

Dom Manipulation

การเขียน Function ที่ดี

- ควรเขียนโดยรับ **parameter**
- เนื่องจากสามารถนำกลับมาใช้ใหม่ได้
- ข้อมูลที่รับควรเป็น ค่าใดๆ ก็ตามที่ได้รับค่าไม่เกิน 7 ตัวแปร
- มิเช่นนั้นก็อาจใช้เป็น **Object** แทน

การสร้าง Object

- ประกาศตัวแปร แล้วสามารถนำไปใช้ได้เลย เช่น
- ก่อนหน้าทำงาน เอาไฟล์เดิมคราวที่แล้วเป็นตัวตั้ง แล้วเอามาสร้าง **repo** ใหม่ นะครับ

```
var student = {};  
student.name = 'คุณเล้ง'  
student.username = 'a@b.com'  
student.gender = 'ชาย'  
  
document.getElementById('output').innerText = student;
```

Jane Doe

@ Username

Choose... ▼

[object Object]

- Object ไม่ใช่ Text
 - แสดงผลออกมาเป็นชนิดของ object
- ตรวจค่าได้จาก **inspector**

ถ้าจะแสดงค่าล่ะ?

- เปิดว่าอยากแสดงแบบไหน ถ้าอยากได้แบบนี้

ชื่อ
รหัส
เพศ

สมชาย
a@b.com
หญิง

```
<div id="output">
  <div class="row">
    <div class="col-1 offset-1">
      | ชื่อ
    </div>
    <div class="col">
      | สมชาย
    </div>
  </div>
  <div class="row">
    <div class="col-1 offset-1">
      | รหัส
    </div>
    <div class="col">
      | a@b.com
    </div>
  </div>
  <div class="row">
    <div class="col-1 offset-1">
      | เพศ
    </div>
    <div class="col">
      | หญิง
    </div>
  </div>
</div>
```

ลองเขียน dom node ของ html นี้

```
<div id="output">
  <div class="row">
    <div class="col-1 offset-1">
      | ชื่อ
    </div>
    <div class="col">
      | สมชาย
    </div>
  </div>
  <div class="row">
    <div class="col-1 offset-1">
      | รหัส
    </div>
    <div class="col">
      | a@b.com
    </div>
  </div>
  <div class="row">
    <div class="col-1 offset-1">
      | เพศ
    </div>
    <div class="col">
      | หญิง
    </div>
  </div>
</div>
```

มาลองเขียน function กัน

- ลบ ค่าที่อยู่ใน **output** ออกไปก่อน

```
function addStudentData(student){
  const output = document.getElementById('output')
  let row = document.createElement('div')
  row.classList.add("row")
  let columnName = document.createElement('div')
  columnName.classList.add("col-1")
  columnName.classList.add("offset-1")
  columnName.innerHTML = 'ชื่อ'
  let columnValue = document.createElement('div')
  columnValue = document.createElement('row')
  columnValue.classList.add('col')
  columnValue.innerHTML = student.name;
  row.appendChild(columnName)
  row.appendChild(columnValue)
  output.appendChild(row)
}

window.addEventListener("load", function(){
  addStudentData(student)
})
```

อธิบาย

```
function addStudentData(student){
  const output = document.getElementById('output')
  let row = document.createElement('div')
  row.classList.add("row")
  let columnName = document.createElement('div')
  columnName.classList.add("col-1")
  columnName.classList.add("offset-1")
  columnName.innerHTML = 'ชื่อ'
  let columnValue = document.createElement('div')
  columnValue = document.createElement('row')
  columnValue.classList.add('col')
  columnValue.innerHTML = student.name;
  row.appendChild(columnName)
  row.appendChild(columnValue)
  output.appendChild(row)
}

window.addEventListener("load", function(){
  addStudentData(student)
})
```

```
<div id="output">
  <div class="row">
    <div class= "col-1 offset-1">
      | ชื่อ
    </div>
    <div class="col">
      | สมชาย
    </div>
  </div>
  <div class="row">
    <div class= "col-1 offset-1">
      | รหัส
    </div>
    <div class="col">
      | a@b.com
    </div>
  </div>
  <div class="row">
    <div class= "col-1 offset-1">
      | เพศ
    </div>
    <div class="col">
      | หญิง
    </div>
  </div>
</div>
```


ลองทำดู

- เพิ่ม `code` ใน `function` เพื่อให้แสดงเพศ และรหัส

ลองทำดูสอง

- เพิ่ม **object** อีก **object** เพื่อที่จะให้หน้าจอแสดงผลได้ทั้งสองข้อมูล (ทั้งนักเรียนแรก และ **object** ของนักเรียนที่สอง)

Array ของ object

- บางครั้งมีข้อมูลมาก
- ควรทำงานกับ **Array**
- ทดลองสร้าง **array**
- แล้วลองดูค่าใน **console**

```
var students = [  
  student,  
  secondStudent, // object ของนักเรียนที่สร้างขึ้นใหม่  
  {  
    name: 'สมรักษ์',  
    username: 'm@n.com',  
    gender: 'ชาย'  
  }  
]
```

ลองเอาค่าออกมาใส่ใน table

- ลองดูตัวอย่าง table ดู

```
<div id="output">
  <table class="table table-hover">
    <thead>
      <tr>
        <th scope="col">#</th>
        <th scope="col">First</th>
        <th scope="col">Last</th>
        <th scope="col">Handle</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <th scope="row">1</th>
        <td>Mark</td>
        <td>Otto</td>
        <td>@mdo</td>
      </tr>
      <tr>
        <th scope="row">2</th>
        <td>Jacob</td>
        <td>Thornton</td>
        <td>@fat</td>
      </tr>
      <tr>
        <th scope="row">3</th>
        <td colspan="2">Larry the Bird</td>
        <td>@twitter</td>
      </tr>
    </tbody>
  </table>
</div>
```

ลองปรับ class นิดหน่อยเพื่อ ความสวยงาม

- ลองสร้าง dom tree

```
<div id="output" class="px-5">
  <table class="table table-hover">
    <thead>
      <tr>
        <th scope="col">#</th>
        <th scope="col">First</th>
        <th scope="col">Last</th>
        <th scope="col">Handle</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <th scope="row">1</th>
        <td>Mark</td>
        <td>Otto</td>
        <td>@mdo</td>
      </tr>
      <tr>
        <th scope="row">2</th>
        <td>Jacob</td>
        <td>Thornton</td>
        <td>@fat</td>
      </tr>
      <tr>
        <th scope="row">3</th>
        <td colspan="2">Larry the Bird</td>
        <td>@twitter</td>
      </tr>
    </tbody>
  </table>
</div>
```

ลองเหลือ หัว table ไว้ เพื่อให้กรอกข้อมูลได้

```
<div id="output" class="px-5">
  <table class="table table-hover">
    <thead>
      <tr>
        <th scope="col">#</th>
        <th scope="col">ชื่อ</th>
        <th scope="col">รหัส</th>
        <th scope="col">เพศ</th>
      </tr>
    </thead>
    <tbody id="tableBody">
    </tbody>
  </table>
</div>
```

ทดลองเขียน function เพื่อเพิ่มข้อมูลของคนๆ เดียวก่อน

```
function addStudentToTable(index, student){  
    const tableBody = document.getElementById('tableBody')  
    let row = document.createElement('tr')  
    let cell = document.createElement('th')  
    cell.setAttribute('score', 'row')  
    cell.innerHTML = index  
    row.appendChild(cell)  
    cell = document.createElement('td')  
    cell.innerHTML = student.name  
    row.appendChild(cell)  
    cell = document.createElement('td')  
    cell.innerHTML = student.username  
    row.appendChild(cell)  
    cell = document.createElement('td')  
    cell.innerHTML = student.gender  
    row.appendChild(cell)  
    tableBody.appendChild(row)  
}
```

```
window.addEventListener("load", function() {  
    |   addStudentToTable(student)  
    })
```

ลองพิมพ์หลายคน

```
function addStudentList(studentList) {  
  let counter = 1  
  for (student of studentList) {  
    addStudentToTable(counter++, student)  
  }  
}
```

```
window.addEventListener("load", function() {  
  addStudentList(students)  
})
```


ลองทำดู

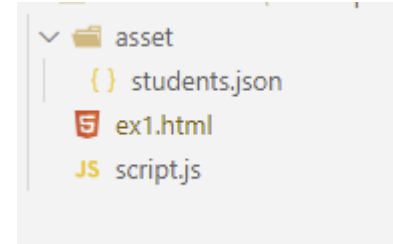
- ลองเขียนโปรแกรมเพื่อรับค่าจาก **input** ที่อยู่ด้านบน โดยเมื่อกด **submit** แล้วให้แสดงค่า ชื่อ รหัส และเพศ ที่กรอกเข้าไป

การดึงข้อมูลจากไฟล์

- บางครั้งเราอาจดึงข้อมูลจากไฟล์อื่นๆ ที่เกี่ยวข้องได้
- สร้าง ไฟล์ นี้
- ลบ **windows add load listener**
 - เรียก **load** ด้วยวิธีนี้แทน

```
<body onload="onLoad()">  
  <div class="container">
```

```
[{  
  {  
    "name": "คามาดิ ทันจิโร่",  
    "username": "kt",  
    "gender": "ชาย"  
  },  
  {  
    "name": "คามาดิ เนซึโกะ",  
    "username": "kn",  
    "gender": "หญิง"  
  },  
  {  
    "name": "อาากาสึมะ เซ็นนิตสึ",  
    "username": "as",  
    "gender": "ชาย"  
  },  
  {  
    "name": "ฮาชิบิระ อินสึเกะ",  
    "username": "hi",  
    "gender": "ชาย"  
  }  
}]
```



แล้วลองโหลดข้อมูล

```
function onLoad() {  
  fetch('asset/students.json').then(data => {  
    console.log(data)  
  })  
}
```

- ลองดูผลลัพธ์ใน development console

ทดลองแบบนี้ซิ

```
function onLoad() {  
  fetch('asset/students.json').then(data => {  
    console.log(data.json())  
  })  
}
```

Code นี้คาดว่าจะเป็นอย่างไ

```
function onLoad() {  
  let students  
  fetch('asset/students.json').then(data => {  
    students = data.json()  
  })  
  console.log(students)  
}
```

Asynchronous programming & Promise object

- Synchronous programming

- โปรแกรมทำงานตามลำดับ

```
let text = inputText.value
console.log(text)
let newButton = document.createElement('button')
newButton.classList.add('btn')
newButton.classList.add('btn-outline-primary')
newButton.classList.add('m-2')
newButton.setAttribute('type', 'button')
newButton.innerText = text
output.appendChild(newButton)
badgeCount.innerText = output.children.length
```

- ปัญหา

- ถ้ามีส่วนใดส่วนหนึ่งนาน จะทำให้ความเร็วทั้งหมด ช้า
 - เช่น การอ่านไฟล์ การดึงข้อมูลจาก **network**

Asynchronous programming

- Asynchronous Programming

- ทิ้ง บางงานให้คนอื่นทำไปก่อน
- ว่างเสร็จแล้วค่อยกลับมาทำนะ

```
function onLoad() {  
  let students  
  fetch('asset/students.json').then(data => {  
    students = data.json()  
  })  
  console.log(students)  
}
```

Promise object


- **Object** แห่งพันธสัญญา
- ฉันทราบไปทำงานนะ
 - ไว้เสร็จเมื่อไหร่มาเจอกัน
- **Promise** จะทำงานไปก่อนเมื่อเสร็จงานแล้วจะกลับมาทำงาน
 - **then** รับ **function** ที่จะทำงานอีกครั้ง ว่าจะให้ทำอะไรบ้าง


```
fetch('asset/students.json').then(data => {  
  |   students = data.json()  
  | })
```


Fetch function

- คืนค่า **promise** ของ **response**
 - คืออ่านค่าที่ได้จากการเรียก **input parameter**

```
fetch('asset/students.json').then(response => {
```

```
Response {type: "basic", url: "http://127.0.0.1:5502/02.%20file%20loader/01.%20simple%20file%20loader.
t/students.json", redirected: false, status: 200, ok: true, ...} 
  body: (...)
  bodyUsed: false
  ▼ headers: Headers
    ► __proto__: Headers
    ok: true
    redirected: false
    status: 200
    statusText: "OK"
    type: "basic"
    url: "http://127.0.0.1:5502/02.%20file%20loader/01.%20simple%20file%20loader/asset/students.json"
    ► __proto__: Response
```



การจะอ่านค่า

- ต้องเรียก `response.json()`
 - เพื่ออ่านค่าในรูปแบบของ `json object`
 - เป็น `Promise` ที่เก็บค่า `json` ไว้
 - อ่านได้ครั้งเดียว ถ้าอ่านแล้วจะเอามาใช้ไม่ได้
 - เพราะว่าสัญญาจะรักกันแค่ครั้งเดียว

การจะอ่านค่า

- ต้องเรียก `response.json()`
 - เพื่ออ่านค่าในรูปแบบของ json object
 - เป็น **Promise** ที่เก็บค่า json ไว้
 - อ่านได้ครั้งเดียว ถ้าอ่านแล้วจะเอามาใช้ไม่ได้
 - เพราะว่าสัญญาจะรักกันแค่ครั้งเดียว
- ต้องใช้ **then** เพื่ออ่านค่า

```
function onLoad() {  
    fetch('asset/students.json').then(response => {  
        return response.json()  
    })  
    .then(data => {  
        console.log(data)  
    })  
}
```

- ลองดูผลลัพธ์

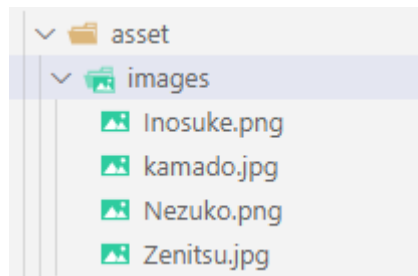
ที่นี้ได้อข้อมูลเป็น json object แล้ว

- นำมาเรียก **function** เพื่อกรอกข้อมูลลงตารางได้

```
function onLoad() {  
    fetch('asset/students.json').then(response => {  
        return response.json()  
    })  
    .then(data => {  
        let students = data  
        addStudentList(data)  
    })  
}
```

ถ้าเปลี่ยน username เป็นรูปละ

- Download file student2.json
- Copy file ภาพไปวางไว้ยังตำแหน่ง



```
[{  
  "name": "คามาดอ ทันจิโร่",  
  "imageLink": "asset/images/kamado.jpg",  
  "gender": "ชาย"  
},  
{  
  "name": "คามาดอ เนซึโกะ",  
  "imageLink": "asset/images/Nezuko.png",  
  "gender": "หญิง"  
},  
{  
  "name": "อาากาสึมะ เซ็นนิตสึ",  
  "imageLink": "asset/images/Zenitsu.jpg",  
  "gender": "ชาย"  
},  
{  
  "name": "ฮาชิบิระ อินสึเกะ",  
  "imageLink": "asset/images/Inosuke.png",  
  "gender": "ชาย"  
}]
```

Html ของภาพที่ต้องการหน้าตาเป็นอย่างไร

ได้ html แล้ว ก็ต้องลองสร้าง dom

แก้ไขตัว function

```
function addStudentToTable(index, student) {  
  const tableBody = document.getElementById('tableBody')  
  let row = document.createElement('tr')  
  let cell = document.createElement('th')  
  cell.setAttribute('score', 'row')  
  cell.innerHTML = index  
  row.appendChild(cell)  
  cell = document.createElement('td')  
  cell.innerHTML = student.name  
  row.appendChild(cell)  
  cell = document.createElement('td')  
  // cell.innerHTML = student.username  
  let img = document.createElement('img')  
  img.setAttribute('src', student.imageLink )  
  cell.appendChild(img)  
  row.appendChild(cell)  
  cell = document.createElement('td')  
  cell.innerHTML = student.gender  
  row.appendChild(cell)  
  tableBody.appendChild(row)  
}
```

```
function onLoad() {  
  
  fetch('asset/students2.json').then(response => {  
    return response.json()  
  })  
  .then(data => {  
    let students = data  
    addStudentList(data)  
  })  
}
```


การบ้าน

- ร้านดอกไม้แสนสวยของเรา ให้ลองเอาเนื้อหาทั้งหมดไปไว้ใน **json file** แล้วค่อยดึงข้อมูลมาแสดงในหน้าจอแสดงร้านดอกไม้ทั้งหมดอีกครั้ง