

The background features a series of colorful rays (green, orange, blue, grey) emanating from a central point on the left. Faint binary code (0s and 1s) is scattered across the background, and a line graph with multiple colored lines is visible on the right side.

## Chapter 2

# CSS

# Cascading Style Sheets

Theerayut Thongkrau

# CSS

- CSS ย่อมาจาก **C**ascading **S**tyle **S**heets
- เป็นภาษาที่ใช้ในการกำหนดรูปแบบการแสดงผลของ HTML Element
- เกิดขึ้นพร้อมกับ HTML 4.0 เพื่อแก้ปัญหการแสดงผล
- สามารถแทรกคำสั่ง CSS ในโค้ด HTML หรือแยกส่วนออกเป็นไฟล์ .css ได้

## ➡ CSS กับ HTML

- HTML ใช้กำหนดโครงสร้างและกำกับส่วนประกอบของหน้าเว็บ
- CSS ใช้กำหนดรูปแบบการแสดงผลของหน้าเว็บ ในแต่ละส่วนของเอกสาร
- แท็ก หรือ attribute ของ HTML ที่ใช้ในการจัดรูปแบบส่วนใหญ่ออกยกเลิกแล้วใน HTML version 5 เช่น <center>, align ซึ่งยังใช้ได้อยู่ แต่เพื่อให้แยกส่วนกันชัดเจนควรใช้ CSS

# ➡ CSS กับ HTML

HTML = การวางโครงสร้างของบ้าน

CSS = การตกแต่งบ้าน



# ➔ CSS Syntax

- CSS ประกอบด้วย 2 ส่วน ได้แก่ Selector และ รายการ Property

Selector

h1 {

color: blue;

font-size: 12px;

}

property

value

- **Selector** คือ ชื่ออิลิเมนต์, ชื่อคลาส หรือ id ที่ต้องการนำมาจัดรูปแบบ
- **CSS Property** คือ ชื่อคุณสมบัติที่ต้องการจัดรูปแบบ
- **Value** คือ ค่าที่ต้องการกำหนดรูปแบบให้กับ Property อาจมีได้หลายค่า แต่ละค่าคั่นด้วยช่องว่าง และปิดท้ายด้วย ;
- เครื่องหมาย { } ใช้กำหนดขอบเขตของ Selector



# CSS Property

	Property
คุณสมบัติเกี่ยวกับสี	color (สีตัวอักษร) background-color (สีพื้นหลัง)
คุณสมบัติเกี่ยวกับการจัดช่องว่าง (Spacing)	margin padding margin-left, margin-right, margin-top, margin-bottom padding-left, padding-right, padding-top, padding-bottom
คุณสมบัติกรอบต่างๆ	border-width border-style border-color border (กำหนดความกว้าง รูปแบบ สีในครั้งเดียว)
การจัดตำแหน่งข้อความ	text-align text-indent word-spacing letter-spacing line-height white-space



# CSS Property

	Property
Font	font-family font-size font-weight font-style font-variant text-decoration
ขนาด	width height
Layout	position left, right float, clear
Graphics	background-image background-repeat background-position

## ➡ คู่มือ CSS

- เราสามารถเลือก CSS Property มาปรับใช้กับเว็บเพจโดยดูจาก CSS Properties Reference

<https://cssreference.io>

<http://devdocs.io/css>

<https://developer.mozilla.org/en-US/docs/Web/CSS/Reference>



# ➡ การแทรกคำสั่ง CSS

- **Internal Style Sheet** - ใส่คำสั่ง CSS **ในแท็ก <head>** ภายใต้อัฒก <style> วิธีนี้เหมาะกับการจัดรูปแบบให้กับเว็บหน้าเดียว
- **External Style Sheet** - ใส่คำสั่ง CSS **ในไฟล์แยกต่างหาก** แล้วอ้างถึงไฟล์ในแท็ก <link> ซึ่งอยู่ภายใต้อัฒก <head> เหมาะกับการจัดรูปแบบให้กับเว็บหลายหน้าที่ใช้รูปแบบเดียวกัน
- **Inline Style Sheet** - ใส่คำสั่ง CSS **ในแท็กที่ต้องการ** โดยตรง เพื่อจัดรูปแบบเฉพาะแท็กนั้นๆ วิธีนี้ควรทำเมื่อมีความจำเป็นเท่านั้น เพราะยากต่อการกลับมาแก้ไขในภายหลัง

หมายเหตุ หากมีชื่อ Selector ซ้ำกัน Browser จะเลือกใช้รูปแบบตามลำดับดังนี้

Inline ➡ Internal ➡ External

# ➔ Internal Style Sheet

```
<!doctype html>
<html>
<head>
```

```
<style>
  p {
    color: maroon;
  }
</style>
```

เพิ่ม CSS ในการ  
กำหนดสี font ของ  
แท็ก <p>

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

```
<h1>Computer Science Department, KKU</h1>
```

```
<p>ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยขอนแก่น จัดตั้งอย่างเป็นทางการเมื่อวันที่ 23 มีนาคม 2537 ผลิตบัณฑิตระดับปริญญาตรีมาแล้ว 21 รุ่น และปริญญาโทมาแล้วมากกว่า 10 รุ่น </p>
```

```
<h2>เป้าหมาย</h2>
```

```
<p>ผลิตบัณฑิตที่มีคุณภาพ มีคุณธรรมจริยธรรม และมีความเชี่ยวชาญเฉพาะด้าน ตรงกับความต้องการของผู้ประกอบการ</p>
```

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

ผลลัพธ์

## Computer Science Department, KKU

ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยขอนแก่น จัดตั้งอย่างเป็นทางการเมื่อวันที่ 23 มีนาคม 2537 ผลิตบัณฑิตระดับปริญญาตรีมาแล้ว 21 รุ่น และปริญญาโทมาแล้วมากกว่า 10 รุ่น

### เป้าหมาย

ผลิตบัณฑิตที่มีคุณภาพ มีคุณธรรมจริยธรรม และมีความเชี่ยวชาญเฉพาะด้าน ตรงกับความต้องการของผู้ประกอบการ

# ➡ External Style Sheet

```
<!doctype html>
<html>
<head>
```

หากอยู่บน server เดียวกันควร  
อ้างอิงโดยใช้ Relative URL

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

```
<style>
  p {
    color: maroon;
  }
</style>
</head>
```

```
<body>
<h1>Computer Science Department, KCU</h1>
<p>ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยขอนแก่น
จัดตั้งอย่างเป็นทางการเมื่อวันที่ 23 มีนาคม 2537 ผลิตบัณฑิตระดับปริญญาตรีมาแล้ว 21 รุ่น และปริญญาโทมาแล้วมากกว่า 10 รุ่น</p>
```

## **mystyle.css**

```
p {
    color: maroon;
}
```

/\* ไม่ต้องใส่แท็ก HTML ใดๆในนี้ \*/

## Inline Style Sheet

- ใช้ Attribute style แทรกในแท็ก แล้วกำหนดคุณสมบัติ CSS

```
<!doctype html>
<html>
<body>
```

```
<h1 style="font-family:sans-serif; color:gray;">
```

Computer Science Department, KKU

```
</h1>
```

```
<p style="color:maroon">ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์</p>
```

```
<h2>เป้าหมาย</h2>
```

```
<p style="color:maroon">ผลิตบัณฑิตที่มีคุณภาพ มีคุณธรรมจริยธรรม</p>
```

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

# ➡ 1 Selector มีได้หลาย Property

กำหนดให้แท็ก <p> มีสีพื้นเป็นสีแดง **และ**

มีกรอบขนาด 1 พิกเซล แบบทึบ สีเทา

p {

**background-color: red;**

**border: 1px solid gray;**

}

} มี 2 property

ค่าของ property สามารถมีได้หลาย  
ค่า โดยคั่นแต่ละค่าด้วยช่องว่าง

# ➡ Selector ที่มีรูปแบบเหมือนกัน

```
h1 {  
  font-family: sans-serif;  
  color: gray;  
}
```

```
h2 {  
  font-family: sans-serif;  
  color: gray;  
}
```



```
h1, h2 {  
  font-family: sans-serif;  
  color: gray;  
}
```

รูปแบบเหมือนกัน สามารถ  
กำหนดชื่อแท็กไว้ด้วยกันได้  
โดยกันด้วย comma

# ➡ กำหนดรูปแบบเพิ่มเติมให้ Selector เดิม

```
h1, h2 {  
  font-family: sans-serif;  
  color: gray;  
  border-bottom: 1px solid black;  
}
```

เพิ่มรูปแบบเส้นใต้ให้กับแท็ก  
<h1> และ <h2>

```
h1 {  
  border-top: 1px solid black;  
}
```

เมื่อต้องการเพิ่มบางรูปแบบให้กับ <h1> ซึ่งถูกกำหนดรูปแบบไปแล้วสามารถทำได้โดยการระบุ Selector h1 เพียงอย่างเดียว

ผลลัพธ์

## Computer Science Department, KCU

ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัย  
ขอนแก่น จัดตั้งอย่างเป็นทางการเมื่อวันที่ 23 มีนาคม 2537 ผลิต  
บัณฑิตระดับปริญญาตรีมาแล้ว 21 รุ่น และปริญญาโทมาแล้ว  
มากกว่า 10 รุ่น

### เป้าหมาย

ผลิตบัณฑิตที่มีคุณภาพ มีคุณธรรมจริยธรรม และมีความเชี่ยวชาญ  
เฉพาะด้าน ตรงกับความต้องการของผู้ประกอบการ

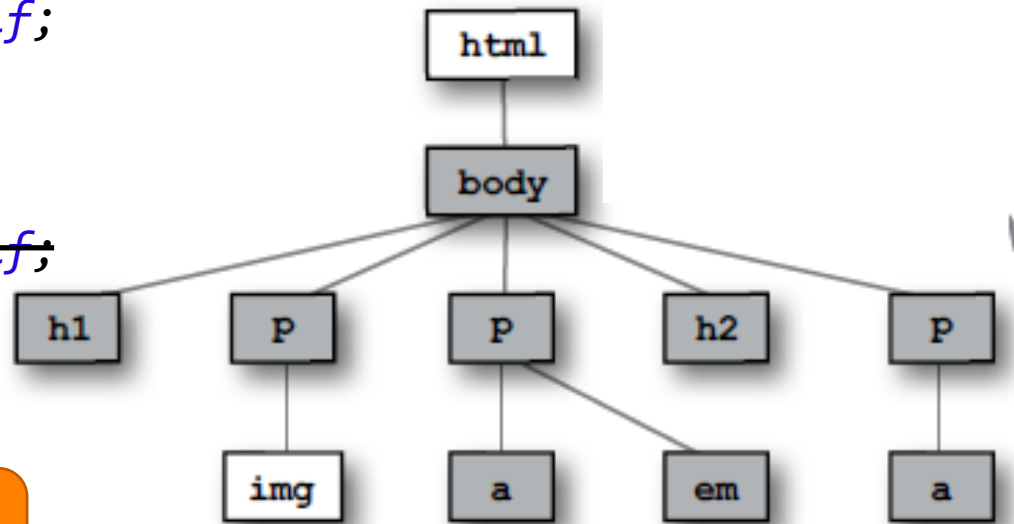
## ➡ การสืบทอด property

- เมื่อมีการกำหนดคุณสมบัติ CSS ในแท็กที่อยู่ระดับเหนือกว่าแท็กที่ซ่อนอยู่จะได้รับคุณสมบัตินั้นไปด้วย

```
body {  
  font-family: sans-serif;  
}
```

```
h1 {  
font-family: sans-serif;  
  color: gray;  
}
```

ไม่จำเป็นต้องใส่ เพราะได้รับการสืบทอด  
จากแท็กที่อยู่ level เหนือกว่าแล้ว





# ➡ ปัญหาการใช้ Selector โดยข้อเท็จ

```
<!doctype html>
<html>
<head>
  <style>
    p {
      color: maroon;
    }
  </style>
</head>
<body>
```

```
  <h1>Computer Science Department, KKU</h1>
  <p>ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์</p>
  <h2>เป้าหมาย</h2>
  <p>ผลิตบัณฑิตที่มีคุณภาพ มีคุณธรรมจริยธรรม และมีความเชี่ยวชาญเฉพาะ</p>
</body>
</html>
```

ถ้าต้องการกำหนดสี font ของแท็ก  
<p> ตัวที่ 2 ให้มีความแตกต่างกับ  
แท็ก <p> ตัวแรกจะอย่างไร ?

## Computer Science Department, KKU

ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัย  
ขอนแก่น จัดตั้งอย่างเป็นทางการเมื่อวันที่ 23 มีนาคม 2537 ผลิต  
บัณฑิตระดับปริญญาตรีมาแล้ว 21 รุ่น และปริญญาโทมาแล้ว  
มากกว่า 10 รุ่น

### เป้าหมาย

ผลิตบัณฑิตที่มีคุณภาพ มีคุณธรรมจริยธรรม และมีความเชี่ยวชาญ  
เฉพาะด้าน ตรงกับความต้องการของผู้ประกอบการ

## ➡ Class Selector

- การกำหนด selector เป็นชื่อ tag จะทำให้ tag เดียวกันมีรูปแบบเหมือนกันหมด ดังนั้นเราจะตั้งชื่อ Class ขึ้น เพื่อใช้ในการจำแนกรูปแบบที่มีความแตกต่างกัน

คลาส red

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

คลาส greentea

```
p.greentea {  
    color: green;  
}
```

# Class Selector

```
<!doctype html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
p.red {  
    color: red;  
}  
  
p.greentea {  
    color: green;  
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<h1>Computer Science Department, KKU</h1>
```

```
<p class="red">ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์</p>
```

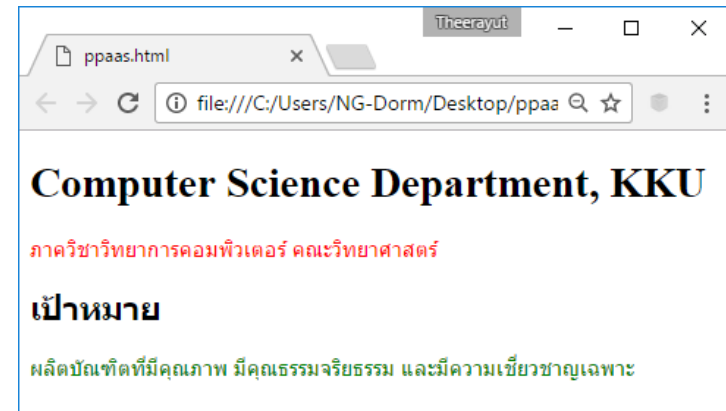
```
<h2>เป้าหมาย</h2>
```

```
<p class="greentea">ผลิตบัณฑิตที่มีคุณภาพ มีคุณธรรมจริยธรรม และมีความเชี่ยวชาญเฉพาะ</p>
```

```
</body>
```

```
</html>
```

ผลลัพธ์



# ➡ class ที่ไม่ผูกกับแท็ก

```
<!doctype html>
```

```
<html>
```

```
<head>
```

```
<style>
```

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

```
.greentea {  
    color: green;  
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<h1 class="greentea">Computer Science Department, KKU</h1>
```

```
<p class="red">ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์</p>
```

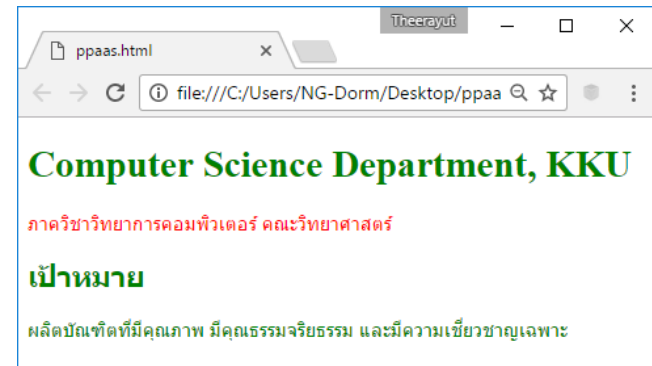
```
<h2 class="greentea">เป้าหมาย</h2>
```

```
<p class="greentea">ผลิตบัณฑิตที่มีคุณภาพ มีคุณธรรมจริยธรรม และมีความเชี่ยวชาญเฉพาะ</p>
```

```
</body>
```

```
</html>
```

ผลลัพธ์

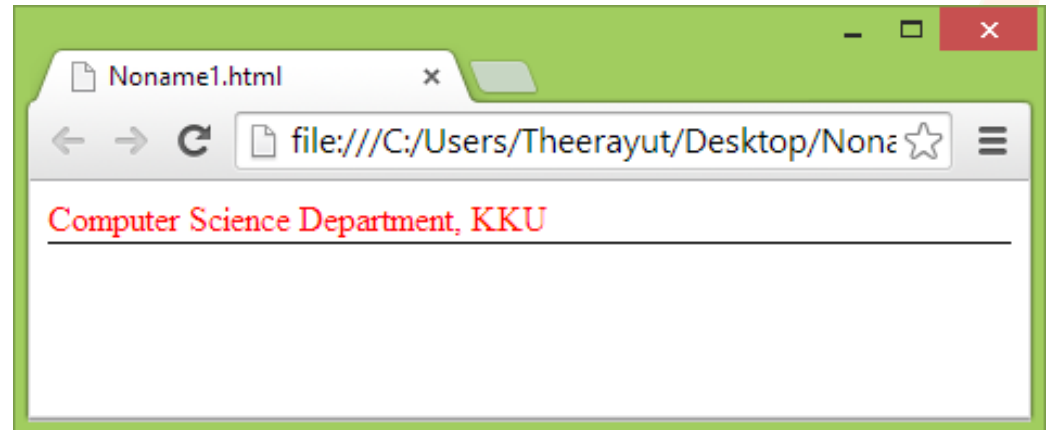


# ➡ การใช้รูปแบบจากคลาส 2 คลาสขึ้นไป

ผลลัพธ์

```
<html>
<head>
<style>
  .red {
    color: red;
  }

  .line {
    border-bottom: 1px solid black;
  }
</style>
</head>
<body>
  <p class="red line">Computer Science Department, KKU</p>
</body>
</html>
```



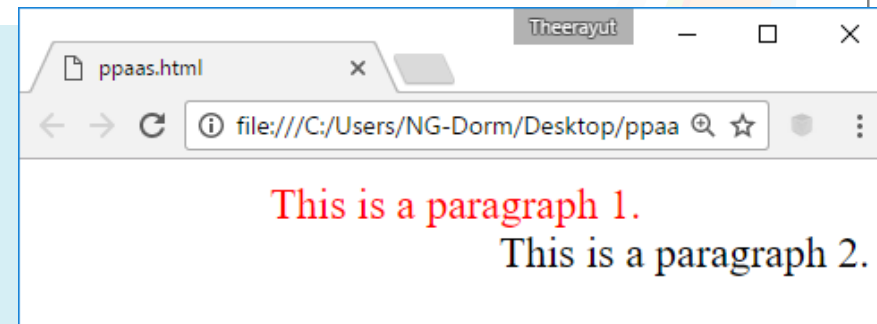
ค้นชื่อคลาสที่นำมาใช้ด้วยช่องว่าง

## ➡ ID Selector

- id ใช้สำหรับกำหนดรูปแบบที่ปรับใช้กับ 1 Element เท่านั้น (single/unique element)

```
<html>
<head>
<style>
  #para1 { text-align:center; color:red; }
  #para2 { text-align:right; }
</style>
</head>
<body>
  <div id="para1">This is a paragraph 1.</div>
  <div id="para2">This is a paragraph 2.</div>
</body>
</html>
```

ผลลัพธ์

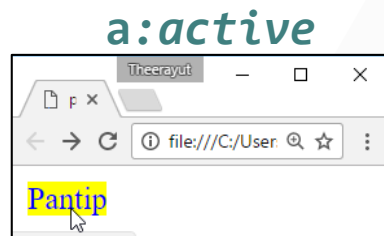
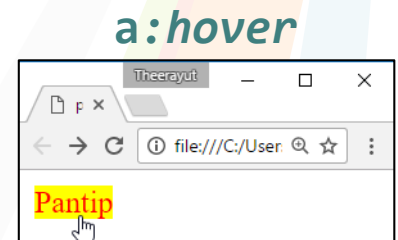


# ➔ Pseudo Selectors

- Element บางตัวสามารถใส่รูปแบบพิเศษได้ เช่น แท็ก `<a>` มี 4 สถานะ ได้แก่ link, visited, hover, และ active การกำหนดรูปแบบให้แตกต่างกันไปในแต่ละสถานะนี้เรียกว่า Pseudo Selectors

```
<!DOCTYPE html>
<html>
<head>
<style>
  a:link      {color:green; text-decoration:none;}
  a:hover     {color:red;background-color:yellow;}
  a:active    {color:blue;}
  a:visited   {color:pink;}
</style>
</head>
<body>
  <a href="http://www.pantip.com">Pantip</a>
</body>
</html>
```

ผลลัพธ์

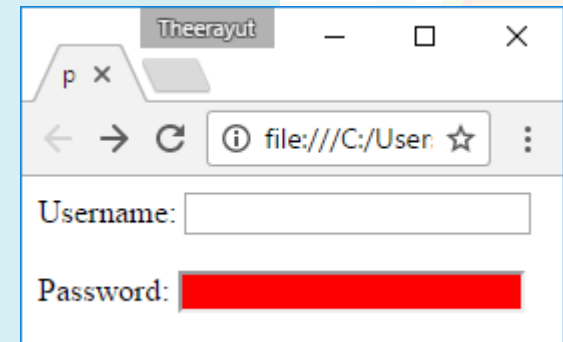


# ➡ Attribute Selectors

- Attribute Selectors คือ การเลือกใส่รูปแบบให้กับ Element ที่มีค่า attribute ตามที่กำหนด

```
<!DOCTYPE html>
<html>
<head>
<style>
  input[type="password"] {
    background-color:red;
  }
</style>
</head>
<body>
  Username: <input type="text" name="username"><br>
  Password: <input type="password" name="pwd">
</body>
</html>
```

ผลลัพธ์



The screenshot shows a web browser window with the title 'Theerayut'. The address bar shows 'file:///C:/User'. The page contains a login form with two fields: 'Username:' followed by a text input field, and 'Password:' followed by a password input field. The password input field has a red background color, which is the result of the CSS attribute selector applied in the code.



# ➡ สรุป Selector

Selectors	CSS code	HTML code
Tag	<code>h1, h2 { color: red; }</code>	<code>&lt;h1&gt;...&lt;/h1&gt;</code> <code>&lt;h2&gt;...&lt;/h2&gt;</code>
Tag ร่วมกับ Class	<code>p.large { font-size: 2em; }</code>	<code>&lt;p class="large"&gt;...&lt;/p&gt;</code>
Class	<code>.large { font-size: 16pt; }</code>	<code>&lt;p class="large"&gt;...&lt;/p&gt;</code> หรือ <code>&lt;div class="large"&gt;...&lt;/div&gt;</code> หรือ <code>&lt;span class="large"&gt;...&lt;/span&gt;</code>
id	<code>#para1 { text-align:center; }</code>	<code>&lt;p id="para1"&gt;...&lt;/p&gt;</code> หรือ <code>&lt;div id="para1"&gt;...&lt;/div&gt;</code> หรือ <code>&lt;span id="para1"&gt;...&lt;/span&gt;</code>
Pseudo	<code>a:link { text-decoration:none; }</code>	<code>&lt;a href="http://www.cc.com"&gt;...&lt;/a&gt;</code>
Attribute	<code>input[type="password"] { background-color:red; }</code>	<code>&lt;input type="password"&gt;</code>

## ➡ CSS Comment

- การอธิบายส่วนต่างๆ ของโค้ด CSS จะช่วยให้การแก้ไขโค้ดง่ายขึ้นในภายหลัง Comment ที่เขียนขึ้น Browser จะไม่สนใจ

```
/*This is a comment*/  
p {  
    text-align: center;  
    /*This is another comment*/  
    color: black;  
    font-family: arial;  
}
```

# ➡ การกำหนดสีใน CSS

- แบบคำที่มีความหมายสี เช่น silver, maroon, yellow

ดูสีอื่นที่ [https://developer.mozilla.org/en/docs/Web/CSS/color\\_value](https://developer.mozilla.org/en/docs/Web/CSS/color_value)

- แบบเลขฐาน 16

#ff0000  
R G B

- แบบเลขฐาน 10

rgb(255, 255, 0)  
R G B

- แบบกำหนด % ความเข้มของสี

rgb(80%, 80%, 100%)

## ตัวอย่าง

```
p {  
  font-family: Arial, sans-serif;  
  background-color: #00ff00;  
  color: rgb(255,255,0);  
  font-size: 2em;  
  font-weight: bold;  
}
```

# ➡ คุณสมบัติ Transparency

- การทำสีแบบโปร่งแสงจะใช้  
วิธีกำหนดค่าสีด้วย

rgba(red, green, blue, ค่า alpha)

ค่า alpha มีค่าในช่วง 0-1

0 - โปร่งใส

1 - สีทึบ

- การทำให้ภาพมีความโปร่ง  
แสงจะใช้คุณสมบัติ

opacity: ค่า alpha

```
<html>
<head>
<meta charset="utf-8">
<style>
  h1 {
    background-color: rgba(34, 234, 0, 0.3);
    border: 1px #336699 solid;
  }
  img:hover {
    opacity: 0.5;
  }
</style>
</head>
<body background="https://dev.cs.kku.ac.th/_temp/lemon-tile.jpg">
<h1>Computer Science Department, KKU</h1>

</body>
</html>
```

# ➡ หน่วยต่างๆใน CSS

ชื่อหน่วย	คำอธิบาย
%	เปอร์เซ็นต์
in	นิ้ว
cm	เซนติเมตร
mm	มิลลิเมตร
em	1em มีขนาดเท่ากับฟอนต์ปัจจุบันของ Element นั้น 2em หมายถึง 2 เท่าของฟอนต์ปัจจุบัน ตัวอย่างเช่น Element นั้นมีขนาดฟอนต์ 12 pt เมื่อกำหนดขนาด 2em จะเพิ่มขนาดเป็น 24 pt
ex	1ex คือ 1/2 ของความสูงของฟอนต์ปกติ
pt	point (1 pt = 1/72 นิ้ว)
pc	pica (1 pc = 12 points)
px	pixels (จุดบนหน้าจอ)

# ➡ หน่วยต่างๆใน CSS

	แนะนำให้ใช้	ใช้เท่าที่จำเป็น	ไม่แนะนำให้ใช้
สำหรับแสดงผลออกทางหน้าจอ	em, px, %	ex	pt, cm, mm, in, pc
สำหรับแสดงผลออกทางกระดาษที่สั่งจากเครื่องพิมพ์	em, cm, mm, in, pt, pc, %	px, ex	

1in = 2.54cm = 25.4mm = 72pt = 6pc

ที่มา:

<https://www.w3.org/Style/Examples/007/units.en.html>

## ➡ คุณสมบัติพื้นหลัง (Background)

- background-color สีพื้นหลัง

**background-color: #e0ffff;**

- background-image ระบุภาพพื้นหลัง

**background-image: url('img\_tree.png');**

- background-size กำหนดขนาดภาพพื้นหลัง (กว้างและสูง) เช่น

**background-size: 80px 60px;**

- background-attachment กำหนดให้ยึดภาพพื้นหลัง ไม่ให้เลื่อนตาม

**background-attachment: fixed;**

## ➡ **background-position** กำหนดตำแหน่งภาพพื้นหลัง

- left top
- left center
- left bottom
- right top
- right center
- right bottom
- center top
- center center
- center bottom



## ➡ คุณสมบัติ background-repeat

- repeat – วางพื้นหลังให้เต็ม (ค่า default)
- repeat-x – วางพื้นหลังเฉพาะในแนวนอน
- repeat-y – วางพื้นหลังเฉพาะในแนวตั้ง
- no-repeat – วางพื้นหลังเพียงครั้งเดียว

```
body {  
    background-image: url('smiley.gif');  
    background-repeat: no-repeat;  
    background-position: left bottom;  
}
```

# ตัวอย่าง

```
<html>
<head>
<style>
.decoratedBox {
  width: 600px;
  height: 130px;
  border: 1px solid gray;
  margin: 20px;
  padding: 20px;
  background-image: url('top-left.png'), url('bottom-right.png');
```

```
  background-position: left top, right bottom;
```

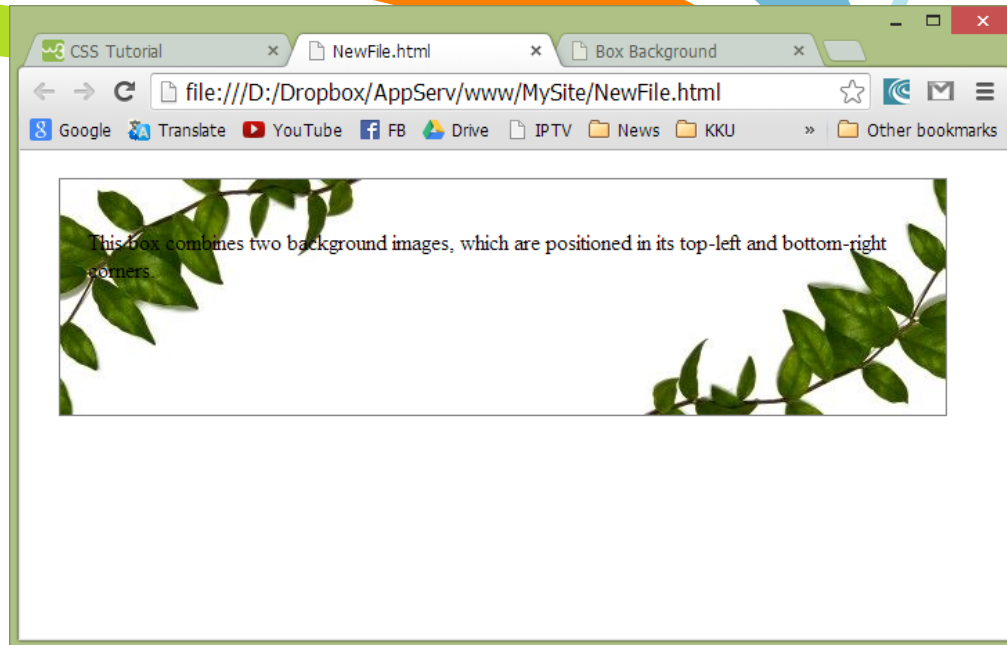
```
  background-repeat: no-repeat, no-repeat;
```

```
}
</style>
</head>
```

ภาพแรก ให้แสดงด้านบน  
ซ้าย แสดงครั้งเดียว

ภาพที่สอง ให้แสดงด้านล่าง  
ขวา แสดงครั้งเดียว

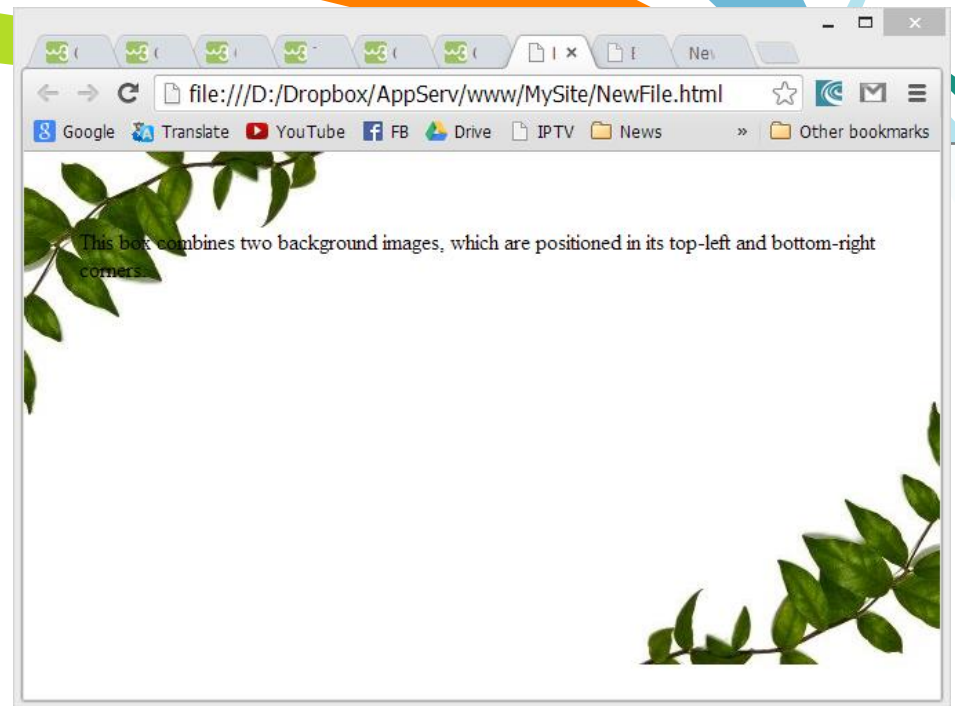
```
<body>
<div class="decoratedBox">
<p>This box combines two background images, which are positioned in its top-left and bottom-right.</p>
</div>
</body>
</html>
```



# ➡ ตัวอย่าง

```
<html>
<head>
<style>
body {
    margin: 20px;
    padding: 20px;
    background-image: url('top-left.png'), url('bottom-right.png');
    background-position: left top, right bottom;
    background-repeat: no-repeat, no-repeat;
}
</style>
</head>

<body>
<p>This box combines two background images, which are positioned in its top-left and bottom-right.</p>
<br><br><br><br><br><br><br><br><br><br><br>
</body>
</html>
```



## ➡ ชนิดของ HTML Element

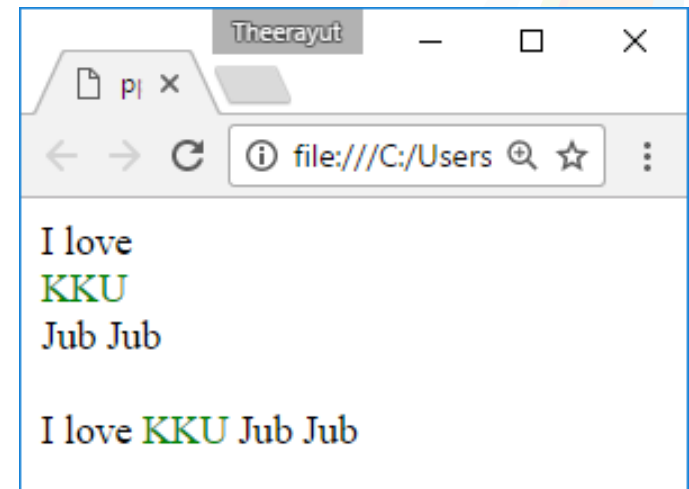
- **Block Element** คือ Element ที่จองพื้นที่บนหน้าเว็บในแนวนอนทั้งหมดของ browser เช่น <h1>, <p>, <ul>, <table>, <div> ซึ่ง browser จะแสดงผลในบรรทัดใหม่เสมอ
- **Inline Element** คือ Element ที่จองพื้นที่บนหน้าเว็บตามขนาดของเนื้อหาภายในแท็ก หรือตามขนาดจริง เช่น <b>, <td>, <a>, <img>, <span> การแสดงผลจะแสดงในแนวนอนเรียงตามลำดับ โดยไม่ขึ้นบรรทัดใหม่ จนกว่าพื้นที่แนวนอนจะเต็ม

## ➡ <div> และ <span>

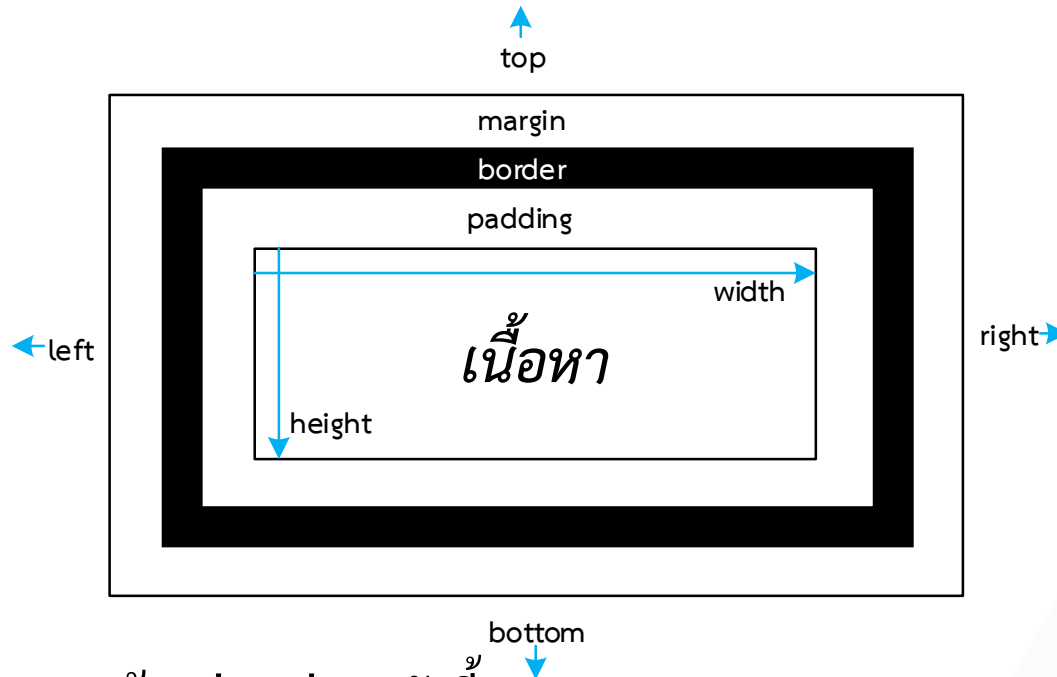
- แท็ก <div> (เป็น Block) และแท็ก <span> (เป็น Inline) มักใช้ในการกำกับ (Markup) หรือครอบส่วนต่างๆ ของหน้าเว็บ เพื่อนำ class หรือ id มาจัดรูปแบบในส่วนที่กำกับ

```
<html>
<head>
  <style>
    .greentea { color: green; }
  </style>
</head>
<body>
  I love <div class="greentea">KKU</div> Jub Jub<br><br>
  I love <span class="greentea">KKU</span> Jub Jub
</body></html>
```

ผลลัพธ์



# ➡ โครงสร้างของ Block Element



Block Element ประกอบด้วยส่วนต่างๆดังนี้

- **Margin** คือ ช่องว่างที่อยู่รอบกรอบ (border) ส่วนนอก ซึ่ง Margin จะไม่มีสีพื้น แต่จะเป็นสีโปร่งใส (Transparent)
- **Border** คือ เส้นกรอบที่ล้อมรอบส่วนที่เป็น Padding และเนื้อหา
- **Padding** คือ ช่องว่างที่อยู่ล้อมรอบเนื้อหา

# ➡ CSS Property สำหรับ Block Element

- การกำหนดพื้นที่ทั้ง 4 ด้าน ใช้เพียง margin หรือ padding

```
margin: 60px; /* margin ทั้งสี่ด้าน 60 pixel เท่ากัน */
```

```
padding: 30px; /* padding ทั้งสี่ด้าน 30 pixel เท่ากัน */
```

```
padding: 15px 30px 15px 30px; /* บน-15 ขวา-30 ล่าง-15 ซ้าย-30 */
```

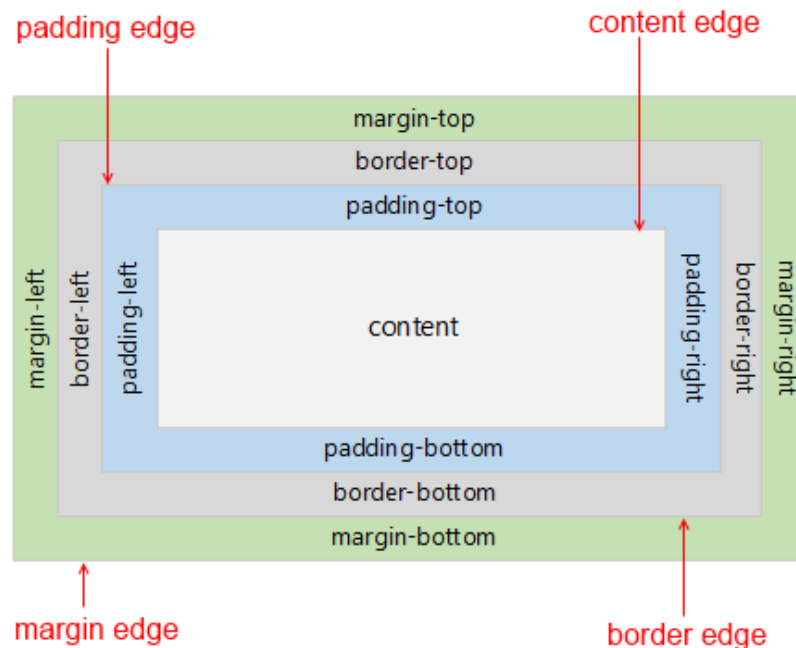
- กำหนดพื้นที่ด้านใดด้านหนึ่ง เช่น

```
margin-left: 50px;
```

```
margin-top: 30px;
```

```
padding-right: 3in;
```

```
padding-bottom: 5cm;
```



# ➔ Property สำหรับ Block Element

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

```
<style>
```

```
.box1 {
```

```
margin: 30px;
```

```
border: 2px solid black;
```

```
padding: 30px;
```

```
width: 400px;
```

```
height: 50px;
```

```
background: gray;
```

```
}
```

```
.rounded {
```

```
border-radius: 20px;
```

```
}
```

```
.shadow {
```

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

```
}
```

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

```
<body>
```

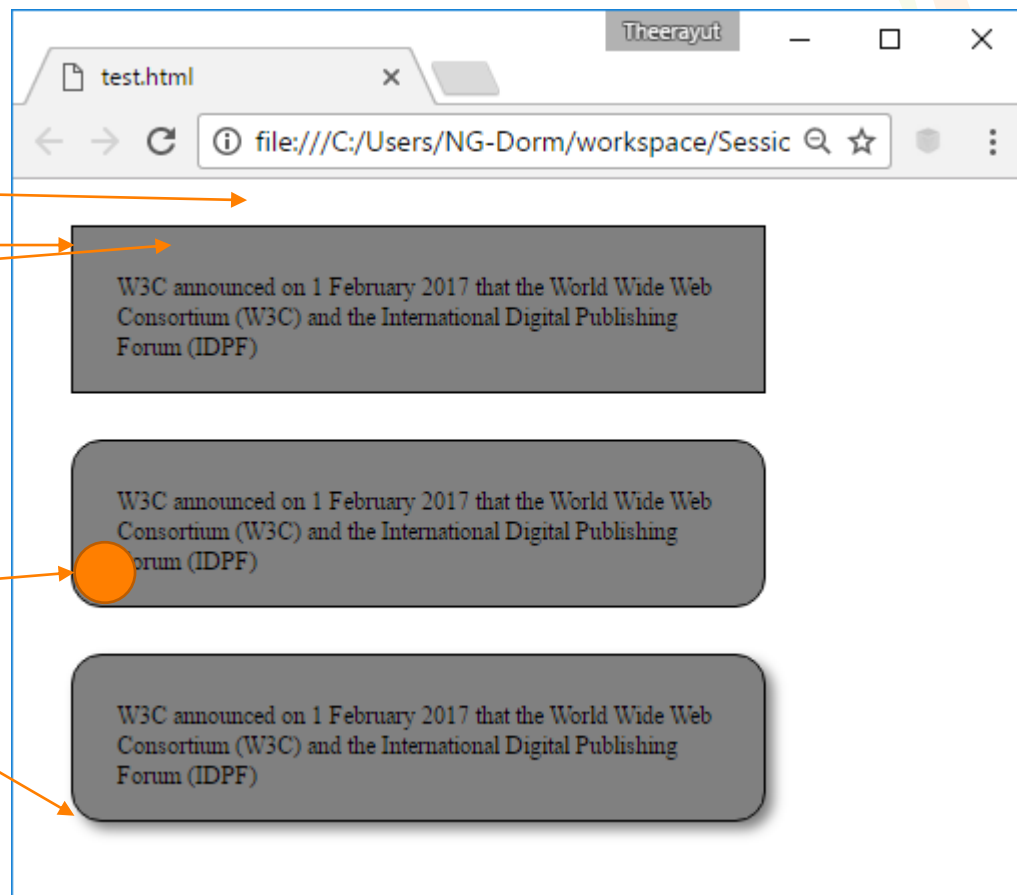
```
<div class="box1">W3C announced on... Forum (IDPF)</div>
```

```
<div class="box1 rounded">W3C announced... Forum (IDPF)</div>
```

```
<div class="box1 rounded shadow">W3C announced... Forum (IDPF)</div>
```

```
</body>
```

```
</html>
```





## ➡ Display Property

- ค่า default ของทุก Element มีค่า display ตามชนิด element อยู่แล้ว
  - Block Element ค่า default คือ **display: block;**
  - Inline Element ค่า default คือ **display: inline;**
- Element ที่เป็น Block สามารถกำหนดให้เป็น inline ได้ เช่น  
**img { display: block; }**
- display property หากกำหนดค่าเป็น none แล้ว element นั้นจะไม่แสดงผล

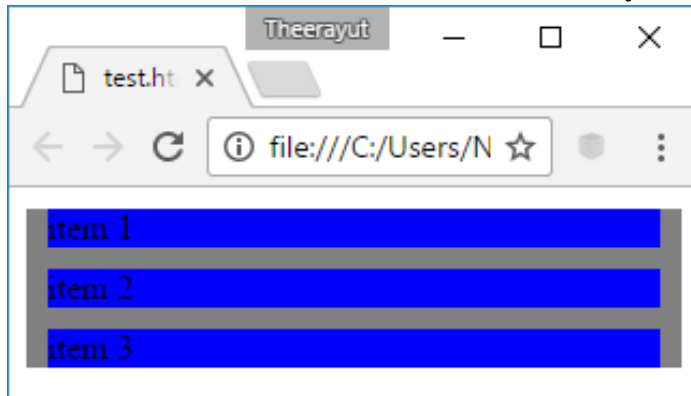
# ➔ display: flex;

- display property ที่มีค่าเป็น flex ใช้ครอบ Block Element ที่ต้องการให้ปรับ width หรือ height ได้ เพราะความกว้างของ Block Element จะใช้พื้นที่ในแนวนอนทั้งหมดเสมอ ส่วนความสูงจะขึ้นกับเนื้อหา

## CSS

```
.flex-container {  
    display: flex;  
    background-color: grey;  
}  
.item {  
    background-color: blue;  
    margin: 10px;  
}
```

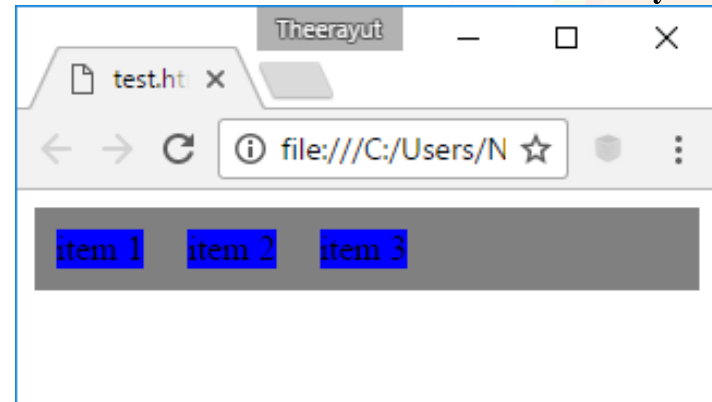
ใช้ Flex Layout



## HTML

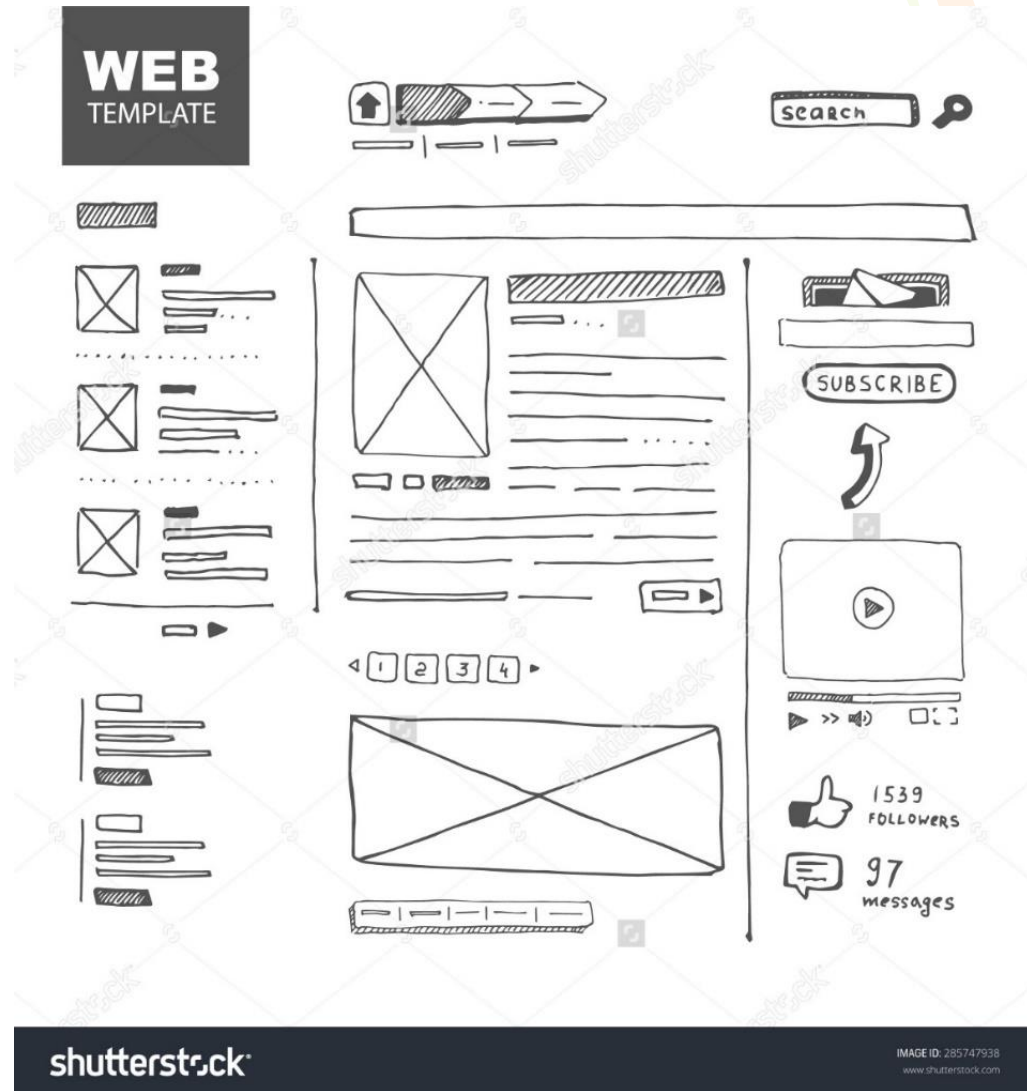
```
<div class="flex-container">  
    <div class="item">item 1</div>  
    <div class="item">item 2</div>  
    <div class="item">item 3</div>  
</div>
```

ไม่ใช่ Flex Layout



# ➡ การจัดโครงสร้างของเว็บ

- การวางโครงสร้างของเว็บเกิดจากการใช้แท็ก `<div>` หรือ Semantic Markup ในการจัดวาง ร่วมกับการใช้คำสั่ง CSS ไม่ควรใช้ `<table>` ในการจัดโครงสร้างเว็บ



# ➡ การจัดโครงสร้างของเว็บด้วย <div>

```
<html><head><style>
body {
  width:500px;
  margin-top: 0px;
  margin-left: auto;
  margin-right: auto;
}
```

```
#header {
  background-color:navy;
  color: white;
  font-size: 25px;
  margin-bottom: 0px;
}
```

```
#section {
  display: flex;
}
```

```
#menu {
  background-color: aqua;
  height: 200px;
  width: 100px;
}
```

```
#content {
  background-color: gray;
  height: 200px;
  width: 400px;
}
```

```
#footer {
  background-color:blue;
  text-align:center;
}
```

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

```
<body>
```

```
<div id="header">My Web Site</div>
```

```
<div id="section">
```

```
<div id="menu">
```

```
<b>Menu</b><br>Add<br>Edit<br>Delete
```

```
</div>
```

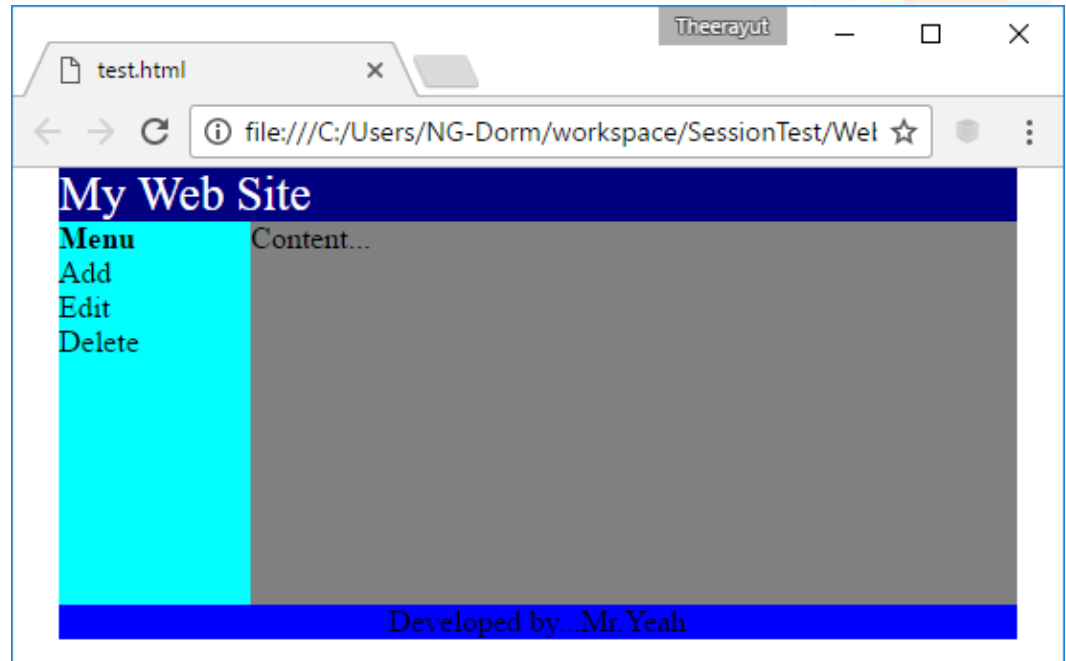
```
<div id="content">Content...</div>
```

```
</div>
```

```
<div id="footer">Developed by...Mr.Yeah</div>
```

```
</body>
```

```
</html>
```



## Semantic Element

- HTML5 มีแท็กที่ใช้ในการกำกับตำแหน่งของหน้าเว็บ ที่มีชื่อที่มีความหมายมากขึ้น

`<article>`, `<section>`, `<aside>`, `<hgroup>`, `<header>`,  
`<footer>`, `<nav>`, `<time>`, `<mark>`, `<figure>`,  
`<figcaption>`

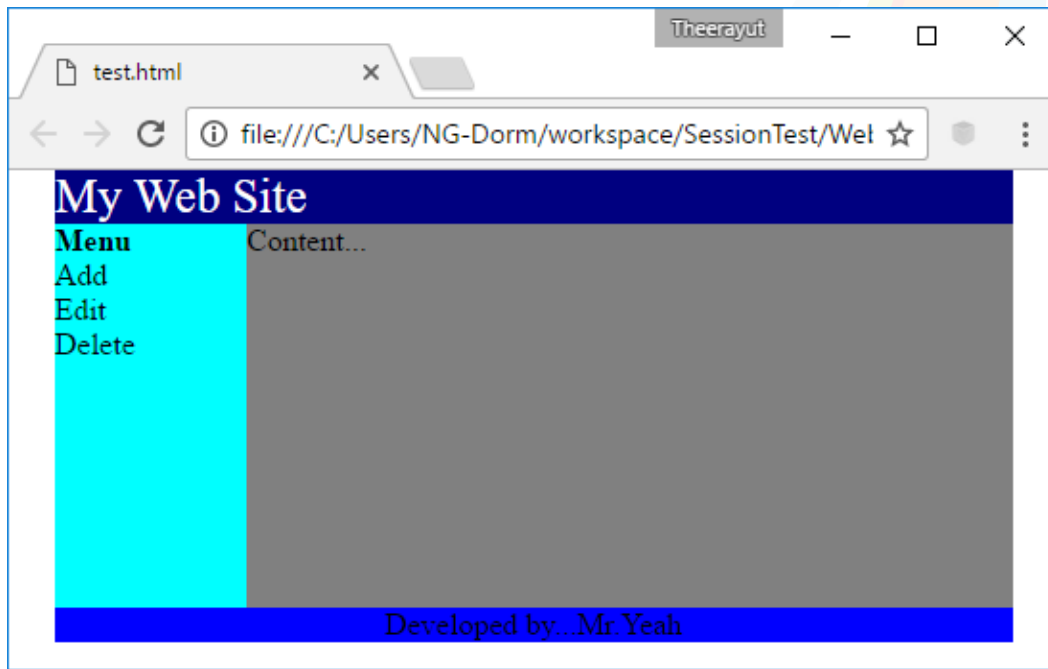
- แท็กเหล่านี้ไม่มีผลต่อการแสดงผลลัพท์ แต่ช่วยให้การกำกับมีความหมายมากขึ้น และยังช่วยให้ Search Engine เข้าใจ
- ศึกษาเพิ่มเติม <https://www.w3.org/TR/html5/sections.html>



# การจัดโครงสร้างของเว็บด้วยแท็ก Semantic

```
<html><head><style>
body {
  width:500px;
  margin-top: 0px;
  margin-left: auto;
  margin-right: auto;
}
header {
  background-color:navy;
  color: white;
  font-size: 25px;
  margin-bottom: 0px;
}
section {
  display: flex;
}
nav {
  background-color: aqua;
  height: 200px;
  width: 100px;
}
article {
  background-color: gray;
  height: 200px;
  width: 400px;
}
footer {
  background-color:blue;
  text-align:center;
}
</style></head>
```

```
<body>
  <header>My Web Site</header>
  <section>
    <nav>
      <b>Menu</b><br>Add<br>Edit<br>Delete
    </nav>
    <article>Content...</article>
  </section>
  <footer>Developed by...Mr.Yeah</footer>
</body>
</html>
```



# ➡ จัดกึ่งกลางข้อความ

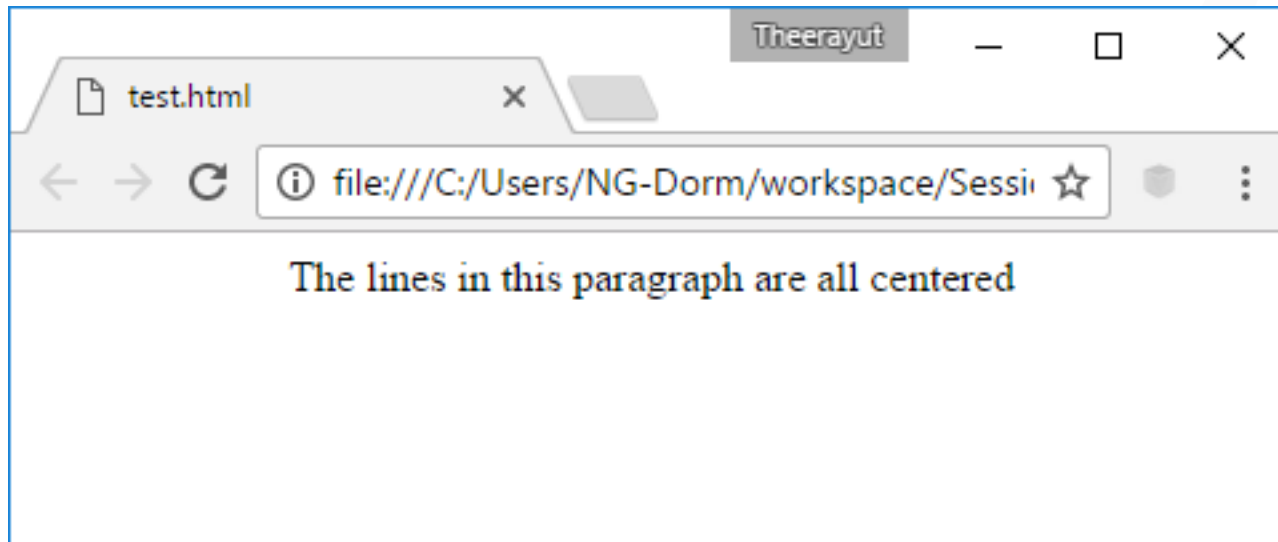
- ใช้แท็กที่เป็น black element และกำหนด text-align เป็น center

## CSS

```
.textcenter {  
    text-align: center;  
}
```

## HTML

```
<div class="textcenter">The  
lines in this paragraph are  
all centered</div>
```



## ➡ จัดกึ่งกลาง block ที่กำหนดขนาด

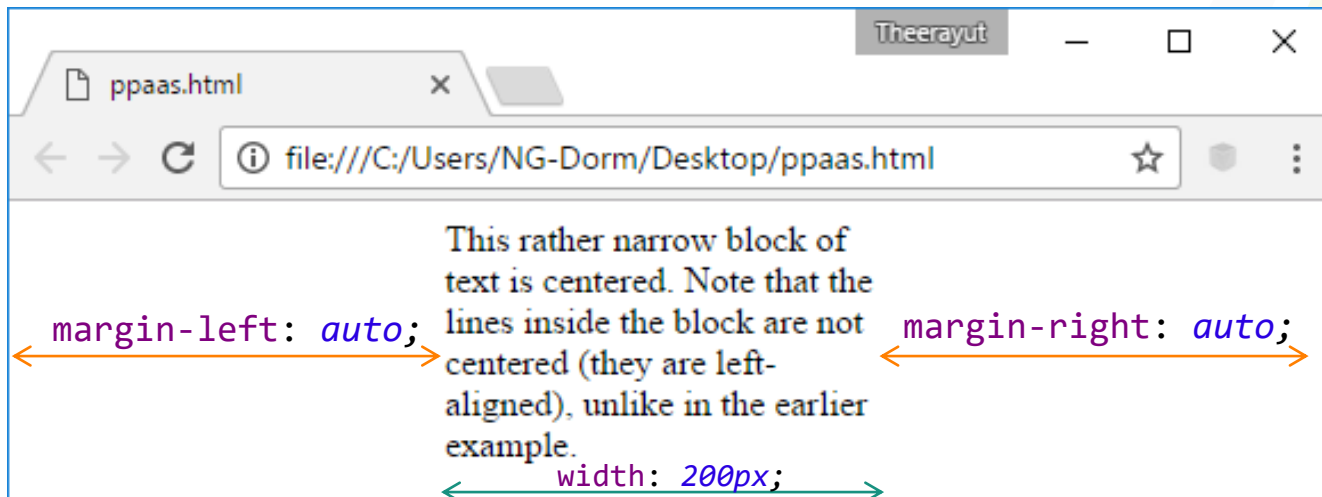
- Block Element ที่ระบุความกว้าง การจัดกึ่งกลางจะกำหนดพื้นที่ด้านซ้ายและขวาให้เท่ากันด้วยการใช้ auto กับ property margin-left และ margin-right

### CSS

```
.blocktext {  
    width: 200px;  
    margin-left: auto;  
    margin-right: auto;  
}
```

### HTML

```
<div class="blocktext">  
    This rather narrow block of  
    text is ...example.  
</div>
```





# ➡ จัดกึ่งกลางรูปภาพ

- กำหนดให้ภาพแสดงผลแบบ Block Element (ปกติภาพเป็นแบบ Inline Element) ด้วย `display: block;` และกำหนด `margin-left` และ `margin-right` เป็น `auto`

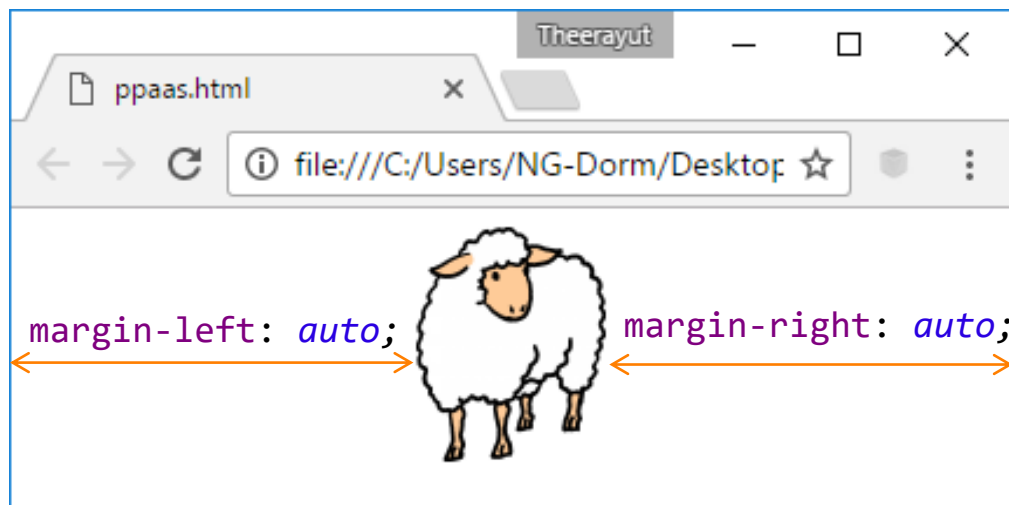
## CSS

```
.center-img {  
  display: block;  
  margin-left: auto;  
  margin-right: auto;  
}
```

## HTML

```

```



# ➡ การจัดกึ่งกลางในแนวตั้ง

- กำหนดส่วนสูงของ Block Element ควรกำหนดเป็น Flex Layout ก่อน แล้วกำหนด align-items: center;

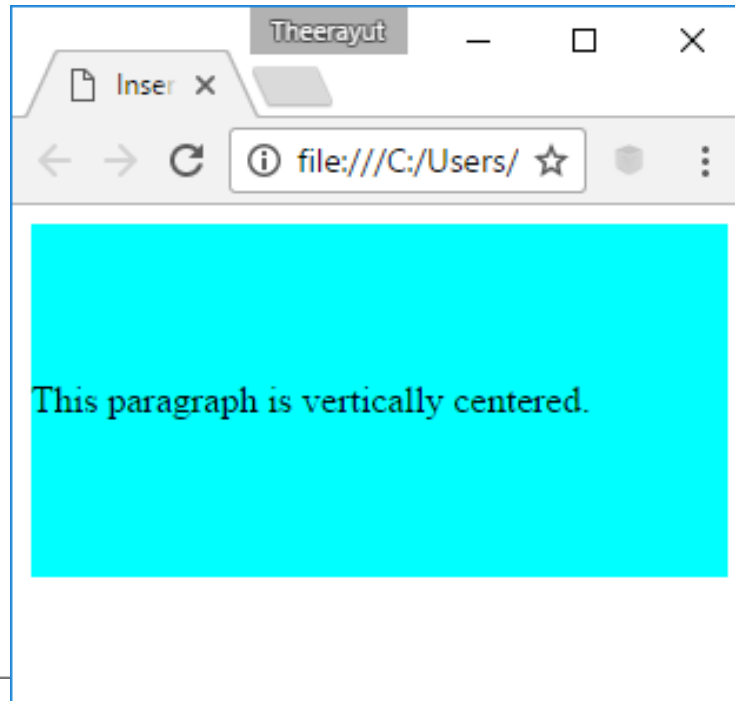
## CSS

```
.vcenter {  
  height: 150px;  
  display: flex;  
  align-items: center;  
  background-color: aqua;  
}
```

## HTML

```
<div class="vcenter">This paragraph is  
vertically centered.</div>
```

height: 150px;



# ➡ การจัดกึ่งกลางทั้งแนวตั้งและแนวนอน

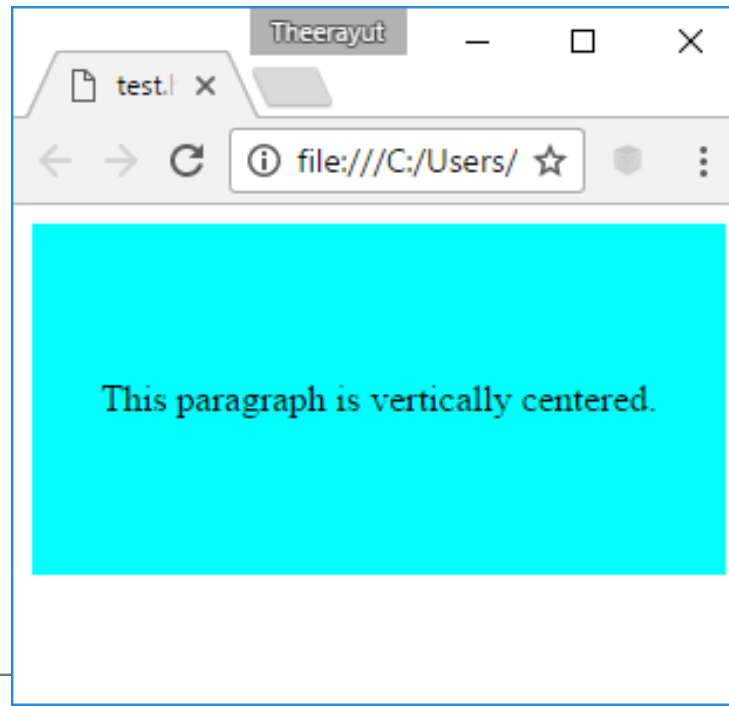
- กำหนดส่วนสูงของ Block Element ควรกำหนดเป็น Flex Layout ก่อน แล้วกำหนด align-items: center;

## CSS

```
.vcenter {  
  height: 150px;  
  display: flex;  
  align-items: center;  
  justify-content: center;  
  background-color: aqua;  
}
```

## HTML

```
<div class="vcenter">This paragraph is  
vertically centered.</div>
```



## ➡ การจัดกึ่งกลางที่ถูกยกเลิกใน HTML5

- attribute align ถูกยกเลิกแล้วใน HTML5 เช่น

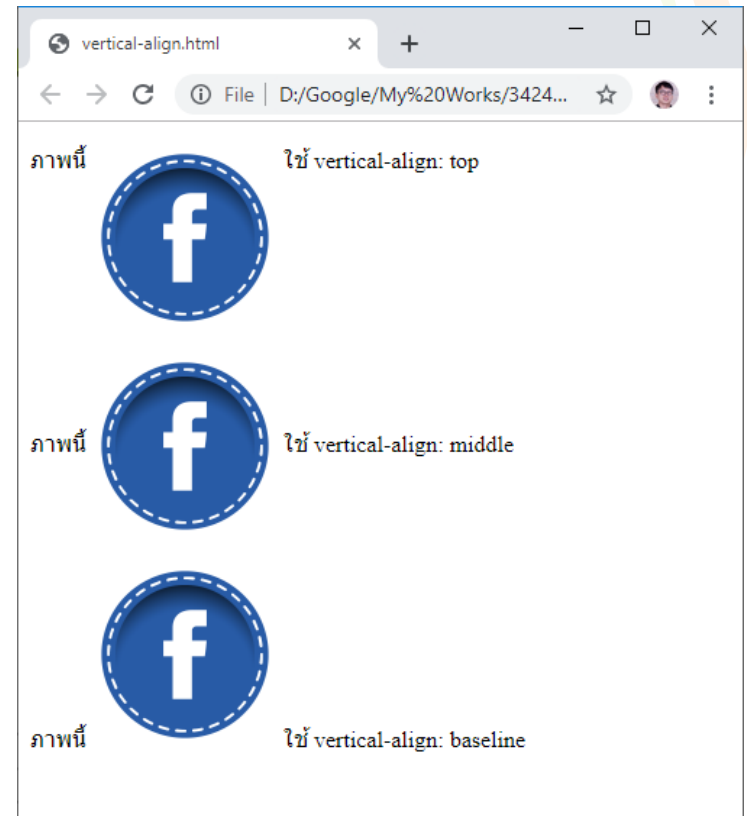
```
<h6 align="center"> ... </h6>
```

- แท็ก center ถูกยกเลิกแล้วใน HTML5 เช่น

```
<center>Hello</center>
```

# ➡ การวางตำแหน่งข้อความและรูปภาพ

```
<!DOCTYPE html>
<html>
<head>
<style>
  #img1 {
    vertical-align: top;
  }
  #img2 {
    vertical-align: middle;
  }
  #img3 {
    vertical-align: baseline;
  }
</style>
</head>
<body>
  <p>ภาพนี้  ใช้ vertical-align: top</p>
  <p>ภาพนี้  ใช้ vertical-align: middle</p>
  <p>ภาพนี้  ใช้ vertical-align: baseline</p>
</body>
</html>
```



## ➡ การจัดวาง Element

Element ในเว็บเพจสามารถจัดวางได้หลายแบบ ดังนี้

- Fixed
- Relative
- Absolute
- Overlapping

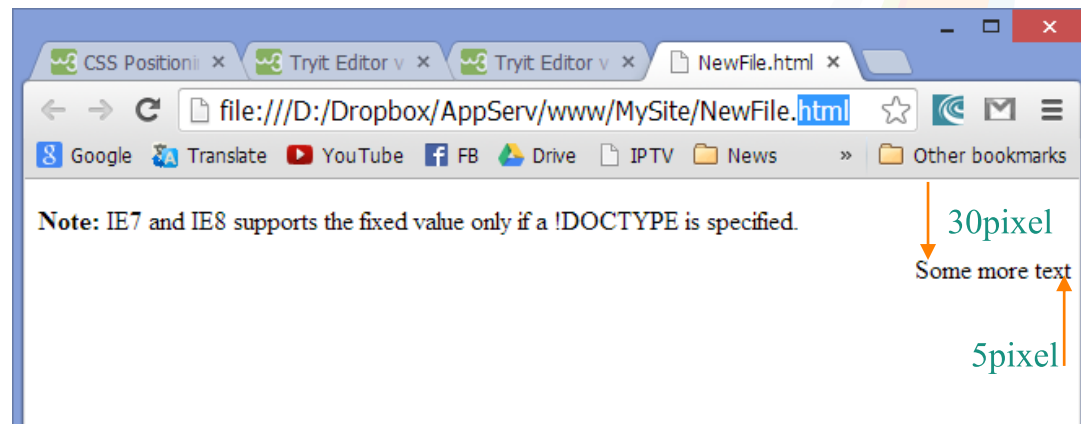
# ➡ การจัดวางแบบ Fixed

- แบบ Fixed คือ การกำหนดตำแหน่งบน Browser โดยที่ Browser จะไม่มีการเคลื่อนที่ Element นั้นเมื่อมีการเลื่อนหน้า

```
<!DOCTYPE html>
<html>
<head>
<style>
  p.pos_fixed {
    position:fixed;
    top:30px;
    right:5px;
  }
</style>
</head>
<body>
```

กำหนดตำแหน่งที่  
จะนำไปวาง

```
<p class="pos_fixed">Some more text</p>
<p><b>Note:</b> IE7 and IE8 supports the fixed value only if a !DOCTYPE is specified.</p>
</body>
</html>
```



# ➡ การจัดวางแบบ Relative

- แบบ Relative คือ การกำหนดตำแหน่งบน Browser โดยเริ่มจากจุดที่ Element นั้นอยู่ เช่น บรรทัดที่ 2 ของหน้า ถ้าต้องการให้เคลื่อนไปทางซ้าย 20pixel ก็จะใช้คำสั่ง left: 20px

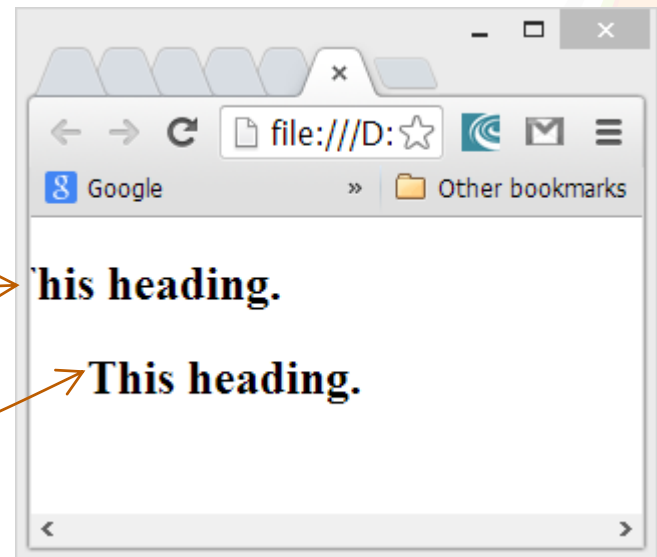
```
<!DOCTYPE html>
<html>
<head><style>
  h2.pos_left {
    position: relative;
    left: -20px;
  }
  h2.pos_right {
    position: relative;
    left: 20px;
  }
</style></head>
<body>
<h2 class="pos_left">This heading.</h2>
<h2 class="pos_right">This heading.</h2>
</body>
</html>
```

ถอยหลังจากขอบ

ทางซ้าย 20 pixel

เดินหน้าจากขอบ

ทางซ้าย 20 pixel

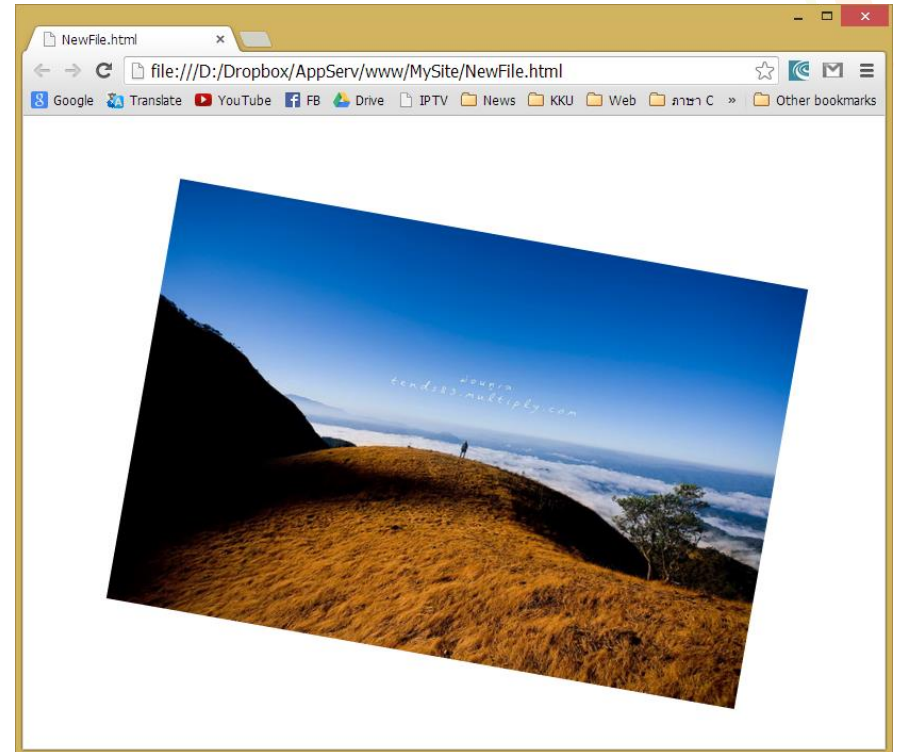




# ➡ ตัวอย่าง

```
<html>
<head>
<style>
img {
    position: relative;
    top: 100px;
    left: 100px;
    transform: rotate(10deg);
}
</style>
</head>
<body>
    
</body>
</html>
```

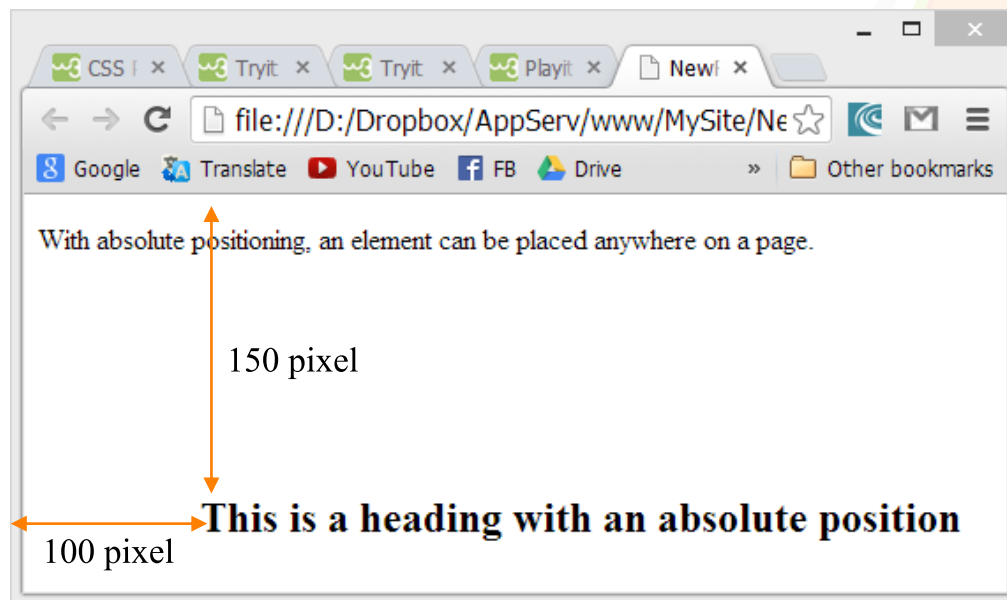
กำหนดให้หมุน 10 องศา  
จากจุดปัจจุบัน



# ➡ การจัดวางแบบ Absolute

- แบบ Absolute คือ การกำหนดตำแหน่งโดยนับจากจุดเริ่มต้นคือด้านบนและด้านข้างของ Browser

```
<!DOCTYPE html>
<html>
<head>
<style>
h2 {
  position: absolute;
  left: 100px;
  top: 150px;
}
</style>
</head>
<body>
<h2>This is a heading with an absolute position</h2>
<p>With absolute positioning, an element can be placed anywhere on a page.</p>
</body>
</html>
```



# ➡ การจัดวางแบบ Overlapping

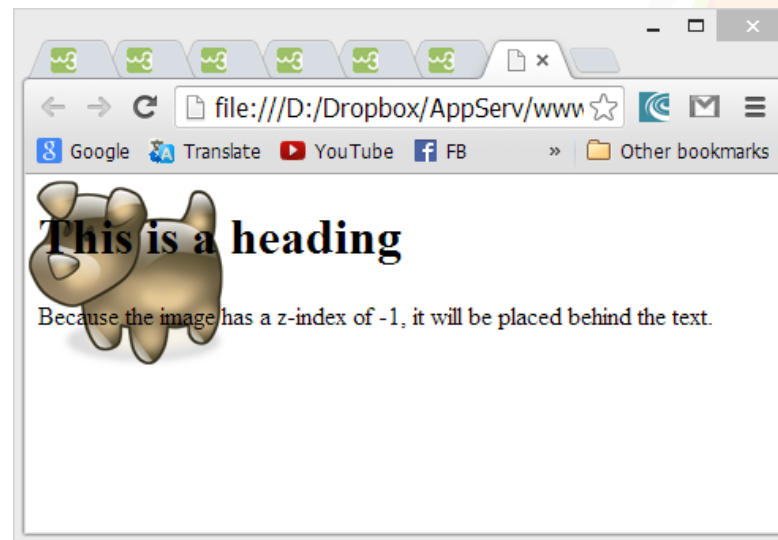
- แบบ Overlapping คือ การจัดลำดับชั้นของ Element ในแนวแกน z โดยกำหนดเป็นค่าบวกหรือลบ ค่าบวกจะทำให้ Element อยู่ด้านบน ค่าลบจะทำให้อยู่ด้านล่าง

```
<!DOCTYPE html>
<html>
<head><style>
img {
  position: absolute;
  left: 0px;
  top: 0px;
  z-index: -1;
}
```

กำหนดตำแหน่งภาพที่  
(0,0) วางไว้ด้านหลัง

```
</style></head>
<body>
<h1>This is a heading</h1>

<p>Because the image has a z-index of -1, it will be placed behind the text.</p>
</body>
</html>
```



# ➡ ตัวอย่าง

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
img {
```

```
    position: relative;
```

```
    top: 100px;
```

```
    left: 100px;
```

กำหนด Pseudo Selector

เพื่อให้ใช้รูปแบบนี้เมื่อมีเมาส์มาวาง

```
}
```

```
img: hover {
```

```
    transform: rotate(10deg) scale(2);
```

```
    z-index: 2;
```

ระบุให้ขยายภาพแนวตั้งและนอน 2 เท่า

```
}
```

```
</style> ให้ภาพอยู่เหนือภาพปัจจุบัน
```

```
</head>
```

```
<body>
```

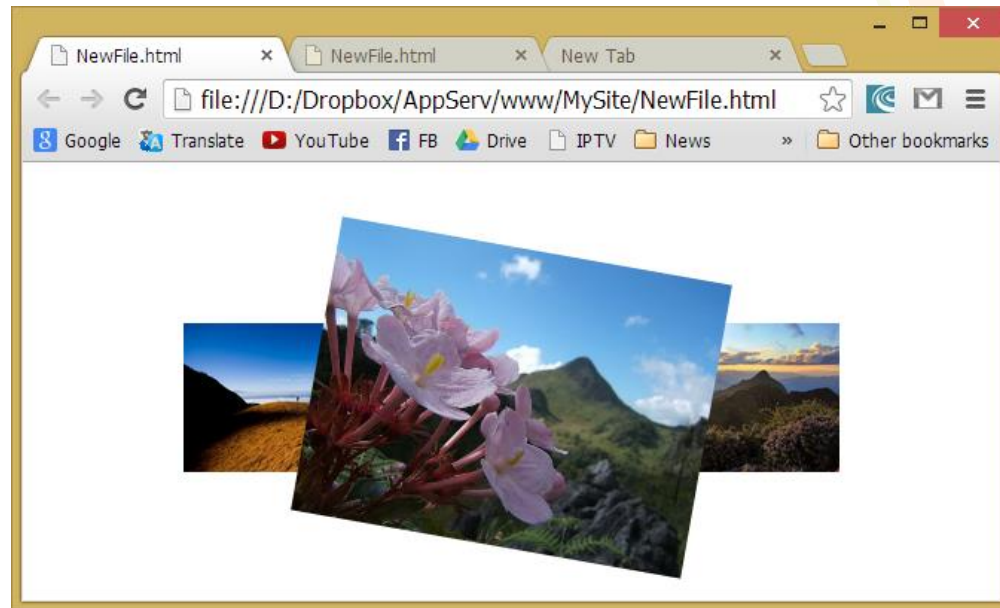
```
    
```

```
    
```

```
    
```

```
</body>
```


```
</html>
```



## ➡ Font

- การกำหนด Font ใน CSS สามารถกำหนดชื่อฟอนต์สำรองไว้ได้หลายชื่อ
- เมื่อ Browser หาฟอนต์บนเครื่องผู้ใช้ไม่เจอ จะค้นหาฟอนต์ที่กำหนดไว้ในลำดับถัดไป

```
body {  
    font-family: Verdana, Geneva, Arial, sans-serif;  
}
```



# ➤ คุณสมบัติ text-shadow

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
#shadow1 {
```

```
font-size: 30px;
```

```
font-weight: bold;
```

```
text-shadow: gray 10px 10px 7px;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<div id="shadow1" align="center">
```

```
Shadow Text
```

```
</div>
```

```
</body>
```

```
</html>
```

สีเงา

ระยะห่างของเงากับข้อความในแนวนอน

ระยะห่างของเงากับข้อความในแนวตั้ง

ค่าการกระจายของเงา

ค่าความ Blur,

## Shadow Text

## ➡ Web Font

- Web Font คือ ฟอนต์ที่เก็บไฟล์ไว้บนเว็บไซต์ของตนเอง เมื่อต้องการใช้ฟอนต์จะโหลดมาพร้อมกับหน้าเว็บเพจ
- Web Font แก้ปัญหาการใช้ฟอนต์ที่มีความแตกต่างกันไปในแต่ละเครื่อง ทำให้ไม่ต้องกังวลว่าเครื่องใดจะไม่มีฟอนต์ติดตั้งอยู่
- Web Font ช่วยให้สามารถใช้ฟอนต์ที่แปลกใหม่ได้ โดยไม่ต้องแปลงข้อความให้เป็นรูปภาพ

## ➡ ชนิดของ Web Font

- TTF, OTF – ใช้ได้กับทุก Browser ยกเว้นบน IE และ IOS
- EOT – ใช้บน IE อย่างเดียว
- WOFF – ใช้ได้ทุก Browser (กำลังจะเป็นมาตรฐาน)
- SVG – ใช้บน IOS เท่านั้น



# ➡ การใช้ Web Font

```
<html>
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<style>
```

```
/* 1. โหลดไฟล์ font */
```

```
@font-face {
```

```
font-family: "myFont";
```

```
src: url("THSarabunNew.ttf");
```

```
}
```

```
/* 2. นำ font ที่โหลดแล้วมาใช้ โดยใช้ชื่อตามที่ตั้งไว้ */
```

```
body {
```

```
font-family: "myFont";
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
ทดสอบ
```

```
</body>
```

```
</html>
```

สามารถเก็บในโฟลเดอร์อื่น และ  
อ้างอิงด้วย Relative URL ได้

```
/* ตั้งชื่อ font */
```

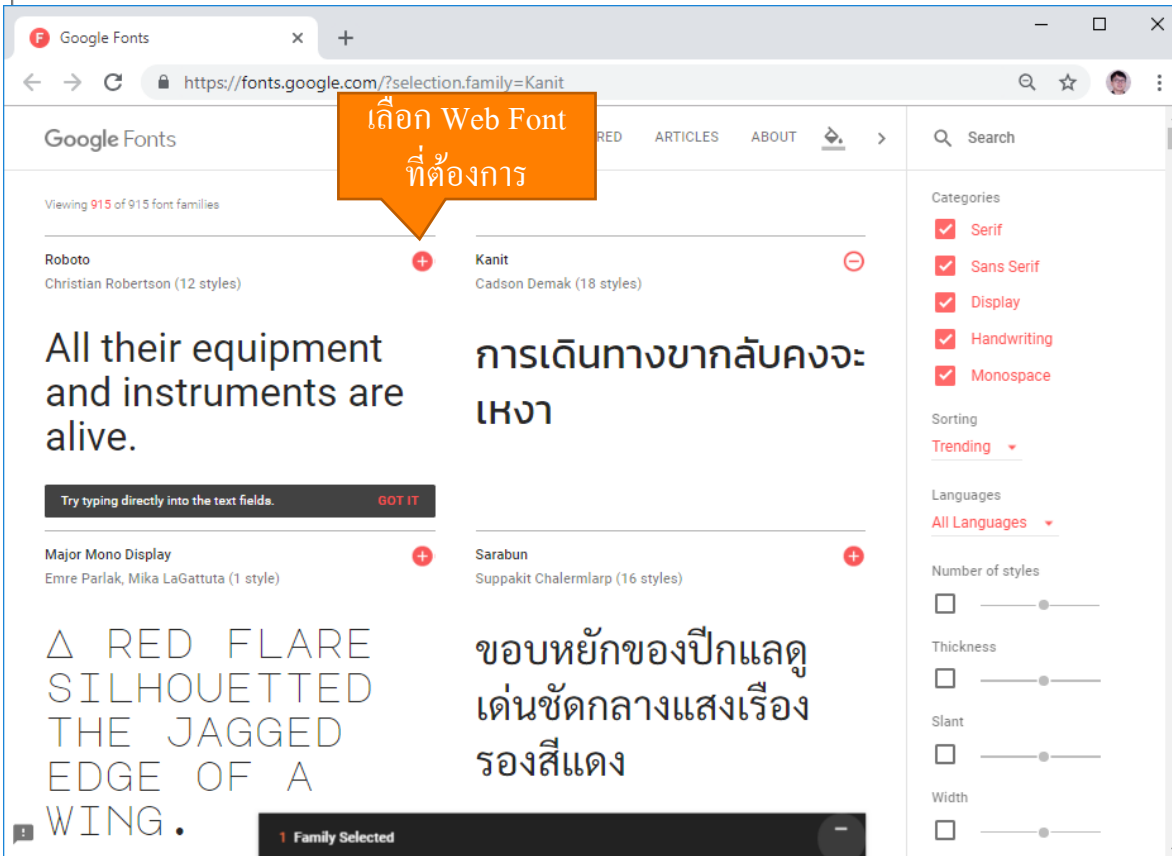
```
/* ระบุชื่อไฟล์ font ที่จะโหลด */
```

# ➡ แหล่ง download font


- Font ภาษาอังกฤษ
  - Font Squirrel, <https://www.fontsquirrel.com>
  - Dafont, <https://www.dafont.com>
  - Everything Fonts, <https://everythingfonts.com/fonts>
- Font ภาษาไทย
  - <https://www.f0nt.com>

# ➔ Google Web Fonts

- Google Web Fonts คือ Web Font แบบ online ที่ให้บริการฟรีแก่นักพัฒนา โดยไม่ต้องโหลดไฟล์ฟอนต์มาไว้ที่เว็บไซต์ของตนเอง นักพัฒนาเพียงระบุฟอนต์ที่ต้องการใช้

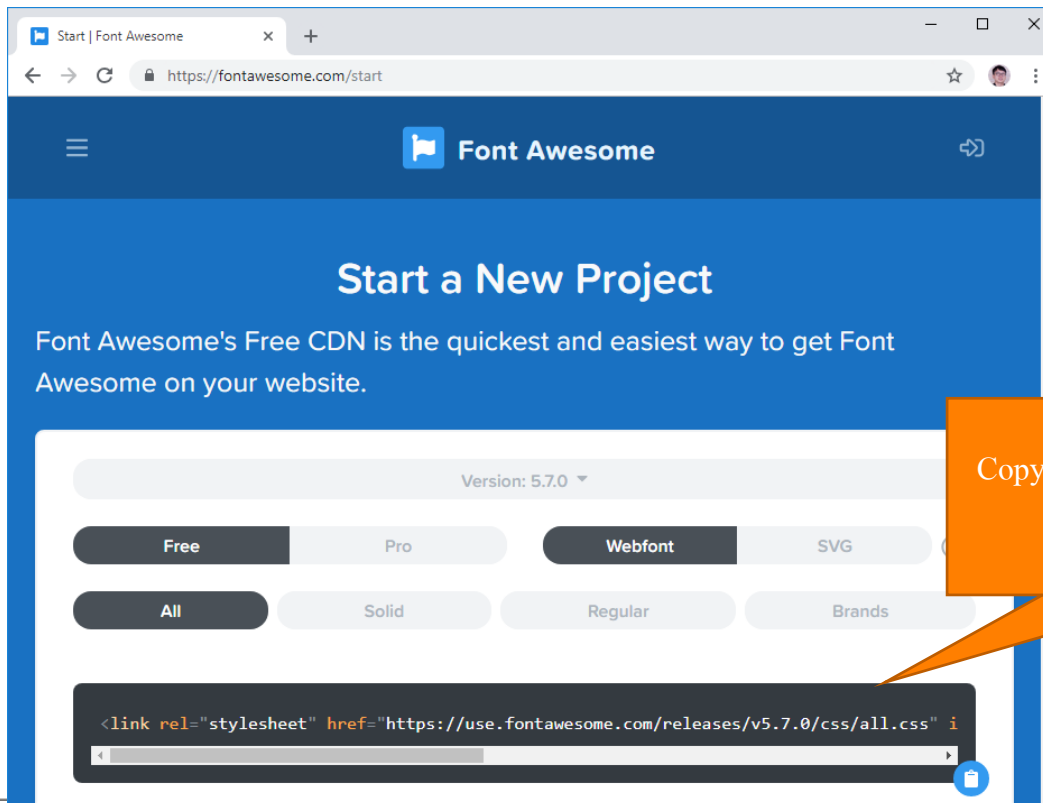


## ขั้นตอนการใช้ Google Web Font

- เลือก Web Font ที่ต้องการจาก <https://fonts.google.com>
- เลือกฟอนต์ที่ต้องการจากไอคอน 
- กดที่แถบ Family Selected จะมีแท็กคำสั่งสำหรับนำไปใช้บนหน้าเว็บ เช่น  
`<link href="https://fonts.googleapis.com/css?family=Kanit" rel="stylesheet">`
- เมื่อต้องการนำฟอนต์มาใช้กับ Selector ใด ให้ใส่ชื่อตามที่ google ระบุ เช่น  
`body {  
 font-family: 'Kanit', sans-serif;  
}`

# Awesome Fonts

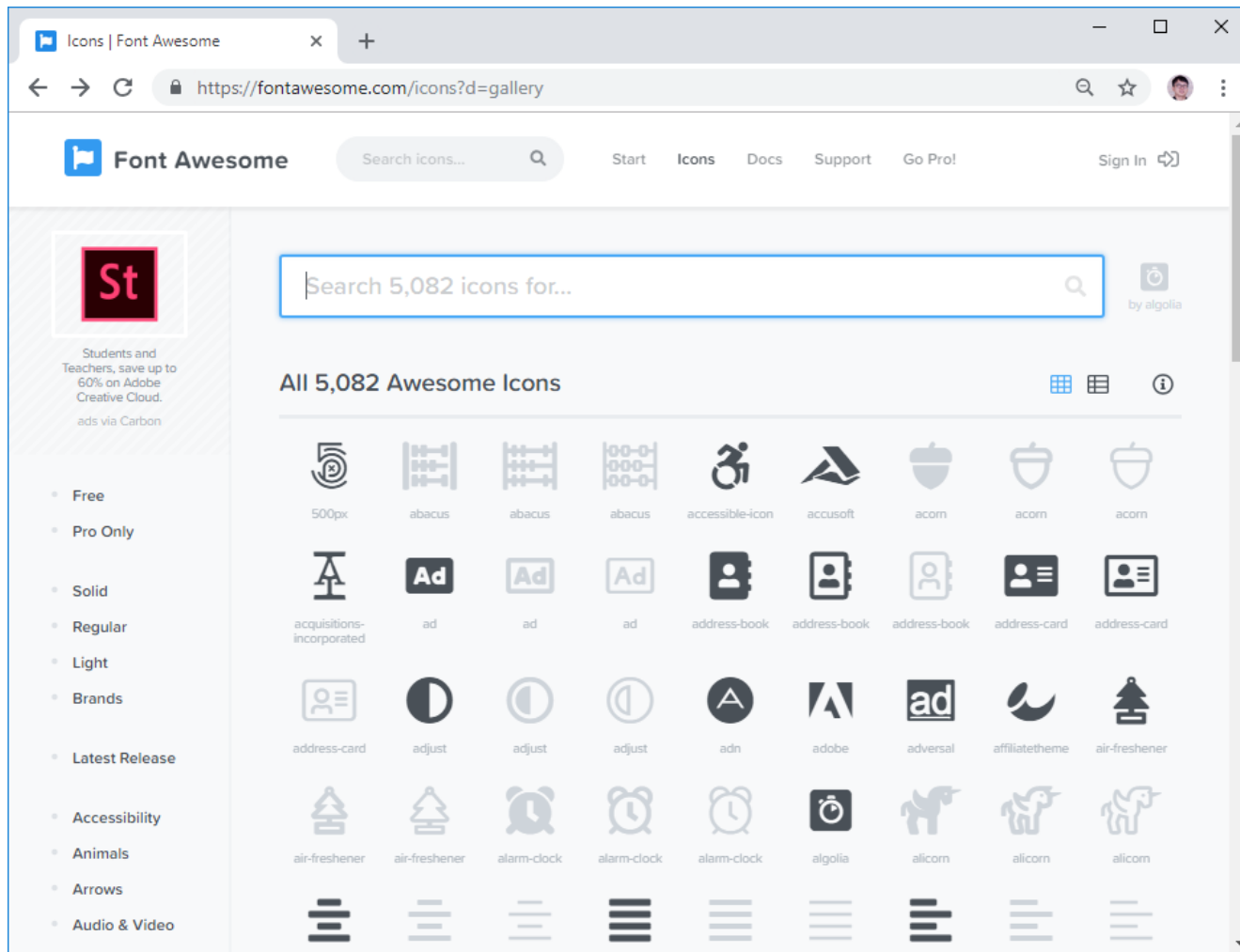
- Awesome Fonts คือ ฟอนต์ที่อยู่ในรูปแบบของ vector icons ใช้สำหรับนักพัฒนาที่ต้องการแสดงไอคอนบนหน้าเว็บ ซึ่งมีความคมชัดสูง
- การอ้างอิงให้กำหนดไฟล์ CSS จากเว็บ <https://fontawesome.com/start>



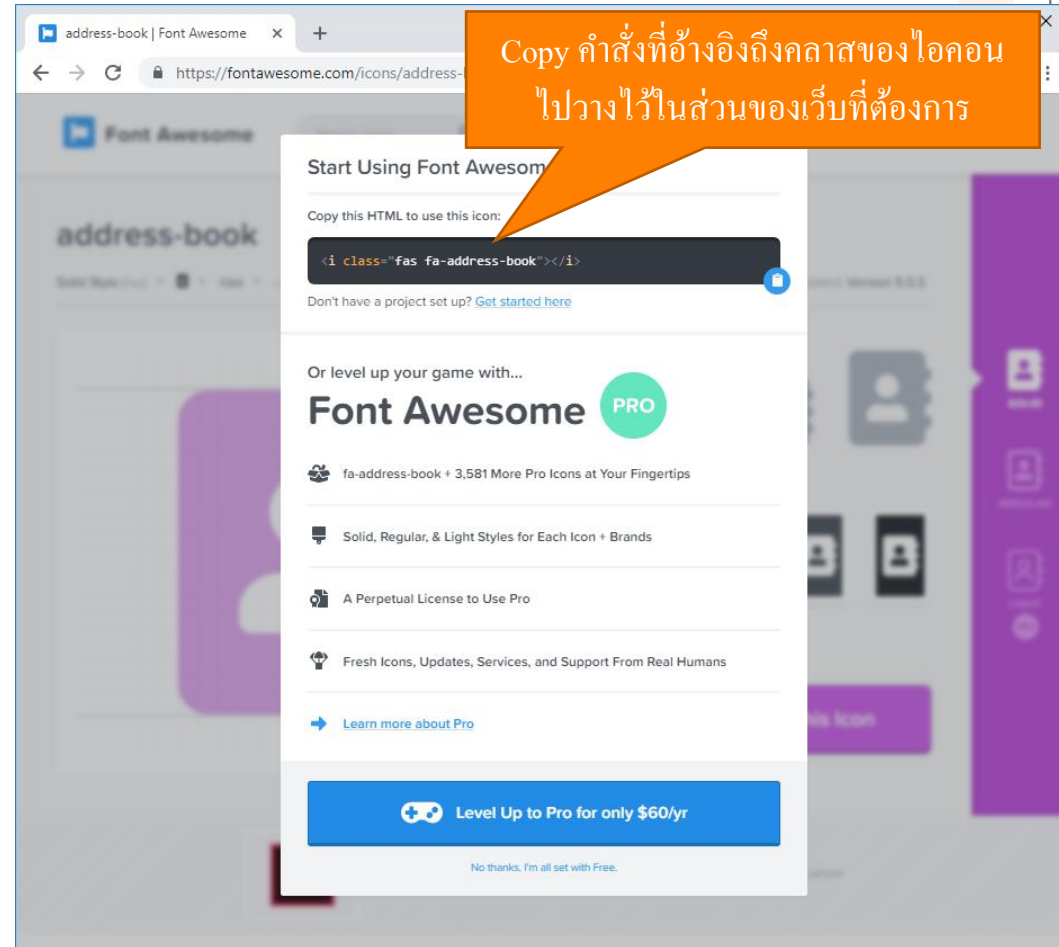
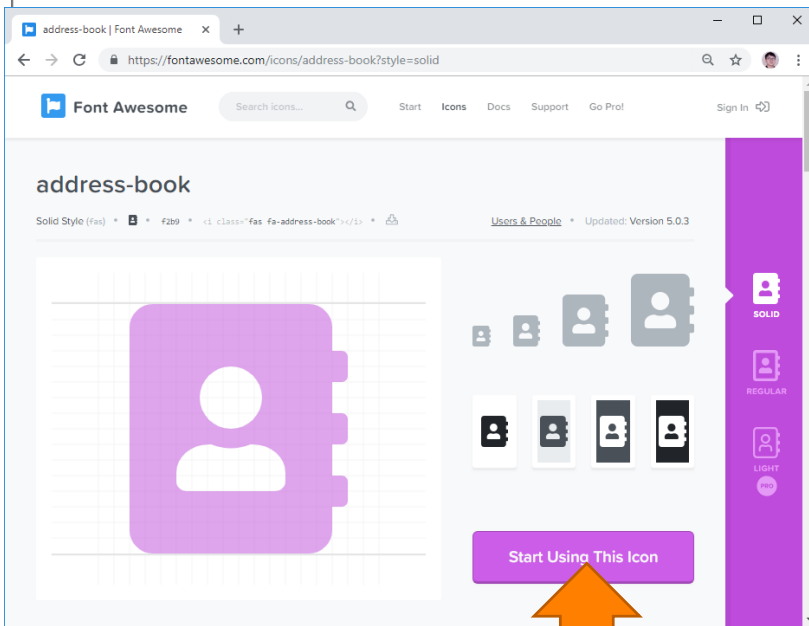
Copy คำสั่งในการโหลด Awesome Fonts  
ไปวางไว้ในแท็ก <head>

# ➡ Awesome Fonts

- เลือกไอคอนที่ต้องการใช้จาก <https://fontawesome.com/icons?d=gallery>



# Awesome Fonts



Copy คำสั่งที่อ้างอิงถึงคลาสของไอคอน  
ไปวางไว้ในส่วนของเว็บที่ต้องการ

# ➡ Responsive Design

- Responsive Design คือ วิธีการพัฒนาเว็บให้สามารถแสดงผลได้ตามขนาดของอุปกรณ์ซึ่งมีอยู่หลากหลาย เช่น เครื่อง PC, Tablet, Smart Phone
- Responsive Design ใช้แนวคิดที่ว่า เขียนครั้งเดียว ทำงานได้ทุกอุปกรณ์ (**write once, run everywhere**)



## ➡ แนวทางในการสร้างเว็บแบบ Responsive Design

- Mobile-first หมายถึง ใช้รูปแบบโครงสร้างเว็บที่มีความเรียบง่ายมากที่สุด โดยออกแบบเริ่มต้นที่หน้าจอมือถือ
- ใช้ค่า CSS Property แบบ Relative (การอ้างอิงจากค่าเริ่มต้น) เช่น ขนาด Font, ขนาดรูปภาพ
- ตรวจสอบหน้าจอด้วย Media Query
- ใช้ CSS Framework ที่รองรับ เช่น Bootstrap, Bulma



# ➡ การกำหนด viewport ด้วยแท็ก <meta>

กำหนดให้ปรับพื้นที่การแสดงผลหน้าเว็บ (viewport) ตามความกว้างของอุปกรณ์ที่ใช้ในการเปิดหน้าเว็บ

```
<meta name="viewport"
```

```
content="width=device-width, initial-scale=1.0, maximum-scale=1.0">
```

คุณสมบัติ	คำอธิบาย
width	ความกว้างของอุปกรณ์ ระบุค่าเป็นตัวเลข (หน่วย pixel) หรือใช้คำว่า "device-width" เพื่อให้ใช้ความกว้างของหน้าจอจริงๆ (หากไม่บอก Browser จะแสดงเหมือนหน้าจอใหญ่ปกติ)
height	ความสูงของอุปกรณ์ ระบุค่าเป็นตัวเลข (หน่วย pixel) หรือใช้คำว่า "device-height" เพื่อให้ใช้ความสูงของหน้าจอจริงๆ
initial-scale	กำหนดการขยาย (zoom) หน้าจอเมื่อมีการเปิดหน้าเว็บ ค่า 1.0 หมายถึงไม่ต้องขยาย
minimum-scale	กำหนดขอบเขตต่ำสุดในการขยาย หากมีค่าเป็น 1.0 ผู้ใช้จะไม่สามารถขยายได้
maximum-scale	กำหนดขอบเขตสูงสุดในการขยาย หากมีค่าเป็น 1.0 ผู้ใช้จะไม่สามารถขยายได้
user-scalable	กำหนดให้ผู้ใช้สามารถขยายหน้าจอได้หรือไม่ ซึ่งมีค่าเป็น yes หรือ no

พื้นฐานของการสร้างเว็บในปัจจุบัน ควรกำหนดค่า meta นี้เสมอ เพื่อรองรับหน้าจอที่มีความแตกต่างกัน

# ➡ การใช้ขนาด Font แบบ Relative

- กำหนดหน่วยของ Font เป็น em แทน px

```
<html>
<head>
  <meta name="viewport" content="width=device-width, initial-
scale=1.0, maximum-scale=1.0">
  <style>
    body      { font-size: 100%; }
    .topic    { font-size: 2em;  }
    .content  { font-size: 1em;  }
  </style>
</head>
<body>
  <div class="topic">หัวข้ออยู่ที่นี้</div>
  <div class="content">เนื้อหา เนื้อหา เนื้อหา เนื้อหา เนื้อหา เนื้อหา เนื้อหา </div>
</body>
</html>
```

กำหนดให้ทั้งเว็บใช้ขนาดฟอนต์มีค่าเริ่มต้นที่ 100%

หากต้องการให้ฟอนต์มีขนาดเพิ่มจากขนาดเริ่มต้นก็เท่า ให้ใช้หน่วย em แทน px เช่น

2 em = 200%

1 em = 100%

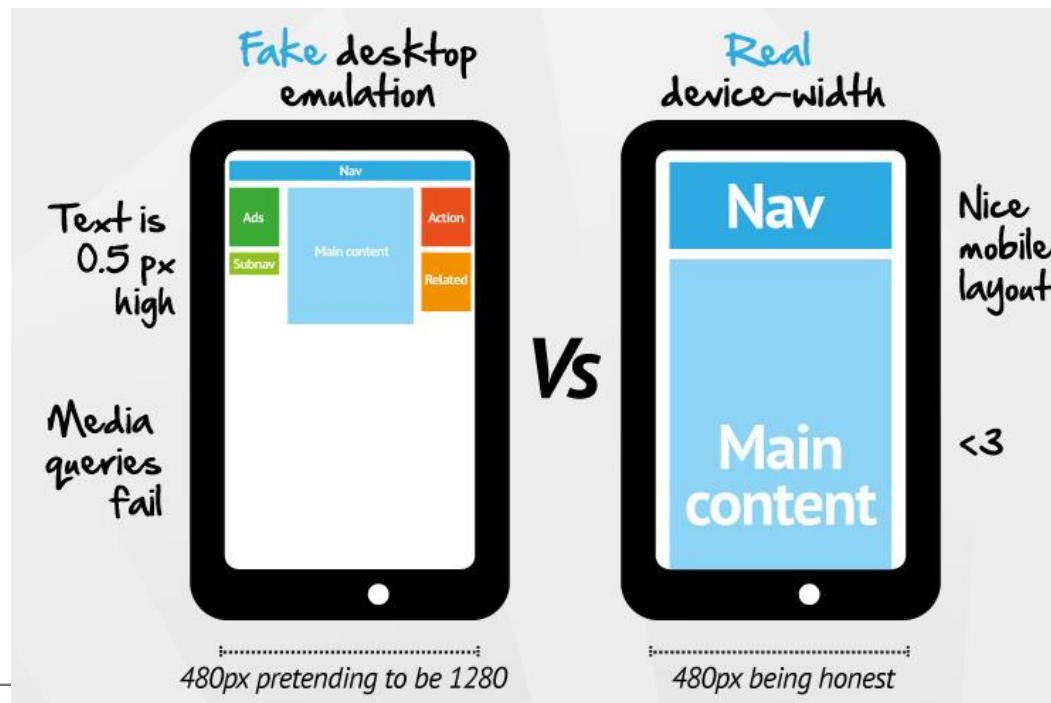
## ➡ การใช้ขนาดรูปภาพ แบบ Relative

- ใช้ max-width เพื่อกำหนดให้แสดงภาพที่ความกว้างเต็มขนาดจอ แต่ความกว้างสูงสุดไม่เกินความกว้างของไฟล์ภาพจริง โดยระบุขนาดเป็นเปอร์เซ็นต์

```
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0">
<style>
    img {
        max-width: 100%;
    }
</style>
</head>
<body>
    <div class="guarantee">This region represents the main content. In this layout, it uses a
    fixed width of 330 pixels. Here is some sample text to make it seem more realistic</div>
    
</body>
</html>
```

# ➡ Media Query

- Media Query คือ คือ การตรวจสอบรายละเอียดของอุปกรณ์ที่ผู้ใช้ ใช้ในการชมเว็บไซต์ เช่น ขนาด, resolution, color capabilities เป็นต้น
- ข้อมูลที่ได้จาก Media query จะช่วยให้นักพัฒนากำหนดรูปแบบ CSS ที่เหมาะกับการแสดงผลได้



## ➡ Media Query

- การกำหนดรูปแบบ CSS ให้กับหน้าจอแต่ละขนาด ทำได้ 2 วิธี ได้แก่
  - แยกไฟล์ CSS สำหรับแต่ละหน้าจอ
  - รวมไฟล์ CSS สำหรับทุกหน้าจอในไฟล์เดียว

## ➡ การแยกไฟล์ CSS สำหรับแต่ละหน้าจอ

```
<link rel="stylesheet" media="screen and (max-width: 480px)" href="smartphone.css">
```

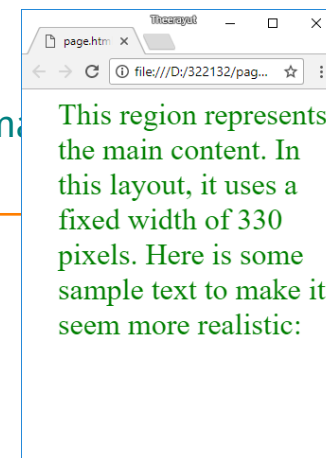
```
<link rel="stylesheet" media="screen and (min-width: 481px)" href="desktop.css">
```

- บรรทัดที่ 1 คือ การกำหนด CSS ให้กับอุปกรณ์ที่มีความกว้างสูงสุด 480 pixel หรือน้อยกว่านั้น
- บรรทัดที่ 2 คือ การกำหนด CSS ให้กับอุปกรณ์ที่มีความกว้างต่ำสุด 481 pixel หรือมากกว่านั้น

# ➡ การรวมไฟล์ CSS สำหรับทุกหน้าอยู่ในไฟล์เดียว

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0, m
<style>
@media screen and (max-width: 480px) {
  .guarantee {
    margin-left: 30px;
    font-size: 2em;
    color: green;
  }
}

@media screen and (min-width: 481px) {
  .guarantee {
    margin-left: 250px;
    color: blue;
  }
}
</style></head>
<body>
<div class="guarantee">This region represents th
this layout, it uses a fixed width of 330 pixels. Here is some sample
text to make it seem more realistic:</div>
</body>
</html>
```



# ➡ การกำหนด Media Query แบบช่วง

*/\* กำหนดรูปแบบของหน้าจอปกติ \*/*

```
@media (min-width: 600px) and (max-width: 700px) {
```

*/\* กำหนดรูปแบบของหน้าจอ 600-700 pixel \*/*

```
}
```

```
@media (min-width: 400px) and (max-width: 599.99px) {
```

*/\* กำหนดรูปแบบของหน้าจอ 400-600 pixel \*/*

```
}
```

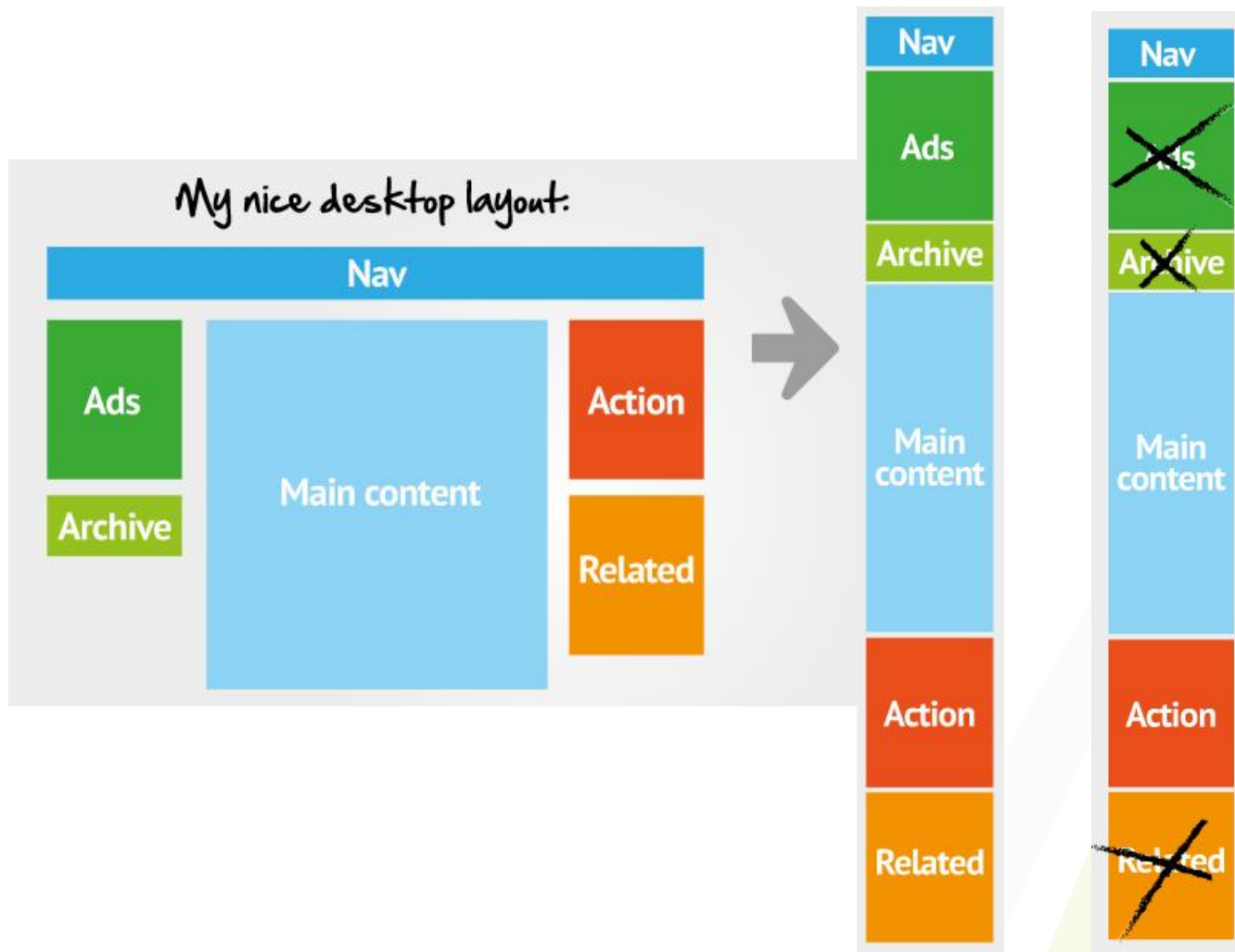
```
@media (max-width: 399.99px) {
```

*/\* กำหนดรูปแบบของหน้าจอที่ต่ำกว่า 400 pixel \*/*

```
}
```



# ➡ การจัด Layout แบบ Responsive Design

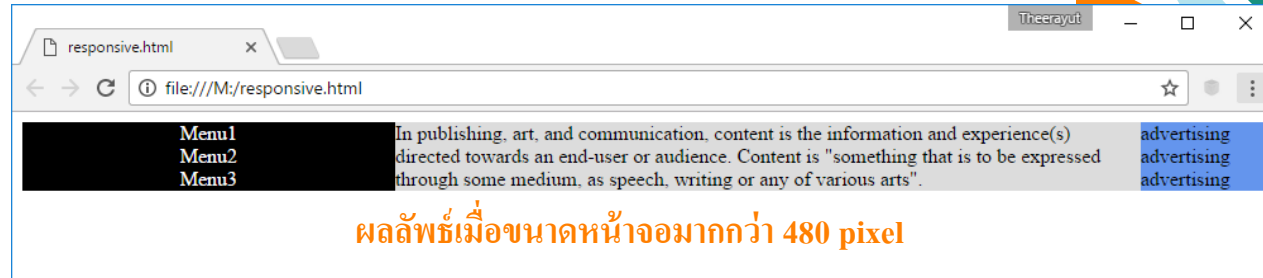


```

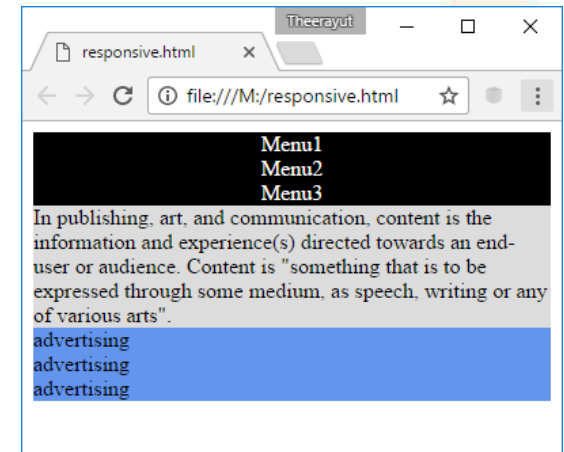
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0">
<style>
  .menu {
    background-color: black;
    color: white;
    text-align: center;
    width: 30%;
  }
  .content {
    background-color: gainsboro;
    width: 60%;
  }
  .ads {
    background-color: cornflowerblue;
    width: 10%;
  }
  .flex { display: flex; }

  @media (max-width : 480px) {
    .flex { display: block; }
    .menu { width: auto; }
    .content { width: auto; }
    .ads { width: auto; }
  }
</style>
</head>
<body>
<div class="flex">
  <div class="menu">Menu1<br>Menu2<br>Menu3<br></div>
  <div class="content">In publishing, art, and communication, ... as
    speech, writing or any of various arts".</div>
  <div class="ads">advertising<br>advertising<br>advertising<br></div>
</div>
</body>
</html>

```



ผลลัพธ์เมื่อขนาดหน้าจอมากกว่า 480 pixel



ผลลัพธ์เมื่อขนาดหน้าจอไม่เกิน 480 pixel