

Google Apps Script

Web app form

4 มิ.ย. 2020

เรียบเรียงโดย
วสันต์ คุณดิลกเศวต

พิมพ์ครั้งที่ 1 : 4 มิ.ย. 2020

Google Apps Script Web App Form

โดย
วสันต์ คุณดิลกเศวต

wasankds@gmail.com

Line ID : wasankds

08-1459-8343

www.poecclub.org

สารบัญ

คำนำ.....	11
-----------	----

Part I - HTML

บทที่ 1 HTML Forms.....	15
1.1. แท็กหรืออิลีเมนต์ <form>	16
1.2. แอทธิบิวต์ action	17
1.3. แอทธิบิวต์ target	17
1.4. แอทธิบิวต์ method	17
1.4.ก.) GET หรือ POST (17)	
1.4.ข.) เมื่อไรใช้ GET เมื่อไรใช้ POST (18)	
1.5. แอทธิบิวต์อื่นๆ	19
1.6. การตั้งค่า CSS สำหรับ <form>	19
บทที่ 2 อิลีเมนต์ <input>.....	21
2.1. อิลีเมนต์ <input>	22
2.2. แอทธิบิวต์ name ของอิลีเมนต์ <input>	22
2.3. Input type "text"	23
2.4. Input type "password"	23
2.5. Input type "submit"	24
2.6. Input type "reset"	25
2.7. Input type "radio"	26
2.8. Input type "checkbox"	26
2.9. Input type "button"	27
2.10. Input type "color"	27
2.11. Input type "date"	28
2.12. Input type "datetime-local"	29
2.13. Input type "email"	30
2.14. Input type "file"	31
2.15. Input type "month"	32
2.16. Input type "number"	33
2.17. Input type "Range"	33
2.18. Input type "search"	34
2.19. Input type "tel"	35
2.20. Input type "time"	36
2.21. Input type "week"	37

บทที่ 3 HTML Form Elements.....	39
3.1. อิเล็มเมนต์นอกเหนือจาก <input>	40
3.2. อิเล็มเมนต์ <select>	40
3.3. อิเล็มเมนต์ <textarea>	41
3.4. อิเล็มเมนต์ <button>	43
3.5. อิเล็มเมนต์ <fieldset> และ <legend>	44
3.6. อิเล็มเมนต์ <datalist> และ <input>	45
3.7. อิเล็มเมนต์ <output>	46
บทที่ 4 แอตทริบิวต์ของ อิเล็มเมนต์ <input> ที่สำคัญ.....	47
4.1. แอตทริบิวต์ (Attributes)	48
4.2. value	49
4.3. readonly	49
4.4. disabled	51
4.5. accept	52
4.6. pattern และ title	53
4.7. placeholder	54
4.8. required	55
4.9. autocomplete	56
4.10. size	57
4.11. maxlength	58
4.12. min และ max	58
4.13. multiple	59
4.14. class	60
4.14.ก.) แอตทริบิวต์ class (60)	
4.14.ข.) แอตทริบิวต์ class แบบ Inline (62)	
4.14.ค.) ใช้หลายคลาส (62)	
4.15. Constraint validation	64
4.15.ก.) checkValidity() (65)	
4.15.ข.) ตัวอย่าง rangeUnderflow (66)	
4.15.ค.) setCustomValidity() (66)	

Part II – Javascript

บทที่ 5 JavaScript HTML DOM.....	71
5.1. The HTML DOM (Document Object Model)	72
5.1.ก.) โมเดล HTML DOM (72)	
5.1.ข.) DOM วัตถุที่สามารถโปรแกรมได้(73)	
5.2. คอนโซลของ Browser	73
บทที่ 6 จั๊บอีเล็มเม้นต์ ด้วยเมธอดต่างๆ.....	75
6.1. วัตถุ HTMLCollection และ NodeList	76
6.1.ก.) วัตถุ HTMLCollection (76)	
6.1.ข.) วัตถุ NodeList (77)	
6.1.ค.) ความแตกต่างระหว่างวัตถุ HTMLCollection และ NodeList (77)	
6.2. getElementById()	77
6.3. getElementsByName	78
6.4. getElementsByTagName()	80
6.5. getElementsByName	81
6.6. querySelector() และ querySelectorAll()	82
6.7. ใช้เมธอดของอาร์เรย์กับวัตถุ NodeList	84
6.7.ก.) Array.prototype (84)	
6.7.ข.) Array.from() (86)	
6.8. innerHTML	88
6.9. parentNode	89
6.10. สร้างอีเล็มเม้นต์ Form ด้วย Javascript	91
บทที่ 7 วัตถุ.....	93
7.1. วัตถุ document	94
7.1.ก.) write() และ writeln() (94)	
7.1.ข.) createElement() (95)	
7.1.ค.) createTextNode() (97)	
7.1.ง.) body (98)	
7.2. คุณสมบัติและเมธอดของอีเล็มเม้นต์	99
7.2.ก.) appendChild() (99)	
7.2.ข.) childNodes (100)	
7.2.ค.) tagName (101)	
7.2.ง.) style (102)	

7.3. วัตถุ Event	103
7.3.ก.) onclick() (104)	
7.3.ข.) Events ที่เป็นส่วนหนึ่งของวัตถุ MouseEvents (105)	
7.3.ค.) target (106)	
7.4. addEventListener()	107
7.4.ก.) addEventListener() (107)	
7.4.ข.) ตัวอย่างที่ 1 (108)	
7.4.ค.) ตัวอย่างที่ 2 (108)	
7.4.ง.) Event Bubbling or Event Capturing?(109)	
7.5. อีเล็มเมนต์ 2 ประเภท	111
7.6. อีเล็มเมนต์ลูก(Child) และ อีเล็มเมนต์แม่(Parent)	111
7.7. วัตถุ this	112
7.7.ก.) การใช้ this ที่จุดต่างๆ (112)	
7.7.ข.) การเชื่อมโยงฟังก์ชัน(Function Binding) (115)	

Part III – CSS

บทที่ 8 การใช้งาน CSS.....	119
8.1. CSS คืออะไร ?	120
8.2. CSS Syntax	121
8.2.ก.) CSS Syntax (121)	
8.2.ข.) Selector, Property, Value (121)	
8.3. ประเภทของ Selectors	122
8.3.ก.) Element selector (<Tage name>) (122)	
8.3.ข.) Id selector (#) (123)	
8.3.ค.) Class selector (.) (123)	
8.3.ง.) Universal Selector (*) (124)	
8.4. การผสมผสานระหว่างประเภทของ Selectors	125
8.5. การใช้ CCS Class หลายตัวกับอีเล็มเมนต์เดียว	125
8.6. การรวมกลุ่ม Selectors	126
8.7. Attribute selectors	127
8.7.ก.) Syntax (128)	
8.8. การใช้งาน CSS ในเอกสาร HTML	128
8.8.ก.) 3 วิธีใส่ Style Sheet ให้กับเอกสาร HTML (128)	
8.8.ข.) Inline CSS (129)	
8.8.ค.) Internal CSS (129)	
8.9. External CSS	130
8.10. Multiple Style Sheets	131
8.11. Cascading Order	132

Part IV – อื่นๆ

บทที่ 9 เครื่องมือและ เทคนิคช่วยสร้างเว็บ.....137

9.1. ทำหน้า Web Apps ด้วย Bootstrap 138

9.2. Bootstrap form 141

9.2.ก.) Stacked Form (.form-control และ .form-group) (141)

9.2.ข.) Inline Form (.form-inline) (142)

9.2.ค.) Inline Form with Utilities (.mr-sm-2 และ .mb-2) (142)

9.2.ง.) Form Row/Grid (.col และ .row) (144)

9.2.จ.) Form validation (.was-validated และ .needs-validation)(146)

9.3. ใช้ฟอนต์ของ Google 149

9.3.ก.) เลือกฟอนต์จากเว็บ fonts.google.com (149)

9.3.ข.) นำฟอนต์ของ Google ไปใช้ในเว็บ (151)

9.4. แยกไฟล์สำหรับ CSS, Javascript แต่อยู่ในโปรเจ็ค 152

9.4.ก.) ขั้นตอนที่ 1 : แยกไฟล์ แยกโค้ด HTML (153)

9.4.ข.) ขั้นตอนที่ 2 : ผัง Scriptlets เพื่อดึงไฟล์ html มารวมในไฟล์หลัก (153)

9.4.ค.) ขั้นตอนที่ 3 : สร้างฟังก์ชันในไฟล์ .js (154)

บทที่ 10 Blob, File และ FileReader.....157

10.1. วัตถุ File และ วัตถุ FileList 158

10.1.ก.) จับไฟล์จาก <input> (158)

10.1.ข.) สร้างวัตถุ File (159)

10.2. จับคุณสมบัติจากไฟล์ที่เลือก 159

10.3. ตัวอย่าง 161

10.3.ก.) เลือกไฟล์ไม่เกินที่กำหนด (161)

10.3.ข.) แสดงพรีวิวของไฟล์ภาพที่เลือก (162)

10.3.ค.) แจ้งขนาดไฟล์และแสดงภาพที่เลือก (163)

10.4. Blob 166

10.4.ก.) Blob คืออะไร ? (166)

10.4.ข.) โครงสร้างของ Blob (166)

10.4.ค.) การสร้าง Blob (167)

10.4.ง.) Blob as URL (167)

10.5. Blob to base64 168

10.5.ก.) Blob to base64 (168)

10.5.ข.) Data URLs (169)

10.5.ค.) การเข้ารหัสข้อมูลเป็น Base64 (169)

10.5.ง.) การใช้ Data URLs กับแท็ก img (170)

10.6. Image to blob	171
10.7. FileReader	172
10.7.ก.) FileReader คืออะไร ? (172)	
10.7.ข.) เมธอดของ FileReader (172)	
10.8. FileReader.readAsDataURL()	176
10.9. ตัวอย่างอัปโหลดไฟล์ลงใน Drive โดยใช้ FileReader	178
10.9.ก.) ไฟล์ forms.html (178)	
10.9.ข.) ไฟล์ Code.gs (179)	
10.9.ค.) ผล (179)	
10.10. เมธอด base64Decode()	180
10.11. newBlob()	181

คำนำ

หนังสือเล่มนี้ เป็นหนึ่งในชุด **การเขียนโปรแกรม Google Apps Script** โดยในเล่มนี้ เป็นเล่มที่ต่อเนื่องจากหนังสือ **"Google Apps Script : Deploy as web app"** อธิบายการสร้าง Web App Form

ผู้เขียน เขียนหนังสือเล่มนี้ จุดประสงค์ดั้งเดิม ก็คือ **เก็บไว้อ่านเอง**

เมื่อผู้เขียนศึกษาเรื่องอะไร ก็จะไปเรียนรู้จากสื่อออนไลน์ในอินเทอร์เน็ต ทั้งคอร์สออนไลน์ วิดีโอ หรือ เอกสารทั้งในแบบฟรีและเสียเงิน

ในยุคปัจจุบันเราต้องเรียนรู้อะไรให้เร็ว โดยเฉพาะเรื่องของ IT ผู้เขียนจึงตั้งใจจะดูวิดีโอ หรือแปลเอกสารเพียงรอบเดียว จึงดูไป อ่านไป พิมพ์สรุปไป เวลาจำอะไรไม่ได้ มาดูจากที่พิมพ์สรุปไว้ง่ายกว่าการไปย้อนดูจากวิดีโอ นอกจากนี้ ก็ยังนำมาทบทวนได้ง่าย ในอนาคตสามารถเพิ่มเติมเสริมแต่งเนื้อหาได้เรื่อยๆด้วย

หนังสือเล่มนี้ ความตั้งใจดั้งเดิมของผู้เขียน ก็คือ **ตั้งใจเก็บไว้อ่านเองคนเดียว เหตุเพราะนำเนื้อหามาจากหลายแหล่ง แม้ผู้เขียนจะเขียนเพิ่มไปด้วยก็ตาม**

อย่างไรก็ดี อดสำหรับพิมพ์ไว้เป็นหนังสือแล้ว จะเก็บไว้อ่านคนเดียวก็รู้สึกเสียดาย ผู้เขียนจึงนำมาแบ่งปัน

เนื่องด้วย ผู้เขียนให้ความสำคัญกับประเด็นด้านลิขสิทธิ์มาก ฉะนั้นจึงขอแจ้งไว้ ณ ที่นี้ ตั้งแต่ต้นก็คือ

1. เนื้อหาในหนังสือ ผู้เขียนรวบรวมมาจากแหล่งต่างๆในอินเทอร์เน็ต ซึ่งจะพยายามให้มากที่สุด ที่จะบอกลิงค์หรือแหล่งที่มาในแต่ละหัวข้อ เพราะหนังสือเล่มนี้ถูกเขียนไว้นานแล้ว บางเรื่องลืมก๊อปปี้ลิงค์มาแปะไว้
2. ผู้เขียนรวบรวมเนื้อหาจากหลายแหล่ง และเพิ่มเติมลงไปด้วย
3. หนังสือเล่มนี้แจกฟรี ผู้เขียนไม่มีรายได้จากหนังสือเล่มนี้

หนังสือเล่มนี้ยังไม่จบเสียทีเดียว หากผู้เขียนว่าง จะมาเขียนเพิ่มเติมเรื่อยๆ ให้ดูเวอร์ชันตามวันที่ที่ปล่อยหนังสือ

เล่มนี้ ผู้เขียนไม่มีเวลาตรวจแก้ไข ถ้าผิดพลาดประการใดขออภัยไว้ ณ ที่นี้

วสันต์ คุณดิลกเสวต
wasankds@gmail.com
081-459-8343
Line ID : wasankds

- Part I -
HTML

บทที่ 1

HTML Forms



HTML Forms

https://www.w3schools.com/html/html_forms.asp

HTML Input form Attributes

https://www.w3schools.com/html/html_form_attributes_form.asp

1.1. แท็กหรืออิลีเมนต์ <form>

HTML <form> Tag

https://www.w3schools.com/tags/tag_form.asp

แท็กหรืออิลีเมนต์ <form> ใช้สร้างฟอร์มเพื่อเก็บการคีย์ข้อมูล(หรืออินพุต) จากยูสเซอร์

โครงสร้างการใช้งาน

```
<form>
...
Form elements เช่น Text fields, Checkboxes, Radio buttons, Submit buttons ...
...
</form>
```

อิลีเมนต์ <form> ประกอบไปด้วย Form elements หรือ องค์ประกอบภายในฟอร์ม เช่น Text fields, Checkboxes, Radio buttons, Submit buttons และ อื่นๆ

องค์ประกอบภายในฟอร์ม เช่น อิลีเมนต์ <input> เป็นอิลีเมนต์สำคัญมาก ในบรรดาองค์ประกอบภายในฟอร์มทั้งหลาย เพราะสามารถเป็นอินพุตได้หลากหลายแบบ ขึ้นอยู่กับการระบุแอตทริบิวต์ **type**

ตัวอย่าง อิลีเมนต์ <input>

Type	Description
<input type="text">	ฟิลด์ Text แบบ 1 บรรทัด
<input type="radio">	ฟิลด์ Radio button (เลือกได้ 1 จากหลายตัวเลือก)
<input type="submit">	ปุ่ม Submit สำหรับส่งฟอร์ม

ตัวอย่างฟอร์ม ที่สร้างด้วย อิลีเมนต์ <form> และ **Form elements**

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname">
</form>
```

จาก HTML ข้างต้นจะได้หน้าเว็บตามภาพดังต่อไปนี้

First name:

Last name:

1.2. แอทรินิวต์ action

แอทรินิวต์ **action** ใช้กำหนดการกระทำเมื่อส่งฟอร์ม โดยปกติก็คือ ส่งข้อมูลไปยังหน้าเพจใน Server ตามที่ระบุ

ตัวอย่าง

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

จากตัวอย่างข้างต้น ข้อมูลจะถูกส่งไปยังหน้าเพจที่ Server ก็คือ **"/action_page.php"** ซึ่งเพจนี้ ประกอบไปด้วย **Server-side script** ที่ใช้จัดการกับข้อมูลที่ส่งเข้ามา

หมายเหตุ

ถ้าส่งไปประมวลผลที่โปรเจ็ค Google Apps Script ให้ระบุเป็นลิงค์เป็น ลิงค์ที่ใช้ Deploy as web app เช่น

```
<form action = "https://script.google.com/macros/s/ABCDEFGHIJKLMNOPQRSTUVWXYZ/exec">
```

ข้างต้น มักใช้กับไฟล์ html ที่ไม่ได้รวมอยู่ในโปรเจ็ค Google Apps Script

1.3. แอทรินิวต์ target

แอทรินิวต์ **target** ใช้กำหนดว่าถ้ามีผลจากการส่งฟอร์ม จะให้เปิดที่ **New browser, A frame** หรือ ใน หน้าต่างปัจจุบัน ซึ่งค่าปริยายก็คือ **_self** (หน้าต่างปัจจุบัน) หากต้องการเปิดใน Browser ใหม่ ให้ใช้ **_blank**

ตัวอย่าง

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

1.4. แอทรินิวต์ method

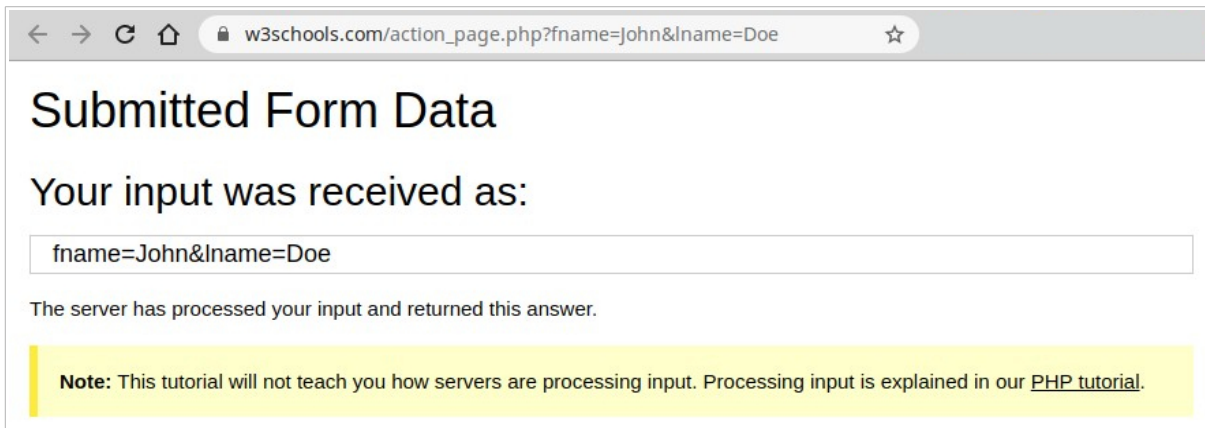
1.4.ก.) GET หรือ POST

แอทรินิวต์ **method** ใช้กำหนด HTTP Request method ว่าจะเป็น **GET** หรือ **POST** เพื่อใช้ในการส่ง ข้อมูลในฟอร์ม

ตัวอย่าง - **GET**

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

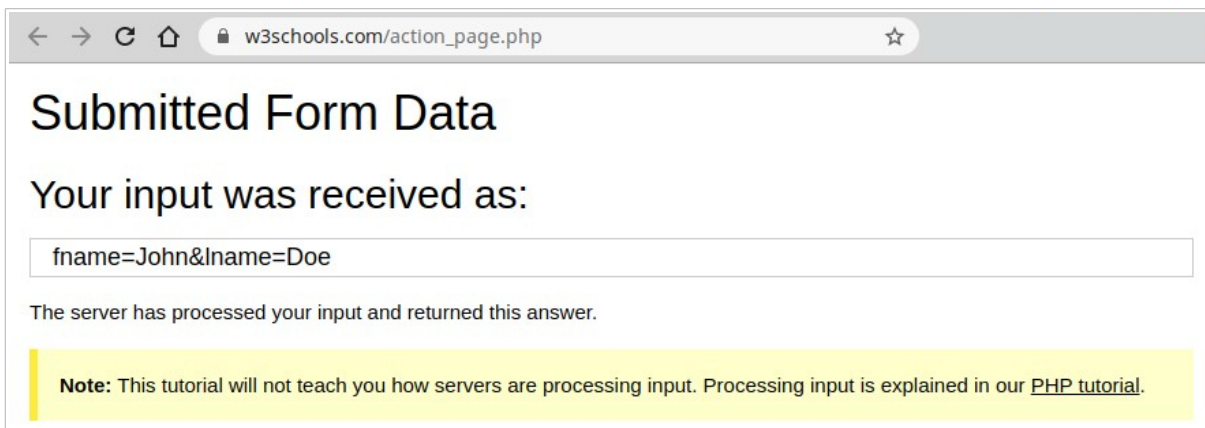
ผล - หลังส่งฟอร์มด้วย GET - สังเกตที่ URL มีการแสดงพารามิเตอร์ให้เห็นด้วย



ตัวอย่าง - POST

`<form action="/action_page.php" method="post">`

ผล - หลังส่งฟอร์มด้วย POST - สังเกตที่ URL ไม่มีการแสดงพารามิเตอร์ให้เห็น



1.4.ข.) เมื่อไรใช้ GET เมื่อไรใช้ POST

คำบรรยายในการส่งฟอร์มก็คือ GET อย่างไรก็ตาม เมื่อใช้ GET ข้อมูลจะถูกมองเห็นในช่องพิมพ์ URL ของ Browser (Page address field) เช่น

`/action_page.php?firstname=Mickey&lastname=Mouse`

หมายเหตุ : สำหรับ GET

1. แบนข้อมูลจากฟอร์มไปกับ URL ในรูปแบบคู่ของ ชื่อตัวแปร และ ค่าของตัวแปร (ตามตัวอย่างข้างต้น)
2. ความยาวของ URL จำกัดที่ 2048 ตัวอักษร
3. ห้ามใช้ GET ส่งข้อมูลที่อ่อนไหว เช่น รหัสผ่าน เพราะมองเห็นใน URL
4. ใช้ดี สำหรับการส่งฟอร์ม และยูสเซอร์ทำ Bookmark เก็บผลลัพธ์นั้นไว้
5. GET ดี สำหรับข้อมูลที่ไม่ต้องการความปลอดภัย

ให้ใช้ POST เสมอ ถ้าข้อมูลในฟอร์มนั้นเป็นข้อมูลอ่อนไหว เช่น ข้อมูลส่วนบุคคล เพราะ POST จะไม่แสดงข้อมูลที่ URL(page address field)

หมายเหตุ : สำหรับ POST

1. POST ไม่จำกัดขนาด จึงถูกใช้กับการส่งข้อมูลขนาดใหญ่
2. ฟอร์มที่ส่งด้วย POST ไม่สามารถทำ Bookmark ได้

1.5. แอธริบิวต์อื่นๆ

ตารางต่อไปนี้ เป็นแอธริบิวต์ทั้งหมดของ `<form>` ทั้งหมดที่อธิบายไปแล้ว และยังไม่ได้อธิบาย

Attribute	Description
accept-charset	Specifies the charset used in the submitted form (default: the page charset).
action	Specifies an address (url) where to submit the form (default: the submitting page).
autocomplete	Specifies if the browser should autocomplete the form (default: on).
enctype	Specifies the encoding of the submitted data (default: is url-encoded).
method	Specifies the HTTP method used when submitting the form (default: GET).
name	Specifies a name used to identify the form (for DOM usage: <code>document.forms.name</code>).
novalidate	Specifies that the browser should not validate the form.
target	Specifies the target of the address in the action attribute (default: <code>_self</code>).

1.6. การตั้งค่า CSS สำหรับ `<form>`

ตัวอย่าง

```
<head>
<style>
  <!-- เว้นระยะโดยรอบฟอร์ม -->
  form {
    display : block ;
    margin-top : 20px ;
    margin-left : 20px ;
    margin-right : 20px ;
    margin-bottom : 20px ;
  }
</style>
</head>
```

ผล - จาก CSS ข้างต้น ตามภาพดังต่อไปนี้ สังเกตว่าโซนของฟอร์มมีการเว้นระยะโดยรอบ

A form element is displayed like this:

First name:	<input type="text" value="Mickey"/>
Last name:	<input type="text" value="Mouse"/>
<input type="button" value="Submit"/>	

Change the default CSS settings to see the effect.

บทที่ 2

อิลีเมนต์

<input>



ที่มา

HTML Input Types

https://www.w3schools.com/html/html_form_input_types.asp

2.1. อิเล็มเมนต์ <input>

อิเล็มเมนต์ <input> เป็นอิเล็มเมนต์สำคัญ ในบรรดาองค์ประกอบภายในฟอร์มทั้งหลาย เพราะสามารถเป็นอินพุตได้หลากหลายแบบ ขึ้นอยู่กับการระบุแอตทริบิวต์ **type** ดังต่อไปนี้

```
<input type="button">
<input type="checkbox">
<input type="color">
<input type="date">
<input type="datetime-local">
<input type="email">
<input type="file">
<input type="hidden">
<input type="image">
<input type="month">
<input type="number">
<input type="password">
<input type="radio">
<input type="range">
<input type="reset">
<input type="search">
<input type="submit">
<input type="tel">
<input type="text">
<input type="time">
<input type="url">
<input type="week">
```

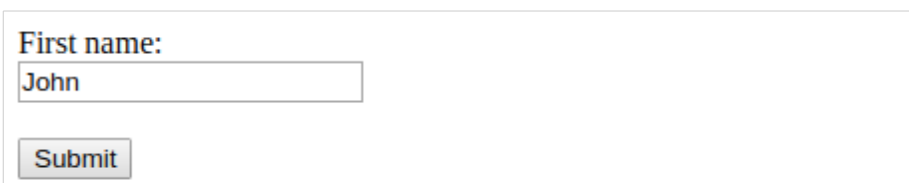
2.2. แอททริบิวต์ name ของอิเล็มเมนต์ <input>

อิเล็มเมนต์ <input> ต้องระบุแอตทริบิวต์ **name** เพื่อจะระบุอิเล็มเมนต์ที่จะส่งไป ถ้าไม่มีแอตทริบิวต์ **name** ข้อมูลในฟิลด์ตัวนั้นจะไม่ถูกส่งไป

ตัวอย่างต่อไปนี้ จะไม่มีข้อมูลจากฟิลด์ First name (**id="fname"**) ส่งไป

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

ผล - จาก HTML ข้างต้นจะได้หน้าเว็บที่มีฟอร์มดังนี้



The screenshot shows a web form with a label "First name:" followed by a text input field containing the text "John". Below the input field is a button labeled "Submit".

ผล - ไม่มีอะไรส่งไป

Your input was received as:

The server has processed your input and returned this answer.

2.3. Input type "text"

`<input type="text">` เป็นฟิลด์ Text สำหรับกรอกข้อมูล 1 บรรทัด

ตัวอย่าง

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname"><br>
</form>
```

ผล - จาก HTML ข้างต้นจะได้หน้าเว็บที่มีฟอร์มดังนี้

First name:

Last name:

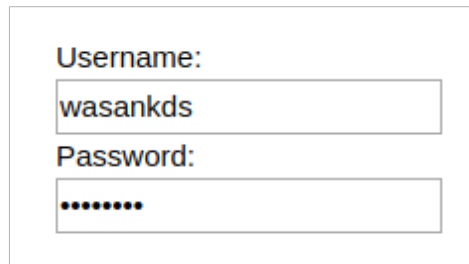
2.4. Input type "password"

`<input type="password">` เป็นฟิลด์สำหรับกรอกพาสเวิร์ด เมื่อกรอกแล้วจะไม่เห็นว่าการกรอกอะไรลงไป

ตัวอย่าง

```
<form>
  <label for="username">Username:</label><br>
  <input type="text" id="username" name="username"><br>
  <label for="pwd">Password:</label><br>
  <input type="password" id="pwd" name="pwd">
</form>
```

ผล - จาก HTML ข้างต้นจะได้หน้าเว็บที่มีฟอร์มดังนี้



A screenshot of a web form. It has two input fields. The first is labeled "Username:" and contains the text "wasankds". The second is labeled "Password:" and contains seven dots, indicating a password field.

2.5. Input type "submit"

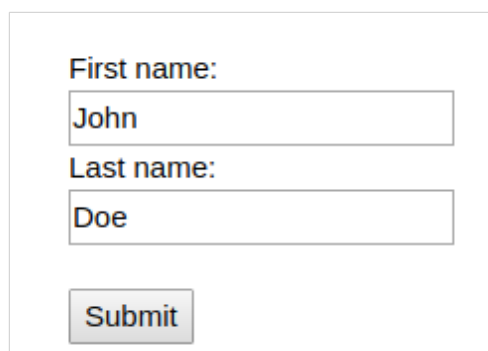
`<input type="submit">` ใช้สร้างปุ่ม สำหรับใช้ส่งข้อมูลในฟอร์มไปประมวลผลยัง **form-handler** ก็คือ ส่งไปยัง **Server page** เพื่อประมวลผลข้อมูลที่ส่งไป (ในกรณีหนังสือเล่มนี้ ก็คือ ส่งข้อมูลในฟอร์มไปประมวลผลด้วย Google Apps Script ซึ่งเป็น Server-side script)

form-handler กำหนดโดยแอททริบิวต์ **action** ของแท็ก `<form>`

ตัวอย่าง

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

ผล - จาก HTML ข้างต้นจะได้หน้าเว็บที่มีฟอร์มดังนี้



A screenshot of a web form. It has two text input fields. The first is labeled "First name:" and contains the text "John". The second is labeled "Last name:" and contains the text "Doe". Below the input fields is a button labeled "Submit".

เมื่อคลิกที่ ปุ่ม Submit จะปรากฏหน้าเพจ ซึ่งฝั่ง Server-side ได้เขียนแจ้งข้อมูลกลับมาตามภาพ

Submitted Form Data

Your input was received as:

fname=John&lname=Doe

The server has processed your input and returned this answer.

2.6. Input type "reset"

`<input type="submit">` ใช้สร้างปุ่ม Reset เพื่อรีเซ็ตค่าที่กรอกลงในฟอร์มให้เป็นค่า Defaults ที่ระบุโดยแอททริบิวต์ `value`

```
<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">
  <input type="reset">
</form>
```

ผล - จาก HTML ข้างต้นจะได้หน้าเว็บที่มีฟอร์มดังนี้ โดยเมื่อกรอกค่าที่ช่อง First name และ Last name จากนั้นคลิกที่ปุ่ม Reset ค่าในฟิลด์ต่างๆจะรีเซ็ตกลับมาเป็นค่า Defaults

First name:

Last name:

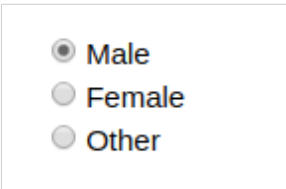
2.7. Input type "radio"

`<input type="radio">` ใช้สร้าง Radio button ยูสเซอร์สามารถเลือกได้เพียง 1 จากหลายตัวเลือก

ตัวอย่าง – สังเกตว่าใช้แอตทริบิวต์ `name` ชื่อเดียวกัน ก็คือ เป็นกลุ่มเดียวกัน (แต่ `value` ต่างกัน)

```
<form>
  <input type="radio" id="male" name="gender" value="male">
  <label for="male">Male</label><br>
  <input type="radio" id="female" name="gender" value="female">
  <label for="female">Female</label><br>
  <input type="radio" id="other" name="gender" value="other">
  <label for="other">Other</label>
</form>
```

ผล – จาก HTML ข้างต้นจะได้หน้าเว็บที่มีฟอร์มดังนี้



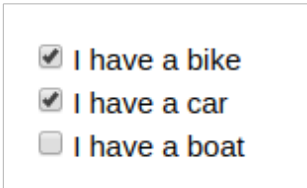
2.8. Input type "checkbox"

`<input type="checkbox">` ใช้สร้าง Checkbox ยูสเซอร์สามารถเลือกได้หลายตัวเลือก จากตัวเลือกทั้งหมด

ตัวอย่าง

```
<form>
  <input type="checkbox" id="vehicle1" name="vehicle1" value="Bike">
  <label for="vehicle1"> I have a bike</label><br>
  <input type="checkbox" id="vehicle2" name="vehicle2" value="Car">
  <label for="vehicle2"> I have a car</label><br>
  <input type="checkbox" id="vehicle3" name="vehicle3" value="Boat">
  <label for="vehicle3"> I have a boat</label>
</form>
```

ผล – จาก HTML ข้างต้นจะได้หน้าเว็บที่มีฟอร์มดังนี้



หมายเหตุ :

แอตทริบิวต์ `for` ของ `<label>` ใช้กำหนด เมื่อคลิกที่ Label เท่ากับคลิกที่ ตัวเลือกที่ระบุใน `for`

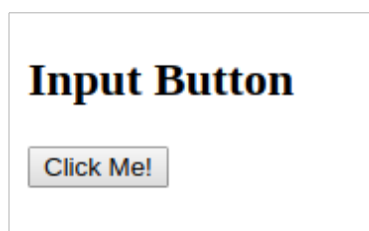
2.9. Input type "button"

`<input type="button">` ใช้สร้างปุ่มเพื่อรันคำสั่งหรือฟังก์ชัน ตามที่ระบุในแอตทริบิวต์ `onclick`

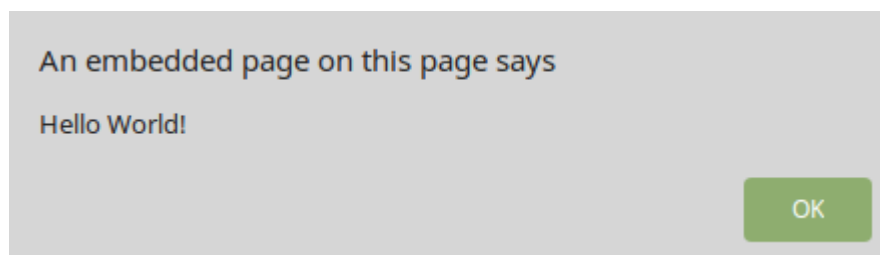
ตัวอย่าง

```
<html>
  <body>
    <h2>Input Button</h2>
    <input type="button" onclick="alert('Hello World!')" value="Click Me!">
  </body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้หน้าเว็บที่มีฟอร์มดังนี้



เมื่อคลิกที่ ปุ่ม "Click Me" จะปรากฏกรอบแจ้งเตือนที่ Browser ตามภาพ



2.10. Input type "color"

`<input type="color">` ใช้สร้างตัวเลือกสี(Color picker) ลักษณะหน้าต่างขึ้นอยู่กับ Browser

ตัวอย่าง

```
<html>
  <body>
    <form action="/action_page.php">
      <label for="favcolor">Select your favorite color:</label>
      <input type="color" id="favcolor" name="favcolor" value="#ff0000">
      <input type="submit" value="Submit">
    </form>
  </body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้หน้าเว็บที่มีฟอร์มดังนี้

Select your favorite color:

เมื่อเลือกสีแล้วคลิกที่ ปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้

Submitted Form Data

Your input was received as:

favcolor=#cd0bc6

The server has processed your input and returned this answer.

2.1.1. Input type "date"

`<input type="date">` ใช้สร้างฟิลด์ที่รับข้อมูลวันที่ และมีตัวเลือกวันที่ (Date picker) โดยลักษณะหน้าตาขึ้นอยู่กับ Browser

ตัวอย่าง

```
<html>
  <body>
    <form action="/action_page.php">
      <label for="birthday">Birthday:</label>
      <input type="date" id="birthday" name="birthday">
      <input type="submit" value="Submit">
    </form>
  </body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มที่มี Date picker ตามภาพ

Birthday:

April 2020

Sun	Mon	Tue	Wed	Thu	Fri	Sat
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2

เมื่อเลือกสีแล้วคลิกที่ปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้

Submitted Form Data

Your input was received as:

birthday=1977-10-13

The server has processed your input and returned this answer.

2.12. Input type "datetime-local"

`<input type="datetime-local">` ใช้สร้างฟิลด์ที่รับข้อมูลวันที่และเวลา แบบไม่มี Time zone และมีตัวเลือกวันที่ (Date time picker) ด้วยลักษณะหน้าตาขึ้นอยู่กับ Browser

ตัวอย่าง

```
<html>
<body>
  <form action="/action_page.php">
    <label for="birthdaytime">Birthday (date and time):</label>
    <input type="datetime-local" id="birthdaytime" name="birthdaytime">
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มที่มี Date time picker ตามภาพ

Birthday (date and time): 10/13/1977, 04:00 AM x ▼ Submit

October 1977 ▼

Sun	Mon	Tue	Wed	Thu	Fri	Sat
25	26	27	28	29	30	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

เมื่อเลือกสีแล้วคลิกที่ ปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้

Submitted Form Data

Your input was received as:

birthdaytime=1977-10-13T04:00

The server has processed your input and returned this answer.

2.13. Input type "email"


`<input type="email">` ใช้สร้างฟิลด์ที่รับข้อมูลเป็นอีเมล ซึ่งจะถูก Validate ตามรูปแบบของอีเมล แต่ก็ขึ้นอยู่กับ Browser

ตัวอย่าง

```
<html>
  <body>
    <form action="/action_page.php">
      <label for="email">Birthday (date and time):</label>
      <input type="email" id="email" name="email">
      <input type="submit" value="Submit">
    </form>
  </body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ เมื่อทดสอบกรอกอีเมลผิด แล้วคลิกที่ ปุ่ม Submit จะปรากฏรอบแจ้งเตือนตามภาพ

Enter your email:

 Please enter a part following '@'. 'abc@' is incomplete.

แต่ถ้ากรอกอีเมลถูกต้อง แล้วคลิกที่ ปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้

Submitted Form Data

Your input was received as:

email=wasankds@gmail.com

The server has processed your input and returned this answer.

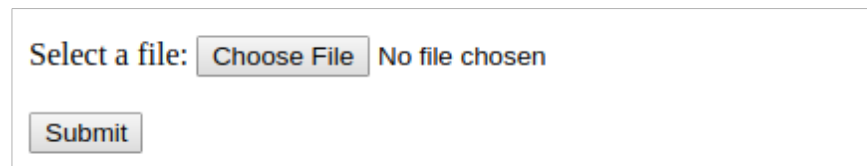
2.14. Input type "file"

`<input type="file">` ใช้สร้างฟิลด์สำหรับสร้างปุ่มเลือกไฟล์ โดยจะเลือกได้เพียงไฟล์เดียว

ตัวอย่าง

```
<html>
<body>
  <form action="/action_page.php">
    <label for="myfile">Select a file:</label>
    <input type="file" id="myfile" name="myfile"><br><br>
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ



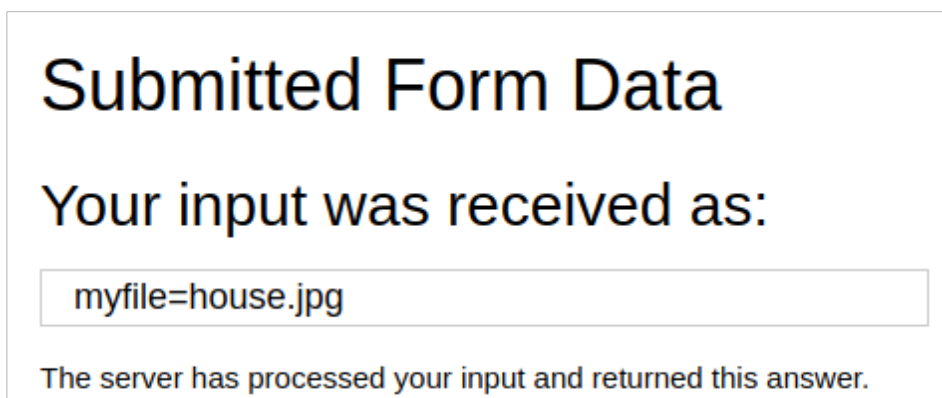
The screenshot shows a web form with the text "Select a file:" followed by a "Choose File" button and the text "No file chosen". Below this is a "Submit" button.

หลังเลือกไฟล์แล้ว จะปรากฏชื่อไฟล์ที่เลือกตามภาพ



The screenshot shows the same web form, but now the "Choose File" button is followed by the text "house.jpg". The "Submit" button remains below.

เมื่อเลือกไฟล์แล้วคลิกที่ ปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้



The screenshot shows a page titled "Submitted Form Data" with the text "Your input was received as:". Below this is a text box containing "myfile=house.jpg". At the bottom, it says "The server has processed your input and returned this answer."

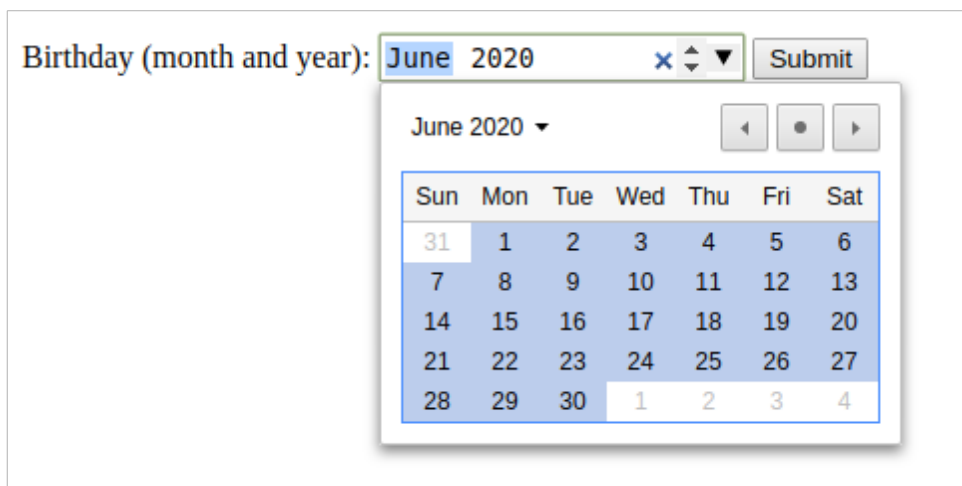
2.15. Input type "month"

`<input type="month">` ใช้สร้างฟิลด์สำหรับเลือกเดือนและปี และมีตัวเลือกว่าวันที่ (Date picker)
ด้วยลักษณะหน้าต่างขึ้นอยู่กับ Browser

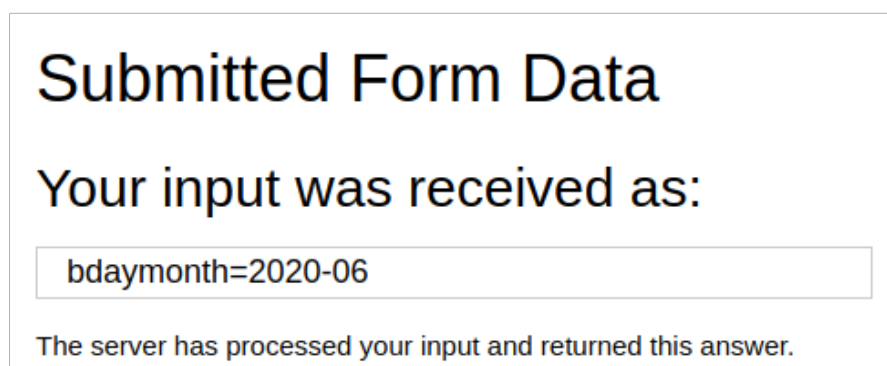
ตัวอย่าง

```
<html>
  <body>
    <form action="/action_page.php">
      <label for="bdaymonth">Birthday (month and year):</label>
      <input type="month" id="bdaymonth" name="bdaymonth">
      <input type="submit" value="Submit">
    </form>
  </body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ



หลังเลือกเดือนและปีแล้ว จากนั้นคลิกปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้



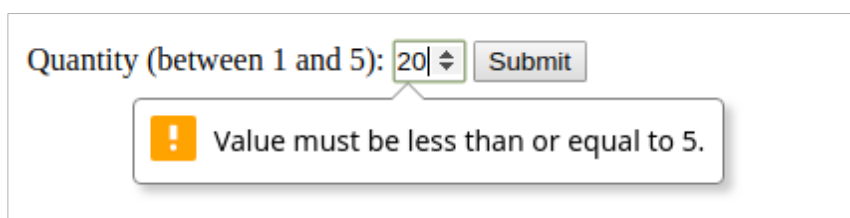
2.16. Input type "number"

`<input type="number">` ใช้สร้างฟิลด์ที่ใช้สำหรับกรอกข้อมูลเป็นตัวเลข โดยเราสามารถจำกัดตัวเลขที่รับได้โดยแอตทริบิวต์ `min` และ `max`

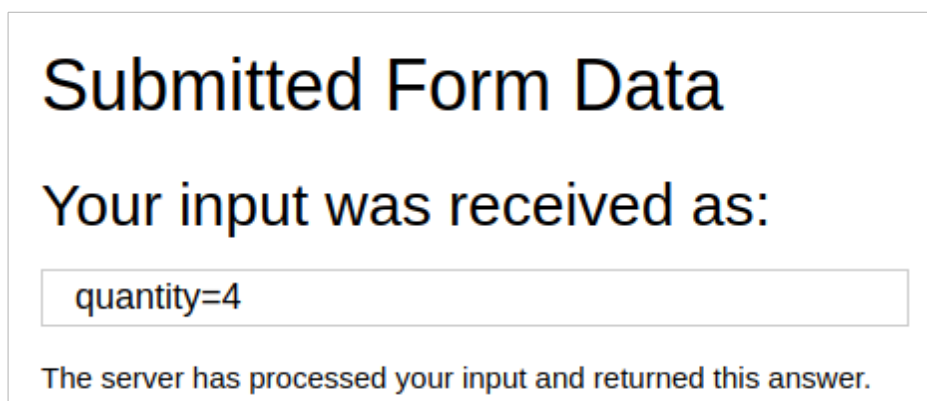
ตัวอย่าง

```
<html>
<body>
  <form action="/action_page.php">
    <label for="quantity">Quantity (between 1 and 5):</label>
    <input type="number" id="quantity" name="quantity" min="1" max="5">
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ เมื่อกรอกตัวเลขนอกเหนือจากที่กำหนดไว้ จากนั้นคลิกปุ่ม `Submit` จะปรากฏกรอบแจ้งเตือน

A screenshot of a web form. The label "Quantity (between 1 and 5):" is followed by a number input field containing the value "20". To the right of the input field is a "Submit" button. Below the input field, a red-bordered error message box is displayed, containing a red exclamation mark icon and the text "Value must be less than or equal to 5."

เมื่อกรอกตัวเลข ถูกจากนั้นคลิกปุ่ม `Submit` จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้

A screenshot of a web page titled "Submitted Form Data". Below the title, it says "Your input was received as:". Underneath this text is a text box containing the string "quantity=4". At the bottom of the page, it says "The server has processed your input and returned this answer."

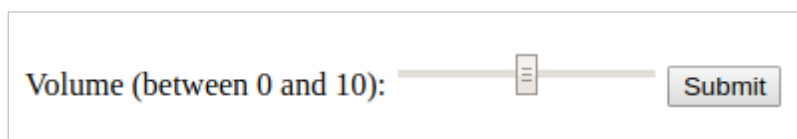
2.17. Input type "Range"

`<input type="range">` ใช้สร้างฟิลด์สไลด์เดอร์ สำหรับเลือกตัวเลข โดยค่าปรัยายก็คือ 0-100 แต่ก็สามารถกำหนดได้โดยแอตทริบิวต์ `min`, `max` และ `step`

ตัวอย่าง

```
<html>
<body>
  <form action="/action_page.php">
    <label for="vol">Volume (between 0 and 10):</label>
    <input type="range" id="vol" name="vol" min="0" max="10" step="0.1">
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ

A screenshot of a web form. It features a label "Volume (between 0 and 10):" followed by a horizontal range slider. The slider has a small vertical bar in the middle, indicating a value of approximately 3.5. To the right of the slider is a button labeled "Submit".

เมื่อเลื่อนสไลด์เตอร์ และคลิก ปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้

Submitted Form Data

Your input was received as:

vol=3.5

The server has processed your input and returned this answer.

2.18. Input type "search"

`<input type="search">` ใช้สร้างฟิลด์ค้นหา (มีการใช้งานเหมือนฟิลด์ Text)

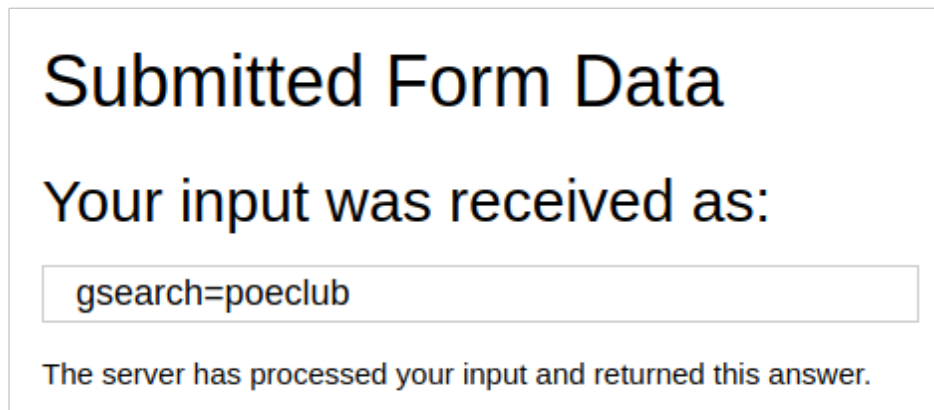
ตัวอย่าง

```
<html>
<body>
  <form action="/action_page.php">
    <label for="gsearch">Search Google:</label>
    <input type="search" id="gsearch" name="gsearch">
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ



เมื่อกรอกข้อมูลลงในฟิลด์ Search และคลิกปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้



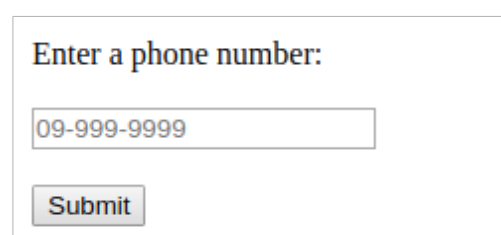
2.19. Input type "tel"

`<input type="tel">` ใช้สร้างฟิลด์สำหรับกรอกข้อมูลเป็นเบอร์โทรศัพท์ แต่จริงๆแล้วเป็นฟิลด์ Text ธรรมดา นี่เอง เพราะตัวที่ทำหน้าที่ Validate เบอร์โทรฯ ก็คือ แอธริบิวต์ `pattern` ที่ใช้ `Regular Expression` เป็นตัวกำหนดรูปแบบของเบอร์โทรฯ ส่วนที่เห็นเป็นตัวเลขสี่เท่าใน ฟิลด์ Tel "09-9999-9999" ตามตัวอย่าง ก็คือ ผลจากแอธริบิวต์ `placeholder`(จองพื้นที่ด้วยค่าที่ระบุ)

ตัวอย่าง

```
<html>
<body>
  <form action="/action_page.php">
    <label for="phone">Enter a phone number:</label><br><br>
    <input type="tel" id="phone" name="phone"
      placeholder="09-9999-9999"
      pattern="[0]{2}-[0-9]{4}-[0-9]{4}" required><br><br>
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ



เมื่อกรอกข้อมูลลงในฟิลด์ Tel และคลิกปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้

Submitted Form Data

Your input was received as:

phone=08-1459-8343

The server has processed your input and returned this answer.

2.20. Input type "time"

`<input type="time">` ใช้สร้างฟิลด์สำหรับกรอกข้อมูลเป็นเวลา แบบไม่มี Time zone

ตัวอย่าง

```
<html>
<body>
  <form action="/action_page.php">
    <label for="appt">Select a time:</label>
    <input type="time" id="appt" name="appt">
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ

Select a time:

เมื่อกรอกข้อมูลลงในฟิลด์ Time และคลิกปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้

Submitted Form Data

Your input was received as:

appt=05:50

The server has processed your input and returned this answer.

2.21. Input type "week"

`<input type="week">` ใช้สร้างฟิลด์สำหรับกรอกข้อมูลเป็นสัปดาห์และปี

ตัวอย่าง

```
<html>
<body>
  <form action="/action_page.php">
    <label for="week">Select a week:</label>
    <input type="week" id="week" name="week">
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ

Week	Sun	Mon	Tue	Wed	Thu	Fri	Sat
14	29	30	31	1	2	3	4
15	5	6	7	8	9	10	11
16	12	13	14	15	16	17	18
17	19	20	21	22	23	24	25
18	26	27	28	29	30	1	2

เมื่อเลือกข้อมูลในฟิลด์ `week` และคลิก ปุ่ม `Submit` จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้

Submitted Form Data

Your input was received as:

week=2020-W16

The server has processed your input and returned this answer.

บทที่ 3

HTML

Form Elements



HTML Form Elements

https://www.w3schools.com/html/html_form_elements.asp

3.1. อิเล็มเมนต์นอกเหนือจาก <input>

Form Elements หรือ องค์ประกอบในฟอร์ม นอกเหนือจากอิเล็มเมนต์ <input> ตามที่อธิบายในบทที่แล้ว ยังมีอิเล็มเมนต์อื่นๆอีก

3.2. อิเล็มเมนต์ <select>

HTML <select> Tag

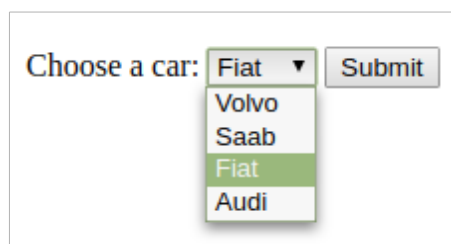
https://www.w3schools.com/tags/tag_select.asp

อิเล็มเมนต์ <select> ใช้สร้าง Drop-down

อิเล็มเมนต์ <option> ใช้สร้างตัวเลือกใน Drop-down ซึ่งค่าปรียาย ตัวเลือกแรกจะถูกเลือก หากต้องการกำหนดเป็นตัวเลือกอื่น(Pre-selected) ให้เพิ่มแอททริบิวต์ **selected** ให้กับ <option> ที่ต้องการเลือกไว้ก่อนตามต้องการ

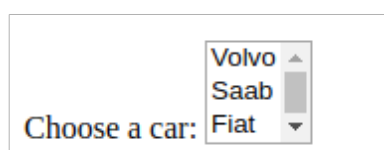
```
<form action="/action_page.php">
  <label for="cars">Choose a car:</label>
  <select id="cars" name="cars">
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
    <option value="fiat" selected>Fiat</option> <!-- เลือกไว้ก่อน -->
    <option value="audi">Audi</option>
  </select>
  <input type="submit">
</form>
```

ผล



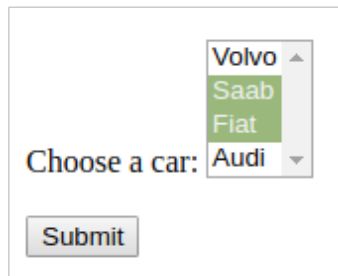
แอททริบิวต์ **size** ใช้กำหนดตัวเลือกที่จะแสดง แต่จาก Drop-down จะกลายเป็นคล้ายๆ Text area ที่มีสกอัลบาร์แทน ตามภาพ

```
<!-- size=3 กำหนดให้แสดง 3 รายการ -->
<select id="cars" name="cars" size="3">
```

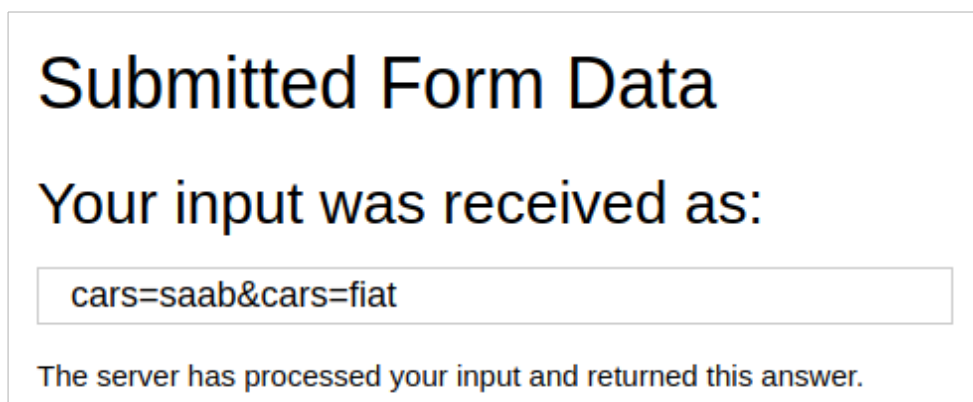


แอททริบิวต์ **multiple** อนุญาตให้เลือกได้มากกว่า 1 ตัวเลือก

```
<!-- ใช้ multiple-->  
<select id="cars" name="cars" size="3" multiple>
```



เมื่อเลือกตัวเลือกใน Drop-down และคลิก ปุ่ม **Submit** จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้



3.3. อีเล็มเมนต์ <textarea>

HTML <textarea> Tag

https://www.w3schools.com/tags/tag_textarea.asp

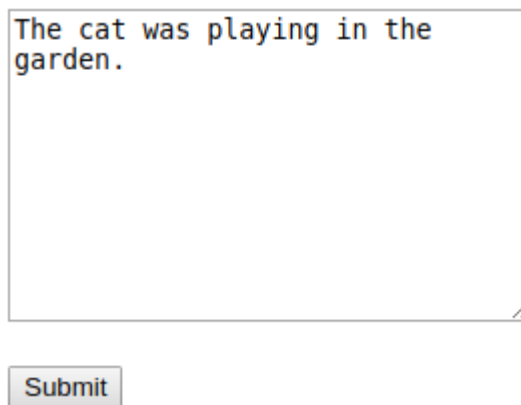
อีเล็มเมนต์ <select> ใช้สร้างกรอบพิมพ์ข้อความแบบหลายบรรทัด(Text area)

แอททริบิวต์ **row** และ **coloum** ใช้กำหนดจำนวนบรรทัด และความกว้างของ Text area ตามลำดับ

ตัวอย่าง

```
<form action="/action_page.php">  
  <textarea name="message" rows="10" cols="30">  
    The cat was playing in the garden.</textarea><br><br>  
  <input type="submit">  
</form>
```

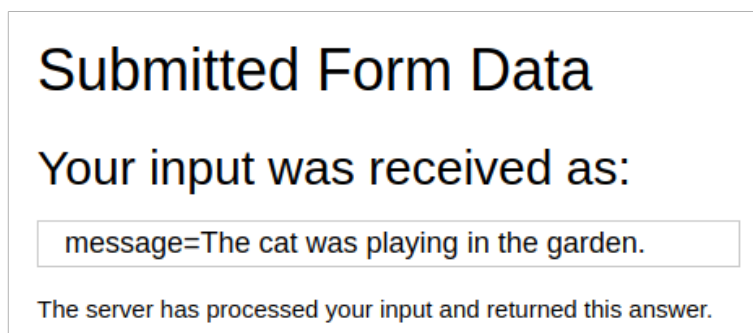
ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ



The cat was playing in the garden.

Submit

เมื่อพิมพ์ข้อความลงใน Text area และคลิก ปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้



Submitted Form Data

Your input was received as:

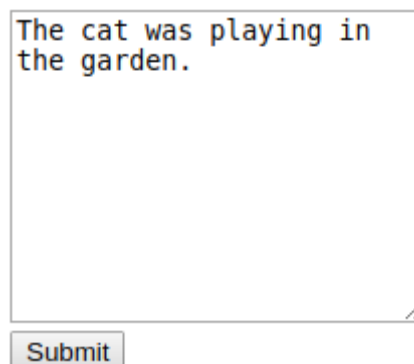
message=The cat was playing in the garden.

The server has processed your input and returned this answer.

เราสามารถตั้งค่า Text area โดยใช้ CSS ได้ด้วยดังนี้

```
<form action="/action_page.php">  
  <textarea name="message" style="width:200px; height:150px;">  
    The cat was playing in the garden.</textarea><br><br>  
  <input type="submit">  
</form>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ



The cat was playing in the garden.

Submit

3.4. อิลีเมนต์ <button>

HTML <button> Tag

https://www.w3schools.com/tags/tag_button.asp

อิลีเมนต์ <button> ใช้สร้างปุ่มเพื่อรันคำสั่งหรือฟังก์ชัน ตามที่ระบุในแอททริบิวต์ [onclick](#) การใช้งานอิลีเมนต์นี้ ไม่ต้องมี อิลีเมนต์ <form> ครอบ

(อิลีเมนต์ตัวนี้ ใช้งานเหมือนกับ อิลีเมนต์ <input> ที่มีแอททริบิวต์เป็น [button](#) ดูเพิ่มเติม ข้อ 2.9 Input type "button" หน้า 27)

ตัวอย่าง

```
<button type="button" onclick="alert('Hello World!')">Click Me!</button>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ

The button Element

Click Me!

เมื่อคลิกที่ ปุ่ม "Click Me!" จะปรากฏกรอบแจ้งข้อความ ตามภาพ

An embedded page on this page says

Hello World!

OK

แอททริบิวต์

Attribute	Value	Description
autofocus	autofocus	Specifies that a button should automatically get focus when the page loads
disabled	disabled	Specifies that a button should be disabled
form	form_id	Specifies which form the button belongs to
formaction	URL	Specifies where to send the form-data when a form is submitted. Only for type="submit"
formenctype	application/x-www-form-urlencoded multipart/form-data text/plain	Specifies how form-data should be encoded before sending it to a server. Only for type="submit"
formmethod	get post	Specifies how to send the form-data (which HTTP method to use). Only for type="submit"
formnovalidate	formnovalidate	Specifies that the form-data should not be validated on submission. Only for type="submit"

Attribute	Value	Description
formtarget	_blank _self _parent _top framename	Specifies where to display the response after submitting the form. Only for type="submit"
name	name	Specifies a name for the button
type	button reset submit	Specifies the type of button
value	text	Specifies an initial value for the button

3.5. อีเล็มเมนต์ <fieldset> และ <legend>

HTML <fieldset> Tag

https://www.w3schools.com/tags/tag_fieldset.asp

อีเล็มเมนต์ <fieldset> ใช้รวมกลุ่มฟิลด์ในฟอร์ม อีเล็มเมนต์ <legend> ใช้ทำแคปชั่นให้กับ

<fieldset>

ตัวอย่าง

```
<form action="/action_page.php">
  <fieldset>
    <legend>Personalia:</legend>
    <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">
  </fieldset>
</form>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ

3.6. อีเล็มเมนต์ <datalist> และ <input>

HTML <datalist> Tag

https://www.w3schools.com/tags/tag_datalist.asp

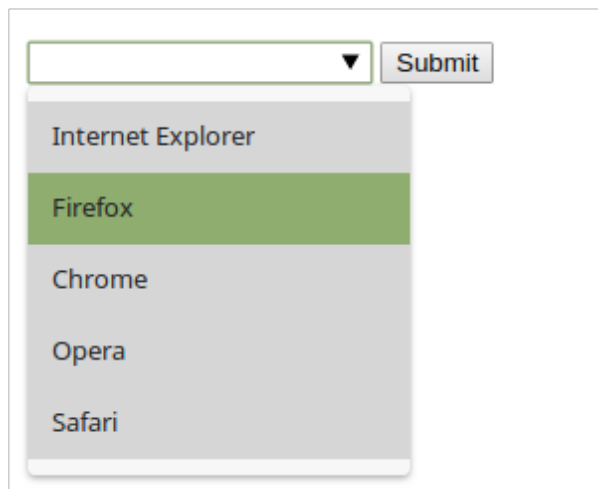
อีเล็มเมนต์ <datalist> ใช้สร้างตัวเลือกให้กับอีเล็มเมนต์ <input> ที่ใช้แอททริบิวต์ `list` ซึ่งจะได้ฟิลด์ที่เป็น Drop-down

การใช้งาน ค่าของแอททริบิวต์ `list` จะต้องอ้างอิงถึง id ของอีเล็มเมนต์ <datalist>

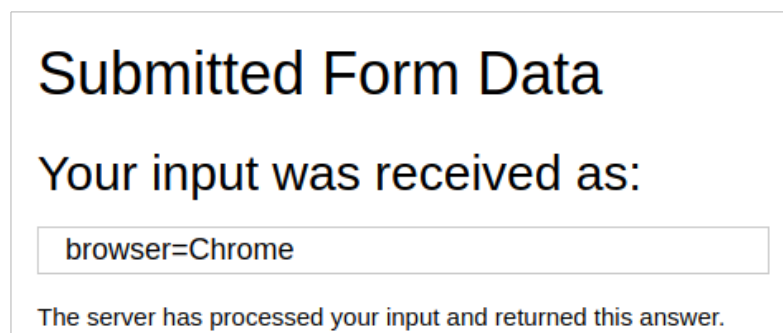
ตัวอย่าง

```
<form action="/action_page.php">
  <input list="browsers" name="browser">
  <datalist id="browsers">
    <option value="Internet Explorer">
    <option value="Firefox">
    <option value="Chrome">
    <option value="Opera">
    <option value="Safari">
  </datalist>
</form>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ

A screenshot of a web form. It features a dropdown menu with a downward arrow on the right. Below the dropdown, a list of browser names is displayed: Internet Explorer, Firefox (highlighted in green), Chrome, Opera, and Safari. To the right of the dropdown is a button labeled "Submit".

เมื่อเลือกตัวเลือกใน Drop-down และคลิก ปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้

A screenshot of a web page titled "Submitted Form Data". Below the title, it says "Your input was received as:". There is a text box containing the text "browser=Chrome". Below this, it says "The server has processed your input and returned this answer."

3.7. อีเล็มเมนต์ <output>

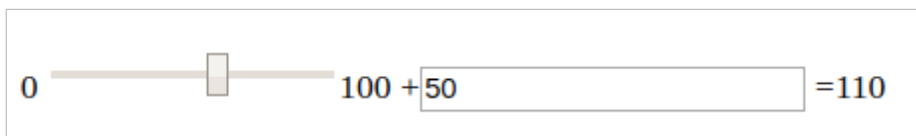
HTML <output> Tag

https://www.w3schools.com/tags/tag_output.asp

อีเล็มเมนต์ <output> ใช้แสดงผลจากการคำนวณ คล้ายกับการทำงานด้วย Script

```
<form action="/action_page.php"
      oninput="x.value=parseInt(a.value)+parseInt(b.value)">
  0
  <input type="range" id="a" name="a" value="50">      <!-- Slider -->
  100 +
  <input type="number" id="b" name="b" value="50">      <!-- ฟิลด์ Number input -->
  =
  <output name="x" for="a b"></output> <br><br>      <!-- แสดงผลการคำนวณ -->
  <input type="submit">
</form>
```

ผล - จาก HTML ข้างต้น จะได้ฟอร์มตามภาพ โดยเมื่อจับ Slider เลื่อน ผลการคำนวณจะปรากฏที่หลังเครื่องหมาย "="



0 ————— 100 + 50 = 110

บทที่ 4

แอตทริบิวต์ของ อีเล็มเมนต์ `<input>` ที่สำคัญ



HTML Input Attributes

https://www.w3schools.com/html/html_form_attributes.asp

4.1. แอทธิบิวต์ (Attributes)

HTML Attributes

https://www.w3schools.com/html/html_attributes.asp

แอทธิบิวต์ (Attributes) ใส่ที่แท็กเปิด เพื่อกำหนดคุณลักษณะให้กับอีเล็มเมนต์ HTML

ตัวอย่างที่ 1

```
<h1 lang="en">Heading 1</h1>
```

lang ก็คือ แอทธิบิวต์

= **"en"** ก็คือ ค่า(Value) ของแอทธิบิวต์

ตัวอย่างที่ 2

```
<h1 lang="en" id="title">Heading 1</h1>
```

<h1> มี 2 แอทธิบิวต์ ตัวแรก ก็คือ **lang** ตัวที่สองคือ **id**

แอทธิบิวต์ ใส่ลงในแท็กใดก็ได้ ตัวอย่างดังต่อไปนี้

```
<html>
  <head> </head>
  <body>
    <h1 lang="en" id="title">Heading 1 </h1>
    <p lang="en">Paragraph 1</p>
    <p>Paragraph 2 <br lang="en">new line</p>
  </body>
</html>
```

ตารางต่อไปนี้ เป็นแอทธิบิวต์ที่ใช้บ่อย

Attribute	Description
alt	Specifies an alternative text for an image, when the image cannot be displayed
disabled	Specifies that an input element should be disabled
href	Specifies the URL (web address) for a link
id	Specifies a unique id for an element
src	Specifies the URL (web address) for an image
style	Specifies an inline CSS style for an element
title	Specifies extra information about an element (displayed as a tool tip)

4.2. value

HTML <input> value Attribute

https://www.w3schools.com/tags/att_input_value.asp

แอททริบิวต์ **value** ใช้กำหนดค่าให้กับอิลีเมนต์ <input> อย่างไรก็ดี ใช้แตกต่างกันสำหรับอิลีเมนต์ input ที่ต่างกัน

สำหรับ button, reset และ submit ใช้กำหนดข้อความบนปุ่ม

สำหรับ text, password และ hidden ใช้กำหนดค่าเริ่มต้นของฟิลด์ข้อความบนปุ่ม

สำหรับ checkbox, radio และ image ใช้กำหนดค่าที่สอดคล้องกับอินพุต ค่านี้จะถูกส่งเมื่อ Submit ด้วย

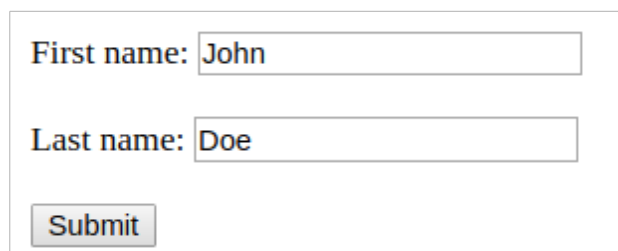
โครงสร้างการใช้งาน

```
<input value="text">
```

ตัวอย่าง

```
<form>
  <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">
  <input type="submit" value="Submit">
</form>
```

ผล



4.3. readonly

HTML <input> readonly Attribute

https://www.w3schools.com/tags/att_input_readonly.asp

แอททริบิวต์ **readonly** เป็นบูลีนที่ใช้กำหนดฟิลด์ตัวนั้นๆว่า จะให้อ่านได้อย่างเดียว (read-only) หรือไม่ ก็คือ ไม่สามารถแก้ไขได้ (แต่สามารถเลือกได้ ก็อปปีได้)

ฟิลด์ที่ **readonly** สามารถส่งค่าไปกับฟอร์มได้เมื่อ Submit แต่ฟิลด์ที่ใช้แอททริบิวต์ disabled จะไม่ส่งค่าไป

หากต้องการเอา **readonly** ออก ระหว่างการใช้งานฟอร์ม สามารถเขียน javascript เอาออกได้

```
<input readonly>
```

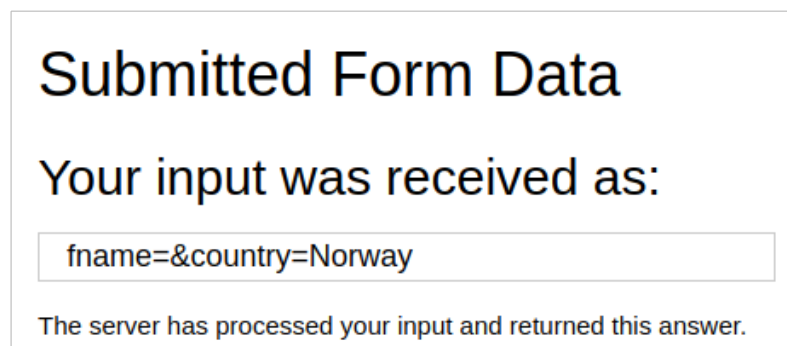
ตัวอย่าง

```
<form action="/action_page.php">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="country">Country:</label>
  <input type="text" id="country" name="country" value="Norway" readonly><br><br>
  <input type="submit" value="Submit">
</form>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ โดยฟิลด์ Country ไม่สามารถแก้ไขข้อมูลได้

A screenshot of a web form. It has two text input fields. The first is labeled 'First name:' and is empty. The second is labeled 'Country:' and contains the text 'Norway'. Below the fields is a button labeled 'Submit'.

เมื่อคลิกที่ ปุ่ม Submit จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้ สังเกตว่าฟิลด์ Country ข้อมูลถูกส่งไปด้วย

A screenshot of a web page showing the result of a form submission. The page has a heading 'Submitted Form Data' and a subheading 'Your input was received as:'. Below this is a text box containing 'fname=&country=Norway'. At the bottom, it says 'The server has processed your input and returned this answer.'

4.4. disabled

HTML <input> disabled Attribute

https://www.w3schools.com/tags/att_input_disabled.asp

แอททริบิวต์ **disabled** เป็นบูลีนที่ใช้กำหนดฟิลด์ตัวนั้นๆจะให้ทำงานหรือไม่(Disable) ถ้ากำหนดเป็น **disabled** ฟิลด์จะใช้งานไม่ได้ เลือกไม่ได้ ก๊อปปี้ไม่ได้ และไม่ส่งข้อมูลไปกับฟอร์ม มักเอาไว้แสดงผลอย่างเดียว

หากต้องการเอา **disabled** ออก ระหว่างการใช้งานฟอร์ม สามารถเขียน Javascript เอาออกได้


โครงสร้างการใช้งาน

```
<input disabled>
```

ตัวอย่าง

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

ผล - จาก HTML ข้างต้น จะได้ฟอร์มตามภาพ โดยฟิลด์ Last name ไม่สามารถแก้ไขข้อมูลได้



เมื่อคลิกที่ ปุ่ม **Submit** จะปรากฏผลลัพธ์ที่ประมวลผลมาจากฝั่ง Server-side ดังต่อไปนี้ สังเกตว่าฟิลด์ Last name ข้อมูลไม่ถูกส่งไปด้วย

Submitted Form Data

Your input was received as:

```
fname=Wasan
```

The server has processed your input and returned this answer.

4.5. accept

HTML <input> accept Attribute

https://www.w3schools.com/tags/att_input_accept.asp

แอททริบิวต์ accept ใช้กำหนดว่าไฟล์ชนิดใด ที่สามารถเลือกได้จาก หน้าต่างเลือกไฟล์

โครงสร้างการใช้งาน

```
<input accept_="file_extensionaudio/* | video/* | image/* | media_type">
```

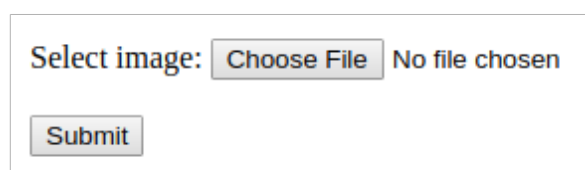
ค่าของแอททริบิวต์ accept

Value	Description
file_extension	นามสกุลของไฟล์ เช่น .gif, .jpg, .png, .doc ที่สามารถเลือกได้
audio/*	เลือกไฟล์เสียงได้
video/*	เลือกไฟล์วิดีโอได้
image/*	เลือกไฟล์ภาพได้
media_type	ชนิดของไฟล์ ดูเพิ่มเติม(IANA Media Types)

ตัวอย่าง

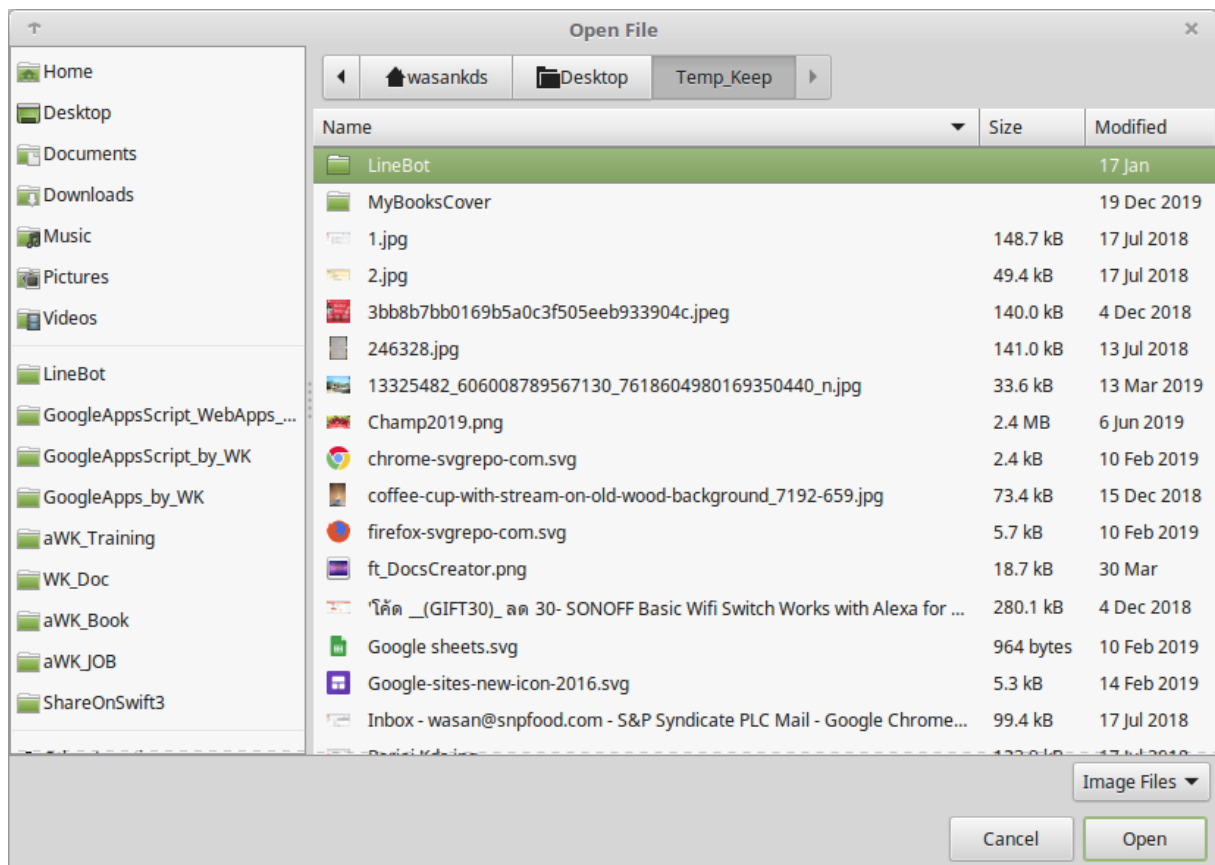
```
<form action="/action_page.php">
  <label for="img">Select image:</label>
  <input type="file" id="img" name="img" accept="image/*">
  <input type="submit">
</form>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ



เมื่อคลิกที่ ปุ่ม "Choose File" จะปรากฏหน้าต่าง Open File ตามภาพถัดไป โดยไฟล์ที่ปรากฏให้เราเลือก มีแต่ไฟล์ภาพเท่านั้น

อย่างไรก็ดี เราสามารถเลือกไฟล์ที่ไม่ใช่ภาพได้ โดยเปลี่ยนตัวกรองไฟล์(ด้านล่างสุด) ฉะนั้นจึงต้องเขียนโปรแกรมเพื่อตรวจสอบอีกที



4.6. pattern และ title

HTML <input> pattern Attribute

https://www.w3schools.com/tags/att_input_pattern.asp

JavaScript Regular Expressions

https://www.w3schools.com/js/js_regexp.asp

HTML title Attribute

https://www.w3schools.com/tags/att_global_title.asp

แอททริบิวต์ **pattern** ใช้ Regular Expression ตรวจสอบค่าในฟิลด์ **pattern** ใช้ได้กับอิลีเมนต์ **<input>** ชนิดต่อไปนี้ text, date, search, url, tel, email และ password

แนะนำให้ใช้แอททริบิวต์ **title** เพื่อช่วยในการใช้งานแอททริบิวต์ **pattern**

โครงสร้างการใช้งาน

```
<input pattern="(regex)">
```

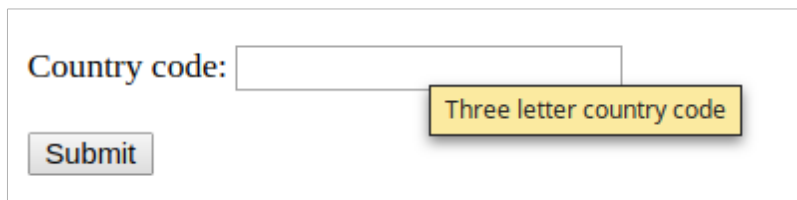
แอททริบิวต์ **title** ใช้ใส่ข้อมูลพิเศษให้กับอิลีเมนต์ ในลักษณะ Tooltip เมื่อชี้เมาส์ที่อิลีเมนต์ จะปรากฏ Tooltip แสดงข้อความตามค่าที่ระบุให้กับแอททริบิวต์ **title**

```
<element title="text">
```

ตัวอย่าง

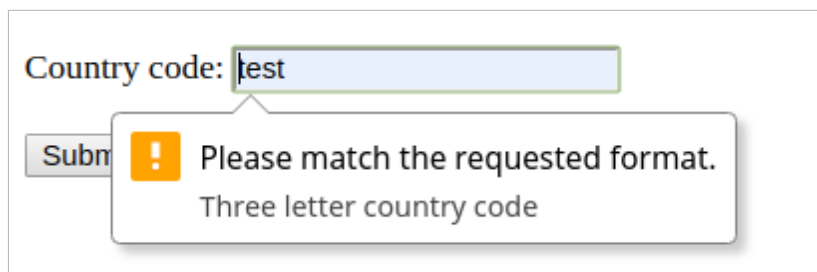
```
<form action="/action_page.php">
  <label for="country_code">Country code:</label>
  <input type="text" id="country_code" name="country_code"
    pattern="[A-Za-z]{3}" <!-- อักษร A-Z พิมพ์เล็กหรือใหญ่ 3 ตัวติดกัน -->
    title="Three letter country code"><br><br>
  <input type="submit">
</form>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ



ถ้าชี้เมาส์ไว้เฉยๆที่ฟิลด์ Country code จะปรากฏข้อความตามภาพ เกิดจากแอททริบิวต์
`title="Three letter country code"`

ถ้ากรอกข้อมูลไม่ตรงกับแพทเทิร์นของ Regular Expression จากนั้นคลิกที่ปุ่ม Submit จะปรากฏ
กรอบแจ้งเตือนตามภาพ



4.7. placeholder

HTML <input> placeholder Attribute

https://www.w3schools.com/tags/att_input_placeholder.asp

แอททริบิวต์ **placeholder** (จองพื้นที่) ใช้แสดงค่าไว้ในฟิลด์ เพื่ออธิบายถึงรูปแบบหรือค่า ที่ยูสเซอร์ควรกรอก เพื่อให้ตรงกับรูปแบบที่ฟิลด์ต้องการ

แอททริบิวต์ **placeholder** ใช้งานได้กับอิลีเมนต์ <input> ชนิดดังต่อไปนี้ text, search, url, tel, email และ password

```
<input placeholder="text">
```

ตัวอย่าง

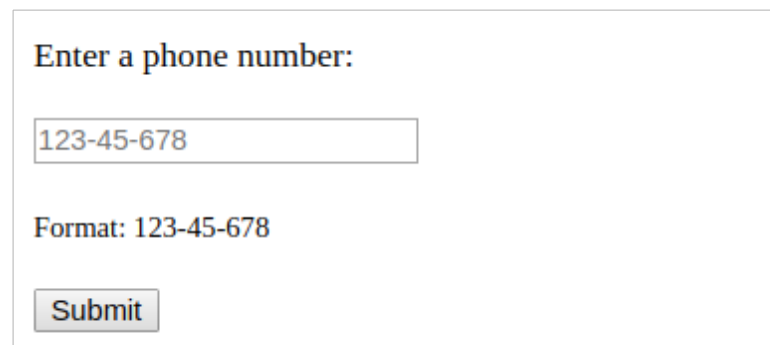
```
<form action="/action_page.php">

  <label for="phone">Enter a phone number:</label><br><br>
  <input type="tel" id="phone" name="phone" placeholder="123-45-678"
    pattern="[0-9]{3}-[0-9]{2}-[0-9]{3}"><br><br>

  <small>Format: 123-45-678</small><br><br>
  <input type="submit">

</form>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ สังเกตค่าที่แสดงในฟิลด์ เป็นข้อความที่เกิดจาก `placeholder="123-45-678"` โดยใช้งานคู่กับแอททริบิวต์ `pattern` เพื่อให้ใช้งานได้อย่างสอดคล้องกัน



4.8. required

HTML <input> required Attribute

https://www.w3schools.com/tags/att_input_required.asp

แอททริบิวต์ `required` เป็นบูลีนที่ใช้กำหนดว่า ฟิลด์นั้นจำเป็นต้องกรอกข้อมูลหรือไม่ ก่อนที่จะ Submit ฟอร์ม

แอททริบิวต์ `required` ใช้งานได้กับอิลีเมนต์ `<input>` ชนิดดังต่อไปนี้ text, search, url, tel, email, password, date pickers, number, checkbox, radio และ file

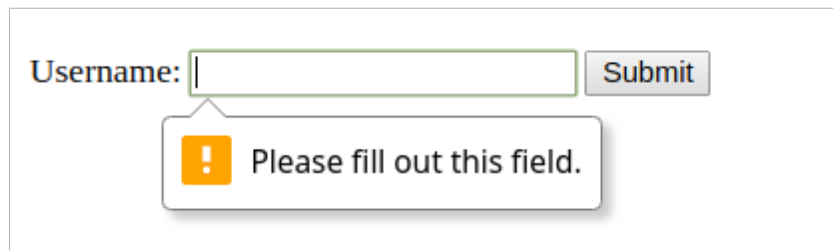
โครงสร้างการใช้งาน

```
<input required>
```

ตัวอย่าง

```
<form action="/action_page.php">
  <label for="username">Username:</label>
  <input type="text" id="username" name="username" required>
  <input type="submit">
</form>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ ถ้าเราคลิกที่ปุ่ม Submit โดยไม่กรอกข้อมูลลงในฟิลด์ ก่อน จะปรากฏกรอบแจ้งเตือนตามภาพ



4.9. autocomplete

HTML <input> autocomplete Attribute

https://www.w3schools.com/tags/att_input_autocomplete.asp

แอททริบิวต์ **autocomplete** ใช้กำหนดว่าจะใช้ Autocomplete กับฟิลด์หรือไม่ Autocomplete จะอนุญาตให้ Browser คาดเดาการกรอกค่าลงในฟิลด์ เมื่อยูสเซอร์เริ่มต้นกรอกค่าลงในฟิลด์ เพื่อช่วยให้ยูสเซอร์กรอกค่าลงในฟิลด์ได้ง่ายขึ้น

(Browser ที่ผู้เขียนใช้ก็คือ Chrome เปิดใช้ **autocomplete** กับทุกฟิลด์อยู่แล้ว โดยไม่ต้องกำหนดแอททริบิวต์ **autocomplete="on"**)

แอททริบิวต์ **autocomplete** ใช้งานได้กับอิลีเมนต์ **<input>** ชนิดดังต่อไปนี้ text, search, url, tel, email, password, datepickers, range และ color

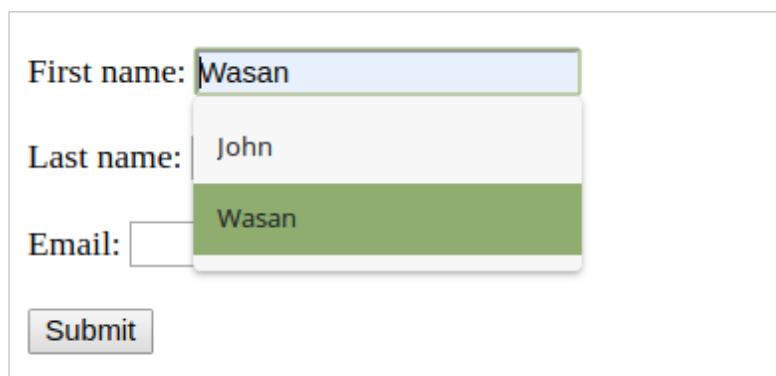
โครงสร้างการใช้งาน

```
<input autocomplete = "on | off">
```


ตัวอย่าง

```
<form action="/action_page.php">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <label for="email">Email:</label>
  <input type="email" id="email" name="email" autocomplete="off"><br><br>
  <input type="submit">
</form>
```

ผล - จาก HTML ข้างต้นจะได้ฟอร์มตามภาพ ไฟล์ First name มีการทำ Autocomplete



4.10. size

HTML <input> size Attribute

https://www.w3schools.com/tags/att_input_size.asp

แอททริบิวต์ **size** ใช้กำหนดความกว้างที่จะให้มองเห็น ในหน่วยจำนวนตัวอักษร ในอิลีเมนต์ <input>

แอททริบิวต์ **size** ใช้งานได้กับอิลีเมนต์ <input> ชนิดดังต่อไปนี้ text, search, tel, url, email และ password

โครงสร้างการใช้งาน

```
<input size="number"> <!-- ค่า Default เป็น 20 -->
```

ตัวอย่าง

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" size="50"><br>
  <label for="pin">PIN:</label><br>
  <input type="text" id="pin" name="pin" size="4">
</form>
```

ผล



4.11. maxlength

HTML <input> maxlength Attribute

https://www.w3schools.com/tags/att_input_maxlength.asp

แอททริบิวต์ **maxlength** ใช้กำหนดจำนวนตัวอักษรสูงสุดที่อนุญาต ในอิลีเมนต์ <input>

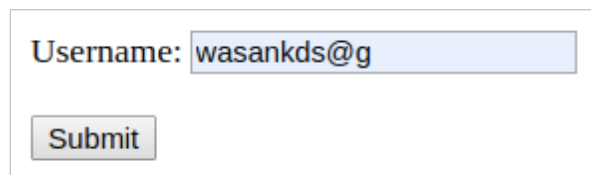
โครงสร้างการใช้งาน

```
<input maxlength="number"> <!-- ค่า Default เป็น 524288 -->
```

ตัวอย่าง

```
<form>
  <label for="username">Username:</label>
  <input type="text" id="username" name="username" maxlength="10"><br><br>
  <input type="submit" value="Submit">
</form>
```

ผล



4.12. min และ max

HTML <input> min Attribute

https://www.w3schools.com/tags/att_input_min.asp

HTML <input> max Attribute

https://www.w3schools.com/tags/att_input_max.asp

แอททริบิวต์ **min** และ **max** ใช้กำหนดค่าต่ำสุดและสูงสุด ตามลำดับ ให้กับอิลีเมนต์ <input>

แอททริบิวต์ **min** และ **max** ใช้งานได้กับอิลีเมนต์ <input> ชนิดดังต่อไปนี้ number, range, date, datetime-local, month, time และ week

```
<input min="number | date">
```

```
<input max="number | date">
```

ตัวอย่าง

```
<form action="/action_page.php">
  <label for="datemax">Enter a date before 1980-01-01:</label>
  <input type="date" id="datemax" name="datemax" max="1979-12-31"><br><br>
  <label for="datemin">Enter a date after 2000-01-01:</label>
  <input type="date" id="datemin" name="datemin" min="2000-01-02"><br><br>
  <label for="quantity">Quantity (between 1 and 5):</label>
  <input type="number" id="quantity" name="quantity" min="1" max="5"><br><br>
  <input type="submit">
</form>
```

ผล - ถ้าเราเลือกค่าจากปุ่มที่ฟอร์มมีให้ ค่าจะไม่อยู่นอกเหนือ min และ max ที่กำหนด แต่ถ้าเรากรอกเอง สามารถกรอกเป็นอะไรก็ได้ แต่จะ Submit ไม่ได้

4.13. multiple

HTML <input> multiple Attribute

https://www.w3schools.com/tags/att_input_multiple.asp

แอททริบิวต์ **multiple** เป็นบูลีนที่อนุญาตให้ผู้ใช้เลือกค่าได้มากกว่า 1 หรือไม่ ในอิลีเมนต์ <input>

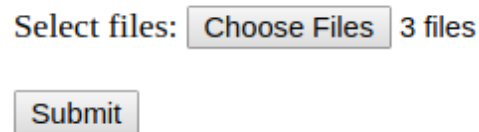
เช่น การกรอกอีเมล <input type="email"> สามารถกรอกแบบนี้ในฟิลด์ได้ mail@example.com, mail2@example.com, mail3@example.com เป็นต้น

```
<input multiple>
```

ตัวอย่าง

```
<form>
  <label for="files">Select files:</label>
  <input type="file" id="files" name="files" multiple><br><br>
  <input type="submit">
</form>
```

ผล



4.14. class

4.14.ก.) แอทธิบิวต์ class

HTML The class Attribute

https://www.w3schools.com/html/html_classes.asp

แอทธิบิวต์ **class** ใช้สำหรับกำหนดคลาสของสไตล์ ให้กับอิลีเมนต์ อิลีเมนต์ที่ใช้คลาสเดียวกัน เวลาแก้ไขสไตล์ก็จะเปลี่ยนตามกัน สะดวกมากในการจัดรูปแบบให้กับอิลีเมนต์

ตัวอย่าง

```
<html>
  <head>
    <style>
      .cities {
        background-color : black ;
        color : white ;
        margin : 10px ;
        padding : 10px ;
      }
    </style>
  </head>
```

```

<body>
  <div class="cities">
    <h2>London</h2>
    <p>London is the capital of England.</p>
  </div>

  <div class="cities">
    <h2>Paris</h2>
    <p>Paris is the capital of France.</p>
  </div>

  <div class="cities">
    <h2>Tokyo</h2>
    <p>Tokyo is the capital of Japan.</p>
  </div>
</body>
</html>

```

ผล - จาก HTML ข้างต้นจะได้หน้าเพจตามภาพ

London

London is the capital of England.

Paris

Paris is the capital of France.

Tokyo

Tokyo is the capital of Japan.

4.14.ข.) แอธรีบิวต์ class แบบ Inline

ตัวอย่าง

```
<html>
<head>
  <style>
    span.note {
      font-size : 120% ;
      color: red ;
    }
  </style>
</head>
<body>
  <h1>My <span class="note">Important</span> Heading</h1>
  <p>This is some <span class="note">important</span> text.</p>
</body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้หน้าเพจตามภาพ

My Important Heading

This is some important text.

4.14.ค.) ใช้หลายคลาส

อิลีเมนต์ HTML สามารถใช้ได้หลายคลาส โดยระบุชื่อคลาสให้กับแอธรีบิวต์ class โดยเว้นช่องว่างระหว่างชื่อ

ตัวอย่าง

```
<html>
<style>
  .city {
    background-color : tomato ;
    color : white ;
    padding : 10px ;
  }
  .main {
    text-align : center ;
  }
</style>
```

```
<body>
  <h2>Multiple Classes</h2>

  <h2 class="city_main">London</h2>      <!-- ใช้หลายคลาส -->
  <h2 class="city">Paris</h2>
  <h2 class="city">Tokyo</h2>
</body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้หน้าเพจตามภาพ



4.15. Constraint validation

JavaScript Validation API

https://www.w3schools.com/js/js_validation_api.asp

HTMLSelectElement.checkValidity()

<https://developer.mozilla.org/en-US/docs/Web/API/HTMLSelectElement/checkValidity>

Constraint validation

https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5/Constraint_validation

HTML5 มีเครื่องมือใหม่สำหรับกระบวนการสร้างฟอร์ม ใช้กับอิลีเมนต์ `<input>` ก็คือ **Constraint validation** (เป็น Built-in API) ซึ่งจะช่วยตรวจสอบเนื้อหาในฟอร์มในฝั่ง Client-side

การทำ Validation ให้กับฟอร์ม สามารถใช้งานโดยไม่ต้องเขียน Javascript ก็ได้ ก็คือ ใช้การกำหนดแอตทริบิวต์ เช่น min, max, required หรือ pattern เป็นต้น ดูรายละเอียดเพิ่มเติมได้จากลิงค์ด้านล่าง

Client-side form validation

https://developer.mozilla.org/en-US/docs/Learn/Forms/Form_validation

กรณีจะเขียน Javascript ก็มี 2 เมธอดสำคัญของ Constraint validation ดังนี้

Constraint Validation DOM Methods

Property	Description
<code>checkValidity()</code>	คืน true ถ้าอิลีเมนต์ input มีข้อมูลที่ Valid
<code>setCustomValidity()</code>	ใช้กำหนดคุณสมบัติให้กับ validationMessage ของอิลีเมนต์ input

Constraint Validation DOM Properties – เป็นคุณสมบัติของอิลีเมนต์ `<input>`

Property	Description
<code>validity</code>	เก็บบูลีน ที่เกี่ยวข้องกับ Validity ของอิลีเมนต์ input (ดูตาราง Validity Properties ต่อ)
<code>validationMessage</code>	เก็บข่าวสารที่ Browser จะแสดงเมื่อ validity เป็น false
<code>willValidate</code>	แสดงถึงว่า อิลีเมนต์ input จะถูก Validate

Validity Properties

คุณสมบัติ Validity ของอิลีเมนต์ input – เป็นคุณสมบัติต่างๆที่เกี่ยวข้องกับผลของ Validity เช่น `rangeOverflow` เป็น true ถ้าค่าของอิลีเมนต์ input มีค่ามากกว่าค่าที่กำหนดโดยแอตทริบิวต์ `max`

Property	Description
<code>customError</code>	Set to true, if a custom validity message is set.
<code>patternMismatch</code>	Set to true, if an element's value does not match its pattern attribute.
<code>rangeOverflow</code>	Set to true, if an element's value is greater than its max attribute.
<code>rangeUnderflow</code>	Set to true, if an element's value is less than its min attribute.
<code>stepMismatch</code>	Set to true, if an element's value is invalid per its step attribute.
<code>tooLong</code>	Set to true, if an element's value exceeds its maxLength attribute.
<code>typeMismatch</code>	Set to true, if an element's value is invalid per its type attribute.
<code>valueMissing</code>	Set to true, if an element (with a required attribute) has no value.
<code>valid</code>	Set to true, if an element's value is valid.

4.15.ก.) checkValidity()

checkValidity() คืน true ถ้าอิลีเมนต์ input มีข้อมูลที่ Valid

โครงสร้างการใช้งาน

```
var result = selectElt.checkValidity() ;
```

ตัวอย่าง การใช้งาน checkValidity() - ถ้าฟิลด์ input มีข้อมูลที่ Invalid ให้แสดงข้อความ

```
<html>
<body>
  <p>Enter a number and click OK:</p>
  <input id="id1" type="number" min="100" max="300" required>
  <button onclick="myFunction()">OK</button>
<script>

function myFunction() {
  var inpObj = document.getElementById("id1");
  if (!inpObj.checkValidity()) {
    document.getElementById("demo").innerHTML = inpObj.validationMessage ;
  } else {
    document.getElementById("demo").innerHTML = "Input OK" ;
  }
}
</script>
</body>
</html>
```

ผล

Enter a number and click OK:

50 OK

Value must be greater than or equal to 100.

Enter a number and click OK:

700 OK

Value must be less than or equal to 300.

Enter a number and click OK:

150 OK

Input OK

4.15.ข.) ตัวอย่าง rangeUnderflow

ตัวอย่าง – rangeUnderflow จะถูกเซ็ตเป็น true เมื่อกรอกค่าต่ำกว่าค่าที่กำหนดในแอททริบิวต์ min

```
<input id="id1" type="number" min="100">
<button onclick="myFunction()">OK</button>
<p id="demo"></p>

<script>
function myFunction() {
  var txt = "";
  if (document.getElementById("id1").validity.rangeUnderflow) {
    txt = "Value too small" ;
  }
  document.getElementById("demo").innerHTML = txt ;
}
</script>
```

ผล



4.15.ค.) setCustomValidity()

Validating forms using JavaScript

https://developer.mozilla.org/en-US/docs/Learn/Forms/Form_validation

setCustomValidity() ใช้กำหนดคุณสมบัติให้กับ validationMessage ของอิลีเมนต์ input

โครงสร้างการใช้งาน


```
selectElt.setCustomValidity("message") ;
```

ตัวอย่าง

```
<form>
  <label for="mail">I would like you to provide me with an e-mail address:</label>
  <input type="email" id="mail" name="mail">
  <button>Submit</button>
</form>

<script>
const email = document.getElementById("mail") ;
email.addEventListener("input", function(event) {
  if (email.validity.typeMismatch) {
    email.setCustomValidity("I am expecting an e-mail address!") ;
  } else {
    email.setCustomValidity("") ;
  }
}); //
</script>
```

I would like you to provide me with an e-mail address:

 I am expecting an e-mail address!

- Part II - Javascript

บทที่ 5

JavaScript

HTML DOM



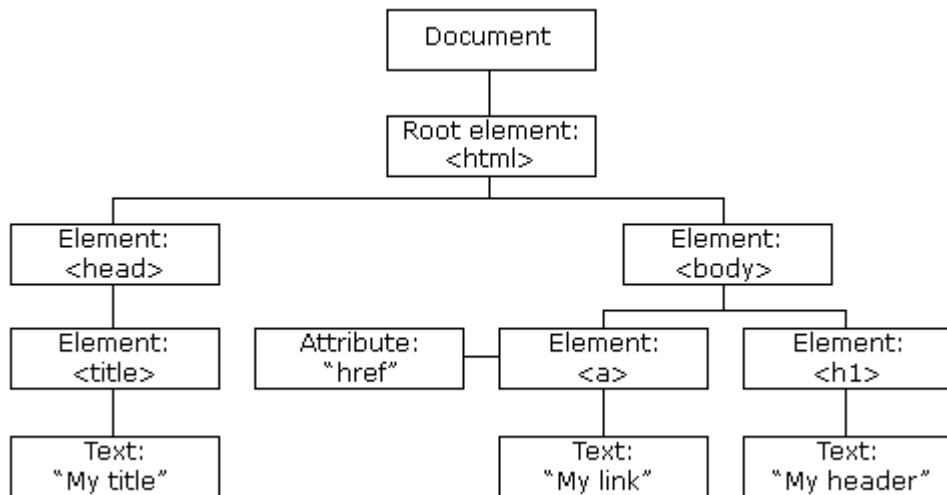
5.1. The HTML DOM (Document Object Model)

5.1.ก.) โมเดล HTML DOM

JavaScript HTML DOM

https://www.w3schools.com/js/js_htmldom.asp

เมื่อหน้าเว็บถูกโหลด Browser จะสร้างหน้าเพจที่เรียกว่า **Document Object Model** (หรือ **HTML DOM**) ขึ้นมา โดย **HTML DOM** มีโครงสร้างตามภาพดังต่อไปนี้



ด้วยโมเดลตามภาพข้างต้น Javascript จะสามารถสร้างหน้าเพจมีชีวิตชีวา หรือ มีความเคลื่อนไหวได้

1. JavaScript สามารถเปลี่ยนทุกอิลีเมนต์ HTML ในหน้าเพจได้
2. JavaScript สามารถเปลี่ยนทุกแอททริบิวต์ HTML ในหน้าเพจได้
3. JavaScript สามารถเปลี่ยนทุก CSS ในหน้าเพจได้
4. JavaScript สามารถลบอิลีเมนต์ HTML ที่ปรากฏอยู่ได้
5. JavaScript สามารถเพิ่มอิลีเมนต์ HTML และแอททริบิวต์ HTML ตัวใหม่ลงไปได้
6. JavaScript สามารถมีปฏิสัมพันธ์กับ HTML Event ที่มีในหน้าเพจได้
7. JavaScript สามารถสร้าง HTML Event ตัวใหม่ในหน้าเพจได้

DOM เป็นมาตรฐานหนึ่งของ **W3C (World Wide Web Consortium)** เป็นมาตรฐานที่ใช้เข้าถึงเอกสาร

มาตรฐาน **DOM** ของ **W3C** แบ่งเป็น 3 ส่วนที่แตกต่างกัน

Core DOM - มาตรฐานโมเดลสำหรับเอกสารทุกชนิด

XML DOM - มาตรฐานโมเดลสำหรับเอกสาร XML

HTML DOM - มาตรฐานโมเดลสำหรับเอกสาร HTML

5.1.ข.) DOM วัตถุที่สามารถโปรแกรมได้

JavaScript - HTML DOM Methods

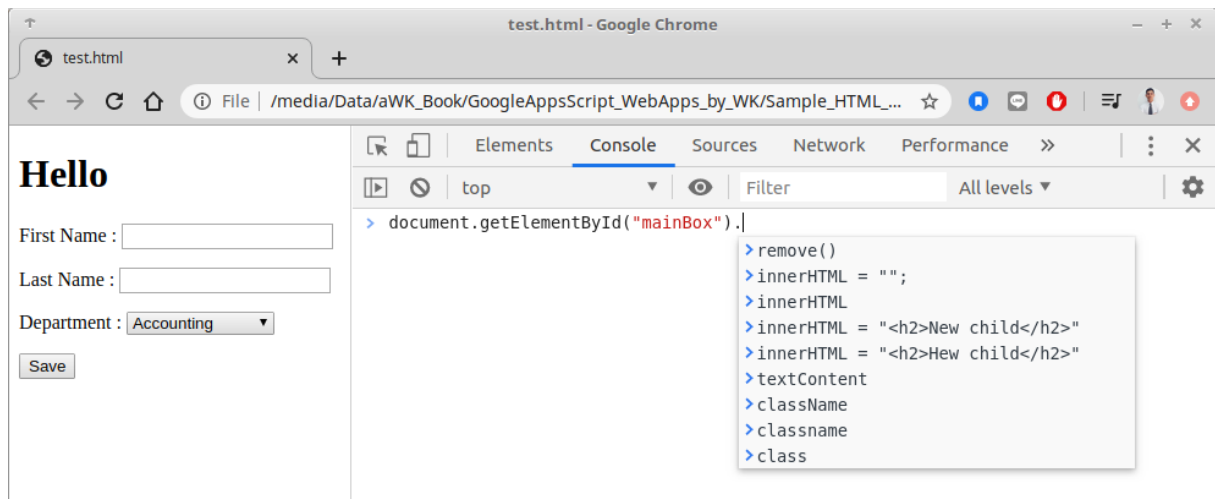
https://www.w3schools.com/js/js_htmldom_methods.asp

ถ้าเรามีพื้นฐานการเขียนโปรแกรมเชิงวัตถุ (OOP) จะเข้าใจ DOM ได้ไม่ยาก เพราะ DOM เป็นวัตถุที่มีเมธอด(Methods) และ คุณสมบัติ(Properties) ที่เราสามารถโปรแกรมได้ เพียงแต่เราต้องมาศึกษาว่า DOM HTML มีเมธอด และคุณสมบัติ อะไรให้เราโปรแกรมได้บ้าง

5.2. คอนโซลของ Browser

ที่ Browser อย่างเช่น Chrome หรือ Firefox ให้เปิดไฟล์ html อะไรก็ได้ จากนั้น [RMB@หน้าเว็บ](#) → [Inspect Element](#) หรือกด `<Ctrl><Shift><I>` จะเปิดกรอบคอนโซลของ Browser ขึ้นมา ตัวอย่างตามภาพถัดไป

ที่คอนโซล เราสามารถทดสอบพิมพ์ Javascript ในลักษณะคอมมานด์ไลน์ ลงไปได้เลย



หมายเหตุ :

ส่วนตัวผู้เขียนแล้ว พบว่าคอนโซลของ Firefox ดูเข้าใจง่ายกว่าของ Chrome

หมายเหตุ :

กรอบคอนโซล มีวิธีการใช้งานหลากหลาย เป็นเครื่องมือที่มีประโยชน์มากสำหรับการพัฒนาเว็บ

ตัวอย่าง - พิมพ์ข้อความหรือตัวแปรลงในคอนโซล

คอนโซล

```
> console.log("Print some text")
<- Print some text

> x = "Some text" ;
> console.log(x) ;
<- some text
```

ตัวอย่าง - พิมพ์คำสั่งที่คอนโซล โดยใช้คำสั่งจับอิลีเมนต์ แล้วดูว่าคอนโซลตอบกลับอย่างไร ตามตัวอย่าง คอนโซลตอบกลับมาเป็นอิลีเมนต์ ที่จับมาได้

คอนโซล

```
> document.getElementById("ln") ;  
<- <input type="text" id="ln">  
  
> document.getElementById("ln").parentNode ;  
<- <div class="form-row">  
    <label>Last Name : </label>  
    <input type="text" id="ln">  
</div>
```

บทที่ 6
จับอิเล็กทรอนิกส์
ด้วยเมทอดต่างๆ



6.1. วัตถุ HTMLCollection และ NodeList

6.1.ก.) วัตถุ HTMLCollection

DOM HTMLCollection

https://www.w3schools.com/jsref/dom_obj_htmlcollection.asp

เมื่อใช้เมธอดจับอิลีเมนต์ จะคืนค่ากลับมาเป็นวัตถุ HTMLCollection เป็นก้อนของอิลีเมนต์ ซึ่งมีพฤติกรรมคล้ายกับอาร์เรย์ ก็คือ สามารถเข้าถึงอิลีเมนต์แต่ละตัวได้โดยวิธีระบุดัชนีลำดับ (เริ่มจาก 0) เช่น `x[0]` เป็นต้น และสามารถใช้ `for` เพื่อวนลูปเข้าไปจัดการกับทุกตัวอิลีเมนต์ข้างใน (ใช้เมธอด `forEach` เหมือนอาเรย์ไม่ได้)

คุณสมบัติของวัตถุ HTMLCollection

Property / Method	Description
<code>item()</code>	คืนค่าเป็น อิลีเมนต์ ตามดัชนีลำดับที่ระบุ
<code>length</code>	คืนค่าเป็น จำนวนอิลีเมนต์ภายในวัตถุ HTMLCollection
<code>namedItem()</code>	คืนค่าเป็น อิลีเมนต์ที่ระบุ พร้อมกับ id หรือ name ของอิลีเมนต์

length

ตัวอย่าง – การใช้งานคุณสมบัติ `length`

```
var x = document.getElementsByClassName("example") ;
document.getElementById("demo").innerHTML = x.length ; // eg. 3
```

for

หากต้องการเข้าไปจัดการกับอิลีเมนต์ทุกตัวในวัตถุ HTMLCollection ให้ใช้ลูป `for` วนเข้าไป

ตัวอย่าง – วนลูปเข้าไปเปลี่ยนสีพื้นหลังของอิลีเมนต์ที่จับมาได้ ให้เป็นสีแดง

```
var x = document.getElementsByClassName("example") ;
var i ;
for (i = 0 ; i < x.length ; i++) {
  x[i].style.backgroundColor = "red" ;
}
```

จับอิลีเมนต์ตามดัชนีลำดับ

ตัวอย่าง

```
<script>
function myFunction() {
  document.getElementsByTagName("P")[0].innerHTML = "Hello World!" ;
}
</script>
```

6.1.ข.) วัตถุ NodeList

JavaScript HTML DOM Node Lists

https://www.w3schools.com/js/js_htmlDOM_nodelist.asp

วัตถุ NodeList เป็นก้อนของโหนดที่ตัดมาจากเอกสาร HTML วัตถุ NodeList **เกือบจะเหมือนกับวัตถุ HTMLCollection**

Browser รุ่นเก่าบางตัว คืนค่ากลับมาเป็น วัตถุ NodeList แทนที่จะเป็น HTMLCollection สำหรับ
เมธอด `getElementsByClassName()`

ทุก Browser คืนค่าเป็นวัตถุ NodeList สำหรับคุณสมบัติ `childNodes` ของอีเล็มเมนต์

เกือบทุก Browser คืนค่าเป็นวัตถุ NodeList สำหรับเมธอด `querySelectorAll()`

6.1.ค.) ความแตกต่างระหว่างวัตถุ HTMLCollection และ NodeList

วัตถุ HTMLCollection เป็นก้อนของอีเล็มเมนต์ HTML ส่วนวัตถุ NodeList เป็นก้อนของโหนดเอกสาร (Document nodes) เกือบจะเหมือนกันทั้งหมด (ผู้เขียนมักเรียกสลับกันไปมา เพราะตีความเป็นอันเดียวกัน เช่น ก้อนของอีเล็มเมนต์หรือโหนด) ต่างกันเพียงนิดหน่อย ดังนี้

วัตถุ HTMLCollection - เข้าถึงอีเล็มเมนต์ ได้โดยใช้แอททริบิวต์ **name**, **id** หรือ **เลขดัชนีลำดับ**

วัตถุ NodeList - เข้าถึงอีเล็มเมนต์ ได้โดยใช้ **เลขดัชนีลำดับ** เพียงอย่างเดียว

วัตถุ NodeList เท่านั้น ที่มีแอททริบิวต์ **nodes** และ **textNodes**

6.2. getElementById()

HTML DOM getElementById() Method

https://www.w3schools.com/jsref/met_document_getelementbyid.asp

`getElementById()` ใช้จับอีเล็มเมนต์ HTML ตามค่าที่ระบุในแอททริบิวต์ **id** ถ้ามีอีเล็มเมนต์ตามที่ระบุ จะคืนค่ากลับมาเป็น **element** ตัวนั้น แต่ถ้าไม่มี จะคืนค่ากลับมาเป็น **null**

`getElementById()` เป็นเมธอดพื้นฐานที่ใช้งานบ่อยมาก

โครงสร้างการใช้งาน

```
document.getElementById(elementID)
```

ตัวอย่าง

เมื่อโหลดหน้าเพจแล้ว โค้ด Javascript จะรอการคลิกปุ่ม เมื่อคลิกปุ่ม "ลิงคคลิกดูซิ" Javascript จะจับอ็
เล็มนัด id="demo" จากนั้นเปลี่ยน innerHTML ไปเป็นอย่างอื่น

```
<html>
<body>

  <p id="demo">คลิกที่ปุ่ม เพื่อเปลี่ยนข้อความนี้</p>
  <button onclick="myFunction()">ลองคลิกดูซิ</button>

  <script><!-- Javascript -->
    function myFunction() {
      document.getElementById("demo").innerHTML = "คุณคลิกปุ่มแล้ว" ;
    }
  </script>

</body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้หน้าเพจตามภาพ

คลิกที่ปุ่ม เพื่อเปลี่ยนข้อความนี้

ลองคลิกดูซิ

เมื่อคลิกที่ปุ่ม "ลิงคคลิกดูซิ" หน้าเพจจะเปลี่ยนไปเป็นตามภาพถัดไป ก็คือเปลี่ยนข้อความภายในอ็
เล็มนัด <p id="demo"></p> เป็นค่าใหม่

คุณคลิกปุ่มแล้ว

ลองคลิกดูซิ

6.3. getElementsByClassName

HTML DOM getElementsByClassName()

https://www.w3schools.com/jsref/met_document_getelementsbyclassname.asp

ใช้จับอ็เล็มนัดที่ใช้คลาสตามที่ระบุ โดยคืนค่าเป็น วัตถุ **HTMLCollection** หรือ ก้อนอ็เล็มนัดที่จับมา
ได้ โดยสามารถเข้าถึงได้ตามดรรชนีลำดับ เหมือนกับอาเรย์ (เริ่มจาก 0)

โครงสร้างการใช้งาน

```
document.getElementsByClassName(classname)
```

ตัวอย่าง

```
<html>
<head>
  <style>
    div {
      border : 1px solid black ;
      margin : 5px ;
    }
  </style>
</head>
<body>

  <div class="example">      <!-- อิเล็มเมนต์ <div> #1 -->
    <p>P element in first div with class="example". Div's index is 0.</p>
  </div>

  <div class="example color">  <!-- อิเล็มเมนต์ <div> #2 -->
    <p>P element in first div with class="example color". Div's index is 0.</p>
  </div>

  <div class="example color">  <!-- อิเล็มเมนต์ <div> #3 -->
    <p>P element in second div with class="example color". Div's index is 1.</p>
  </div>

  <button onclick="myFunction()">Try it</button>

  <script>
    function myFunction() {
      // 2 อิเล็มเมนต์ <div> ล่างที่ถูกจับ
      var x = document.getElementsByClassName("example color") ;
      x[0].style.backgroundColor = "red" ;
    }
  </script>
</body>
</html>
```

ผล - จาก HTML ข้างต้นจะได้หน้าเพจตามภาพ

P element in first div with class="example". Div's index is 0.

P element in first div with class="example color". Div's index is 0.

P element in second div with class="example color". Div's index is 1.

Try it

หลังคลิกที่ปุ่ม Try it หน้าเพจจะเปลี่ยนไปเป็นตามภาพ

P element in first div with class="example". Div's index is 0.

P element in first div with class="example color". Div's index is 0.

P element in second div with class="example color". Div's index is 1.

Try it

6.4. getElementsByTagName()

HTML DOM getElementsByTagName()

https://www.w3schools.com/jsref/met_document_getelementsbytagname.asp

ใช้จับอิลเมนต์ที่ใช้ชื่อแท็กตามที่ระบุ โดยคืนค่าเป็น วัตถุ HTMLCollection หรือ ก้อนอิลเมนต์ที่จับมาได้ โดยสามารถเข้าถึงได้ตามดรรชนีลำดับ เหมือนกับอาร์เรย์ (เริ่มจาก 0)

โครงสร้างการใช้งาน

```
document.getElementsByTagName(tagname)
```

การระบุพารามิเตอร์ เช่น P, LI, DIV เป็นต้น

ตัวอย่าง

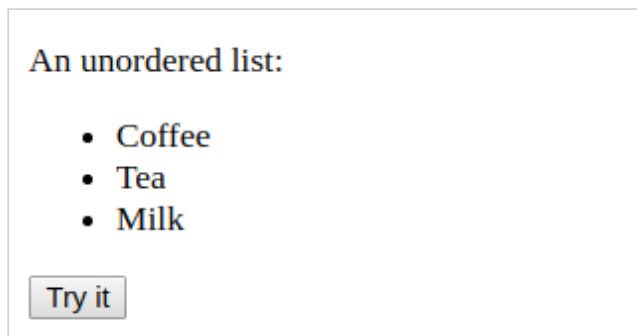
```
<html>
  <body>
    <p>An unordered list:</p>
    <ul>
      <li>Coffee</li>
      <li>Tea</li>
      <li>Milk</li>
    </ul>

    <button onclick="myFunction()">Try it</button>

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

    <script>
      function myFunction() {
        var x = document.getElementsByTagName("LI") ;
        document.getElementById("demo").innerHTML = x.length ;
      }
    </script>
  </body>
</html>
```


ผล - จาก HTML ข้างต้นจะได้หน้าเพจตามภาพ



หลังคลิกที่ปุ่ม `try it` จะปรากฏ เลข 3 ที่ด้านล่างปุ่ม ก็คือ มีอิลีเมนต์ที่ใช้แท็ก `` 3 อิลีเมนต์

หมายเหตุ

พารามิเตอร์ `*` (Asterisk) หมายถึงทุกแท็ก ใช้จับทุกอิลีเมนต์ภายใน HTML

```
var x = document.getElementsByTagName("*") ;
```

6.5. `getElementsByName`

HTML DOM `getElementsByName()`

https://www.w3schools.com/jsref/met_doc_getelementsbyname.asp

ใช้จับอิลีเมนต์ตามชื่อ ที่ระบุด้วยแอททริบิวต์ `name` โดยคืนค่าเป็น วัตถุ `HTMLCollection` หรืออิลีเมนต์ที่จับมาได้ โดยสามารถเข้าถึงได้ตามดัชนีลำดับ เหมือนกับอาร์เรย์ (เริ่มจาก 0)

โครงสร้างการใช้งาน

```
document.getElementsByName(fname)
```

ตัวอย่าง

```
<p>First Name: <input name="fname" type="text" value="Michael"></p>
<script>
  function myFunction() {
    var x = document.getElementsByName("fname")[0].tagName ;
    document.getElementById("demo").innerHTML = x ; // INPUT
  }
</script>
```

6.6. querySelector() และ querySelectorAll()

CSS Syntax

https://www.w3schools.com/css/css_syntax.asp

CSS Selector Reference

https://www.w3schools.com/cssref/css_selectors.asp

HTML DOM querySelector() Method

https://www.w3schools.com/jsref/met_document_queryselector.asp

HTML DOM querySelectorAll() Method

https://www.w3schools.com/jsref/met_document_queryselectorall.asp

querySelector() ใช้ค้นหาและจับอิลีเมนต์ตัวแรกที่พบ โดยจะค้นหาโดยใช้ CSS Selector(s)

หากต้องการจับอิลีเมนต์ทุกตัวที่เข้ากับเงื่อนไข ให้ใช้ querySelectorAll() โดยจะคืนค่ากลับมาเป็นวัตถุ NodeList หรือก้อนของโหนด ที่สามารถเข้าถึงแต่ละโหนดโดยใช้ดัชนีลำดับ (เริ่มต้นจาก 0) ได้เหมือนกับตัวแปรอาเรย์

โครงสร้างการใช้งาน

```
document.querySelector(CSS selectors)
```

querySelector() คืนค่าเป็นวัตถุ NodeList ของอิลีเมนต์แรกแมทช์กับ CSS Selector ที่ถ้าไม่พบจะคืนค่ากลับมาเป็น Null

```
document.querySelectorAll(CSS selectors)
```

querySelectorAll() คืนค่าเป็นวัตถุ NodeList เป็นก้อนของทุกอิลีเมนต์ที่แมทช์กับ CSS Selector ที่ถ้าไม่พบจะคืนค่ากลับมาเป็น Null

ตัวอย่าง - การใช้ querySelector

```
// จับอิลีเมนต์ <p> ตัวแรก  
document.querySelector("p") ;
```

```
// จับอิลีเมนต์ <p> ที่ใช้คลาส example ตัวแรก  
document.querySelector("p.example") ;
```

```
// จับอิลีเมนต์ที่ใช้มี id="demo" - เปลี่ยนข้อความใน Inner HTML  
document.querySelector("#demo").innerHTML = "Hello World!" ;
```

```
// จับอิลีเมนต์<p> ตัวแรก ที่มีอิลีเมนต์แม่เป็นอิลีเมนต์ <div>  
document.querySelector("div > p") ;
```

```
// จับอิลีเมนต์ <a> ตัวแรก ที่มีแอททริบิวต์ target  
document.querySelector("a[target]") ;
```

```
// ระบุอีเล็มเมนต์ 2 ตัวใน querySelector – ตัวแรกในลำดับของ HTML จะถูกจับก่อน
<h3>A h3 element</h3> <!-- ตัวนี้ถูกจับ แม่ใน querySelector ถูกระบุทีหลัง -->
<h2>A h2 element</h2>
<script>
    document.querySelector("h2, h3").style.backgroundColor = "red" ;
</script>
```

ตัวอย่าง - การใช้ querySelectorAll

```
// จับอีเล็มเมนต์ <p> ทั้งหมด
var x = document.querySelectorAll("p") ;

// เปลี่ยนแบ็คกราวนด์แรกของอีเล็มเมนต์ในวัตถุ NodeList เป็นสีแดง
x[0].style.backgroundColor = "red" ;
```

```
// จับอีเล็มเมนต์ <p> ที่ใช้คลาส example ทั้งหมด
var x = document.querySelectorAll("p.example") ;

// Logs จำนวนอีเล็มเมนต์ในตัวแปรวัตถุ NodeList
console.log(x.length) ;
```

```
// วนลูปเข้าไปเปลี่ยนพื้นหลังของอีเล็มเมนต์ที่ใช้คลาส example ให้เป็นสีแดง
var x = document.querySelectorAll("p.example") ;
var i ;
for (i = 0 ; i < x.length ; i++) {
    x[i].style.backgroundColor = "red" ;
}
```

```
// วนลูปเข้าไปขีดเส้นขอบให้ อีเล็มเมนต์ <a> ที่มีแอททริบิวต์ target
var x = document.querySelectorAll("a[target]") ;
var i ;
for (i = 0 ; i < x.length ; i++) {
    x[i].style.border = "10px solid red" ;
}
```

```
// วนลูปเข้าไปเปลี่ยนพื้นหลัง ให้กับอีเล็มเมนต์ <p> ที่มีอีเล็มเมนต์แม่เป็น <div>
var x = document.querySelectorAll("div > p") ;
var i ;
for (i = 0 ; i < x.length ; i++) {
    x[i].style.backgroundColor = "red" ;
}
```

```
// วนลูปเข้าไปเปลี่ยนพื้นหลัง ให้กับอีเล็มเมนต์ <h2>,<div> และ <span> ทั้งหมด
// ในเอกสาร HTML
var x = document.querySelectorAll("h2, div, span") ;
var i ;
for (i = 0 ; i < x.length ; i++) {
    x[i].style.backgroundColor = "red" ;
}
```

6.7. ใช้เมธอดของอาเรย์กับวัตถุ NodeList

NodeList

<https://developer.mozilla.org/en-US/docs/Web/API/NodeList>

เมธอดอย่าง `querySelectorAll()` คืนค่ากลับมาวัตถุ NodeList ซึ่งเป็นก้อนของโหนด(Nodes collection) แต่เนื่องจาก NodeList ไม่ใช่อาเรย์ เราจึงใช้เมธอดของอาเรย์วนลูปเข้าไปในก้อนโหนดไม่ได้

แม้ Javascript รุ่นใหม่ มีเมธอด `forEach()` สำหรับ NodeList แล้ว แต่อีกหลายๆตัวยังไม่มี เช่น `filter()` เป็นต้น

อย่างไรก็ดีเราสามารถใช้อะไรก็ได้

6.7.ก.) Array.prototype

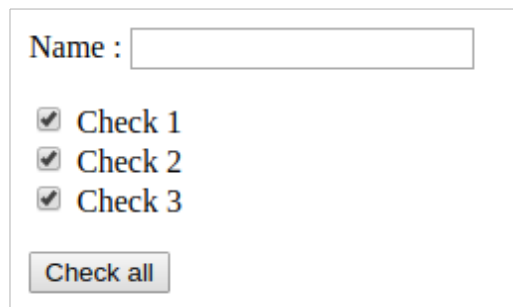
โครงสร้างการใช้งาน

```
Array.prototype.(ArrayMethod).call(NodeLists, function(AnElement) {  
    // Code  
});
```

ตัวอย่างที่ 1 : ใช้ `forEach`

```
<body>  
  <form>  
    <label>Name : </label>  
    <input type="text" id="Name"><p></p>  
  
    <input type="checkbox" id="check1">  
    <label for="check1">Check 1 </label><br>  
  
    <input type="checkbox" id="check2">  
    <label for="check2">Check 2 </label><br>  
  
    <input type="checkbox" id="check3">  
    <label for="check3">Check 3 </label><p></p>  
  
    <input type="button" id="btn" value="Click all">  
  </form>  
  
  <script>  
  
    document.getElementById("btn").addEventListener("click", myfunction) ;  
  
    function myfunction(){  
      var checkBoxes = document.querySelectorAll('input[type=checkbox]') ;  
  
      Array.prototype.forEach.call(checkBoxes, function(checkbox) {  
        checkbox.checked = true ;  
      }) ;  
    }  
  </script>  
</body>
```

ผล - เมื่อคลิกที่ปุ่ม Check all เช็คบ็อกซ์ทั้งหมดจะถูกติ๊ก



Name :

☒ Check 1
☒ Check 2
☒ Check 3

ตัวอย่างที่ 2 : ใช้ filter()

```
<!DOCTYPE html>
<html>
<head>
  <base target="_top">
</head>
<body>
  <form>
    <label for="name">Name : </label>
    <input type="text" id="name" required><br><br>

    <label for="lastname">lastname : </label>
    <input type="text" id="lastname" required><br><br>

    <label for="age">Age : </label>
    <input type="number" id="age" min="15" max="90" required><br><br>

    <input type="button" id="btn" value="Submit">
  </form>

  <p id="res"></p> <!-- สำหรับพิมพ์ผลลัพธ์ลงไป innerHtml -->

  <script>

document.getElementById("btn").addEventListener("click",myfunction) ;

function myfunction(){
    // จับไปที่ฟิลด์ input ที่ type เป็น text และ number
    var inputFields = document.querySelectorAll('input[type=text],input[type=number]');

    var res = [] ;
    var isAllFilled = Array.prototype.every.call(inputFields, function(field,i) {
        res.push(field.checkValidity()) ;
        return field.checkValidity() ; // คืน true ถ้าฟิลด์ Valid
    }) ;

    document.getElementById("res").innerHTML = res ;
  }
</script>

</body>
</html>
```

ผล – ฟิวด์ที่ 3 ก็คือ Age ไม่ Valid เพราะกำหนดค่าของแอททริบิวต์ min และ max ระหว่าง 15-90
เมื่อดูด checkValidity() ของอิลีเมนต์จึงคืนค่า false กลับมา

Name : Wasan

lastname : Kds

Age : 900

Submit

true,true,false

6.7.ข.) Array.from()

ตัวอย่างที่ 1 : ใช้ forEach

```
<body>
  <form>
    <label>Name : </label>
    <input type="text" id="Name"><p></p>

    <input type="checkbox" id="check1">
    <label for="check1">Check 1 </label><br>

    <input type="checkbox" id="check2">
    <label for="check2">Check 2 </label><br>

    <input type="checkbox" id="check3">
    <label for="check3">Check 3 </label><p></p>

    <input type="button" id="btn" value="Click all">
  </form>

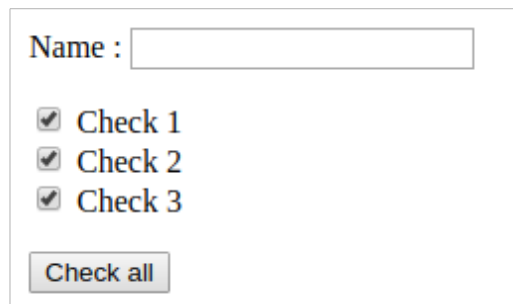
  <script>

document.getElementById("btn").addEventListener("click",myfunction) ;

function myfunction(){
  var checkBoxes = document.querySelectorAll('input[type=checkbox]') ;

  Array.from(checkBoxes).forEach(function(checkbox) {
    checkbox.checked = true ;
  }) ;
} // close - function
</script>
</body>
```

ผล - เมื่อคลิกที่ปุ่ม Check all เช็คบ็อกซ์ทั้งหมดจะถูกติ๊ก



Name :

☒ Check 1
☒ Check 2
☒ Check 3

ตัวอย่างที่ 2 : ใช้ filter()

```
<!DOCTYPE html>
<html>
<head>
  <base target="_top">
</head>
<body>
  <form>
    <label for="name">Name : </label>
    <input type="text" id="name" required><br><br>

    <label for="lastname">lastname : </label>
    <input type="text" id="lastname" required><br><br>

    <label for="age">Age : </label>
    <input type="number" id="age" min="15" max="90" required><br><br>

    <input type="button" id="btn" value="Submit">
  </form>

  <p id="res"></p> <!-- สำหรับพิมพ์ผลลัพธ์ลงไป innerHtml -->

  <script>
document.getElementById("btn").addEventListener("click",myfunction) ;

function myfunction(){
    // จับไปที่ฟิลด์ input ที่ type เป็น text และ number
    var inputFields = document.querySelectorAll('input[type=text],input[type=number]');

    var res = [] ;
    Array.from(inputFields).filter((field) => {
        res.push(field.checkValidity()) ;
        return field.checkValidity() ;
    }) ;

    document.getElementById("res").innerHTML = res ;
  }
</script>
</body>
</html>
```

ผล - ฟิวด์ที่ 3 ก็คือ Age ไม่ Valid เพราะกำหนดค่าของแอททริบิวต์ min และ max ระหว่าง 15-90
เมธอด checkValidity() ของอิลีเมนต์จึงคืนค่า false กลับมา

Name : Wasan

lastname : Kds

Age : 900

Submit

true,true,false

6.8. innerHTML

HTML DOM innerHTML Property

https://www.w3schools.com/jsref/prop_html_innerhtml.asp

คุณสมบัติ innerHTML ใช้กำหนดหรือคืนค่ากลับมาเป็น ข้อความระหว่างแท็ก HTML หรือค่า Inner HTML

โครงสร้างการใช้งาน - จับค่า Inner HTML

```
HTMLElementObject.innerHTML
```

โครงสร้างการใช้งาน - เซ็ตค่า Inner HTML

```
HTMLElementObject.innerHTML = text
```

ตัวอย่าง

```
<html>
  <body>
    <p id="demo" onclick="myFunction()">คลิกเพื่อเปลี่ยน Inner HTML</p>

    <script>
      function myFunction() {
        document.getElementById("demo").innerHTML = "เปลี่ยน Inner HTML แล้ว" ;
      }
    </script>

  </body>
</html>
```


ผล - จาก HTML ข้างต้นจะได้หน้าเพจตามภาพ

คลิกเพื่อเปลี่ยน Inner HTML

เมื่อคลิกบนข้อความตามภาพ ข้อความจะเปลี่ยนไปเป็น ตามภาพต่อไปนี้

เปลี่ยน Inner HTML แล้ว

6.9. parentNode

HTML DOM parentNode Property

https://www.w3schools.com/jsref/prop_node_parentnode.asp

โครงสร้างการใช้งาน

```
node.parentNode
```

คุณสมบัติ parentNode คืนค่ากลับมาเป็น เป็นวัตถุ Parent node

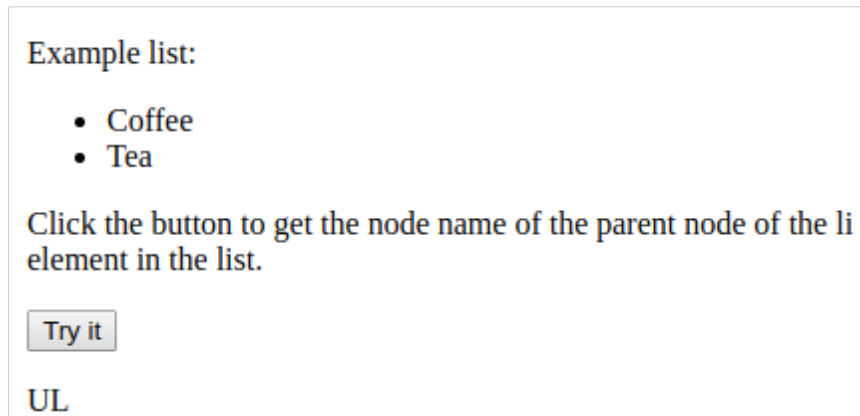
ใน HTML วัตถุ document ก็คือ parentNode ขององค์ประกอบใน HTML HEAD, BODY จึงเป็น Child nodes ขององค์ประกอบ HTML

ตัวอย่างที่ 1 - https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_node_parentnode

```
<!DOCTYPE html>
<html>
  <body>
    <p>Example list:</p>
    <ul>
      <li id="myLI">Coffee</li>
      <li>Tea</li>
    </ul>
    <p>Click the button to get the node name of the parent node of the li
      element in the list.</p>
    <button onclick="myFunction()">Try it</button>
    <p id="demo"></p> // แสดงผลลัพธ์จากการคลิกปุ่ม

    <script>
      function myFunction() {
        var x = document.getElementById("myLI").parentNode.nodeName ;
        document.getElementById("demo").innerHTML = x ;
      }
    </script>
  </body>
</html>
```

ผล - เมื่อคลิกที่ปุ่ม Try it จะปรากฏข้อความ UL องค์ประกอบ **myLI** หรือแท็ก มี Parent node เป็นแท็ก ผลลัพธ์จึงออกมาเป็น UL



ตัวอย่างที่ 2 - แก้โค้ดในตัวอย่างที่ 1 โดยเมื่อกดปุ่ม จากนั้นส่งพารามิเตอร์ `this.parentNode` ไปให้กับฟังก์ชัน `myFunction` ที่จะรันด้วย

```
<!DOCTYPE html>
<html>
  <body>
    <p>Example list:</p>
    <ul>
      <li id="myLI">Coffee</li>
      <li>Tea</li>
    </ul>
    <p>Click the button to get the node name of the parent node of the li
      element in the list.</p>
    <button onclick="myFunction(this.parentNode)">Try it</button>
    <p id="demo"></p> // แสดงผลลัพธ์จากการคลิกปุ่ม

    <script>
      function myFunction(para) {
        document.getElementById("demo").innerHTML = para.nodeName ;
      }
    </script>
  </body>
</html>
```

ผล – องค์ประกอบ Body เป็น Parent node ของปุ่ม try it (แท็ก <button>)

Example list:

- Coffee
- Tea

Click the button to get the node name of the parent node of the li element in the list.

Try it

BODY

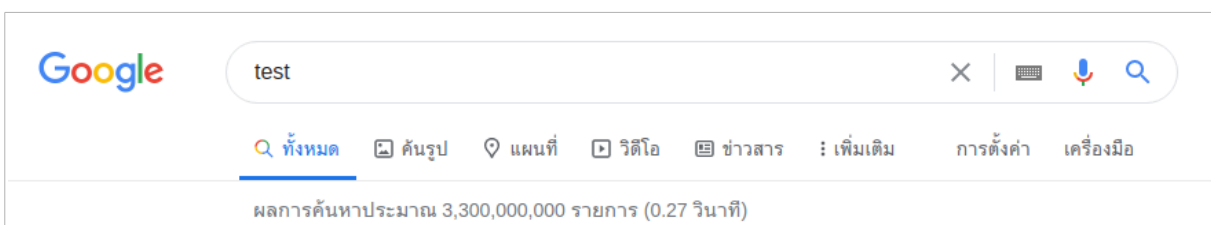
6.10. สร้างอีเล็มเมนต์ Form ด้วย Javascript

Forms: event and method submit
<https://javascript.info/forms-submit>

โค้ด

```
<html>
  <head></head>
  <body></body>
  <script>
    let form = document.createElement('form') ;
    form.action = 'https://google.com/search' ;
    form.method = 'GET' ;
    form.innerHTML = '<input name="q" value="test">' ;
    // the form must be in the document to submit it
    document.body.append(form) ;
    form.submit() ;
  </script>
</html>
```

ผล



บทที่ 7

วัตถุ



7.1. วัตถุ document

The HTML DOM Document Object

https://www.w3schools.com/jsref/dom_obj_document.asp

JavaScript HTML DOM Document

https://www.w3schools.com/js/js_htmlDOM_document.asp

เมื่อเอกสาร HTML ถูกโหลดไปไว้ใน Browser ก็จะกลายเป็นวัตถุ document ซึ่งก็คือ หน้าเว็บนั่นเอง และเป็น โหนดราก(Root node) หรือ วัตถุราก(Root object) เพื่อจะโยงต่อไปยังวัตถุลูกวัตถุหลานที่อยู่ข้างใน

วัตถุ document มีคุณสมบัติและเมธอดหลายตัว (ให้ดูจากลิงค์ข้างบน) ตัวอย่างดังต่อไปนี้

7.1.ก.) write() และ writeln()

HTML DOM write() Method

https://www.w3schools.com/jsref/met_doc_write.asp

HTML DOM writeln() Method

https://www.w3schools.com/jsref/met_doc_writeln.asp

write() ใช้เขียนข้อความ HTML หรือโค้ด Javascript ลงในเอกสาร HTML

writeln() ต่างจาก write() ที่ writeln() จะขึ้นบรรทัดใหม่ ในแต่ละครั้งที่เรียกใช้ แต่ write() จะเขียนข้อมูลต่อกันในบรรทัดเดียวกันไปเรื่อยๆ

หากใช้เมธอด write() หลังจากหน้าเว็บถูกโหลดแล้ว อีเล็มเมนต์ภายใน HTML จะถูกลบออกทั้งหมด ก่อนจะเขียนด้วย write()

ตัวอย่าง

```
<script>
  document.write("Hello World!") ;
</script>
```

```
<script>
  document.write("<h1>Hello World!</h1><p>Have a nice day!</p>") ;
</script>
```

ตัวอย่าง - ตัวอย่างต่อไปนี้ เป็นการใช้ write() หลังจากหน้าเว็บถูกโหลดแล้ว

```
<body>
  <h1>My First Web Page</h1>
  <p>My first paragraph.</p>
  <button type="button" onclick="myFunction()">Click me!</button>
  <script>
    function myFunction() {
      document.write("Hello World") ;
    }
  </script>
</body>
```

ผล – จาก HTML ข้างต้น จะได้หน้าเว็บตามภาพ

My First Web Page

My first paragraph.

Click me!

เมื่อคลิกที่ปุ่ม Click me! จะได้หน้าเว็บที่เปลี่ยนไปเป็นตามภาพ

Hello World

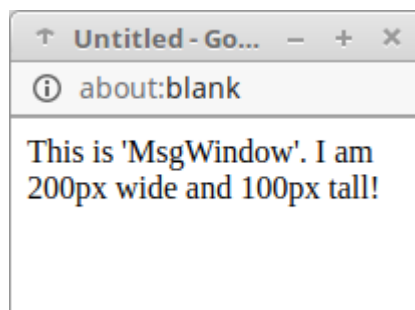
ตัวอย่าง – สร้างหน้าเว็บใหม่ แล้วใช้ write() เขียนข้อมูลลงไป

```
<body>
  <button onclick="myFunction()">Try it</button>
  <script>
    function myFunction() {
      var myWindow = window.open("", "MsgWindow", "width=200,height=100") ;
      myWindow.document.write("<p>This is 'MsgWindow'.
                               I am 200px wide and 100px tall!</p>") ;
    }
  </script>
</body>
```

ผล – จาก HTML ข้างต้น จะได้หน้าเว็บตามภาพ (มีแต่ปุ่ม Try it)

Try it

เมื่อคลิกที่ปุ่ม Try it จะปรากฏเว็บหน้าใหม่ มีลักษณะตามภาพ



7.1.ข.) createElement()

HTML DOM createElement() Method

https://www.w3schools.com/jsref/met_document_createelement.asp

createElement() ใช้สร้างโหนดหรืออิลีเมนต์ ตามที่ระบุ

โครงสร้างการใช้งาน

```
document.createElement(nodename)
```

ตัวอย่าง – สร้างอิลีเมนต์ <p> และ ใส่ข้อความ Inner HTML (คุณสมบัติ innerText) จากนั้นนำไปใส่แนบท้ายในส่วน body ของเอกสาร HTML

```
<script>
var para = document.createElement("P") ;    // สร้างอิลีเมนต์ <p>
para.innerText = "This is a paragraph" ;    // ใส่ข้อความให้อิลีเมนต์ <p>
document.body.appendChild(para) ;
</script>
```

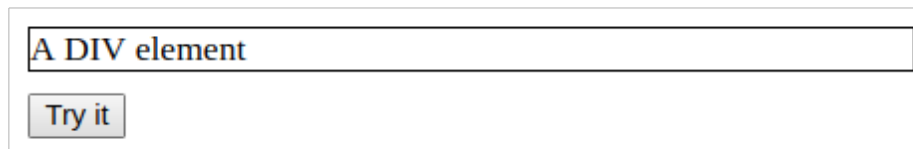

ตัวอย่าง – สร้างอิลีเมนต์ `<p>` และ ใส่ข้อความ Inner HTML (คุณสมบัติ `innerHTML`) จากนั้นนำไปใส่ไว้ในอิลีเมนต์ที่มี `id="myDIV"`

```
<html>
<head>
  <style>
    #myDIV { border: 1px solid black ; margin-bottom : 10px ; }
  </style>
</head>
<body>
  <div id="myDIV">A DIV element</div>
  <button onclick="myFunction()">Try it</button>

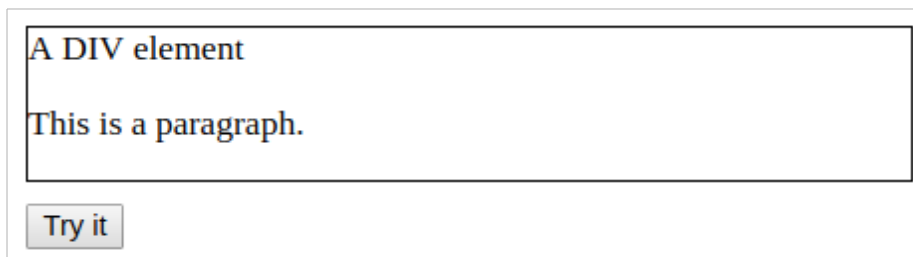
  </script>
  var para = document.createElement("P") ; // สร้างอิลีเมนต์ <p>
  para.innerHTML = "This is a paragraph." ; // ใส่ข้อความให้อิลีเมนต์ <p>
  // Append <p> to <div> with id="myDIV"
  document.getElementById("myDIV").appendChild(para) ;
</script>

</body>
</html>
```

ผล – จาก HTML ข้างต้น จะได้หน้าเว็บตามภาพ



เมื่อคลิกที่ปุ่ม `Try it` หน้าเว็บจะเปลี่ยนไปเป็นตามภาพต่อไปนี้



7.1.ค.) `createTextNode()`

HTML DOM `createTextNode()` Method

https://www.w3schools.com/jsref/met_document_createtextnode.asp

`createTextNode()` ใช้สร้างโหนด Text (ข้อความธรรมดา)

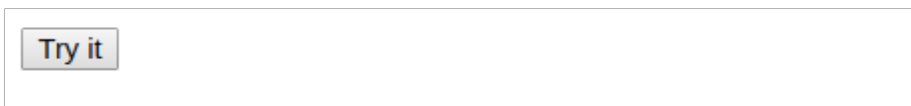
โครงสร้างการใช้งาน

```
document.createTextNode(text)
```

ตัวอย่าง

```
<html>
<body>
  <button onclick="myFunction()">Try it</button>
  <script>
    function myFunction() {
      var h = document.createElement("H1");           // สร้างอีเล็มเมนต์ <h1>
      var t = document.createTextNode("Hello World"); // สร้างโหนด Text
      h.appendChild(t);                                // แแนบ Text ไปกับ <h1>
      document.body.appendChild(h);                   // แแนบ <h1> ไปกับ <body>
    }
  </script>
</body>
</html>
```

ผล - จาก HTML ข้างต้น จะได้หน้าเว็บตามภาพ (มีแต่ปุ่ม Try it)



เมื่อคลิกที่ปุ่ม Try it หน้าเว็บจะเปลี่ยนเป็นดังต่อไปนี้ คลิกปุ่มไปเรื่อยๆ ข้อความ Hello world จะปรากฏไปเรื่อยๆ



7.1.ง.) body

HTML DOM body Property

https://www.w3schools.com/jsref/prop_doc_body.asp

โครงสร้างการใช้งาน

กรณี On return

```
document.body
```

กรณี On set

```
document.body = newContent
```

คุณสมบัติ **body** (กรณี On set) ใช้กำหนดคุณสมบัติให้กับส่วน Body หรือ (กรณี On return) ใช้จับส่วน Body ของเอกสาร HTML โดยจะคืนค่ากลับมาเป็น อีเล็มเมนต์ **<body>** ของเอกสาร HTML ตัวปัจจุบัน

ความแตกต่างระหว่าง คุณสมบัติ **body** และ **documentElement** ก็คือ จะคืนค่ากลับมาเป็นอีเล็มเมนต์ **<body>** และ **<html>** ตามลำดับ

ตัวอย่าง - ข้อความในหน้าเว็บจะมีเพียง "Some new HTML content"

```
document.body.innerHTML = "Some new HTML content" ;
```

ตัวอย่าง - แบนอีเล็มเมนต์ <p> ไว้ต่อท้ายภายในอีเล็มเมนต์ <body>

```
var x = document.createElement("P") ; // สร้างอีเล็มเมนต์ <p>
var t = document.createTextNode("This is a paragraph.") ; // สร้างอีเล็มเมนต์ Text
x.appendChild(t) ; // แบน Text ไปกับอีเล็มเมนต์ <p>
document.body.appendChild(x) ; // แบนอีเล็มเมนต์ <p> ไปกับ <body>
```

7.2. คุณสมบัติและเมทอดของอีเล็มเมนต์

The HTML DOM Element Object

https://www.w3schools.com/jsref/dom_obj_all.asp

ในโมเดล HTML DOM อีเล็มเมนต์ จะหมายถึงแท็ก HTML อย่างเช่น <p>, <div>, <a> หรือ <table> เป็นต้น

อีเล็มเมนต์ มีคุณสมบัติและเมทอดต่าง ๆ มากมาย (ให้ดูจากลิงค์ข้างบน) ตัวอย่างเช่น

7.2.ก.) appendChild()

HTML DOM appendChild() Method

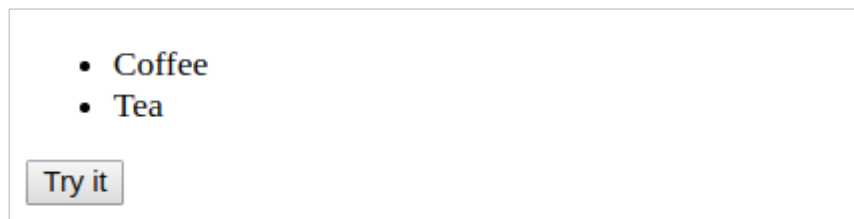
https://www.w3schools.com/jsref/met_node_appendchild.asp

ใช้เพิ่มโหนดลูก(หรืออีเล็มเมนต์ลูก) ให้กับอีเล็มเมนต์แบบต่อท้าย

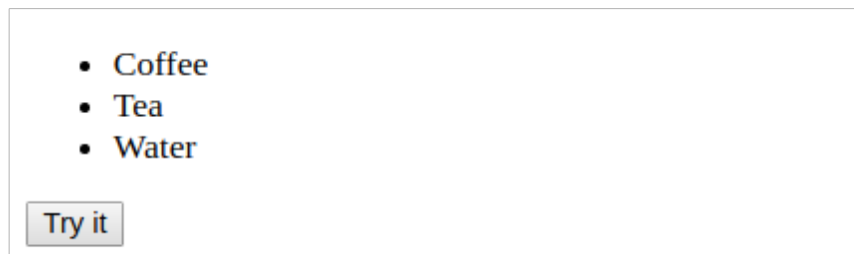
ตัวอย่างการใช้งาน appendChild()

```
<body>
  <ul id="myList">
    <li>Coffee</li>
    <li>Tea</li>
  </ul>
  <button onclick="myFunction()">Try it</button>
  <script>
    function myFunction() {
      var node = document.createElement("LI") ; // สร้างโหนด <li>
      var textnode = document.createTextNode("Water") ; // สร้างโหนด Text
      node.appendChild(textnode) ; // แบนโหนด Text ให้กับ <li>
      // แบนโหนด <li> ให้กับ <ul> ที่มี id="myList"
      document.getElementById("myList").appendChild(node) ;
    }
  </script>
</body>
```

ผล - จาก HTML ข้างต้น จะได้หน้าเว็บตามภาพ



เมื่อคลิกที่ปุ่ม Try it จะได้หน้าเว็บที่เปลี่ยนไป เป็นตามภาพ



7.2.ข.) childNodes

HTML DOM childNodes Property

https://www.w3schools.com/jsref/prop_node_childnodes.asp

เป็นคุณสมบัติที่คืนค่ากลับมาเป็นวัตถุ NodeList ที่เป็นก้อนของโหนดลูก ของโหนดที่ระบุ

ตัวอย่าง

```
<body><!-- This is a comment node! -->
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>

<script>

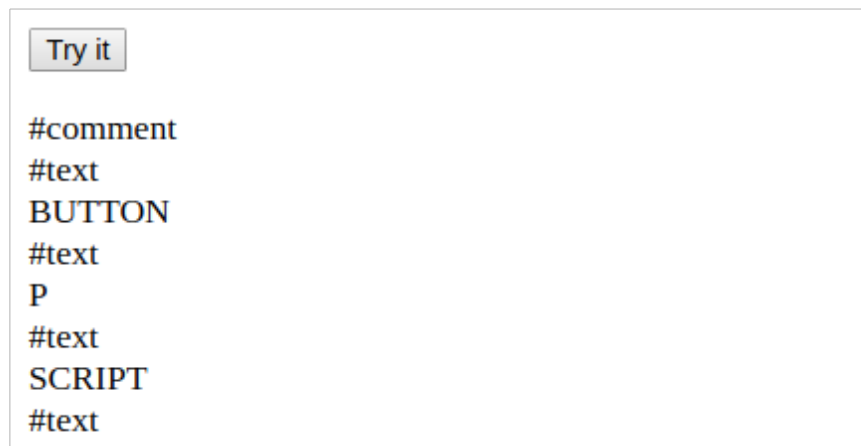
function myFunction() {
  var c = document.body.childNodes ; // จับโหนดลูกทั้งหมดของโหนด Body
  var txt = "" ;
  var i ;
  for (i = 0 ; i < c.length ; i++) {
    txt = txt + c[i].nodeName + "<br>" ;
  }
  document.getElementById("demo").innerHTML = txt ;
}

</script>
</body>
```

ผล - จาก HTML ข้างต้น จะได้หน้าเว็บตามภาพ



เมื่อคลิกที่ปุ่ม `Try it` จะได้หน้าเว็บที่เปลี่ยนไป เป็นตามภาพ แสดงถึงโหนดที่อยู่ภายใน `<body>`



7.2.ค.) tagName

HTML DOM tagName Property

https://www.w3schools.com/jsref/prop_element_tagname.asp

tagName คืค่าเป็นชื่อแท็กของอิลีเมนต์ ซึ่งได้เป็นตัวพิมพ์ใหญ่เสมอ เช่น `P`, `DIV`, `TABLE` เป็นต้น

โครงสร้างการใช้งาน

```
element.tagName
```

ตัวอย่าง – จับชื่อแท็กของอิลีเมนต์ `id="myP"`

```
var x = document.getElementById("myP").tagName ; // ผล x = P
```

ตัวอย่าง

```
<html>

<!-- ถ้ามี Event คลิก ให้ส่ง Event เป็นพารามิเตอร์ไปรันในฟังก์ชันที่ระบุ -->
<body onclick="myFunction(event)">
  <h1>This is a heading</h1>
  <button>This is a button</button>
  <span>A span element</span>
  <p id="demo"></p> <!-- สำหรับแสดงผลลัพธ์จากการคลิก -->

  <script>
    function myFunction(event) {
      var x = event.target ;
      document.getElementById("demo").innerHTML = "Triggered by a " +
                                                    x.tagName + " element" ;
    }
  </script>

</body>
</html>
```

This is a heading

This is a button A span element

ถ้าคลิกโดนอิลีเมนต์ตัวใด ก็แสดงผลว่าคลิกอิลีเมนต์ตัวนั้น เช่น

ถ้าคลิกที่ส่วนของ Body (ที่ว่าง) จะปรากฏข้อความต่อท้ายเป็น Triggered by **BODY** element

ถ้าคลิกที่ข้อความ A span element จะปรากฏข้อความต่อท้ายเป็น Triggered by a **SPAN** element

ถ้าคลิกปุ่ม This is a button จะปรากฏข้อความต่อท้ายเป็น Triggered by **BUTTON** element

เป็นต้น

7.2.ง.) style

HTML DOM style Property

https://www.w3schools.com/jsref/prop_html_style.asp

HTML DOM Style Object

https://www.w3schools.com/jsref/dom_obj_style.asp

The CSSStyleDeclaration Object

https://www.w3schools.com/jsref/obj_cssstyledeclaration.asp

คุณสมบัติ style คืค่าเป็นวัตถุ CSSStyleDeclaration ซึ่งเป็นก้อนของแอทริบิวต์ที่เก็บค่ารูปแบบของอิลีเมนต์

โครงสร้างการใช้งาน

กรณี On set

```
element.style.property = value
```

กรณี On return

```
element.style.property
```

(แอทริบิวต์ของ style มีอะไรบ้างดูได้จากลิงค์ข้างบน)

ตัวอย่าง - กรณี On set

```
// เซ็ตพื้นหลังให้กับอิลีเมนต์ เป็นสีแดง  
element.style.backgroundColor = "red" ;
```

ตัวอย่าง - กรณี On return

```
// จับคุณสมบัติเส้นขอบของอิลีเมนต์ id="myP"  
var x = document.getElementById("myP").style.borderTop ; // eg. 5px solid red
```

ตัวอย่าง

```
<html>
<head>
<style>
  body {
    background-color : yellow ;
    color : red ;
    font-size: 15px ;
  }
</style>
</head>
<body>
  <button onclick="myFunction()">Try it</button>
  <p id="demo"></p>
  <script>
    function myFunction() {
      var x = document.getElementsByTagName("STYLE")[0] ;
      document.getElementById("demo").innerHTML = x.innerHTML ;
    }
  </script>
</body>
</html>
```

จาก HTML ข้างต้นได้หน้าเว็บดังต่อไปนี้



เมื่อคลิกปุ่ม Try it จะได้หน้าเว็บดังต่อไปนี้



7.3. วัตถุ Event

HTML DOM Events

https://www.w3schools.com/jsref/dom_obj_event.asp

JavaScript Events

https://www.w3schools.com/js/js_events.asp

The Event Object

https://www.w3schools.com/jsref/obj_event.asp

Event Objects

https://www.w3schools.com/jsref/obj_events.asp

เมื่อมี **Event (เหตุการณ์)** ปรากฏใน HTML เช่น เมื่อโหลดหน้าเว็บเสร็จ, เมื่อค่าในฟิลด์ input เปลี่ยนแปลง, เมื่อคลิกปุ่ม เป็นต้น **Event** นั้นจะกลายเป็นส่วนหนึ่งของ **วัตถุ Event** เช่น **การคลิกเมาส์ (onClick)** เป็นส่วนหนึ่งของวัตถุ **MouseEvent** (วัตถุ **MouseEvent** มี **Events** หลายอย่างที่เกิดจากเมาส์)

HTML DOM events อนุญาตให้ Javascript เข้าไปจัดการอ็อบเจกต์ในเอกสาร HTML ตามแต่ Events ที่เกิดขึ้น

Events มักใช้คู่กับ ฟังก์ชัน โดยฟังก์ชันจะทำงานเมื่อมี Event เกิดขึ้น เช่น เมื่อคลิกเมาส์ฟังก์ชันจึงจะทำงาน

HTML DOM events มีอะไรบ้างดูได้จากลิงค์ข้างบน และ เนื่องจาก **Events** เป็นวัตถุที่มีคุณสมบัติและเมธอดต่างๆ สามารถดูได้จากลิงค์ข้างบนเช่นกัน (**HTML DOM Event Properties and Methods**)

ตัวอย่าง Events และ คุณสมบัติหรือเมธอดของวัตถุ Events ต่างๆ ดังต่อไปนี้

7.3.ก.) onclick()

onclick Event – เป็นส่วนหนึ่งของวัตถุ **MouseEvent**
https://www.w3schools.com/jsref/event_onclick.asp

HTML onclick Attribute
https://www.w3schools.com/tags/att_onclick.asp

The MouseEvent Object
https://www.w3schools.com/jsref/obj_mouseevent.asp

onclick Event จะปรากฏเมื่อมีเหตุการณ์คลิกที่อ็อบเจกต์

โครงสร้างการใช้งาน

1. ใน HTML – แอธริบิวต์ **onclick**

```
<element onclick="myScript">
```

2. ใน Javascript – เมธอด **onclick**

```
object.onclick = function(){ myScript } ;
```

3. ใน Javascript โดยใช้เมธอด **addEventListener()**
(รายละเอียดในข้อ 7.4 **addEventListener()** หน้า 107)

```
object.addEventListener("click", myScript) ;
```

ตัวอย่างที่ 1 – ใส่แอธริบิวต์ **onclick** ให้กับปุ่มเพื่อรันโค้ดที่ระบุเป็นค่าของแอธริบิวต์ **onclick**

```
<html>
<body>
  <button onclick="getElementById('demo').innerHTML=Date()">What is the time?
</button>
<p id="demo"></p>
</body>
```



```
</html>
```

เมื่อคลิกที่ ปุ่ม **What is the time?** ในหน้าเว็บ จะปรากฏข้อความแจ้งวันที่ด้านล่างของปุ่ม ตามภาพ

What is the time?

Fri Apr 10 2020 15:11:08 GMT+0700 (Indochina Time)

ตัวอย่างที่ 2 – เหมือนตัวอย่างที่ 1 ใช้ Event ในโค้ด HTML แต่ตัวอย่างนี้ส่งอิลีเมนต์ และ ค่าเป็นพารามิเตอร์ไปรันในฟังก์ชัน Javascript

```
<html>
<body>
  <!-- วัตถุ this ก็คือ อิลีเมนต์ <p> คือตัวมันเอง -->
  <p onclick="myFunction(this, 'red')">Click me to change my text color.</p>
  <script>
    function myFunction(elmnt,clr) {
      elmnt.style.color = clr ;
    }
  </script>
</body>
</html>
```

เมื่อคลิกข้อความ Click me to change my text color. ในหน้าเว็บ จะเปลี่ยนเป็นสีแดงตามภาพ ตามภาพ

Click me to change my text color.

ตัวอย่างที่ 3 – ใช้ Event **onclick** กับวัตถุใน Javascript เมื่อคลิกที่ส่วน Body ของ HTML สีพื้นหลังจะเปลี่ยนเป็นสีเหลือง

```
<script>
  window.onclick = myFunction ;
  function myFunction() {
    document.getElementsByTagName("BODY")[0].style.backgroundColor = "yellow" ;
  }
</script>
```

7.3.ข.) Events ที่เป็นส่วนหนึ่งของวัตถุ **MouseEvents**

นอกจาก **onclick** แล้ววัตถุ **MouseEvent** ยังมี Event ที่เกี่ยวกับเมาส์อีกหลายตัวดังตารางต่อไปนี้

Event	Description
onclick	The event occurs when the user clicks on an element
oncontextmenu	The event occurs when the user right-clicks on an element to open a context menu
ondblclick	The event occurs when the user double-clicks on an element
onmousedown	The event occurs when the user presses a mouse button over an element
onmouseenter	The event occurs when the pointer is moved onto an element
onmouseleave	The event occurs when the pointer is moved out of an element
onmousemove	The event occurs when the pointer is moving while it is over an element
onmouseout	The event occurs when a user moves the mouse pointer out of an element, or out of one of its children
onmouseover	The event occurs when the pointer is moved onto an element, or onto one of its children
onmouseup	The event occurs when a user releases a mouse button over an element

7.3.ค.) target

target Event Property

https://www.w3schools.com/jsref/event_target.asp

target เป็นคุณสมบัติของวัตถุ **MouseEvents** โดยจะคืนค่ากลับมาเป็น อีเล็มเมนต์ที่เกิด Event

โครงสร้างการใช้งาน

```
event.target
```

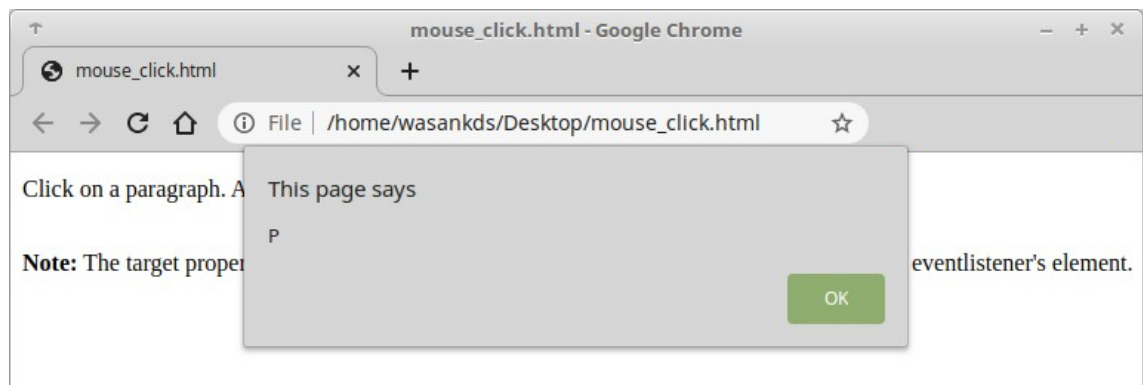
ตัวอย่าง – เมื่อเกิด Event **onclick** วัตถุ **MouseEvents** จะถูกส่งไปประมวลผลในฟังก์ชัน

myFunction โดยคุณสมบัติ **target** ของวัตถุ **MouseEvents** ก็คือ อีเล็มเมนต์ที่โดนคลิก เช่น **<body>** หรือ **<p>** เป็นต้น

```
<body onclick="myFunction(event)">
  <p>A Paragraph.</p>
  <script>
    function myFunction(event) {
      alert(event.target.nodeName) ; // ใช้ event.target.tagName ก็ได้
    }
  </script>
</body>
```

เมื่อคลิกที่อีเล็มเมนต์ จะปรากฏหน้าต่าง **alert** มาแจ้งว่า อีเล็มเมนต์ชนิดไหนโดนคลิก ตัวอย่างตาม

ภาพ

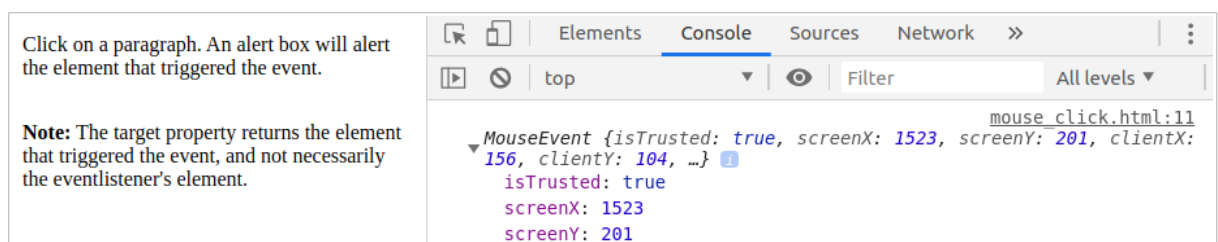


หมายเหตุ :

ใช้คอนโซลของ Browser จะช่วยได้มากกว่า ข้อมูลจาก Event มีอะไรส่งมาบ้าง ให้ลองแก้โค้ดในฟังก์ชันเป็น

```
console.log(event) ;
```

จากนั้นลองคลิกอีเล็มเมนต์ต่างๆดู พร้อมกับเปิดดูคอนโซล จะได้ผลลัพธ์ตามภาพดังต่อไปนี้ สังเกตดูว่าเมื่อมีการคลิก สิ่งที Logs ออกมาคือ วัตถุ **MouseEvent** ซึ่งมีคีย์และค่าต่างๆมากมาย



7.4. addEventListener()

7.4.ก.) addEventListener()

JavaScript HTML DOM EventListener

https://www.w3schools.com/js/js_html_dom_eventlistener.asp

addEventListener() เป็นเมธอดที่ใช้แนบตัวจัดการ Event ให้กับอีเล็มเมนต์ โดยเราสามารถแนบ Event ให้กับอีเล็มเมนต์ตัวเดียวกัน ได้หลาย Event

สามารถถอดตัวจัดการ Event ออกจากอีเล็มเมนต์ได้โดยใช้เมธอด removeEventListener()

โครงสร้างการใช้งาน

```
element.addEventListener(event, function, useCapture) ;
```

event : ชนิดของ Event เช่น **click, mouseover, mouseout, resize**

(ดูเพิ่มเติม [HTML DOM Event](#))

function : ฟังก์ชันที่จะเรียกใช้งานเมื่อเกิด Event

useCapture : เป็นบูลีน เพื่อที่จะใช้ event bubbling หรือ event capturing

7.4.ข.) ตัวอย่างที่ 1

โค้ด HTML

```
<html>
  <body>
    <h2>JavaScript addEventListener()</h2>
    <button id="myBtn">คลิกที่นี่</button>

    <script>
      document.getElementById("myBtn").addEventListener("click", function() {
        alert("Hello World!");
      });
    </script>
  </body>
</html>
```

ผล - เมื่อเปิดไฟล์ HTML ข้างต้น จะปรากฏหน้าเว็บดังต่อไปนี้

JavaScript addEventListener()

คลิกที่นี่

เมื่อคลิกที่ปุ่ม "**คลิกที่นี่**" จะปรากฏกรอบแจ้งเตือนตามภาพ

An embedded page on this page says

Hello World!

OK

7.4.ค.) ตัวอย่างที่ 2

โค้ด HTML

```
<html>
  <body>
    <h2>คลิกที่ปุ่มเพื่อคำนวณ</h2>
```

```

<button id="myBtn">คำนวณ</button>
<p id="demo"></p>

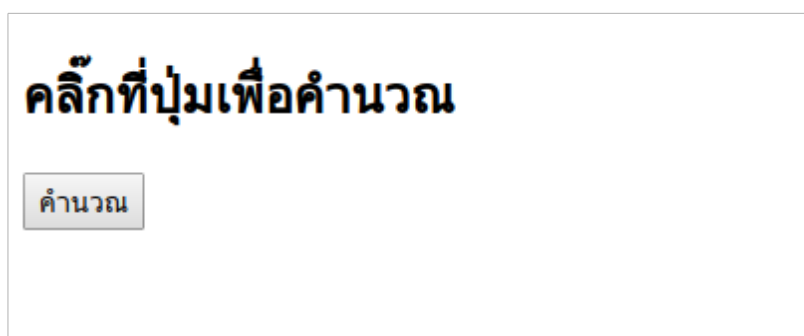
<script>
    var p1 = 5 ;
    var p2 = 7 ;

    document.getElementById("myBtn").addEventListener("click", function() {
        myFunction(p1, p2) ;
    }) ;

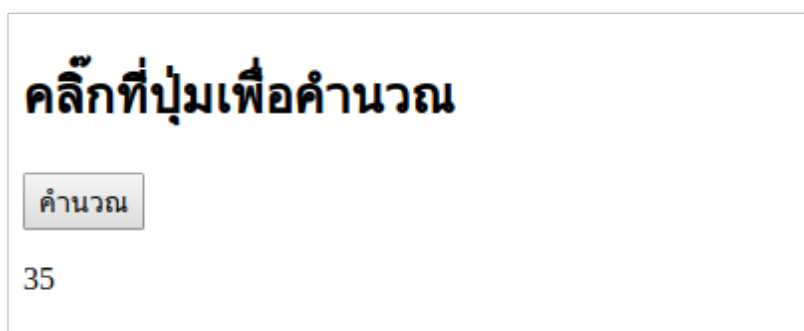
    function myFunction(a, b) {
        var result = a * b ;
        // innerHTML ก็คือ ข้อความที่อยู่ในแท็ก
        document.getElementById("demo").innerHTML = result ;
    }
</script>
</body>
</html>

```

ผล - เมื่อเปิดไฟล์ HTML ข้างต้น จะปรากฏหน้าเว็บดังต่อไปนี้



เมื่อคลิกที่ปุ่ม "คลิกที่นี่" จะปรากฏกรอบแจ้งเตือนตามภาพ



7.4.ง.) Event Bubbling or Event Capturing?

มี 2 วิธีในการขยายการจัดการ Event ออกไป(Event propagation) ใน HTML DOM ก็คือ **Bubbling** และ **Capturing**

Event propagation เป็นวิธีในการหาลำดับอีเล็มเมนต์ เมื่อเกิด Event เช่น ถ้าเรามีอีเล็มเมนต์ <p> ที่อยู่ใน <div> จากนั้นเราคลิกที่ <p> คำถามก็คือ “อีเล็มเมนต์ตัวไหนเป็นตัวที่ถูกคลิก ?”

Bubbling อีเล็มเมนต์ที่อยู่ข้างใน จะถูกจับเป็นลำดับแรก จากนั้นก็เป็นด้านนอก จากคำถามข้างต้น ก็คือ <p> ถูกจับเป็นลำดับแรก จากนั้นเป็น <div>

Capturing อีเล็มเมนต์ที่อยู่ข้างนอก จะถูกจับเป็นลำดับแรก จากนั้นก็เป็นด้านใน จากคำถามข้างต้น ก็คือ <div> ถูกจับเป็นลำดับแรก จากนั้นเป็น <p>

ตัวอย่าง

```
<html>
<head>
  <style>
    #myDiv1, #myDiv2 { background-color: coral ; padding: 20px ; }

    #myP1, #myP2 { background-color: white ; font-size: 20px ;
                  border: 1px solid ; padding: 20px ;}
  </style>
  <meta content="text/html; charset=utf-8" http-equiv="Content-Type">
</head>
<body>

  <h2>JavaScript addEventListener()</h2>

  <div id="myDiv1">
    <h2>Bubbling:</h2>
    <p id="myP1">Click me!</p>
  </div><br>

  <div id="myDiv2">
    <h2>Capturing:</h2>
    <p id="myP2">Click me!</p>
  </div>

  <script>
    document.getElementById("myP1").addEventListener("click", function() {
      alert("You clicked the white element!") ;
    }, false) ;

    document.getElementById("myDiv1").addEventListener("click", function() {
      alert("You clicked the orange element!") ;
    }, false);

    document.getElementById("myP2").addEventListener("click", function() {
      alert("You clicked the white element!") ;
    }, true);

    document.getElementById("myDiv2").addEventListener("click", function() {
      alert("You clicked the orange element!") ;
    }, true);
  </script>
</body>
```

</html>

ผล - เมื่อเปิดไฟล์ HTML จะได้ผลลัพธ์ตามภาพ จากนั้นทดสอบคลิกที่พื้นหลังสีส้มหรือสีขาวในกรอบต่างๆ จะปรากฏหน้าต่างมาแจ้งว่าอิลีเมนต์ถูกจับเป็นลำดับแรก

JavaScript addEventListener()

Bubbling:

Click me!

Capturing:

Click me!

7.5. อิลีเมนต์ 2 ประเภท

แท็กหรืออิลีเมนต์ HTML มี 2 ประเภท ก็คือ

แท็กหรืออิลีเมนต์แบบครอบ เช่น `<p> ... </p>` มีแท็กเปิดข้างหน้า ก็คือ `<p>` และ แท็กปิดข้างหลัง ก็คือ `</p>` ระหว่างแท็กก็คือ **Inner HTML** เป็นเนื้อหาที่อยู่ระหว่างแท็กเปิดและปิด

แท็กหรืออิลีเมนต์ว่าง(Empty elements) เช่น `
` มีเพียงแท็กเปิด ไม่มีปิด และไม่มี Inner HTML

7.6. อิลีเมนต์ลูก(Child) และ อิลีเมนต์แม่(Parent)

HTML DOM parentElement Property

https://www.w3schools.com/jsref/prop_node_parentelement.asp

HTML DOM parentNode Property

https://www.w3schools.com/jsref/prop_node_parentnode.asp

โครงสร้างของเอกสาร HTML มีลักษณะเป็น อิเล็มเมนต์ลูก(child) และ อิเล็มเมนต์แม่(Parent-ภาษาไทยอาจไม่ตรงเสียทีเดียว แต่เพื่อให้เข้าใจได้ง่าย) ของกันและกัน ตัวอย่างดังต่อไปนี้

```
<html> อิเล็มเมนต์แม่ใหญ่(Main parent)

<head> เป็นอิเล็มเมนต์ลูกโดยตรง(Direct child)ของ <html>
      ( ใช้ใส่ Style, Meta เป็นต้น ซึ่งจะไม่ปรากฏอะไรให้ยูสเซอร์เห็น )

<body> เป็นอิเล็มเมนต์ลูกโดยตรง(Direct child)ของ <html>

      <div> เป็นอิเล็มเมนต์ลูกโดยตรง(Direct child)ของ <body>

            <h1> เป็นอิเล็มเมนต์ลูกโดยตรง(Direct child)ของ <div>
```

ระบบนี้มีผลต่อการจับอิเล็มเมนต์ในลักษณะความสัมพันธ์ เพื่อนำมาโปรแกรมต่อยอดด้วย เช่น คุณสมบัติ parentNode เป็นอิเล็มเมนต์แม่ของวัตถุที่ถูกจับ เป็นต้น

หมายเหตุ : คุณสมบัติต่อไปนี้ ใช้จับโหนดหรืออิเล็มเมนต์ จากความสัมพันธ์

HTML DOM firstChild Property

https://www.w3schools.com/jsref/prop_node_firstchild.asp

HTML DOM lastChild Property

https://www.w3schools.com/jsref/prop_node_lastchild.asp

HTML DOM parentNode Property

https://www.w3schools.com/jsref/prop_node_parentnode.asp

HTML DOM nextSibling Property

https://www.w3schools.com/jsref/prop_node_nextsibling.asp

HTML DOM previousSibling Property

https://www.w3schools.com/jsref/prop_node_previousibling.asp

HTML DOM nodeName Property

https://www.w3schools.com/jsref/prop_node_nodename.asp

7.7. วัตถุ this

The JavaScript this Keyword

https://www.w3schools.com/js/js_this.asp

this เป็นคีย์เวิร์ดที่หมายถึง **วัตถุตัวเอง this** เป็นไปได้หลากหลายขึ้นอยู่กับว่า ถูกเรียกใช้ที่ไหน

7.7.ก.) การใช้ this ที่จุดต่างๆ

ตัวอย่าง - **this** โดดๆ ก็คือ วัตถุ Window

```
<html>
<body>
  <h2>The JavaScript <i>this</i> Keyword</h2>
  <p>In this example, <b>this</b> refers to the window Object:</p>
  <p id="demo"></p>
  <script>
```



```

var x = this ;
document.getElementById("demo").innerHTML = x ;
x.alert("Some text alert") ; // แสดงกรอบแจ้งเตือนที่มีข้อความตามที่ระบุ
</script>
</body>
</html>

```

ผล - จาก HTML ข้างต้นจะได้หน้าเพจตามภาพ

The JavaScript *this* Keyword

In this example, **this** refers to the window Object:

[object Window]

ตัวอย่าง - **this** ในฟังก์ชัน ก็คือ Global object ในที่นี้ก็คือ วัตถุ Window เหมือนกับข้างบน (ค่า Default เป็น Binding)

```

<p id="demo"></p>
<script>
  document.getElementById("demo").innerHTML = myFunction() ;

  function myFunction() {
    return this ;
  }
</script>

```

ผล - จาก HTML ข้างต้นจะได้หน้าเพจตามภาพ

[object Window]

ตัวอย่าง - **this** อยู่ใน Event หมายถึง อีเล็มเมนต์ที่รับ Event ไป

```

<html>
<body>
  <h2>The JavaScript <i>this</i> Keyword</h2>
  <button onclick="this.style.display='none'">Click to Remove Me!</button>
</body>
</html>

```

ผล - จาก HTML ข้างต้นจะได้หน้าเพจตามภาพ โดยเมื่อคลิกปุ่ม "Click to Remove Me!" ปุ่มดังกล่าวจะหายไป (**this** ก็คือปุ่มนี้)

The JavaScript *this* Keyword

Click to Remove Me!

ตัวอย่าง - `this` อยู่ในเมธอดของวัตถุ หมายถึง วัตถุที่เป็นเจ้าของเมธอด

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

<script>

function myFunction() {


    var person = {
        firstName: "John" ,
        lastName : "Doe" ,
        id      : 5566 ,
        fullName : function() {
            return this.firstName + " " + this.lastName ;
        } // Close - function
    } ; // Close - person

    console.log(person.firstName) ;           // พิมพ์ : John
    console.log(person.lastName) ;           // พิมพ์ : Doe
    console.log(person.id) ;                 // พิมพ์ : 5566.0
    console.log(person.fullName()) ;         // พิมพ์ : John Doe
}

document.getElementById("demo").innerHTML = person.fullName() ;

</script>
```

ผล - จาก HTML ข้างต้นจะได้หน้าเพจตามภาพ



John Doe

7.7.ข.) การเชื่อมโยงฟังก์ชัน(Function Binding)

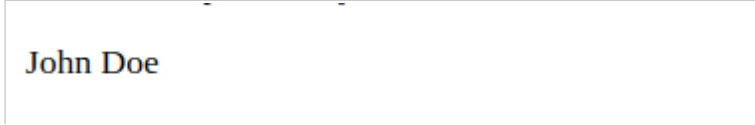
เมธอด `call()` และ `apply()` ทั้งคู่ใช้เรียกเมธอดของวัตถุจากวัตถุอื่นมาใช้ ในฐานะที่เป็นอากิวเมนต์

ตัวอย่าง – เมื่อเรียกเมธอด `person1.fullName()` โดยเอาวัตถุ `person2` มาใส่เป็นอากิวเมนต์ `this` จึงหมายถึง `person2` แม้จะเป็นเมธอดของ `person1` ก็ตาม

```
<p id="demo"></p>
<script>
    var person1 = {
        fullName: function() {
            return this.firstName + " " + this.lastName ;
        }
    }
    var person2 = {
        firstName : "John" ,
        lastName : "Doe" ,
    }
    var x = person1.fullName.call(person2) ;

    document.getElementById("demo").innerHTML = x ;
</script>
```

ผล - จาก HTML ข้างต้นจะได้หน้าเพจตามภาพ



John Doe

- Part III -
CSS

บทที่ 8

การใช้งาน CSS



8.1. CSS คืออะไร ?

CSS Tutorial

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

CSS คืออะไร?

http://www.enjoyday.net/webtutorial/css/css_chapter01.html

CSS (Cascading Style Sheets) เป็นภาษาที่ใช้พรรณารูปแบบหรือสไตล์ของเอกสาร HTML

CSS ช่วยลดงานจัดรูปแบบได้จำนวนมาก ที่สำคัญสามารถควบคุมรูปแบบของหน้าเว็บหลายๆหน้า ได้ที่

จุดเดียว

CSS จะเขียนไว้กับไฟล์ HTML ก็ได้ หรือจะแยกเป็นไฟล์ CSS ต่างหากก็ได้

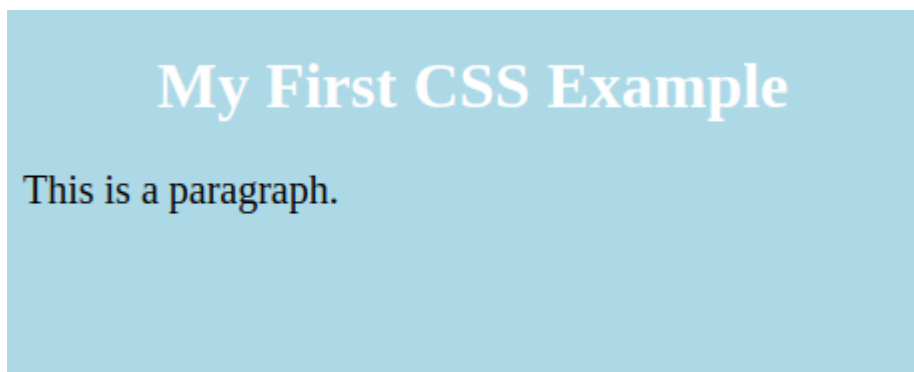
ตัวอย่าง

```
<html>
<head>
  <style>
    body {
      background-color: lightblue ;
    }

    h1 {
      color: white ;
      text-align: center ;
    }

    p {
      font-family: verdana ;
      font-size: 20px ;
    }
  </style>
</head>
<body>
  <h1>My First CSS Example</h1>
  <p>This is a paragraph.</p>
</body>
</html>
```

ผล



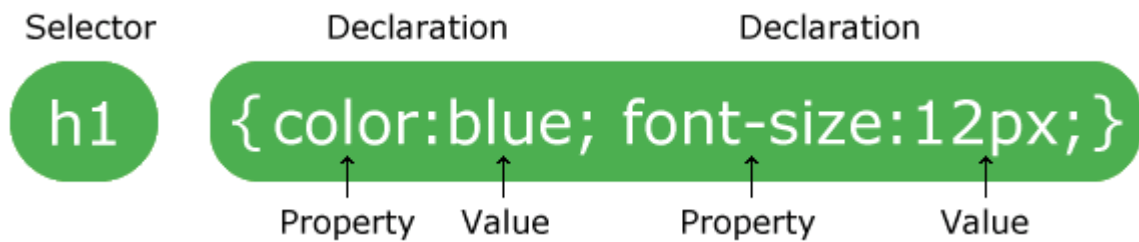
8.2. CSS Syntax

8.2.ก.) CSS Syntax

CSS Syntax

https://www.w3schools.com/css/css_syntax.asp

โครงสร้างของ CSS ประกอบด้วย **Selector**, **Property** และ **Values**



เวลาเขียนจะมักจะเขียนในลักษณะต่อไปนี้

```
/* มีคอมเมนต์มีแบบนี้แบบเดียว แบบ // (Double slashes) ใช้ใน CSS ไม่ได้ */  
p {  
  color: red ;  
  text-align: center ;  
}
```

8.2.ข.) Selector, Property, Value

Selector

หรือ **ตัวเลือกอิลีเมนต์** สามารถเป็นแท็ก HTML เช่น `<body>`, `<p>` หรือเป็น **class name** หรือ **id** ที่เราตั้งชื่อให้ก็ได้

Property

คุณสมบัติที่กำหนดการแสดงผล เช่น **color** (กำหนดสี), **font-size** (กำหนดขนาดตัวอักษร)

Value

เป็นค่าที่กำหนดให้กับ **Property** เช่น **color:white**, **font-size:14px**

8.3. ประเภทของ Selectors

CSS Selectors

https://www.w3schools.com/css/css_selectors.asp

Selectors มีอยู่หลายประเภทด้วยกัน ใช้งานแตกต่างกันไป หลักๆก็คือ จะเลือกอิเล็มเมนต์ในเอกสาร HTML อย่างไร เพื่อนำมาจัดรูปแบบ ด้วย **Properties** และ **Values** ที่อยู่ใน { }

ประเภทของ Selectors มีดังต่อไปนี้

1. Element selector (<Tage name>)
2. Id selector (#)
3. Class selector (.)
4. Universal Selector (*)

นอกจากนี้ก็ยังสามารถใช้งานผสมผสานกันได้ด้วย

8.3.ก.) Element selector (<Tage name>)

Element selector เลือกอิเล็มเมนต์ตามชื่อของแท็ก HTML มาจัดรูปแบบ

ตัวอย่าง - เลือกแท็ก <p> ทั้งหมดจะจัดรูปแบบด้วย CSS ต่อไปนี้

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

<body>
<p>Every paragraph will be affected by the style.</p>
<p id="para1">Me too!</p>
<p>And me!</p>
</body>
```

ผล

Every paragraph will be affected by the style.

Me too!

And me!

8.3.ข.) Id selector (#)

Id selector เลือกกิลี้มเน้นต์ตามแอธทริบิวต์ **id** มาจัดรูปแบบ

ตัวอย่าง - เลือกกิลี้มเน้นต์ที่มี **id** ชื่อ **para1** มาจัดรูปแบบด้วย CSS ต่อไปนี้

```
</head>
<style>
#para1 {
  color: red ;
  text-align: center ;
}
</style>
</head>

<body>
<p id="para1">Hello World!</p>
<p>This paragraph is not affected by the style.</p>
</body>
```

ผล

Hello World!

This paragraph is not affected by the style.

8.3.ค.) Class selector (.)

Class selector (.) เลือกกิลี้มเน้นต์ตามแอธทริบิวต์ **class** (ตามชื่อคลาส) มาจัดรูปแบบ

ตัวอย่าง - เลือกกิลี้มเน้นต์ที่มี **class** ชื่อ **center** มาจัดรูปแบบด้วย CSS ต่อไปนี้

```
</head>
<style>
.center {
  color: red ;
  text-align: center ;
}
</style>
</head>

<body>
<h1 class="center">Red and center-aligned heading</h1>
<p class="center">Red and center-aligned paragraph.</p>
</body>
```

ผล - แม้จะเป็นแท็กคนละประเภท แต่ใช้คลาสเดียวกัน จึงจัดรูปแบบด้วย CSS ตัวเดียวกัน

Red and center-aligned heading

Red and center-aligned paragraph.

8.3.ง.) Universal Selector (*)

CSS * Selector

https://www.w3schools.com/cssref/sel_all.asp

universal selector (*) เลือกทุกอิลีเมนต์ใน HTML มาจัดรูปแบบ

Selector * ใช้เลือกทุกอิลีเมนต์ รวมไปถึงอิลีเมนต์ข้างในด้วย

ตัวอย่าง - เลือกทุกอิลีเมนต์ และกำหนดสีพื้นหลังเป็นสีเหลือง

```
* {  
  background-color: yellow ;  
}
```

ตัวอย่าง - เลือกทุกอิลีเมนต์ภายใน <div> และกำหนดสีพื้นหลังเป็นสีเหลือง

```
div * {  
  background-color: yellow ;  
}
```

ตัวอย่าง

```
</head>  
<style>  
  * {  
    color: blue ;  
    text-align: center ;  
  }  
</style>  
</head>  
  
<body>  
  <h1>Hello world!</h1>  
  <p>Every element on the page will be affected by the style.</p>  
  <p id="para1">Me too!</p>  
  <p>And me!</p>  
</body>
```

ผล

Hello world!

Every element on the page will be affected by the style.

Me too!

And me!

8.4. การผสมผสานระหว่างประเภทของ Selectors

เราสามารถใช้ Selectors ผสมผสานเพื่อเลือกอีเล็มเมนต์ มาจัดรูปแบบได้ เช่น `p.center{ }`

ตัวอย่าง – เฉพาะอีเล็มเมนต์ `<p>` ที่มี `class="center"` จะถูกจัดเข้ากลาง

```
</head>
<style>
  p.center {
    color: red ;
    text-align: center ;
  }
</style>
</head>

<body>
  <h1 class="center">This heading will not be affected</h1>
  <p class="center">This paragraph will be red and center-aligned.</p>
</body>
```

ผล

This heading will not be affected

This paragraph will be red and center-aligned.

8.5. การใช้ CCS Class หลายตัวกับอีเล็มเมนต์เดียว

อีเล็มเมนต์ 1 ตัว สามารถใช้ `Class selector` ได้หลายตัว เพื่อจัดรูปแบบด้วยคลาสที่ระบุทั้งหมด

ตัวอย่าง – อีเล็มเมนต์ `<p>` สามารถจัดรูปแบบโดยใช้ CSS ทั้งของคลาส `center` และ `large` พร้อมกัน

```
<p class="center large">This paragraph refers to two classes.</p>
```

ตัวอย่างที่ 2

```
</head>
<style>
  p.center {                /* อักษรสีแดง + เข้ากลาง */
    color: red ;
    text-align: center ;
  }
  p.large {                 /* ขนาดฟอนต์ 300% */
    font-size: 300% ;
  }
</style>
</head>
<body>
  <h1 class="center">This heading will not be affected</h1>
  <p class="center">This paragraph will be red and center-aligned.</p>
  <p class="center large">This paragraph will be red, center-aligned, and in a large font-size.</p>
</body>
```

ผล

This heading will not be affected

This paragraph will be red and center-aligned.

**This paragraph will be red, center-aligned,
and in a large font-size.**

8.6. การรวมกลุ่ม Selectors

Selectors ที่มีสไตล์แบบเดียวกัน (Properties และ Value) สามารถเขียนรวมเป็นกลุ่มได้

ตัวอย่าง h1, h2 และ p มีสไตล์เหมือนกันตามตัวอย่าง

```
h1 {
  text-align: center ;
  color: red ;
}
h2 {
  text-align: center ;
  color: red ;
}
p {
  text-align: center ;
  color: red ;
}
```

ข้างต้น สามารถเขียนรวมกลุ่ม Selectors แบบนี้ได้

```
h1, h2, p {  
  text-align: center ;  
  color: red ;  
}
```

8.7. Attribute selectors

Attribute selectors

https://developer.mozilla.org/en-US/docs/Web/CSS/Attribute_selectors

Attribute selectors (เครื่องหมาย []) เหมือนเป็นส่วนต่อขยายของ Element selectors เพราะเวลาใช้งาน จะระบุ Element selectors ไว้ก่อน จากนั้นตามด้วย Attribute selectors เพื่อเลือกเจาะจงไปที่แอตทริบิวต์ และค่าของแอตทริบิวต์

ให้สังเกตจากตัวอย่าง แถวเครื่องหมาย = มีเครื่องหมาย *, ^, \$, ~ และ อื่นๆ

ตัวอย่าง

```
/* อีเล็มเมนต์ <a> ที่มีแอตทริบิวต์ title */  
a[title] {  
  color: purple ;  
}  
  
/* อีเล็มเมนต์ <a> ที่มีแอตทริบิวต์ href ที่ค่าเท่ากับ "https://example.org" */  
a[href="https://example.org"] {  
  color: green ;  
}  
  
/* อีเล็มเมนต์ <a> ที่มีแอตทริบิวต์ href ที่ค่าประกอบไปด้วย "example" */  
a[href*="example"] {  
  font-size: 2em ;  
}  
  
/* อีเล็มเมนต์ <a> ที่มีแอตทริบิวต์ href ที่ค่าลงท้ายด้วย ".org" */  
a[href$=".org"] {  
  font-style: italic ;  
}  
  
/* อีเล็มเมนต์ <a> ที่มีแอตทริบิวต์ class ที่ค่าประกอบไปด้วย "logo" */  
a[class~="logo"] {  
  padding: 2px ;  
}
```

8.7.ก.) Syntax

(ไม่มีเวลาแปลจ้... ขออภัย)

[attr] : Represents elements with an attribute name of attr.

[attr=value] : Represents elements with an attribute name of attr whose value is exactly value.

[attr~=value] : Represents elements with an attribute name of attr whose value is a whitespace-separated list of words, one of which is exactly value.

[attr|=value] : Represents elements with an attribute name of attr whose value can be exactly value or can begin with value immediately followed by a hyphen, - (U+002D). It is often used for language subcode matches.

[attr^=value] : Represents elements with an attribute name of attr whose value is prefixed (preceded) by value.

[attr\$=value] : Represents elements with an attribute name of attr whose value is suffixed (followed) by value.

[attr*=value] : Represents elements with an attribute name of attr whose value contains at least one occurrence of value within the string.

[attr operator value i] : Adding an i (or I) before the closing bracket causes the value to be compared case-insensitively (for characters within the ASCII range).

[attr operator value s] : Adding an s (or S) before the closing bracket causes the value to be compared case-sensitively (for characters within the ASCII range).

8.8. การใช้งาน CSS ในเอกสาร HTML

How To Add CSS

https://www.w3schools.com/css/css_howto.asp

8.8.ก.) 3 วิธีใส่ Style Sheet ให้กับเอกสาร HTML

เมื่อ Browser อ่าน Style Sheet Browser จะจัดรูปแบบเอกสาร HTML ตามข้อมูลใน Style Sheet โดยเรามี 3 วิธีในการใส่ Style Sheet ให้กับเอกสาร HTML

1. **Inline CSS**
2. **Internal CSS**
3. **External CSS**

8.8.ข.) Inline CSS

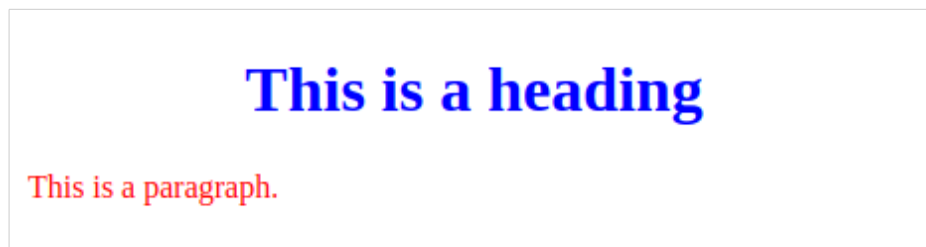
Inline CSS หรือ Inline Style ใช้กำหนดสไตล์ให้กับ อีเล็มเมนต์เดียว

การใช้งาน Inline Style สามารถทำได้โดยใช้แอททริบิวต์ **style** ใส่ลงในแท็ก HTML โดยค่าของแอททริบิวต์ **style** ก็คือ **property:value** ของ CSS

ตัวอย่าง

```
<!DOCTYPE html>
<html>
  <body>
    <h1 style="color:blue;text-align:center;">This is a heading</h1>
    <p style="color:red;">This is a paragraph.</p>
  </body>
</html>
```

ผล



8.8.ค.) Internal CSS

Internal CSS หรือ Internal style sheet ใช้กำหนดสไตล์ให้กับ อีเล็มเมนต์ในเอกสาร HTML หนึ่งเดียว

การใช้งาน Internal style sheet สามารถทำได้โดย เขียนโค้ด CSS ไว้ในบล็อกของแท็ก **<style>** ซึ่งอยู่ภายในแท็ก **<head>** อีกที

ตัวอย่าง

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      body {
        background-color: linen ;
      }
      h1 {
        color: maroon ;
        margin-left: 40px ;
      }
    </style>
  </head>
```

```

<body>
  <h1>This is a heading</h1>
  <p>This is a paragraph.</p>
</body>
</html>

```

ผล



8.9. External CSS

External CSS หรือ **External style sheet** ใช้กำหนดสไตล์ให้กับ **อิลีเมนต์**ในเอกสาร HTML ทั้งเว็บไซต์ โดยใช้ไฟล์ CSS เพียงไฟล์เดียว

การใช้งาน **Internal style sheet** สามารถทำได้โดยใช้แท็ก **<link>** ซึ่งอยู่ภายในแท็ก **<head>** อีกที่อ้างอิงหรือระบุลิงค์ไปยังไฟล์ CSS

ตัวอย่าง

ไฟล์ "index.html"

```

<html>
  <head>
    <link rel="stylesheet" type="text/css" href="mystyle.css">
    <!-- อยู่โฟลเดอร์เดียวกับไฟล์ .css -->
  </head>
  <body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
  </body>
</html>

```

ไฟล์ "mystyle.css"

```

body {
  background-color: lightblue ;
}

h1 {
  color: navy ;
  margin-left: 20px ;
}

```

This is a heading

This is a paragraph.

8.10. Multiple Style Sheets

หาก Property เดียวกัน ถูกกำหนดไว้กับ Selector ตัวเดียวกัน แต่อยู่ใน Style Sheets ที่ต่างกัน และ Value ของ Property ก็ไม่เหมือนกัน ผลก็คือ Value ที่ได้จากการอ่าน style sheet ตัวสุดท้ายจะถูกใช้งาน

ตัวอย่าง

ไฟล์ "mystyle.css"

```
/* External stylesheet */
h1 {
  color: navy ;
}
```

ไฟล์ "Index.html" - Internal CSS อยู่ด้านล่างของ External CSS

```
<head>
  <!-- External -->
  <link rel="stylesheet" type="text/css" href="mystyle.css">
  <style>
    h1 { /* Internal */
      color: orange ;
    }
  </style>
</head>
```

ผล - ทำ Internal CSS

This is a heading

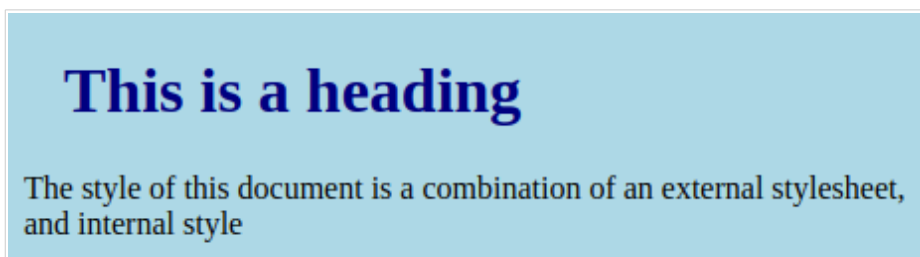
The style of this document is a combination of an external stylesheet, and internal style

ในกรณีกลับกัน ย้าย External CSS ลงมาด้านล่าง

ไฟล์ "Index.html"

```
<head>
  <style>
    h1 { /* Internal */
      color: orange ;
    }
  </style>
  <link rel="stylesheet" type="text/css" href="mystyle.css"> <!-- External -->
</head>
```

ผล - ทำ External CSS



8.1.1. Cascading Order

สไตล์ตัวไหนจะถูกใช้ ถ้าสไตล์หลายตัวถูกใช้กับอิลีเมนต์ ???

ทุกสไตล์ในหน้าเพจ จะมีน้ำหนักตามลำดับ Priority ดังนี้ โดยเรียงจาก Priority มากไปน้อย

1. **Inline style** (ในแท็ก HTML) – มี Priority สูงสุด
2. **External และ Internal style sheets** (ในบล็อกของ <header>)
3. **Browser default**

```
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="mystyle.css">
    <style>
      body { background-color: linen ; }
    </style>
  </head>

  <body style="background-color: lavender"> <!-- ทำสไตล์นี้ -->
    <h1>Multiple Styles Will Cascade into One</h1>
    <p>Here, the background color of the page is set with inline CSS, and also with
an internal CSS, and also with an external CSS.</p>
    <p>Try experimenting by removing styles to see how the cascading stylesheets
work (try removing the inline CSS first, then the internal, then the external).</p>
  </body>
</html>
```

Multiple Styles Will Cascade into One

Here, the background color of the page is set with inline CSS, and also with an internal CSS, and also with an external CSS.

Try experimenting by removing styles to see how the cascading stylesheets work (try removing the inline CSS first, then the internal, then the external).

- Part IV -
อื่นๆ

บทที่ 9
เครื่องมือและ
เทคนิคช่วยสร้างเว็บ



9.1. ทำหน้า Web Apps ด้วย Bootstrap

Bootstrap 4 Tutorial

<https://www.w3schools.com/bootstrap4/>

การสร้างเว็บมีอะไรให้ต้องทำมากมาย โดยเฉพาะการออกแบบหน้าตาและลูกเล่นต่างๆ ซึ่งต้องใช้ทั้ง HTML, CSS และ Javascript

Bootstrap เป็น โครงสร้างสำหรับการสร้างเว็บไซต์(Framework) ที่ทำให้การสร้างเว็บง่ายขึ้น เพราะ Bootstrap ทำโครง HTML, CSS และ Javascript มาให้แล้ว หากเราจะสร้างเว็บของเราเอง ก็เพียงสร้าง HTML และตั้งค่าต่างๆตามแนวทางของ Bootstrap

ขอไม่อธิบายรายละเอียดของ Bootstrap ในหนังสือเล่มนี้ เพราะมีเนื้อหามากมายบนเว็บ ทั้งที่เป็นภาษาไทยและอังกฤษ

ตัวอย่างนี้ ใช้งาน Bootstrap แบบ CDN (Content Delivery Network) ก็คือ ลิงค์ไปยังไฟล์ของ CSS และ Javascript ของ Bootstrap โดยจะวางลิงค์ต่างๆไว้ที่แท็ก `<head>` จากนั้น ก็ตั้งค่าการใช้งานตามคำแนะนำของ Bootstrap ถ้าสังเกตจากตัวอย่าง จะพบว่าแอททริบิวต์ `class` คือตัวตั้งค่าสำคัญ

เริ่มต้นสร้างเว็บด้วย Bootstrap เราต้องวางเลเอาท์ของเว็บก่อน จะมีกี่คอลัมน์กี่แถว ขนาดของคอลัมน์ประมาณเท่าไร

โค้ดของไฟล์ html ตามตัวอย่างถัดไป วางเลเอาท์ไว้ดังนี้

```
<!-- ส่วนหัวของเว็บ -->
<div class="jumbotron text-center">
  ...
</div>

<!-- ส่วนเนื้อหาของเว็บ -->
<div class="container-sm">                                <!-- ห่อหุ้มนอกสุด -->
  <div class="row border">                                <!-- ห่อหุ้มแถว -->
    <div class="col-sm-12 p-3">                            <!-- 1 -->
      ...
    </div>
    ...
    <div class="col-sm-12 p-3">                            <!-- 2 -->
      ...
    </div>
    ...
    <div class="col-sm-12 p-3">                            <!-- 3 -->
      ...
    </div>
  </div>                                <!-- ห่อหุ้มแถว -->
</div>                                <!-- ห่อหุ้มนอกสุด -->
```

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- ลิ้งค์ไปยัง CSS และ Javascript ของ Bootstrap4 -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"></script>

    <title>ข้อมูลผู้ใช้งาน</title>
  </head>
  <body>

    <div class="jumbotron text-center">                                <!-- ส่วนหัวของเว็บ -->
      <h1>โปรดกรอกข้อมูลของท่าน</h1>
      <p>(พัฒนาฟอร์มโดย วสันต์ คุณติลกเสวต)</p>
    </div>

    <div class="container-sm">                                         <!-- ห่อหุ้มนอกสุด -->
      <div class="row border">                                         <!-- ห่อหุ้มแถว -->

        <div class="col-sm-12 p-3">                                    <!-- 1 : First name-->
          <label class="control-label text-primary">ชื่อ :</label>&nbsp;
          <input class="form-control form-inline" type="text" id="fn">
        </div>                                                        <!-- 1 -->

        <div class="col-sm-12 p-3">                                    <!-- 2 : Last name -->
          <label class="control-label text-primary">นามสกุล :</label>&nbsp;
          <input class="form-control form-inline" type="text" id="ln">
        </div>                                                        <!-- 2 -->

        <div class="col-sm-12 p-3">                                    <!-- 3 : Department -->
          <label class="control-label text-primary">แผนก : </label>
          <select class="form-control" id="department">
            <option>Accounting</option>
            <option>Marketing</option>
            <option>Design</option>
            <option>Construction</option>
            <option>Human resource</option>
            <option>Factory</option>
            <option selected>IT</option>
          </select>                                                    <!-- 3 -->
        </div>

        <div class="col-sm-12 p-3">                                    <!-- 4 : ปุ่ม Save-->
          <button class="btn btn-primary btn-block" id="btn">บันทึก</button>
        </div>                                                        <!-- 4 -->

      </div>                                                         <!-- ห่อหุ้มแถว -->
    </div>                                                         <!-- ห่อหุ้มนอกสุด -->
  </body>
</html>
```

โปรดกรอกข้อมูลของท่าน

(พัฒนาฟอร์มโดย วสันต์ คุณดิลกเสวต)

ชื่อ :

นามสกุล :

แผนก :

บันทึก

อีกดีไซน์หนึ่ง (ไม่ได้แสดงโค้ดไว้)

โปรดกรอกข้อมูลของท่าน

(พัฒนาฟอร์มโดย วสันต์ คุณดิลกเสวต)

ชื่อ :

นามสกุล :

แผนก :

บันทึก

หมายเหตุ :

นอกจาก Bootstrap แล้ว อีกเจ้าหนึ่งที่นิยมใช้กันก็คือ Materialize

ดูรายละเอียดได้ที่เว็บ <https://materializecss.com/>

9.2. Bootstrap form

Bootstrap 4 Forms

https://www.w3schools.com/bootstrap4/bootstrap_forms.asp

9.2.ก.) Stacked Form (.form-control และ .form-group)

คำปรียายตั้งต้นของ Form elements เช่น อีเล็มเมนต์ `<input>`, `<textarea>` และ `<select>` ก็คือ คลาส `.form-control` ซึ่งมีความกว้าง 100% คลาส `.form-group` เป็นตัวห่อหุ้มรอบอีเล็มเมนต์แต่ละตัว เพื่อให้แน่ใจว่าจะมีระยะ Margin ที่เหมาะสม หน้าตาของฟอร์มที่ออกมาจะมีลักษณะเรียงกันเป็นชั้นๆ

ตัวอย่าง

```
<div class="container">
  <form action="/action_page.php">
    <div class="form-group">                                <!-- Email: -->
      <label for="email">Email:</label>
      <input type="email" class="form-control" id="email" placeholder="Enter email" name="email">
    </div>
    <div class="form-group">                                <!-- Password: -->
      <label for="pwd">Password:</label>
      <input type="password" class="form-control" id="pwd" placeholder="Enter password" name="pswd">
    </div>
    <div class="form-group form-check">                      <!-- Checkbox -->
      <label class="form-check-label">
        <input class="form-check-input" type="checkbox" name="remember">Remember me
      </label>
    </div>
    <button type="submit" class="btn btn-primary">Submit</button>
  </form>
</div>
```

ผล

Email:

Password:

☐ Remember me

9.2.ข.) Inline Form (.form-inline)

Inline Form ฟิลด์จะเรียงอยู่ในแถวเดียวกัน โดยหน้าจอสองแสดงผลจะต้องไม่น้อยกว่า 576px ถ้าน้อยกว่านี้จะเห็นฟิลด์เรียงเป็นชั้นกันลงมา (Stacked Form)

เราสามารถทำ Inline Form ได้โดยใช้คลาส **.form-inline** กับอิลีเมนต์ **<form>**

```
<div class="container">
  <form class="form-inline" action="/action_page.php">
    <!-- Email: -->
    <label for="email">Email:</label>
    <input type="email" class="form-control" id="email"
      placeholder="Enter email" name="email">

    <!-- Password: -->
    <label for="pwd">Password:</label>
    <input type="password" class="form-control" id="pwd"
      placeholder="Enter password" name="pwd">

    <!-- Checkbox -->
    <div class="form-check">
      <label class="form-check-label">
        <input class="form-check-input" type="checkbox" name="remember">
        Remember me
      </label>
    </div>

    <button type="submit" class="btn btn-primary">Submit</button>
  </form>
</div>
```

ผล

Email: Password: ☐ Remember me

9.2.ค.) Inline Form with Utilities (.mr-sm-2 และ .mb-2)

Bootstrap 4 Utilities

https://www.w3schools.com/bootstrap4/bootstrap_utilities.asp

Inline Form ในข้อก่อนหน้าดูแน่นไปหน่อย จะดีกว่านี้ถ้าเราเพิ่มช่องว่างเข้าไปด้วย ให้ใช้คลาส **.mr-sm-2** สำหรับแต่ละอิลีเมนต์ (สำหรับอุปกรณ์จอเล็กขึ้นไป) และเพิ่มระยะ Margin ด้านล่างโดยใช้คลาส **.mb-2** กรณีหน้าจามีขนาดแคบเกินไป จะถูกตัดเป็นชั้นๆ กลายเป็น Stacked form

ตัวอย่าง

```
<div class="container">

  <form class="form-inline" action="/action_page.php">

    <!-- Email: -->
    <label for="email2" class="mb-2 mr-sm-2">Email:</label>
    <input type="text" class="form-control mb-2 mr-sm-2" id="email2"
      placeholder="Enter email" name="email">

    <!-- Password: -->
    <label for="pwd2" class="mb-2 mr-sm-2">Password:</label>
    <input type="text" class="form-control mb-2 mr-sm-2" id="pwd2"
      placeholder="Enter password" name="pswd">

    <!-- Checkbox -->
    <div class="form-check mb-2 mr-sm-2">
      <label class="form-check-label">
        <input type="checkbox" class="form-check-input" name="remember">
        Remember me
      </label>
    </div>

    <button type="submit" class="btn btn-primary mb-2">Submit</button>

  </form>
</div>
```

ผล - ฟอรมมีช่องระหว่างฟิลด์ ดูดีขึ้น

Email:	<input type="text" value="Enter email"/>	Password:	<input type="text" value="Enter password"/>	<input type="checkbox"/> Remember me	<input type="submit" value="Submit"/>
--------	------------------------------------------	-----------	---------------------------------------------	--------------------------------------	---------------------------------------

เมื่อนำจอแคบ จะตัดขึ้นแถวใหม่ ให้ผลเหมือนเป็น Stacked form เหมาะกับทุกหน้าจอ

Email:
<input type="text" value="Enter email"/>
Password:
<input type="text" value="Enter password"/>
<input type="checkbox"/> Remember me
<input type="submit" value="Submit"/>

9.2.ง.) Form Row/Grid (.col และ .row)

Bootstrap 4 Grid System

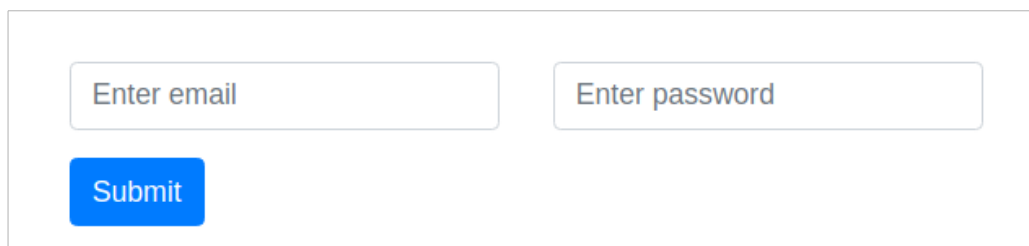
https://www.w3schools.com/bootstrap4/bootstrap_grid_system.asp

เราสามารถใช้คลาส **.col** ควบคุมความกว้างและการเรียงฟิลด์ในฟอร์ม โดยไม่จำเป็นต้องใช้เครื่องมือ Space utilities อย่าง **.mr-sm-2** หรือ **.mb-2** เพียงแต่ต้องจำไว้ว่า ต้องวางอิลีเมนต์ไว้ในอิลีเมนต์ที่ใช้คลาส **.row** ซึ่งเป็นอิลีเมนต์บรรทัด

ตัวอย่างที่ 1 – ตัวอย่างนี้มีฟิลด์ในฟอร์ม แบ่งเป็น 2 คอลัมน์ ไม่ว่าขนาดหน้าจะเป็นเท่าไรก็ตาม

```
<div class="container">
  <form action="/action_page.php">
    <div class="row">
      <!-- Email -->
      <div class="col">
        <input type="text" class="form-control" id="email"
          placeholder="Enter email" name="email">
      </div>
      <!-- Password -->
      <div class="col">
        <input type="password" class="form-control"
          placeholder="Enter password" name="pswd">
      </div>
    </div>
    <button type="submit" class="btn btn-primary mt-3">Submit</button>
  </form>
</div>
```

ผล



ถ้าต้องการกริดที่มี Margin น้อยๆ (เขียนทับค่าปรีายายช่องว่างระหว่างคอลัมน์) ให้ใช้คลาส

.form-row แทนคลาส **.row**

ตัวอย่างที่ 2

```
<div class="container"> <!-- Container -->

  <p>Create two form elements that appear side by side with .row and .col:</p>

  <form>                                <!-- Form #1 -->
    <div class="row">
      <div class="col">                  <!-- Email -->
        <input type="text" class="form-control" id="email"
          placeholder="Enter email" name="email">
      </div>
      <div class="col">                  <!-- Password -->
        <input type="password" class="form-control"
          placeholder="Enter password" name="pswd">
      </div>
    </div>
  </form><br>

  <p>Create two form elements that appear side by side with .form-row and .col:</p>

  <form>                                <!-- Form #2 -->
    <div class="form-row">
      <div class="col">                  <!-- Email -->
        <input type="text" class="form-control" id="email"
          placeholder="Enter email" name="email">
      </div>
      <div class="col">                  <!-- Password -->
        <input type="password" class="form-control"
          placeholder="Enter password" name="pswd">
      </div>
    </div>
  </form>
</div> <!-- Container -->
```

ผล - ฟอรม์ก๊อคนที่ 2 มีระยะระหว่างคอลัมน์น้อยกว่า ฟอรม์ก๊อแรก

Create two form elements that appear side by side with .row and .col:

Create two form elements that appear side by side with .form-row and .col:

9.2.จ.) Form validation (.was-validated และ .needs-validation)

เราสามารถ useClass สำหรับทำ Validation ที่แตกต่างกันเพื่อความเหมาะสม โดยใช้คลาส

.was-validated หรือ **.needs-validation** กับอ็ลเมนต์ **<form>** ขึ้นอยู่กับว่าจะทำ Validation ก่อนหรือหลัง Submit ฟอรั่ม

ถ้าฟิลด์ถูกกรอกข้อมูล ที่เหมาะสมแล้ว จะเป็นสีเขียว แต่ถ้าไม่ใช่ จะเป็นสีแดง

นอกจากนี้ เรายังสามารถใช้คลาส **.valid-feedback** หรือ **.invalid-feedback** แจ้งข้อมูลกับยูสเซอร์ว่าสิ่งที่ผิดคืออะไร

ตัวอย่างที่ 1

ตัวอย่างนี้ใช้ **.was-validated** ผลก็คือ ทำ Validation ก่อนคลิกที่ปุ่ม Submit

```
<div class="container">
  <form action="/action_page.php" class="was-validated">
    <div class="form-group">
      <label for="uname">Username:</label>
      <input type="text" class="form-control" id="uname"
        placeholder="Enter username" name="uname" required>
      <div class="valid-feedback">Valid.</div>
      <div class="invalid-feedback">Please fill out this field.</div>
    </div>

    <div class="form-group">
      <label for="pwd">Password:</label>
      <input type="password" class="form-control" id="pwd"
        placeholder="Enter password" name="pswd" required>
      <div class="valid-feedback">Valid.</div>
      <div class="invalid-feedback">Please fill out this field.</div>
    </div>

    <div class="form-group form-check">
      <label class="form-check-label">
        <input class="form-check-input" type="checkbox"
          name="remember" required> I agree on blabla.
        <div class="valid-feedback">Valid.</div>
        <div class="invalid-feedback">Check this checkbox to continue.</div>
      </label>
    </div>

    <button type="submit" class="btn btn-primary">Submit</button>
  </form>
</div>
```

ผล – ถ้ากรอกได้เหมาะสม **กรอบของฟิลด์จะเป็นสีเขียว** และมีข้อความ **Valid** แจ้ง(กำหนดได้) แต่ถ้าไม่เหมาะสม **กรอบฟิลด์จะเป็นสีแดง** และจะมีข้อความสีแดงแจ้งเช่นเดียวกัน(กำหนดได้) ถ้าเราคลิกปุ่ม **Submit** โดยที่ยังกรอกฟิลด์ไม่สมบูรณ์ จะปรากฏกรอบแจ้งเตือน(กำหนดข้อความได้)

The screenshot shows a web form with two input fields. The first field, labeled 'Username:', contains the text 'wasan' and has a green border with a green checkmark icon on the right, indicating it is valid. Below this field, the word 'Valid.' is displayed in green. The second field, labeled 'Password:', contains the placeholder text 'Enter password' and has a red border with a red exclamation mark icon on the right, indicating it is invalid. Below this field, the text 'Please fill out this field.' is displayed in red. A tooltip with a yellow exclamation mark icon and the text 'Please fill out this field.' is shown next to the password field. Below the password field, there is a checkbox labeled 'I agree on blabla.' with the text 'Check this checkbox to continue.' below it. At the bottom of the form is a blue 'Submit' button.

ตัวอย่างที่ 2

ตัวอย่างนี้ใช้ **.needs-validation** ซึ่งจะเพิ่มเอเพ็กต์หลังจาก **Submit** ฟอर्मแล้ว (ถ้ามีอะไรขาดไป) โค้ด HTML เหมือนกับตัวอย่างก่อนหน้านี้ เพียงแต่เปลี่ยนแอตทริบิวต์ **<from>** ใช้คลาสต่างกัน และมีสคริปต์ **jQuery** เพิ่มเข้ามา

```
<div class="container">
  <form action="/action_page.php" class="needs-validation" novalidate>
    <div class="form-group">                                <!-- Username: -->
      ...
    </div>

    <div class="form-group">                                <!-- Password: -->
      ...
    </div>

    <div class="form-group form-check">                    <!-- I agree on blabla. -->
      ...
    </div>

    <button type="submit" class="btn btn-primary">Submit</button>

  </form>
</div>

<!-- มีต่อ -->
```

```

<!-- ต่อ -->
<script>
// Disable form submissions if there are invalid fields
(function() {
  'use strict' ;
  window.addEventListener('load', function() {

    // Get the forms we want to add validation styles to
    var forms = document.getElementsByClassName('needs-validation') ;

    // Loop over them and prevent submission
    var validation = Array.prototype.filter.call(forms, function(form) {
      form.addEventListener('submit', function(event) {
        if (form.checkValidity() === false) {
          event.preventDefault() ;
          event.stopPropagation() ;
        }
        form.classList.add('was-validated') ;
      }, false) ;
    }, false) ;
  })() ;
})() ;
</script>

```

ผล - เมื่อกรอกแบบฟอร์ม ฟิลด์จะยังไม่ถูกทำ Validation แต่จะทำหลังจากคลิกที่ปุ่ม Submit


Username:

Password:

☒ I agree on blabla.


หลังคลิกปุ่ม Submit ฟิลด์ถูก Validation

Username:

Valid.

Password:

Please fill out this field.

☒ I agree on blabla.

Valid.

9.3. ใช้ฟอนต์ของ Google

Google Fonts

<https://fonts.google.com/>

CSS API update

<https://developers.google.com/fonts/docs/css2>

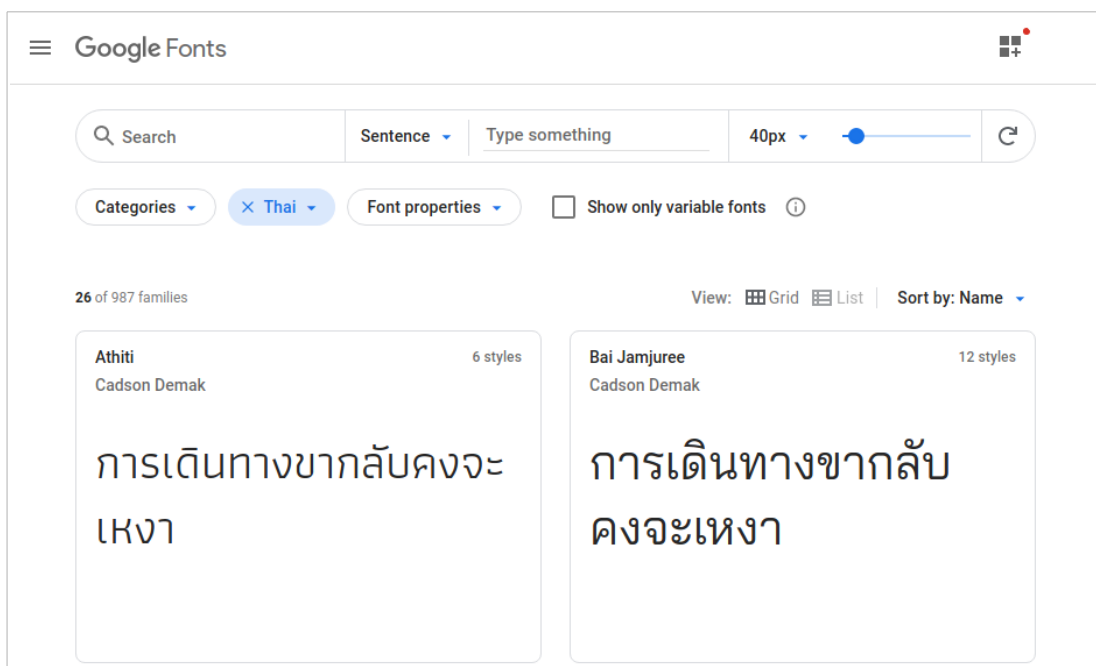
การใช้ฟอนต์ในเว็บแบบกำหนดเอง ถ้าเราจะเขียน CSS และ HTML เพื่อแนบฟอนต์ที่เราต้องการ เป็นเรื่องยุ่งยาก ผู้เขียนนานๆทำทีบางทีก็ลืม และรู้สึกว่ามันยุ่งยาก และก็ไม่ถนัดที่จะมาตีไซส์หน้าตาของ Web App สักเท่าไร เพราะเน้นไปที่การใช้งาน แต่อย่างไรก็ดี หน้าตาของเว็บนั้นก็สำคัญ โดยเฉพาะฟอนต์

มีวิธีง่ายๆในการนำฟอนต์สวยๆ มาใช้ในเว็บของเรา ก็คือ ใช้ฟอนต์ของ Google เพราะการนำมาใช้นั้นง่ายมาก

9.3.ก.) เลือกฟอนต์จากเว็บ fonts.google.com

ไปที่เว็บ <https://fonts.google.com/>


จากนั้นค้นหาฟอนต์ที่ต้องการ ตัวอย่าง ผู้เขียนค้นหาฟอนต์ที่เป็นภาษาไทย (ตามภาพ)

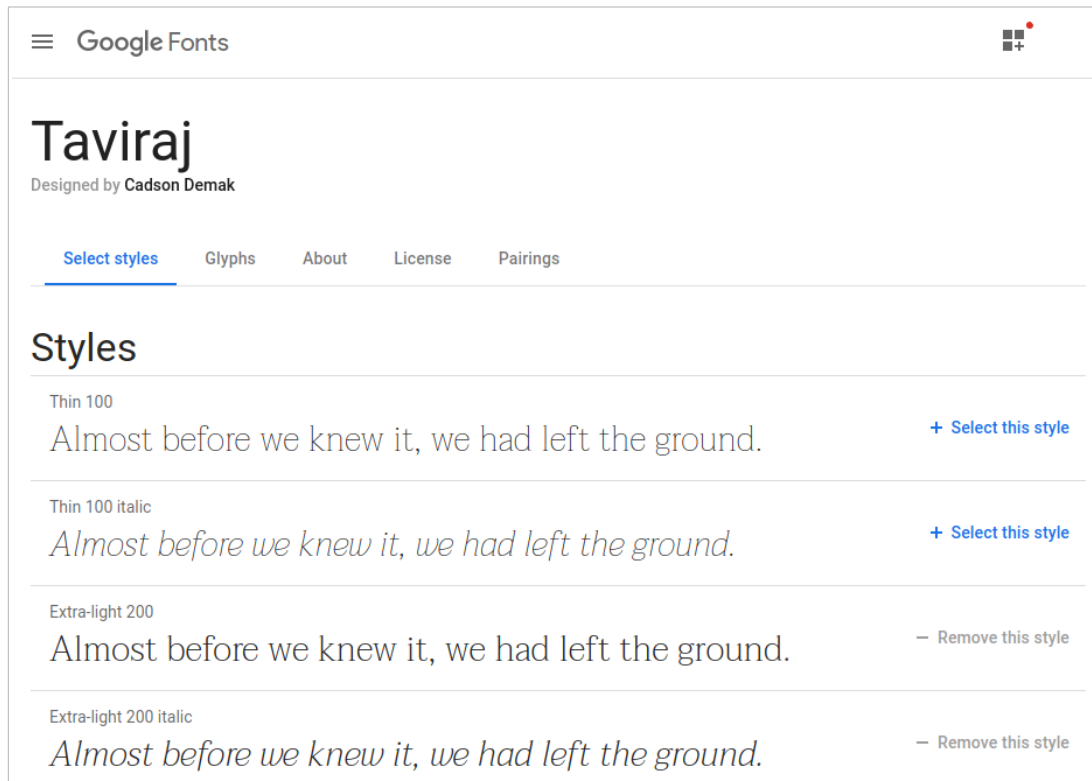


ถ้าเจอฟอนต์ที่พอใจแล้ว ให้คลิกเลือกฟอนต์ จะปรากฏหน้าต่างแสดงรายละเอียดของฟอนต์ตามภาพถัดไป ซึ่งเป็นสไลด์ต่างๆของฟอนต์ตัวนั้น เช่น ความหนา ความเอียง เป็นต้น

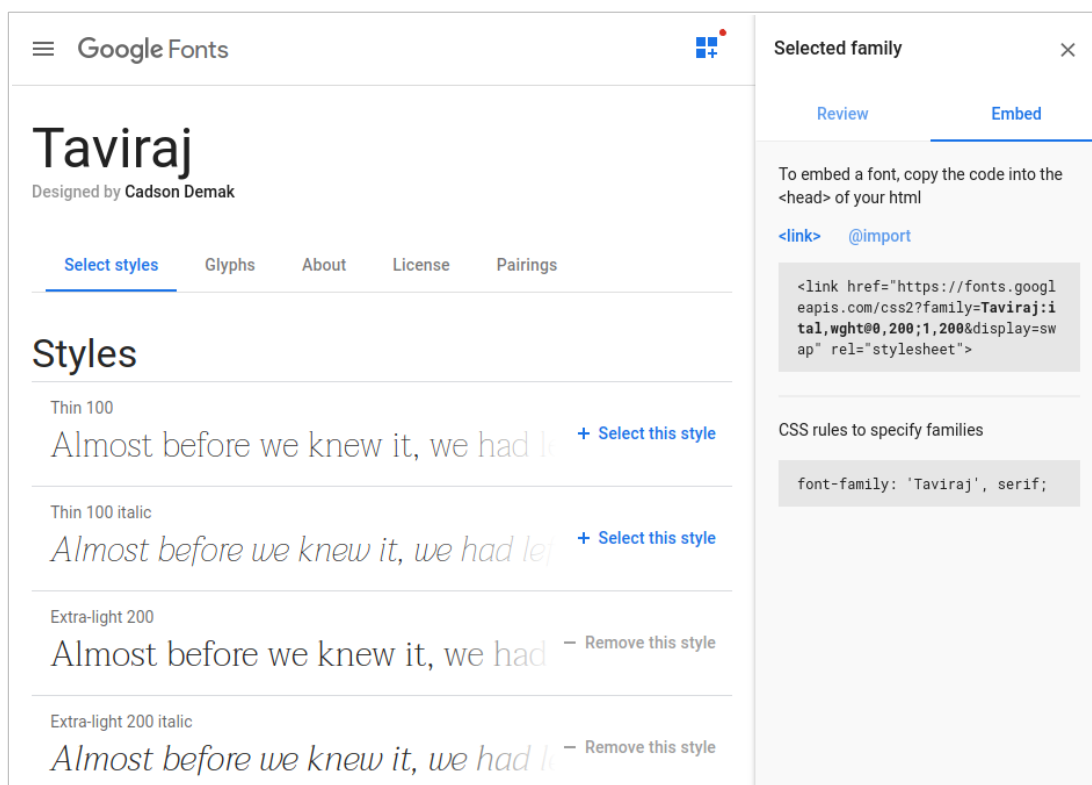
(ตามภาพถัดไป) ฟอนต์ที่ผู้เขียนเลือก ก็คือ Taviraj

เมื่อเจอสไตล์ของฟอนต์ที่พอใจแล้ว ให้คลิกที่ **+Select this style** เพื่อเลือกสไตล์และฟอนต์ดังกล่าว สามารถเลือกได้หลายตัว

จากนั้น คลิกปุ่ม  (View you selected families) ที่มุมขวาบนสุด จะปรากฏกรอบ **Selected family** ตามภาพถัดไป



(ตามภาพถัดไป) ให้ดูที่แท็บ **Embed** กรอบนี้มีข้อมูลการนำฟอนต์ไปใช้ในเว็บของเรา



ลิงค์ไปยังฟอนต์

```
<link href="https://fonts.googleapis.com/css2?
family=Taviraj:ital,wght@0,200;1,200&display=swap" rel="stylesheet">
```

และ CSS สำหรับ Font families

```
font-family: 'Taviraj', serif ;
```

9.3.ข.) นำฟอนต์ของ Google ไปใช้ในเว็บ

ที่ไฟล์ html ให้นำโค้ดที่ได้ไปใส่ โดยใส่ไว้ภายในแท็ก <head> ดังตัวอย่างต่อไปนี้

ส่วนของลิงค์วางไว้ภายในแท็ก <head>

CSS ที่ได้มาเป็น Font families ให้เขียน CSS วางไว้ภายในแท็ก <style> เพื่อกำหนดว่าอิลีเมนต์ใดจะใช้ Font families ตัวนั้นบ้าง

ไฟล์ .html

```
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">

<!-- ลิงค์ไปยัง CSS และ Javascript ของ Bootstrap4 -->
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"></script>

<!-- ลิงค์ไปยังฟอนต์ของ Google -->
<link href="https://fonts.googleapis.com/css2?family=Taviraj:ital,wght@0,300;1,300&display=swap" rel="stylesheet">

<style>
// กำหนด Font families ให้กับอิลีเมนต์ต่างๆ
// หรือใช้ * { font-family: 'Taviraj', serif ; } ที่เดียวจบเลยก็ได้
body {
    font-family: 'Taviraj', serif ;
    /* font-size: 20px; */
}

p {
    font-family: 'Taviraj', serif ;
    /* font-size: 18px; */
}

h1,h2,h3,h4,h5,h6 {
    font-family: 'Taviraj', serif ;
}
</style>

<title>ข้อมูลผู้ใช้งาน</title>
</head>
<!-- อื่นๆ -->
```

โปรดกรอกข้อมูลของท่าน

(พัฒนาฟอร์มโดย วสันต์ คุณดิลกเสวต)

ชื่อ :

นามสกุล :

แผนก :

IT

บันทึก

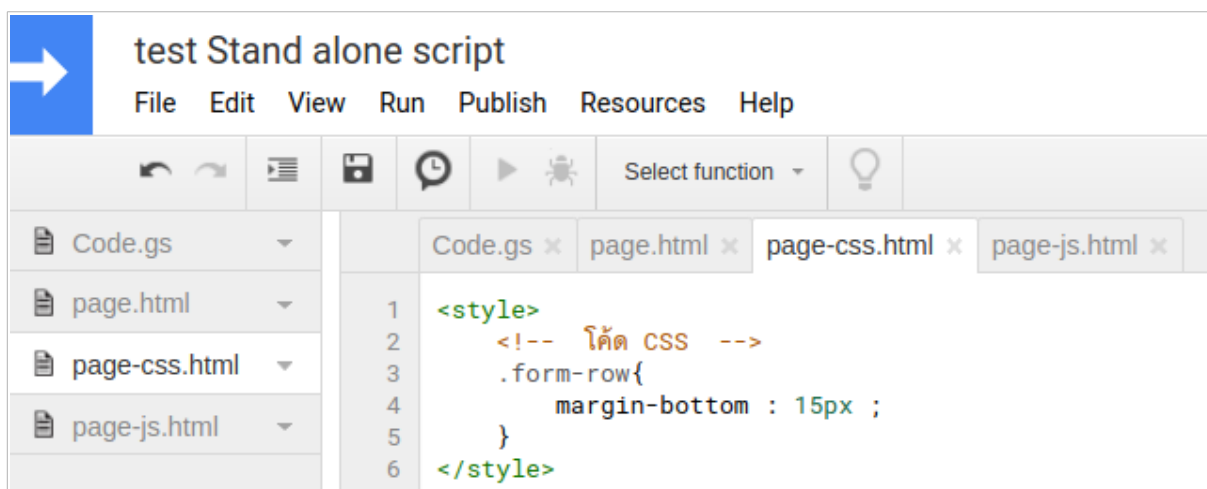
9.4. แยกไฟล์สำหรับ CSS, Javascript แต่อยู่ในโปรเจ็ค

Template Partial & Passing Objects - Google Apps Script Web App Tutorial - Part 2
<https://www.youtube.com/watch?v=1toLqGwMRVc>

โปรเจ็ค Google Apps Script ที่ใช้งานแบบ Web App อย่างน้อยที่สุดต้องมี 2 ไฟล์ ก็คือ ไฟล์ .gs และ .html

กรณีไฟล์ .gs เราจะสร้างก็ไฟล์ก็ได้ เขียนโค้ด Google Apps Script ลงในไฟล์ไหนก็ได้ ไม่มีปัญหา

แต่... ถ้าเราจะแยกเนื้อหาในไฟล์ html ไปไว้ในไฟล์อื่น เช่น แยก CSS หรือ Javascript ไปไว้ในไฟล์ html อีกไฟล์ ตัวอย่างตามภาพ เราต้องใช้เทคนิคกันหน่อย



9.4.ก.) ขั้นตอนที่ 1 : แยกไฟล์ แยกโค้ด HTML

ที่ไฟล์ .html ไฟล์ที่ 2 (ไฟล์ page-css.html) เขียนโค้ดดังนี้ สำหรับเก็บโค้ด CSS โดยเฉพาะ
สังเกตว่ายกแท็ก <style> มาทั้งยวง

```
<style>
<!-- โค้ด CSS -->
.form-row{
    margin-bottom : 15px ;
}
</style>
```

ที่ไฟล์ .html ไฟล์ที่ 3 (ไฟล์ page-js.html) สำหรับเก็บโค้ด Javascript สังเกตว่ายกแท็ก <script>
มาทั้งยวง

```
<script>

// โค้ด Javascript – พิมพ์ข้อความง่ายๆ ให้กับอิลีเมนต์ demo
//
document.getElementById("demo").innerHTML = "This text is from Javascript" ;

</script>
```

9.4.ข.) ขั้นตอนที่ 2 : ผัง Scriptlets เพื่อดึงไฟล์ html มารวมในไฟล์หลัก

ย้อนกลับไปไฟล์ .html ไฟล์ที่ 1 (ไฟล์ page.html) ที่เป็นไฟล์หลัก ให้ฝัง Scriptlets แบบ Force-
printing ก็คือ <?!= // Apps Script Code ?> เพื่อรันฟังก์ชัน include() ที่อยู่ใน ไฟล์ .gs ซึ่งฟังก์ชันนี้จะจับ
โค้ด HTML ในไฟล์อื่น มารวมกับไฟล์หลัก

```
<html>
<head>
    <base target="_top">

    <!-- รันฟังก์ชัน include ที่อยู่ในไฟล์ .gs โดยส่งพารามิเตอร์ชื่อไฟล์ไป -->
    <!-- แท็ก <style> ต้องอยู่ใน <head> -->
    <?!= include("page-css") ; ?>

</head>

<body>

    <h1>Test</h1>
    <p id="demo"></p>

    <!-- รันฟังก์ชัน include ที่อยู่ในไฟล์ .gs โดยส่งพารามิเตอร์ชื่อไฟล์ไป -->
    <!-- แท็ก <script> อยู่บนหรือล่าง แล้วแต่ว่า ต้องการให้รันตอนไหน แต่มักอยู่ล่าง -->
    <?!= include("page-js"); ?>

</body>
</html>
```

9.4.ค.) ขั้นตอนที่ 3 : สร้างฟังก์ชันในไฟล์ .gs

ในไฟล์ .gs สร้างฟังก์ชัน include()

```
function doGet() {  
    // ทดสอบ Logs ดูโค้ด HTML ทั้งหมดที่รวมกันแล้ว  
    // ดูผลที่ Logs ----- > [ 01 ]  
    //  
    Logger.log(HtmlService.createTemplateFromFile("page").evaluate().getContent());  
  
    // คืค่าเป็น HTML ที่สร้างจากไฟล์ page.html  
    // เนื่องจากไฟล์ page.html มี Scriptlets จึงต้องรัน Scriptlets เพื่อแปลงเป็นโค้ด HTML ก่อน  
    // ในที่นี้ จะรันฟังก์ชัน include() เพื่อดึง HTML จากไฟล์ต่างๆมารวมกัน  
    //  
    return HtmlService.createTemplateFromFile("page").evaluate()  
}  
  
// ฟังก์ชันนี้ไม่ได้มีใช้งานอะไร  
// เพียงแต่ทดสอบดูว่า include() คืค่าเป็นอะไรกลับมา  
//  
function abc() {  
    Logger.log(include('page-css')); // ดูผลที่ Logs ----- > [ 02 ]  
}  
  
//  
// พารามิเตอร์ filename เป็นชื่อไฟล์ html เช่น page-css หรือ page-js  
  
// ฟังก์ชัน include คืค่าเป็น โค้ด HTML ในไฟล์ html ที่ระบุ  
// ผล → เอาโค้ดในไฟล์ page-css หรือ page-js มาแปะลงไปในไฟล์ html ไฟล์หลัก  
//  
function include(filename) {  
    return HtmlService.createHtmlOutputFromFile(filename).getContent();  
}
```

เท่านี้ก็ใช้งานได้แล้ว

เมื่อ Deploy as web app จะได้ผลลัพธ์ที่หน้า Web App ตามภาพ

Test

This text is from Javascript

Logs จากฟังก์ชัน doGet() เป็นโค้ด HTML ที่รวมมาจากไฟล์ html ทั้ง 3 ไฟล์

Logs

```
[ 01 ] <!DOCTYPE html>
      <html>
      <head>
      <base target="_top">
      <style>
      .form-row{
      margin-bottom : 15px ;
      }
      </style>
      </head>
      <body>
      <h1>Test</h1>
      <p id="demo"></p>
      <script>
      document.getElementById("demo").innerHTML = "This text is form Javascript" ;
      </script>
      </body>
      </html>
```

Logs จากฟังก์ชัน abc() ที่เรียกใช้ฟังก์ชัน include() เพื่อทดสอบดูว่า include() คืนค่าเป็นอะไร

Logs

```
[ 02 ] <style>
      <!-- โค้ด CSS -->
      .form-row{
      margin-bottom : 15px ;
      }
      </style>
```


บทที่ 10

Blob, File และ FileReader



10.1. วัตถุ File และ วัตถุ FileList

10.1.ก.) จับไฟล์จาก <input>

File

<https://developer.mozilla.org/en-US/docs/Web/API/File>

File.File()

<https://developer.mozilla.org/en-US/docs/Web/API/File/File>

วัตถุ File โดยปกติจับมาได้จากวัตถุ FileList ที่คืนกลับมาจากการเลือกไฟล์ โดยใช้ไอเล็มเมนต์ <input type="file">

วัตถุ File เป็น Blob แบบเจาะจง และสามารถถูกใช้ในแบบที่ Blob สามารถทำได้ โดยเฉพาะอย่างยิ่ง FileReader, URL.createObjectURL(), createImageBitmap() และ XMLHttpRequest.send() รับทั้ง Blobs และ Files

FileList

<https://developer.mozilla.org/en-US/docs/Web/API/FileList>

วัตถุ FileList คืนมาจากคุณสมบัติ files ของไอเล็มเมนต์ <input>

คุณสมบัติ files จะช่วยให้เราเข้าถึงไฟล์ต่างๆที่ถูกเลือกด้วยไอเล็มเมนต์ <input type="file">

การใช้งาน

```
// HTML
<input id="fileItem" type="file">
```

```
// Javascript - จับไฟล์แรก
var file = document.getElementById('fileItem').files[0] ;
```

```
// Javascript - จับไฟล์ทั้งก้อนมาเป็นวัตถุ FileList
var filesInput = document.getElementById('fileItem').files ;
```

อีกวิธี จับไฟล์จาก Event ก็ได้ ดังตัวอย่างต่อไปนี้

```
<input type='file' onchange='onFilesSelected(event)' multiple>
<script>
  var onFilesSelected = function(event) {
    var input = event.target ; // ไฟล์ถูกส่งผ่านมากับ Event - อันนี้แปลกติ
    for (var i = 0; i < input.files.length; i++) {
      console.log(input.files[i].name) ; // Logs ชื่อไฟล์ที่เลือกทั้งหมด
    } // for
  } ;
</script>
```

10.1.ข.) สร้างวัตถุ File

File and FileReader
<https://JavaScript.info/file>

File : Blob
<https://www.Javascripture.com/File>

Constructor ของ File มีลักษณะคล้ายกับ Blob ก็คือ

```
new File(fileParts, fileName, [options])
```

fileParts : อาร์เรย์ของ Blob/BufferSource/String

fileName : ชื่อไฟล์

options : (optional)

- **lastModified** : วันที่ของการแก้ไขล่าสุด

File เป็นวัตถุที่สืบทอดมาจากวัตถุ Blob ดังนั้นวัตถุ File จึงมีคุณสมบัติเหมือน Blob แต่จะเพิ่มคุณสมบัติต่อไปนี้เข้าไป ก็คือ **name** และ **lastModified**

บ่อยๆที่เราจะได้รับไฟล์จาก `<input type="file">` , การลากวาง หรือ จาก Interfaces อื่นๆของ Browser

ตัวอย่างต่อไปนี้ รับไฟล์จาก `<input type="file">` จากนั้นหน้าต่าง Alert จะแสดงชื่อไฟล์และการแก้ไขล่าสุดขึ้นมา

```
<input type="file" onchange="showFile(this)">
<script>
// มีอะไรอยู่ใน Input มากมาย ดูใน console ของ Firefox จะเห็นหมด
function showFile(input) {
    var file = input.files[0] ; // จับไฟล์แรกเท่านั้น
    alert('File name:\f' + file.name + '\nLast modified:\f' + file.lastModified) ;
}
</script>
```

10.2. จับคุณสมบัติจากไฟล์ที่เลือก

`<input type="file">`
<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/file>

ไฟล์ที่ถูกเลือก คืนค่ากลับมาเป็นคุณสมบัติ `HTMLInputElement.files` ซึ่งเป็นวัตถุ **FileList** ที่บรรจุไปด้วยวัตถุ **File** ต่างๆ วัตถุ **FileList** มีพฤติกรรมคล้ายกับอาร์เรย์ เราจึงสามารถนับจำนวนไฟล์ได้โดยใช้คุณสมบัติ `length`

วัตถุ **File** ที่บรรจุอยู่ใน **FileList** ประกอบไปด้วยข้อมูลดังต่อไปนี้

name : ชื่อไฟล์

lastModified : ตัวเลขระบุวันที่และเวลาที่แก้ไขไฟล์ล่าสุด ในหน่วยมิลลิวินาที โดยนับเริ่มต้นจาก 1 มกราคม 1970 เวลาเที่ยงคืน

lastModifiedDate : วัตถุเวลา แสดงถึงวันที่และเวลาที่แก้ไขไฟล์ล่าสุด (ยกเลิกการใช้งานไปแล้ว)

size : ขนาดของไฟล์ในหน่วย Bytes.

type : MIME type ของไฟล์

webkitRelativePath : String ระบุพารของไฟล์ ตัวนี้ไม่มาตรฐานควรใช้อย่างระมัดระวัง

ตัวอย่าง – โค้ดต่อไปนี้ Logs รายละเอียดของไฟล์

(**File.name** – <https://developer.mozilla.org/en-US/docs/Web/API/File/name>)

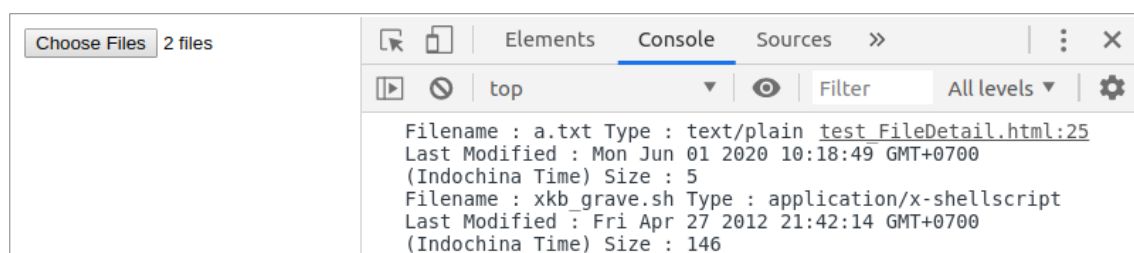
```
<html lang="en">
  <head>
    <meta charset="UTF-8">
  </head>
  <body>
    <input type="file" multiple onchange="processSelectedFiles(this)">
    <script>

      function processSelectedFiles(fileInput) {

        // จับไฟล์จาก input ไว้ในตัวแปรก่อน *****
        var files = fileInput.files ;
        var fileDetails = " " ;

        // วนลูปเข้าไปจับข้อมูลของแต่ละไฟล์
        for (var i = 0 ; i < files.length ; i++) {
          fileDetails += "Filename : " + files[i].name +
            " Type : " + files[i].type +
            " Last Modified : " + files[i].lastModifiedDate +
            " Size : " + files[i].size + '\n' ;
        }
        console.log(fileDetails) ;
      }
    </script>
  </body>
</html>
```

ผล



10.3. ตัวอย่าง

10.3.ก.) เลือกไฟล์ไม่เกินที่กำหนด

File.size

https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5/Constraint_validation

โค้ดต่อไปนี้จะสร้างฟิลด์เลือก File ที่ตรวจสอบขนาดไฟล์ ไม่ให้เกินกำหนด ก็คือ ไม่เกิน 75Kb

```
<html>
<head>
  <script type="text/javascript">

    function checkFileSize() {
      var FS = document.getElementById("FS") ;
      var files = FS.files ;

      if (files.length > 0) { // If there is (at least) one file selected
        if (files[0].size > 75*1024) { // Check the constraint
          FS.setCustomValidity("The selected file must not be larger than 75 kB") ;

          console.log("FS Non valid") ;
          return ;
        }
      }

      console.log("FS valid") ;
      FS.setCustomValidity("") ; // No custom constraint violation
    }

    // Hook ฟังก์ชันเข้ากับ Event
    window.onload = function() {
      document.getElementById("FS").onchange = checkFileSize ;
    }
  </script>
</head>
<body>
  <label for="FS">Select a file smaller than 75 kB : </label>
  <input type="file" id="FS">
</body>
</html>
```

ผล - เมื่อเลือกขนาดไฟล์มากกว่า 75 Kb จะปรากฏกรอบแดง บ่งบอกว่าไม่ OK (ขึ้นเฉพาะใน Firefox ในโค้ดข้างต้นจึงต้องทดสอบ Logs ผลลัพธ์ออกมาด้วย)

Select a file smaller than 75 kB : blender

10.3.ข.) แสดงพรีวิวของไฟล์ภาพที่เลือก

โค้ด

```
<body>
  <form>
    <div>
      <label for="file">Choose file to upload</label>
      <input type="file" id="file" name="file" multiple onchange="handleFiles(this)">
    </div>
    <div id="preview"></div>
    <div><button>Submit</button></div>
  </form>
  <script>
    function handleFiles(inputList) {

      var preview = document.getElementById('preview') ;
      files = inputList.files

      for (let i = 0 ; i < files.length ; i++)
      {
        const file = files[i] ;
        if (!file.type.startsWith('image/')){ continue }
        const img = document.createElement("img") ;
        img.classList.add("obj") ;
        img.file = file ;
        preview.appendChild(img) ;
      } // for
    } // function

  </script>
</body>
```

ผล

Choose file to upload ft_DocsCreator.png



10.3.ค.) แจ้งขนาดไฟล์และแสดงภาพที่เลือก

<input type="file">

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/file>

[https://mdn.mozillademos.org/en-US/docs/Web/HTML/Element/input/file\\$samples/Examples?revision=1616361](https://mdn.mozillademos.org/en-US/docs/Web/HTML/Element/input/file$samples/Examples?revision=1616361)

โค้ด

```
<html>
<head>
  <meta charset="utf-8">
  <link rel = "stylesheet" type = "text/css" href = "form.css" />
</head>
<body>

  <form method="post" enctype="multipart/form-data">
    <div>
      <label for="image_uploads">Choose images to upload (PNG, JPG)</label>
      <input type="file" id="image_uploads" name="image_uploads"
        accept=".jpg, .jpeg, .png" multiple>
    </div>

    <div class="preview">
      <p>No files currently selected for upload</p>
    </div>

    <div>
      <button>Submit</button>
    </div>
  </form>

  <script>
const input = document.querySelector('input');
const preview = document.querySelector('.preview');

input.style.opacity = 0 ;
input.addEventListener('change', updateImageDisplay);

function updateImageDisplay() {
  // ล้างโซน Preview ไปก่อน
  while(preview.firstChild) {
    preview.removeChild(preview.firstChild) ;
  }

  const curFiles = input.files ;

  // ยังไม่ได้เลือกไฟล์ - แสดงข้อความไว้ที่โซนพร็ว
  if(curFiles.length === 0)
  {
    const para = document.createElement('p') ;
    para.textContent = 'No files currently selected for upload' ;
    preview.appendChild(para) ;
  }
}
```

```

// เลือกไฟล์แล้ว – เลือกได้มากกว่า 1
else
{
    const list = document.createElement('ol') ;
    preview.appendChild(list) ;

    for(const file of curFiles)
    {
        const listItem = document.createElement('li') ;
        const para = document.createElement('p') ;

        // เลือกไฟล์แล้ว – และ... ตรงชนิด
        if(validFileType(file))
        {
            para.textContent = `File name ${file.name}, file size ${returnFileSize(file.size)}.` ;
            const image = document.createElement('img') ;
            image.src = URL.createObjectURL(file) ;

            listItem.appendChild(image) ;
            listItem.appendChild(para) ;
        }
        // เลือกไฟล์แล้ว – แต่... ไม่ตรงชนิด
        else
        {
            para.textContent = `File name ${file.name}: Not a valid file type. Update your selection.` ;
            listItem.appendChild(para) ;
        }

        list.appendChild(listItem) ;
    } // for
} // else - เลือกไฟล์แล้ว – เลือกได้มากกว่า 1
} // function // https://developer.mozilla.org/en-US/docs/Web/Media/Formats/Image_types

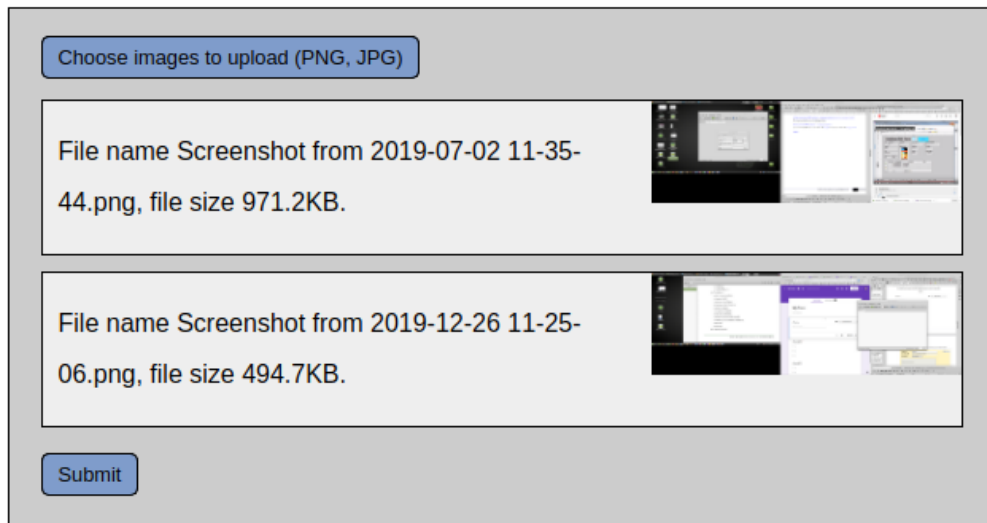
const fileTypes = [ "image/apng", "image/bmp", "image/gif", "image/jpeg",
                    "image/pjpeg", "image/png", "image/svg+xml", "image/tiff",
                    "image/webp", "image/x-icon" ] ;

function validFileType(file) {
    return fileTypes.includes(file.type) ;
}

function returnFileSize(number) {
    if(number < 1024) {
        return number + 'bytes' ;
    } else if(number >= 1024 && number < 1048576) {
        return (number/1024).toFixed(1) + 'KB' ;
    } else if(number >= 1048576) {
        return (number/1048576).toFixed(1) + 'MB' ;
    }
} // Close function

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

```



CSS สำหรับโค้ดข้างต้น

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

form { width: 580px; background: #ccc ;
      margin: 0 auto; padding: 20px ;
      border: 1px solid black ; }

form ol { padding-left: 0 ; }

form li, div > p { background: #eee ;
                  display: flex ;
                  justify-content: space-between ;
                  margin-bottom: 10px ;
                  list-style-type: none ;
                  border: 1px solid black ; }

form img { height: 64px; order: 1 ; }

form p { line-height: 32px; padding-left: 10px ; }

form label, form button { background-color: #7F9CCB ;
                          padding: 5px 10px ;
                          border-radius: 5px ;
                          border: 1px ridge black ;
                          font-size: 0.8rem ;
                          height: auto ; }

form label:hover, form button:hover { background-color: #2D5BA3 ;
                                      color: white ; }

form label:active, form button:active { background-color: #0D3F8F ;
                                       color: white ; }
```

10.4. Blob

10.4.ก.) Blob คืออะไร ?

Blob

<https://Javascript.info/blob>

Blob

<https://developer.mozilla.org/en-US/docs/Web/API/Blob>

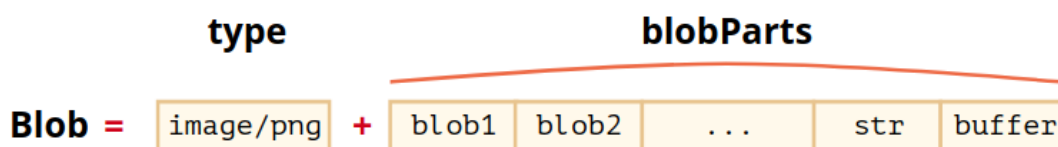
วัตถุ Blob ก็หมายถึง Blob ซึ่งมีลักษณะคล้ายวัตถุ File ที่ไม่เปลี่ยนรูป เป็นข้อมูลดิบ ซึ่งสามารถถูกอ่านเป็นข้อมูลแบบ Text หรือข้อมูลแบบ Binary ได้ หรือสามารถแปลงเป็น **ReadableStream**(ข้อมูล Stream) ฉะนั้นเมทอดของ Blob จึงถูกใช้เพื่อประมวลผลข้อมูล

10.4.ข.) โครงสร้างของ Blob

Blob สามารถใช้แทนข้อมูลใน Javascript ได้เลย เพราะวัตถุ File มีพื้นฐานมาจาก Blob

Blob ประกอบไปด้วย 2 ส่วนตามภาพ ก็คือส่วนของ

1. **type** เป็น String ของหรือ MIME-type
2. **blobParts** เป็นลำดับของวัตถุ Blob, String และ BufferSource



MIME Type (Multipurpose Internet Mail Extensions)

https://en.wikipedia.org/wiki/Media_type

MIME Type เป็นข้อความ 2 ส่วน ก็คือ **Type** และ **Subtype** เช่น **text/html** ที่ใช้ระบุ **ชนิดของไฟล์**(File format) และ **ชนิดของเนื้อหา**(Format contents) ที่สื่อสารกันในอินเทอร์เน็ต เช่น การส่งไฟล์แนบไปกับอีเมล เป็นต้น

ตัวอย่างการระบุ **MIME Type**

- text/html
- image/jpeg
- audio/mpeg
- application/pdf
- application/vnd.oasis.opendocument.text (.odt)

เป็นต้น

10.4.ค.) การสร้าง Blob

Blob()

<https://developer.mozilla.org/en-US/docs/Web/API/Blob/Blob>

โครงสร้างการใช้งาน

```
var newBlob = new Blob(blobParts, options) ;
```

blobParts : เป็นอาร์เรย์ของค่า Blob/BufferSource/String

options (Optional) : วัตถุตัวเลือก ซึ่งระบุข้อมูลดังต่อไปนี้

- **type**(Optional) : ชนิดของ Blob ปกติก็คือ MIME-type เช่น image/png

- **endings**(Optional) : ใช้กำหนดว่าจะแปรการขึ้นบรรทัดใหม่(\r\n หรือ \n) ของ Blob หรือไม่ ถ้าข้อมูลเป็น Text ค่าปริยายก็คือ transparent" (ไม่ทำอะไร) แต่สามารถระบุเป็น "native" (transform) ได้

Blob คืนค่ากลับมาเป็นวัตถุ Blob ที่เก็บข้อมูลแบบจำเพาะเจาะจง

ตัวอย่าง

```
// สร้าง Blob จากข้อความ
// Argument ตัวแรกจะต้องเป็นอาร์เรย์ [...]
let blob = new Blob( ["<html>...</html>"], {type: 'text/html'} ) ;
```

ตัวอย่าง

```
// สร้าง Blob จาก อาร์เรย์และข้อความ
let hello = new Uint8Array( [72, 101, 108, 108, 111] ) ; // "Hello" in binary form
let blob = new Blob( [hello, ' ', 'world'], {type: 'text/plain'}) ;
```

10.4.ง.) Blob as URL

Blob สามารถถูกใช้ได้ง่าย ในลักษณะ URL สำหรับแท็ก `<a>`, `` หรืออย่างอื่นเพื่อแสดงเนื้อหา

เราสามารถ Upload/Download วัตถุ Blob ได้โดยใช้ type (content type)

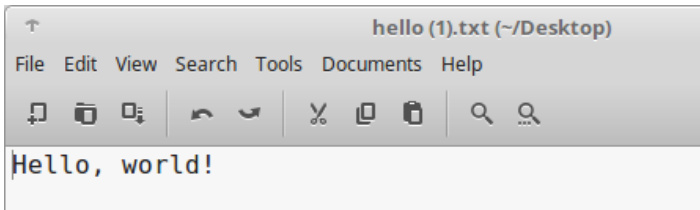
ตัวอย่าง

```
<!-- download attribute forces the browser to download instead of navigating -->
<a download="hello.txt" href="#" id="link">Download</a>

<script>
  let blob = new Blob(["Hello, world!"], {type: 'text/plain'}) ;
  link.href = URL.createObjectURL(blob) ; // สร้างลิงค์
</script>
```

ผล – เมื่อคลิกที่ "Download" ไฟล์ hello.txt จะถูกดาวน์โหลดลงมา โดยมีเนื้อหาข้างในเป็น Hello, world!

[Download](#)



`URL.createObjectURL` เอา Blob ไปสร้าง URL เพื่อลิงค์ไปที่ Blob ในรูปแบบ `blob:<origin>/<uuid>` ฉะนั้น `link.href` จึงมีลักษณะเช่น `blob:https://Javascript.info/1e67e00e-860d-40a5-89ae-6abocbee6273`

URL ที่ถูกสร้างใช้ได้เฉพาะใน Document ปัจจุบันเท่านั้น และขณะที่มันเปิด และอนุญาตให้อ้างอิงถึง Blob ใน ``, `<a>`

กระบวนการ mapping ข้างต้น URL ที่จับไปที่ Blob ซึ่ง Blob อยู่ในเมมโมรี Browser จึงยังปล่อยทิ้งไม่ได้ แต่จะถูกเคลียร์ออกเมื่อ `Unload Document` (ปิดแท็บ)

ฉะนั้นเมื่อเราสร้าง URL มาสักตัวหนึ่ง Blob จะค้างอยู่ในเมมโมรี แม้ว่าเราจะไม่ต้องการมันอีกแล้วก็ตาม

ตัวอย่าง – แค่เปิดลิงค์ก็ดาวน์โหลดไฟล์

ตัวอย่างนี้ เราตั้งใจจะใช้ Blob ครั้งเดียวเพื่อดาวน์โหลด ดังนั้นจึงใช้ `URL.revokeObjectURL(url)` ทันที

```
<!-- download attribute forces the browser to download instead of navigating -->
<script>
  let link = document.createElement('a') ;      // สร้างองค์ประกอบ <a>
  link.download = 'hello.txt' ;                  // กำหนด attribute download ให้กับ <a>

  let blob = new Blob(['Hello, world!'], {type: 'text/plain'}) ;
  link.href = URL.createObjectURL(blob) ;

  link.click() ;                                // ใช้ Method click() ที่ลิงค์
  URL.revokeObjectURL(link.href) ;              // เปิดลิงค์
</script>
```

10.5. Blob to base64

(อันนี้ใช้บ่อย ใช้ดี ในการสร้างไฟล์จากการอัปโหลดไปที่ Google Drive)

10.5.ก.) Blob to base64

อีกทางเลือกหนึ่ง สำหรับ `URL.createObjectURL` ก็คือ การเข้ารหัสหรือแปลง Blob ไปเป็น `base64-encoded string`

การเข้ารหัสข้างต้น จะแสดงผลเป็นข้อมูล Binary แบบ String ซึ่งปลอดภัยสำหรับการอ่าน (คนอ่านไม่รู้เรื่อง) เพราะเป็น ตัวอักษร ASCII จาก 0-64 ที่สำคัญ เราสามารถใช้ตัวนี้เข้ารหัสในลักษณะ **Data URLs** ได้ด้วย

10.5.ข.) Data URLs

Data URLs

https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Data_URIs

Data URLs มีรูปแบบดังนี้

```
data:[<mediatype>][;base64],<data>
```

Data URLs เริ่มต้นด้วย **data:**

mediatype ก็คือ MIME type string เช่น image/jpeg ใช้บ่งบอกชนิดของข้อมูล ถ้าเว้นไว้ค่าปริยายจะเป็น text/plain;charset=US-ASCII

ดู MIME Type ได้ที่

https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types/Common_types
https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types

ส่วน **data** ถ้าเป็น Text ก็แนบเป็น Text ติดไป ถ้าเป็นอย่างอื่นๆ เราสามารถใช้ Base64 เข้ารหัสข้อมูล Binary แล้วแนบไปได้

ตัวอย่างที่ 1 – แนบเป็น Text ตรงๆ ส่วนที่เป็นสัญลักษณ์แปลกๆ เราจะใช้ percent-encoding แทน เช่น Space ก็คือ %20

```
data:,Hello%2C%20World!
```

ตัวอย่างข้างต้น ถ้าเข้ารหัสเป็น Base64 จะได้ดังนี้

```
data:text/plain;base64,SGVsbG8sIFdvcmxkIQ==
```

ตัวอย่างที่ 2 – เอกสาร Html ที่มีเนื้อหาเป็น <h1>Hello, World!</h1>

```
data:text/html,%3Ch1%3EHello%2C%20World!%3C%2Fh1%3E
```

ตัวอย่างที่ 3 – เอกสาร Html ที่ทำคำสั่ง Javascript alert

```
data:text/html,<script>alert('hi');</script>
```

10.5.ค.) การเข้ารหัสข้อมูลเป็น Base64

Base64

<https://developer.mozilla.org/en-US/docs/Glossary/Base64>

Base64 เป็นกลุ่มของรูปแบบการเข้ารหัสจาก Binary ไปเป็น Text ที่ใช้แสดงแทนข้อมูล Binary ด้วย ASCII String โดยการแปลงไปเป็น radix-64 (มี 64 ตัวอักษร)

ตัวอย่าง - เข้ารหัส Base64 ใน Linux เปิด Terminal จากนั้นพิมพ์คำสั่ง เช่น

```
echo -n lolbase64
# outputs to console: aGVsbG8=
```

ตัวอย่าง - อ่าน Text จากไฟล์ แล้วแปลงเป็น Base64

```
echo -n hello>a.txt
base64 a.txt
# outputs to console: aGVsbG8=
```

ตัวอย่าง - อ่าน Text จากไฟล์ แล้วแปลงเป็น Base64 เก็บไว้ในอีกไฟล์หนึ่ง

```
base64 a.txt>b.txt
# outputs to file b.txt: aGVsbG8=
```

10.5.ง.) การใช้ Data URLs กับแท็ก img

เราสามารถใช้ Data URLs นี้ที่ไหนก็ได้ เช่นเดียวกันกับ Url ทั่วไป

ตัวอย่าง - ต่อไปนี้เป็น Data URLs ที่มีข้อมูลเป็นภาพ Smiley โดย Browser จะถอดรหัส(decode) ออกมาเป็นภาพ 😊

การใช้งานก็แค่ใส่แท็ก ไปวางใน <body> ในไฟล์ html

```

```

สำหรับการแปลง Blob ไปเป็น base64 เราจะใช้เมธอดของ FileReader เข้าไปอ่าน Blob เพราะสามารถอ่านข้อมูลใน Blob ได้ในหลากหลายรูปแบบ

ตัวอย่าง - เมื่อเปิดไฟล์ .html จะดาวน์โหลดไฟล์ลงมาทันที

```
<script>
let link = document.createElement('a') ;    // สร้างแท็ก <a>
link.download = 'hello.txt' ;                // เชื่อม Attribute download

let blob = new Blob(['Hello, world!'], {type: 'text/plain'}) ; // สร้าง Blob จาก String

let reader = new FileReader() ;               // สร้างวัตถุ FileReader

reader.readAsDataURL(blob); // เข้าไปอ่าน Blob แล้วแปลงเป็น base64 (DataURL)
// เรียกใช้งานตอน onload

reader.onload = function() {
    link.href = reader.result ; // กำหนด data url ให้กับ
    // Attribute href ของวัตถุ link (<a>)

    link.click() ;
} ;
</script>
```

10.6. Image to blob

Image to blob

<https://javascript.info/blob#image-to-blob>

เราสามารถสร้าง Blob ของภาพ บางส่วนของภาพ หรือ ภาพแคปจากหน้าเว็บ ที่ง่ายในการอัปโหลดไป
สักที

การดำเนินการกับภาพ สามารถทำได้ผ่านอีเล็มেন্ট <canvas>

```
<script>
  'use strict' ;

  // take any image
  let img = document.querySelector('img') ;

  // make <canvas> of the same size
  let canvas = document.createElement('canvas') ;
  canvas.width = img.clientWidth ;
  canvas.height = img.clientHeight ;

  let context = canvas.getContext('2d') ;

  // copy image to it (this method allows to cut image)
  context.drawImage(img, 0, 0) ;
  // we can context.rotate(), and do many other things on canvas

  // toBlob is async operation, callback is called when done
  canvas.toBlob(function(blob) {

    // blob ready, download it
    let link = document.createElement('a') ;
    link.download = 'example.png' ;

    link.href = URL.createObjectURL(blob) ;
    link.click();

    // delete the internal blob reference, to let the browser clear memory from it
    URL.revokeObjectURL(link.href) ;
  }, 'image/png') ;
</script>
```

10.7. FileReader

(ที่เราต้องศึกษา FileReader เพราะต้องใช้เขียนโค้ดเพื่ออัปโหลดไฟล์จากฟอร์มลงใน Google Drive)

10.7.ก.) FileReader คืออะไร ?

EventTarget : Object - <https://www.javascripture.com/EventTarget>

FileReader : EventTarget - <https://www.Javascripture.com/FileReader>

FileReader - <https://developer.mozilla.org/en-US/docs/Web/API/FileReader>

FileReader() - <https://developer.mozilla.org/en-US/docs/Web/API/FileReader/FileReader>

Using files from web applications

https://developer.mozilla.org/en-US/docs/Web/API/File/Using_files_from_web_applications

โครงสร้างการใช้งาน - วิธีสร้างวัตถุ FileReader

```
var reader = new FileReader() ;
```

วัตถุ FileReader ใช้อ่านเนื้อหาใน Blob หรือ File โดยคืนค่ากลับมาเป็น วัตถุตัวใหม่ของ FileReader (Constructed)

FileReader เป็นวัตถุ EventTarget หรือวัตถุที่มากับ Event ส่งผ่านข้อมูลโดยใช้ Event โดยอ่านข้อมูลจาก Disk ของผู้ใช้งาน

โดยปกติ EventTarget จะมีคุณสมบัติ on+event ต่างๆ สำหรับแต่ละ Event ที่เราสามารถผูกกับฟังก์ชันเพื่อทำคำสั่งเมื่อ Event นั้นถูกยิงออกมา หรือเราสามารถที่จะใช้เมธอด addEventListener() เพื่อผูก Event กับหลายวัตถุก็ได้

10.7.ข.) เมธอดของ FileReader

readAsArrayBuffer(blob) – read the data in binary format ArrayBuffer.

readAsText(blob, [encoding]) – read the data as a text string with the given encoding (utf-8 by default). – อ่านข้อมูลมาเป็น Text และเข้ารหัส

readAsDataURL(blob) – อ่านข้อมูล Binary และเข้ารหัสเป็น Base64 Data url

abort() – ยกเลิกการทำงาน

ขณะอยู่ในระหว่างกระบวนการอ่าน มี Event ดังต่อไปนี้

(on)loadstart – loading started.

(on)progress – occurs during reading.

(on)load – no errors, reading complete.

(on)abort – abort() called.

(on)error – error has occurred.

(on)loadend – reading finished with either success or failure.

เมื่ออ่านไฟล์จบแล้ว เราสามารถเข้าถึง result ได้โดย

reader.result ผลการอ่าน (ถ้าสำเร็จ)

reader.error เป็น Error (ถ้าล้มเหลว)

หมายเหตุ :

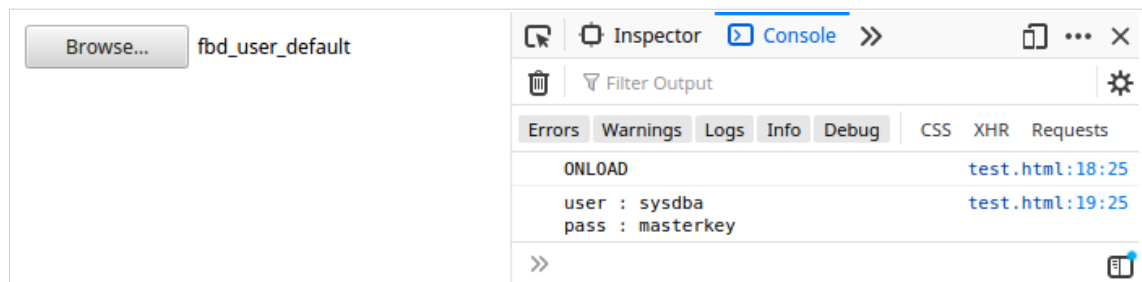
เราสามารถจับไฟล์ได้ โดยใช้ `HTMLInputElement.files` (การเลือกไฟล์จากฟิลด์ `<input>`) หรือ เมธอด `DataTransferItem.getAsFile()` จากนั้นใช้ `FileReader` อ่านเนื้อหาภายในไฟล์

ตัวอย่าง – อ่านข้อมูลในไฟล์ Text

```
<input type="file" onchange="readFile(this)">
<script>
function readFile(input) {
    let file = input.files[0]; // จับไฟล์แรก ไฟล์เดียวที่มาจากอิลิเมนต์ input
    let reader = new FileReader();
    reader.readAsText(file); // ใช้เมธอด readAsText ของ FileReader
                                // อ่านข้อมูลในไฟล์ที่ระบุ

    reader.onload = function() { // หากอ่านสำเร็จ
        console.log("ONLOAD"); // แสดงผลการอ่านที่ console
        console.log(reader.result);
    };
    reader.onerror = function() { // หากอ่านไม่สำเร็จ
        console.log("ONERROR"); // แสดง Error ที่ console
        console.log(reader.error);
    };
};
</script>
```

ผล – ดูที่ console จะปรากฏ Text ที่อยู่ในไฟล์ ทั้งนี้เพราะไฟล์ที่ไปอ่านเป็น Text file แต่ถ้าเป็นไฟล์ภาพ จะได้เป็นตัวอักษรต่างดาว



ตัวอย่าง – คล้ายกับข้างบน เพียงแต่ตัวอย่างนี้ จะแสดงรายละเอียดไว้ที่ console เพื่อให้เข้าใจยิ่งขึ้น

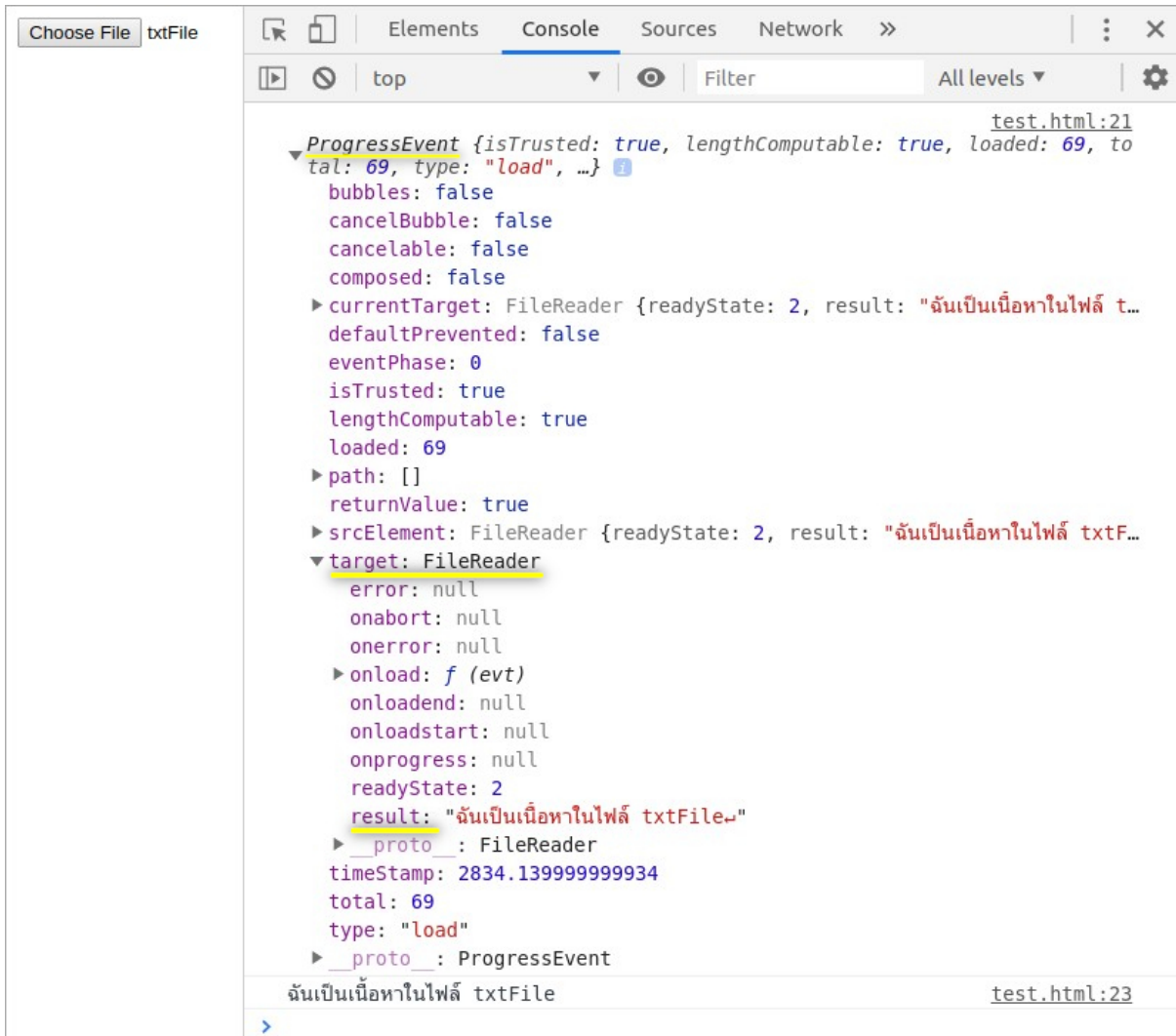
```
function printFile(input) {
    let file = input.files[0];
    var reader = new FileReader();
    reader.readAsText(file);

    reader.onload = function(evt) {
        console.log(evt); // ดูผลที่ console (แสดงสิ่งต่างๆที่มากับ Event)
        console.log(evt.target.result); // ดูผลที่ console (แสดงเนื้อหาภายในไฟล์)
    };
}
```

ผล - วัตถุ **ProgressEvent** ปรากฏเมื่อเราโหลดเอกสารข้างนอก ถ้าดูที่ **target** จะพบว่าเป็นวัตถุ **FileReader** และถ้าดูต่อไปที่ **result** จะเห็นเนื้อหาภายในไฟล์ (ตัวอย่างไฟล์ที่ทดสอบเป็น Text file) นอกจากนี้จะเห็น Event ต่างๆของวัตถุ FileReader ด้วยตัวอย่างเช่น onload, onprogress เป็นต้น

HTML DOM ProgressEvent

https://www.w3schools.com/jsref/obj_progressesevent.asp



The screenshot shows a web browser's developer console with the 'Console' tab selected. The console displays a **ProgressEvent** object. The event's **target** property is a **FileReader** object. The **result** property of the **FileReader** object contains the text: "ฉันเป็นเนื้อหาในไฟล์ txtFile". The console also shows the **onload** event handler being called.

```
test.html:21
ProgressEvent {isTrusted: true, lengthComputable: true, loaded: 69, total: 69, type: "load", ...}
  bubbles: false
  cancelBubble: false
  cancelable: false
  composed: false
  ▶ currentTarget: FileReader {readyState: 2, result: "ฉันเป็นเนื้อหาในไฟล์ t...
  defaultPrevented: false
  eventPhase: 0
  isTrusted: true
  lengthComputable: true
  loaded: 69
  ▶ path: []
  returnValue: true
  ▶ srcElement: FileReader {readyState: 2, result: "ฉันเป็นเนื้อหาในไฟล์ txtF...
  ▶ target: FileReader
    error: null
    onabort: null
    onerror: null
    ▶ onload: f (evt)
      onloadend: null
      onloadstart: null
      onprogress: null
      readyState: 2
      result: "ฉันเป็นเนื้อหาในไฟล์ txtFile"
    ▶ __proto__: FileReader
    timeStamp: 2834.139999999934
    total: 69
    type: "load"
    ▶ __proto__: ProgressEvent
  ฉันเป็นเนื้อหาในไฟล์ txtFile
test.html:23
```

ตัวอย่าง – เมื่อเลือกไฟล์โดยใช้ อีเล็มเมนต์ <input> ก็ให้แสดงภาพด้วย

```
<input type='file' accept='image/*' onchange='openFile(event)'><br>
<img id='output'>

<script>

var openFile = function(event) {

var input = event.target ;    // event.target ก็คืออีเล็มเมนต์ <input>
var reader = new FileReader() ;

reader.onload = function(){

    var dataURL = reader.result ;

    // console.log(dataURL); // data:image/jpeg;base64,/9j/4AAQSkZJRg...
    var output = document.getElementById('output') ;
    output.src = dataURL ;    // ใส่ src ให้กับแท็ก <img id='output'>

} ; // function - reader.onload

reader.readAsDataURL(input.files[0]) ;
// บรรทัดนี้ ย้ายไปไว้ก่อน reader.onload = function(){... ก็ได้ น่าจะเข้าใจง่ายกว่า
// อ่านเนื้อหาในไฟล์เป็น Data Urls – เอาไปใช้กับอีเล็มเมนต์ img เพื่อแสดงภาพได้

} ; // function- openFile

</script>
```

ผล – เมื่อเลือกไฟล์โดยใช้ปุ่ม Choose File จะปรากฏภาพของไฟล์ที่เลือก ตามภาพ



10.8. FileReader.readAsDataURL()

FileReader.readAsDataURL()

<https://developer.mozilla.org/en-US/docs/Web/API/FileReader/readAsDataURL>

readAsDataURL() ใช้อ่านเนื้อหาใน Blob หรือ File เมื่ออ่านเสร็จแล้ว คุณสมบัติ **readyState** ของ FileReader จะกลายเป็น DONE (มี 3 สถานะ EMPTY, LOADING และ DONE) และ Event **loadend** จะถูกยิงออกมา และ ณ เวลานั้น คุณสมบัติ **result** ก็จะเก็บข้อมูลไว้ในลักษณะ **Data URL**

โครงสร้างการใช้งาน

```
instanceOfFileReader.readAsDataURL(blob) ;
```

ตัวอย่าง

Html

```
<input type="file" onchange="previewFile()"><br>
<img src="" height="200" alt="Image preview...">
```

JavaScript

```
function previewFile() {
    const preview = document.querySelector('img');
    const file = document.querySelector('input[type=file]').files[0] ;
    const reader = new FileReader() ;

    reader.addEventListener("load", function() {
        preview.src = reader.result ;      // convert image file to base64 string
    }, false) ;

    if (file) {
        reader.readAsDataURL(file);
    }
}
```

ผล - เมื่อเลือกไฟล์โดยใช้ปุ่ม Choose File จะปรากฏภาพของไฟล์ที่เลือก ตามภาพ



ตัวอย่างที่ 2

Html

```
<input id="browse" type="file" onchange="previewFile()" multiple><br>
<div id="preview"></div>
```

JavaScript

```
function previewFile() {

    var preview = document.querySelector('#preview') ;
    var files = document.querySelector('input[type=file]').files ;

    // #1
    function readAndPreview(file) {

        // ทำให้แน่ใจว่า file.name แตกต่างที่เราติกรอบไว้ ก็คือ ไฟล์ PNG,JPGmJPEG,GIF เท่านั้น
        if ( /\.?(jpe?g|png|gif)$/i.test(file.name) ) {

            var reader = new FileReader() ;

            reader.addEventListener("load", function () {
                var image = new Image() ;
                image.height = 100 ;
                image.title = file.name ;
                image.src = this.result ;

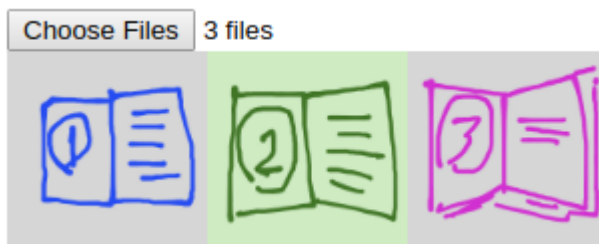
                preview.appendChild(image) ;

            }, false) ;

            reader.readAsDataURL(file) ;
        }
    }

    // #2
    if (files) {
        [].forEach.call(files, readAndPreview) ;
        // เหมือนกับ Array.prototype.forEach.call() - เรียกใช้เมธอดของอาเรย์เพื่อใช้กับวัตถุ NodeList
        // วนลูปเข้าไปอ่านทีละไฟล์
    }
}
```

ผล - เลือก 3 ภาพ ปรากฏภาพขนาดเล็กๆ ซึ่งกำหนดความสูงไว้ 100 ของทั้ง 3 ตามภาพ



10.9. ตัวอย่างอัปโหลดไฟล์ลงใน Drive โดยใช้ FileReader

(ปรับแต่งจาก) Upload Files to Google Drive with Google Apps Script

<https://www.labnol.org/code/19747-google-forms-upload-files>

ทดลองใช้งาน - <https://script.google.com/macros/s/AKfycbxQho1lqXJEnNkuEO-QLxdFoE5eHAIRcRBlbMNAQ-k3zSzyUKX/exec>

โค้ด - <https://script.google.com/d/1jtCiAqq7pS124ba-fmFc9zmOpjhCcTwbzFh-Z38DmAAxOvTNP8NS-lw/edit?usp=sharing>

10.9.ก.) ไฟล์ forms.html

โค้ด

```
<html>
<head>
  <base target="_top">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <style>div,input,button{margin: 5px 5px 5px 5px}</style>
</head>

<body>
  <input id="name" type="text" placeholder="Your Name"><br>
  <input id="email" type="email" placeholder="Your Email"><br>

  <input id="file" type="file"><br>
  <button onclick="submitForm()">Submit</button><br>
  <div id="progress"></div>

  <script>
    var file ;
    var reader = new FileReader() ;

    // Upload the file to Google Drive
    reader.onloadend = function(e) {
      google.script.run
        .withSuccessHandler(showMessage)
        .uploadFileToGoogleDrive( e.target.result ,           // 1
                                file.name ,                 // 2
                                $('#input#name').val() ,     // 3
                                $('#input#email').val()       // 4
                                ) ;

    } ;

    // Read the file on form submit
    function submitForm() {                                // เมื่อคลิกปุ่ม Submit
      file = $('#file')[0].files[0] ;                      // จับไฟล์ดรอปนี้ลำดับที่ 0
      showMessage("Uploading file.." ) ;                  // แสดงข้อความ Uploading ที่ progress
      reader.readAsDataURL(file) ;                         // อ่านไฟล์เป็น Data Url
    }

    function showMessage(e) {                              // หาก Upload สำเร็จ
      $('#progress').html(e) ;                             // แสดงข้อความที่มาจาก Apps Script (OK)
    }
  </script>
</body>
</html>
```

10.9.ข.) ไฟล์ Code.gs

โค้ด

```
function doGet(e) {
    return HtmlService.createHtmlOutputFromFile('forms.html')
        .setTitle("Upload a file to Google Drive") ;
}

function uploadFileToGoogleDrive(data, file, name, email) {

    try {
        var dropbox = "WK_Dropbox" ;
        var folder, folders = DriveApp.getFoldersByName(dropbox) ;

        if (folders.hasNext()) {           // ถ้าไม่มีโฟลเดอร์ตามที่ระบุ ก็ให้สร้างใหม่
            folder = folders.next() ;
        } else {
            folder = DriveApp.createFolder(dropbox) ;
        }

        // ตัดส่วนของ MIME Type ออกมา (เริ่มจาก อักขระตัวที่ 5 ไปจนก่อนถึง ;
        // เช่น data:text/plain;base64,SGVsbG8sIFdvcmxkIQ==
        var contentType = data.substring(5, data.indexOf(';')) ;

        // ตัด Data ที่เป็น Base64 ออกมา ก็คือ หลังเครื่องหมาย , ทั้งหมด
        // เช่น data:text/plain;base64,SGVsbG8sIFdvcmxkIQ==
        // จากนั้น ถอดรหัสโดยใช้ Utilities.base64Decode คืนค่ากลับมาเป็น Byte[]
        var bytes = Utilities.base64Decode(data.substr(data.indexOf('base64,')+7)) ;

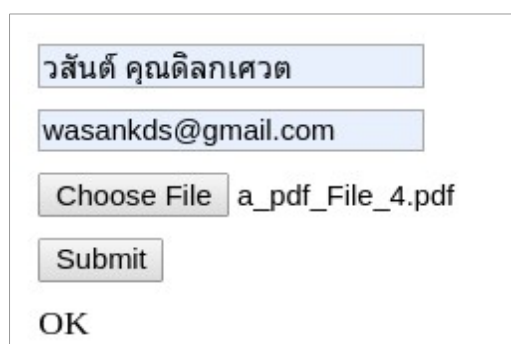
        // สร้าง Blob ใหม่                                     // ชื่อไฟล์
        var blob = Utilities.newBlob(bytes, contentType, file) ;

        folder.createFolder([name, email].join(" ")).createFile(blob) ;

        return "OK" ;
    } catch (f) {
        return f.toString() ;
    }
}
```

10.9.ค.) ผล

หน้าฟอร์มของโปรเจ็ค มีลักษณะตามภาพ – เมื่อกรอกข้อมูล และเลือกไฟล์ที่จะอัปโหลด จากนั้นคลิกปุ่ม Submit ไฟล์ถูกอัปโหลด และเมื่ออัปโหลดสำเร็จจะมีข้อความแจ้ง (OK)




wasankds@gmail.com

Choose File a_pdf_File_4.pdf

Submit

OK

มาดูที่ฝั่ง Google Drive – ไฟล์จะถูกอัปโหลดไว้ในโฟลเดอร์ตามที่กำหนดในโค้ด(WK_Dropbox) โดยโฟลเดอร์ใหม่จะถูกสร้างลงไป โดยตั้งชื่อตาม ชื่อที่กรอกและอีเมลที่กรอกลงในฟอร์ม จากนั้นไฟล์จะถูกอัปโหลดไว้ที่นี่

My Drive > ... > WK_Dropbox > วสันต์ คุณดิลกเศวต wasankds@gmail.com		
Name ↑	Owner	Last modified by me
 a_pdf_File_4.pdf	me	6:45 AM

10.10. เมธอด `base64Decode()`

`base64Decode(encoded)` – เมธอดในคลาส Utilities

<https://developers.google.com/apps-script/reference/utilities/utilities#base64decodeencoded>

`base64Decode(encoded, charset)` – เมธอดในคลาส Utilities

<https://developers.google.com/apps-script/reference/utilities/utilities#base64decodeencoded,-charset>

`base64Decode()` ใช้ถอดรหัสข้อมูลที่เข้ามาแบบ Base64 string (พารามิเตอร์ `encoded`) โดยถอดออกมาเป็น `Byte[]` หรือ ข้อมูล Bytes ในก้อนอาเรย์ โดยเราสามารถกำหนด Character set ที่จะถอดออกมาได้ กำหนดโดยพารามิเตอร์ `charset` ถ้าไม่กำหนดพารามิเตอร์นี้ค่า Default จะเป็น UTF-8

ตัวอย่าง

```
var base64data = "SGVsbG8sIFdvcmxkIQ==" ;
var decodedAsBytes = Utilities.base64Decode(base64data, Utilities.Charset.UTF_8) ;
Logger.log(decodedAsBytes) ; // ดูผลที่ Logs ----- > [01]
var dataString = Utilities.newBlob(decodedAsBytes).getDataAsString() ;
Logger.log(dataString) ; // ดูผลที่ Logs ----- > [02]
```

ผล

Logs

```
[01] [72.0, 101.0, 108.0, 108.0, 111.0, 44.0, 32.0, 87.0, 111.0, 114.0, 108.0, 100.0, 33.0]
[02] Hello, World!
```

10.11. newBlob()

คลาส Utilities

<https://developers.google.com/apps-script/reference/utilities/utilities>

เครื่องมือต่างๆ ที่ใช้สำหรับเข้ารหัสหรือถอดรหัส จัดรูปแบบวันที่ จัดการ JSON และอื่นๆ

`newBlob(data, contentType, name)` - เมธอดในคลาส Utilities

https://developers.google.com/apps-script/reference/utilities/utilities#newblobdata,-contenttype,-name_1

ใช้สร้างวัตถุ Blob จาก `String`, `Content type` และ `ชื่อ`

พารามิเตอร์

Name	Type	Description
<code>data</code>	<code>String</code>	the string for the blob, assumed UTF-8
<code>contentType</code>	<code>String</code>	the content type of the blob - can be null
<code>name</code>	<code>String</code>	the name of the blob - can be null

ตัวอย่างการสร้างวัตถุ Blob อย่างง่าย

```
var blob = Utilities.newBlob( 'Insert any HTML content here', // ข้อความในไฟล์ HTML
                             'text/html',                    // MIME type แบบ text/html
                             'my_document.html' );           // ชื่อไฟล์

// จากนั้นเอาไปสร้างเป็นไฟล์ต่อได้ เช่น
// var file = DriveApp.createFile(blob);
```

หมายเหตุ : เกี่ยวกับการอัปโหลดไฟล์ลงใน Google Drive ที่น่าสนใจ

Uploading Multiple Files From Local To Google Drive using Google Apps Script

<https://tanaikech.github.io/2018/12/22/uploading-multiple-files-from-local-to-google-drive-using-google-apps-script/>

Uploading Multiple Files to Google Drive with Google App Script

<https://stackoverflow.com/questions/31126181/uploading-multiple-files-to-google-drive-with-google-app-script>