

- `selector { property: value }`
- `body { background-color: lime }`
- `h1, h2, h3 { font-size: x-large; background-color: yellow }`



# CSS fields

Here is a list of the most common CSS fields and an example:

- `color: #FF0000; red; rgba(255,00,100,1.0);` //different ways to specify colors
- `background-color: red;`
- `background-image: url('file.png');`
- `font: 18px 'Tahoma';`
- `border: 2px solid black;`
- `border-top: 2px solid red;`
- `border-radius: 2px;` //to remove corners and make them more round
- `margin: 10px;` //distance from the border to the outer elements
- `padding: 2px;` //distance from the border to the inner elements
- `width: 100%; 300px; 1.3em;` //many different ways to specify distances
- `height: 200px;`
- `text-align: center;`
- `box-shadow: 3px 3px 5px black;`
- `cursor: pointer;`
- `display: inline-block;`
- `overflow: hidden;`



# CSS how to add it

There are four ways to add **CSS rules** to your website:

- Inline

```
<p style="color: blue; margin: 10px">
```

- Embedded

```
<style>
```

```
  p { color: blue }
```

```
</style>
```

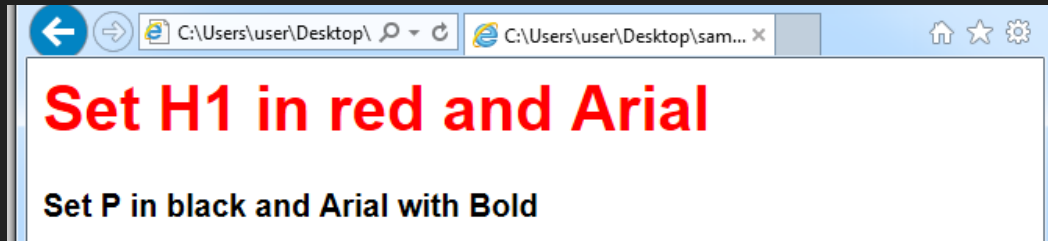
- Linked

```
<link href="style.css" rel="stylesheet" />
```

- Using Javascript (we will see this one later).

# CSS : Inline

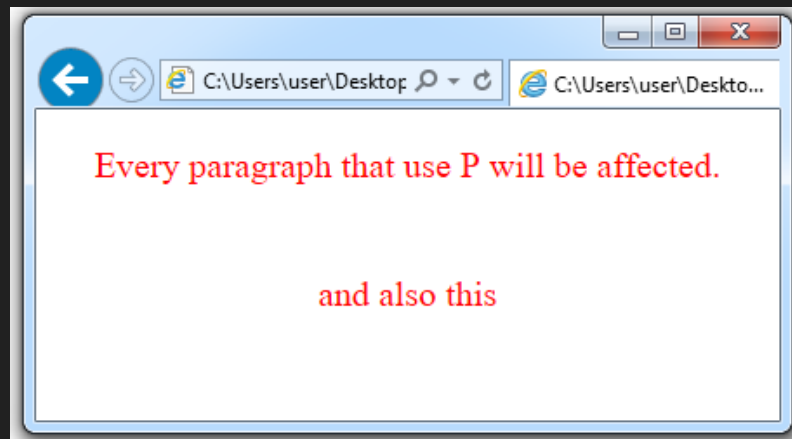
```
<html>
<body>
  <h1 style = "color:red; font-family:Arial">
    Set H1 in Arail red </h1>
  <p style = "color:black; font-family:Arial; font-weight:bold">
    Set P in black, Arial with bold </p>
</body>
</html>
```



# CSS : Embedded

```
<html>
<head>
<style>
    p {          text-align: center;
                  color: red;
                  font-size: 25;
    }
</style>
</head>

<body>
    <p> Every paragraph that use P tag will be affected. </p>
    <p> and also this </p>
</body>
</html>
```



# CSS : Linked

## sample1.CSS

```
body {  
    background-color: lightblue;}  
  
h1 {  
    color: navy;  
    margin-left: 20px;  
}
```

## index.html

```
<html>  
  
  <head>  
    <link rel="stylesheet" type="text/css" href="sample1.css">  
  </head>  
  
  <body>  
    <h1>This is a heading</h1>  
    <p>This is a paragraph.</p>  
  
  </body>  
  
</html>
```







# CSS : 2 in 1

ใน sample1.CSS

```
h1 { color: blue; }
```

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
<style>
h1 {
  color: orange;
}
</style>
</head>
<body>

</body>
</html>
```

**This is a heading**



# CSS : 2 in 1

ใน sample1.CSS

```
h1 { color: blue; }
```

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  color: orange;
}
</style>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
<body>

<h1>This is a heading</h1>

</body>
</html>
```

**This is a heading**

# CSS : id selector

```
<html>  
<head>  
<style>
```

```
    #para1 {    text-align: center;  
                color: red;  
                font-size: 25;  
            }
```

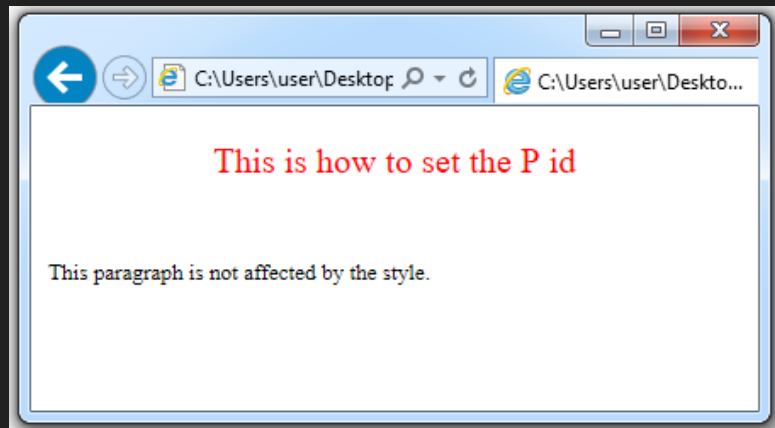
```
</style>  
</head>  
<body>
```

```
<p id = "para1"> This is how to set the P id </p>
```

```
<p> This paragraph is not affected by the style </p>
```

```
</body>  
</html>
```

#para1 => id selector



# CSS : Class selector

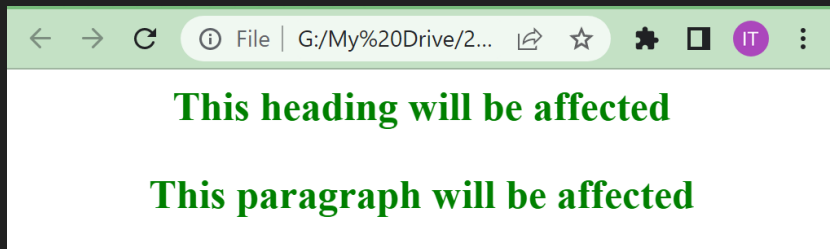
.center => class selector

```
<html>
<head> <style>
    .center {    text-align: center;
                  color: green;
                  font-size: 25;
                }
</style> </head> <body>
```

```
<h1 class = "center"> This heading will be affected </p>
```

```
<p class = "center"> This paragraph will be affected </p>
```

```
</body> </html>
```



# CSS : Class selector (2)

.center => class selector

```
<html>
<head> <style>
    p.center { text-align: center;
               color: red; }
    p.large { font-size: 300% }
</style> </head> <body>
```

```
<h1 class = "center"> This heading will not be affected </p>
```

```
<p class = "center"> This paragraph will be affected in red-center </p>
```

```
<p class = "center large"> This paragraph will be affected all </p>
```

```
</body> </html>
```



# CSS : Group selector

h1, h2, p { } => group selector

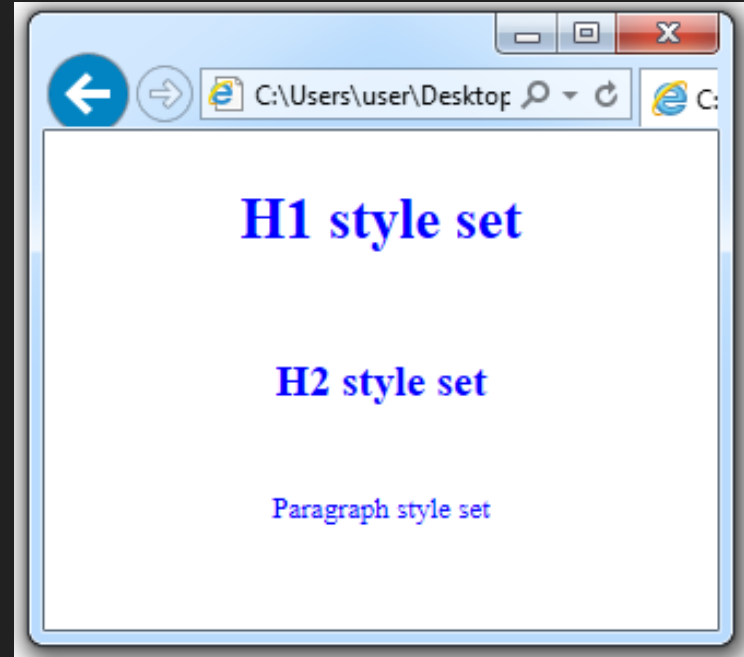
```
<html>
<head> <style>
    h1, h2, p { text-align: center;
                color: blue;
            }
</style> </head> <body>

    <h1> heading 1 </h1>

    <h2> heading 1 </h2>

    <p> paragraph </p>

</body> </html>
```



# CSS : Border

- `p.dotted {border-style: dotted;}`
- `p.dashed {border-style: dashed;}`
- `p.solid {border-style: solid;}`
- `p.double {border-style: double;}`
- `p.groove {border-style: groove;}`
- `p.ridge {border-style: ridge;}`
- `p.inset {border-style: inset;}`
- `p.outset {border-style: outset;}`
- `p.none {border-style: none;}`
- `p.hidden {border-style: hidden;}`
- `p.mix {border-style: dotted dashed solid double;}`

A dotted border.

A dashed border.

A solid border.

A double border.

A groove border. The effect depends on the border-color value.

A ridge border. The effect depends on the border-color value.

An inset border. The effect depends on the border-color value.

An outset border. The effect depends on the border-color value.

No border.

A hidden border.

A mixed border.



# CSS : Margins

```
<style>
```

```
div {
```

```
  border: 1px solid black;
```

```
  margin-top: 100px;
```

```
  margin-bottom: 100px;
```

```
  margin-right: 150px;
```

```
  margin-left: 80px;
```

```
  background-color: lightblue;
```

```
}
```

```
</style>
```

or top, right, bottom, left

```
margin: 100px 150px 100px 80px;
```

This div element has a top margin of 100px, a right margin of 150px, a bottom margin of 100px, and a left margin of 80px.





# CSS : Padding

```
<style>
```

```
div {
```

```
  border: 1px solid black;
```

```
  background-color: lightblue;
```

```
  padding-top: 100px;
```

```
  padding-right: 30px;
```

```
  padding-bottom: 50px;
```

```
  padding-left: 80px;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<div>This div element has a top padding of 50px, a right padding of 30px, a bottom padding of 50px, and a left padding of 80px.</div>
```

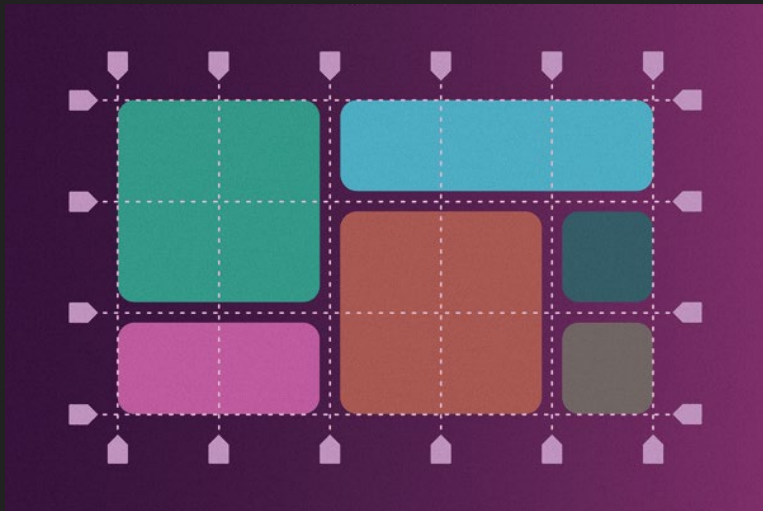
```
or or top, right, bottom, left
```

```
padding: 100px 30px 50px 80px;
```

This div element has a top padding of 50px, a right padding of 30px, a bottom padding of 50px, and a left padding of 80px.



# Grid system



## HTML

```
<div class="grid-container">  
  <div class="grid-item1">1</div>  
  <div class="grid-item2">2</div>  
</div>
```

## CSS

```
.grid-container {  
  display: grid;  
  grid-template-rows: 100px; 100px;  
  grid-template-columns: 100px; 100px; 100px;  
  grid-gap: 5px;  
}  
  
.grid-item1 {  
  background: blue;  
  border: black 5px solid;  
  grid-column-start: 1;  
  grid-column-end: 5;  
  grid-row-start: 1;  
  grid-row-end: 3;  
}
```

# CSS : !important

```
<style>
#myid {
  background-color: blue;
}
```

```
.myclass {
  background-color: gray;
}
```

```
p {
  background-color: red ;
}
```

```
</style>
</head>
<body>
```

```
<p>This is some text in a paragraph.</p>
<p class="myclass">This is some text in a paragraph.</p>
<p id="myid">This is some text in a paragraph.</p>
```

This is some text in a paragraph.

This is some text in a paragraph.

This is some text in a paragraph.

```
p {
  background-color: red !important;
}
```

This is some text in a paragraph.

This is some text in a paragraph.

This is some text in a paragraph.



# Technologies

- HTML
- CSS
- Javascript





# Javascript: insert code

There is three ways to execute javascript code in a website:

- **Embed** the code in the HTML using the `<script>` tag.

```
<script> /* some code */ </script>
```

- **Import** a Javascript file using the `<script>` tag:

```
<script src="file.js" />
```

- **Inject** the code on an event inside a tag:

```
<button onclick="javascript: /*code*/">press me</button>
```



# Javascript: Syntax

Very similar to C++ or Java but much simpler.

```
var my_number = 10; //this is a comment
var my_string = "hello"; /* this is also comment */
var my_array = [10,20,"name",true];
var my_object = { name: "javi", city: "Barcelona" };

function say( str )
{
    for(var i = 0; i < 10; ++i)
        console.log(" say: " + str );
}
```

# Javascript example

```
<html>  
  <body>  
    <h1>This is a title</h1>  
  
  </body>  
</html>
```

# Javascript API

Javascript comes with a rich API to do many things like:

- Access the DOM (HTML nodes)
- Do HTTP Requests
- Play videos and sounds
- Detect user actions (mouse move, key pressed)
- Launch Threads
- Access the GPU, get the Webcam image, ...

And the API keeps growing with every new update of the standard.

Check the [WEB API reference](#) to know more



# Javascript: retrieving element

You can get elements from the DOM (HTML tree) using different approaches.

- **Crawling the HTML tree** (starting from the body, and traversing its children)
- **Using a selector** (like in CSS)
- **Attaching events listeners** (calling functions when some actions are performed)

# Javascript: crawling the DOM

From javascript you have different variables that you can access to get information about the website:

- `document`: the DOM information (HTML)
- `window`: the browser window

The document variable allows to crawl the tree:

```
document.body.children[0] // returns the first node inside body tag
```



# Basic command : Document Object

Object.Method(" ")

document.write("Hello !!");

```
<html>
<head>
<title>This is a JavaScript example</title>
</head>
<body>
  <script>
    document.write("<h1> This is my first JavaScript Page</h1>");
  </script>
</body>
</html>
```





# Basic command :Prompt()

```
prompt ("message", "value")
```

```
<script >
```

```
window.prompt("please enter user name")
```

```
window.prompt("please enter user name", "name")
```

```
</script>
```

This page says

please enter user name

OK Cancel

This page says

please enter user name

OK Cancel



# Basic command :Alert

<script>

```
window.alert("Script tag with ALERT");  
alert("Script tag with ALERT2");
```

</script>

**This page says**

Script tag with ALERT

OK

This page says

Script tag with ALERT2

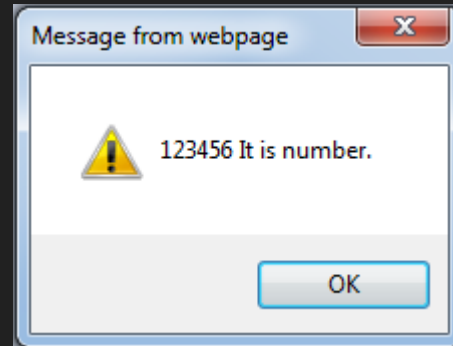
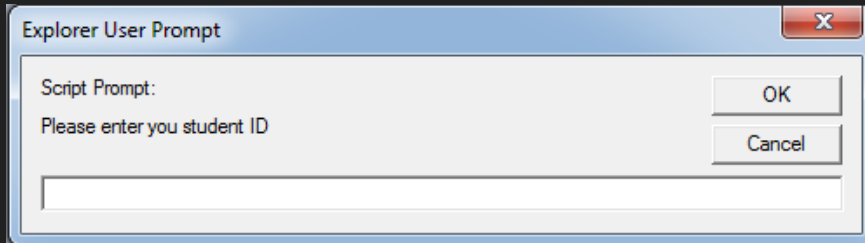
OK



# Basic command :isNaN ()

- isNaN(testValue);
- To check the input value is number or not

```
<script>  
  var x = prompt("Please enter you student ID","");  
  if (isNaN(x))  
    alert(x+" It is text, not number");  
  else  
    alert(x+" It is number.");  
</script>
```



# getElementById ()

```
<body>
```

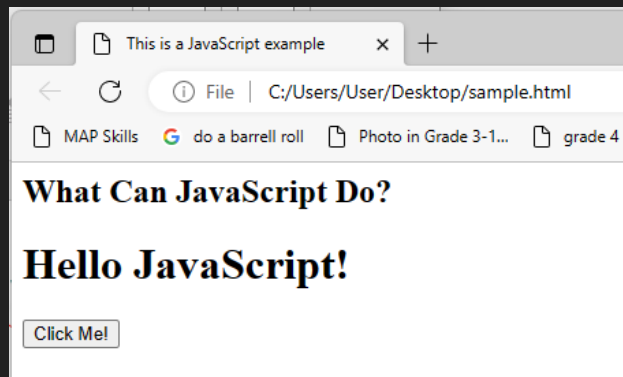
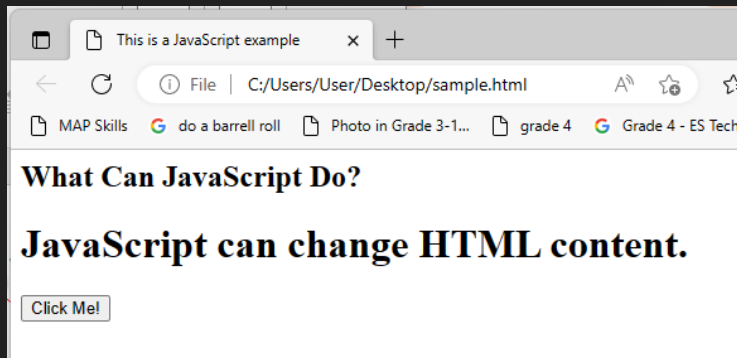
```
<h2>What Can JavaScript Do?</h2>
```

```
<h1 id="demo">JavaScript can change HTML content.</h1>
```

```
<button type="button"
```

```
onclick='document.getElementById("demo").innerHTML = "Hello JavaScript!'">Click Me!</button>
```

```
</body>
```



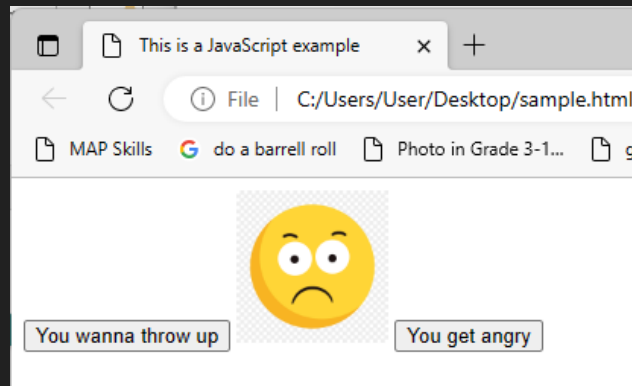
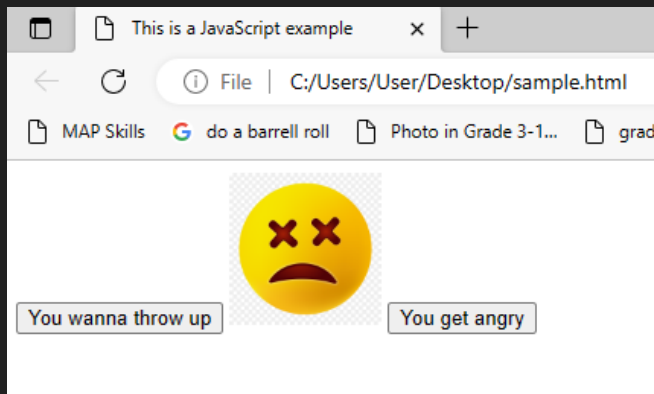
# getElementById () => image change

```
<button onclick="document.getElementById('myImage').src='sad.png'">You wanna throw up</button>
```

```

```

```
<button onclick="document.getElementById('myImage').src='frown.png'">You get angry</button>
```







# Function

```
<!DOCTYPE html>  
<html>  
<body>
```

```
<p id="demo"></p>
```

```
<script>  
function myFunction(p1, p2) {  
  return p1 * p2;  
}  
document.getElementById("demo").innerHTML = myFunction(4, 3);  
</script>
```

```
</body>  
</html>
```

12



# Local variable

```
<p id="demo1"></p>
```

```
<p id="demo2"></p>
```

```
<script>  
myFunction();
```

string Volvo

```
function myFunction() {  
  let carName = "Volvo";  
  document.getElementById("demo1").innerHTML = typeof carName + " " + carName ;  
}
```

```
document.getElementById("demo2").innerHTML = typeof carName ;
```

```
</script>
```

Undefined



# Event object

- `onClick()`

```
<html><body>
```

```
<button onclick="myFunction()">Click me</button>
```

```
<p id="demo"></p>
```

```
<script>
```

```
    function myFunction() {  
        document.getElementById("demo").innerHTML = "Hello World";  
    }
```

```
</script></body></html>
```



Click me



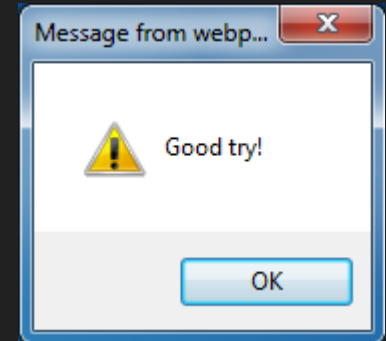
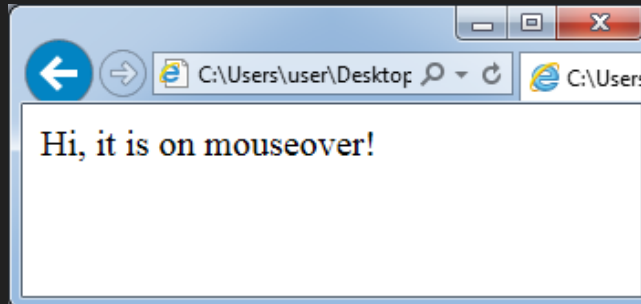
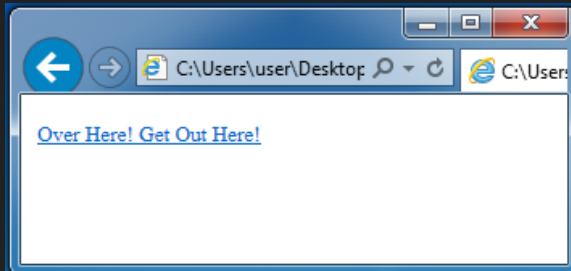
Click me

Hello World

# Mouse Event

- onclick
- ondblclick
- onmousedown
- onmousemove
- onmouseover
- onmouseout
- onmouseup

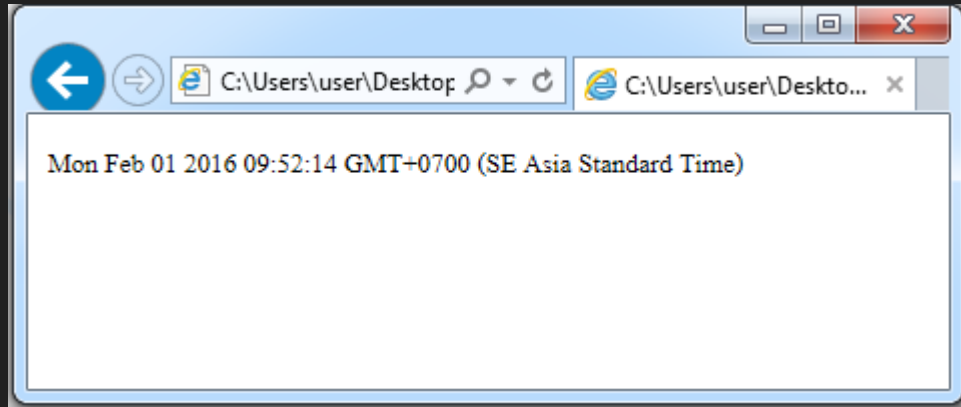
```
<a href="#" onMouseOver="document.write  
('Hi, nice to see you!')">Over Here!</a>  
<a href="#" onMouseOut="alert('Good try!')">  
Get Out Here!</a>
```



# Date time

- getDate()
- getDay()
- getHours()
- getMinutes()
- getSeconds()
- getMonth()
- getYear()
- getTime()

```
<SCRIPT>  
    var x= Date();  
    document.write (x);  
</SCRIPT>
```



# Validate form

```
<html>
<body>
<script>
function validateform(){
var name=document.myform.name.value;
var password=document.myform.password.value;

if (name==null || name==""){
  alert("Name can't be blank");
  return false;
}else if(password.length<6){
  alert("Password must be at least 6 characters long.");
  return false;
}
}
</script>
```

```
<body>
<form name="myform" method="post"
onsubmit="return validateform()" >
Name: <input type="text" name="name"><br/>
Password: <input type="password"
name="password"><br/>
<input type="submit" value="register">
</form>
</body>
</html>
```

Name:

Password:

This page says

Name can't be blank

# Example of a website

## HTML in index.html

```
<link href="style.css" rel="stylesheet"/>
<h1>Welcome</h1>
<p>
    <button>Click me</button>
</p>
<script src="code.js"/>
```

## CSS in style.css

```
h1 { color: #333; }
button {
    border: 2px solid #AAA;
    background-color: #555;
}
```

## Javascript in code.js

```
//fetch the button from the DOM
var button = document.querySelector("button");

//attach and event when the user clicks it
button.addEventListener("click", myfunction);

//create the function that will be called when the
button is pressed
function myfunction()
{
    //this shows a popup window
    alert("button clicked!");
}
```

# Github

- <https://git-scm.com/downloads>
- cmd => git --version
- git config --global user.email "pikulkaew.ta@kmitl.ac.th"
- git config --global user.name "pikulkaew"
- git config --list
- เข้าเว็บ github.com=>sign in => new => create repo

optional

git remote -v

git remote rm origin



# Github2

- เปิด vscode => create index.html => open terminal
- git init
- git add .
- git commit -m "first commit"
- git branch -M main
- git remote add origin <https://github.com/>....
- git push -u origin main => auto pop up ให้ sign in ในเวป
- check in github
- add file hello.html
- git add .
- git commit -m "second commit"
- git push -u origin main

# Github3

- build html
- go to repo that you've just upload your html
- Pages
  - Branch => main=>root =>save
  - pikulkaew.github.io/test => index.html
  - pikulkaew.github.io/test/hello.html => index.html

# Deploy html on google cloud

- Create Virtual Machine
- `sudo apt-get update`
- `sudo apt install nginx`
  
- `vi test.html`
- `34.xx.xx.xx/test.html`
- `sudo apt-get install git`
- `git clone https://github.com/...`