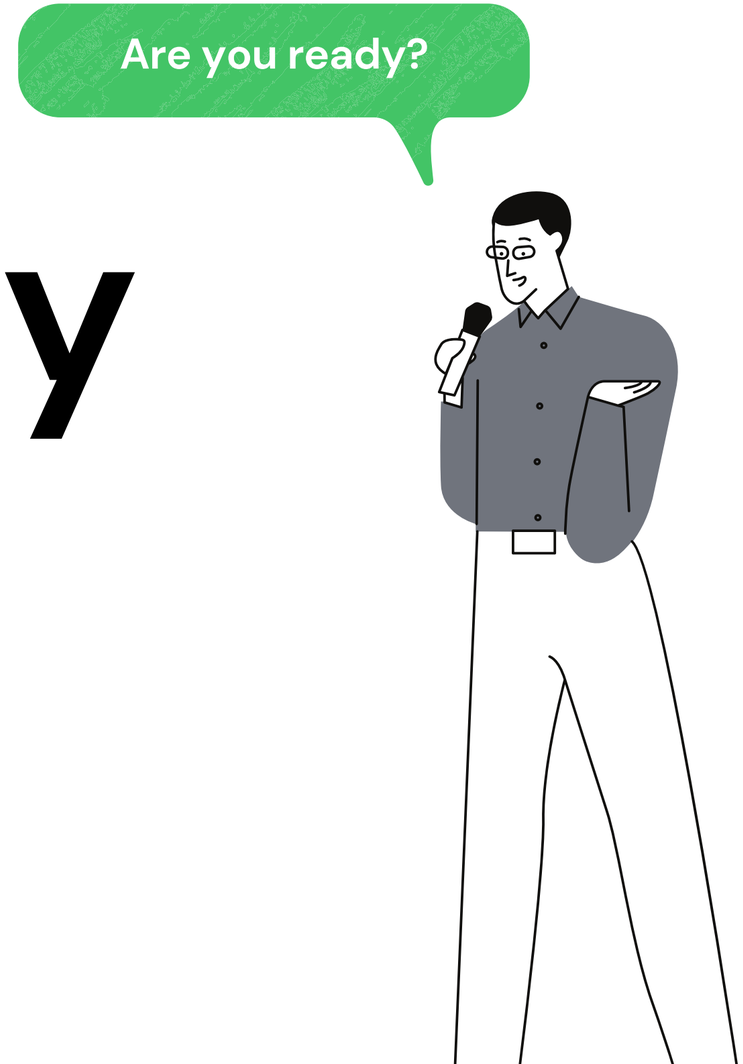
# jQuery



# jQuery คือ ?

**jquery** คือ **Library** ของ **JavaScript**

การใช้ jQuery ช่วยให้การเขียนชุดคำสั่งของภาษา JavaScript ซึ่งต้องพิมพ์ตัวอักษรจำนวนมาก สั้น และกระทัดรัดขึ้น

jQuery


JS

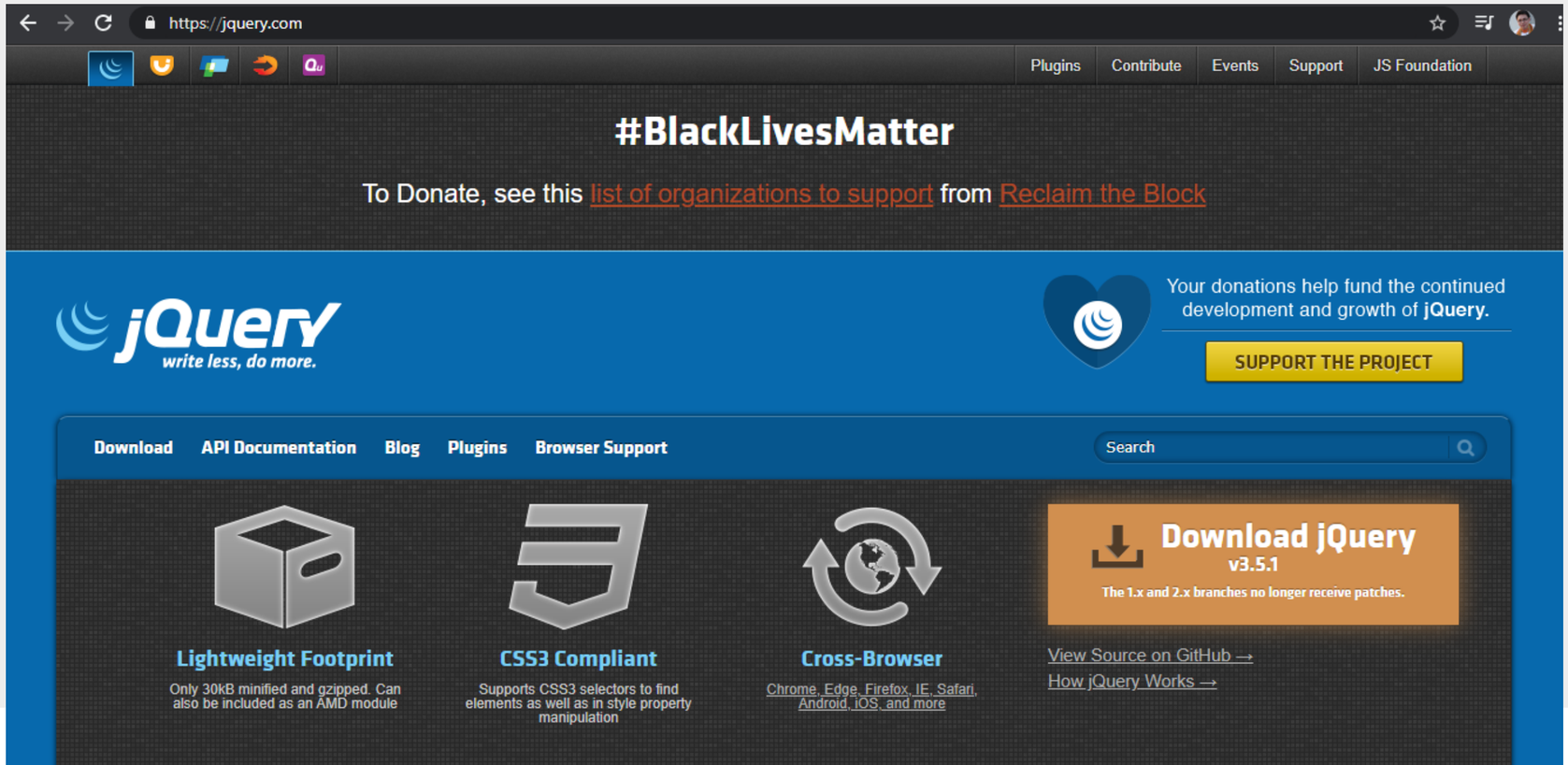
# jQuery

## คุณสมบัติของ

- สามารถใช้กับบราวเซอร์ตัวใดก็ได้ เช่น IE, Safari, FireFox, Chrome
- ช่วยให้การเลือกและจัดการเอลิเมนต์ของ HTML สามารถแก้ไขคุณสมบัติได้ง่ายขึ้น
- ช่วยให้จัดการอีเวนต์ (Event) ได้ง่ายขึ้น โดยไม่ต้องตรวจสอบบราวเซอร์ก่อน
- ใช้ Ajax ในการโหลดข้อมูลของเว็บเพจ โดยไม่ต้องโหลดหน้าเว็บทั้งหน้า
- มี Plugin ให้เลือกใช้จำนวนมาก
- มีรูปแบบการเขียนที่สั้นกว่า JavaScript

# jQuery

## การติดตั้ง



# jQuery

## การนำไปใช้

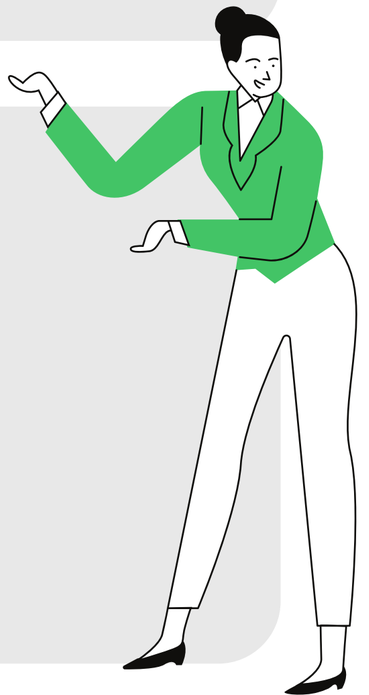
- นำไฟล์ไลบรารีวางไว้ภายในโฟลเดอร์ของเว็บไซต์
- กำหนดที่อยู่สำหรับเรียกใช้งานไลบรารีในส่วนของแท็ก Header

```
<html>
    <head>
        <script src="jquery-3.5.1.min.js" type="text/javascript"></script>
        <script src="myScript.js"></script>
    </head>
    <body><body>
</html>
```

# jQuery VS JavaScript

```
<body>
    <input id=“txtName” />
    <script>
        document.getElementById("txtName").value = "Thailand";
    </script>
</body>
```

```
<body>
    <input id="txtName" />
    <script>
        jQuery("#txtName").val("Thailand");
        $("#txtName").val("Thailand");
    </script>
</body>
```



# Ready

```
window.onload = function(){

    $("#myId").text(“Hello jQuery”);

}
```

```
$(document).ready(function(){

    $("#myId").text("Hello jQuery");

});
```

```
$(function() {

    $("#myId").text("Hello jQuery");

});
```

## OnClick

```
<button type="button" id="myBtn">Click me</button>

<script>
	$("#myBtn").click(function() {
		alert( "OK!" );
	});
</script>
```

# Ajax

## คืออะไร ?

**หลักการทำงานพื้นฐาน**

- ทำการร้องขอข้อมูลจากเซิร์ฟเวอร์ (Server) ซึ่งเรียกว่าการ Request
- รอการตอบสนองจากเซิร์ฟเวอร์ (Server) และ เมื่อ Ajax ได้รับการตอบรับจากเซิร์ฟเวอร์ จึงเขียนชุดคำสั่งเพื่อนำข้อมูลที่ได้รับไปใช้งาน เรียกว่า Asynchronous

# Ajax

เริ่มต้นใช้งาน

```
var  parm = new Object();
      parm.id = '123';
var JSON_DATA = JSON.stringify(parm);
```

```
$.ajax({
      url: URL ,
      type: "POST" ,
      dataType: 'json' ,
      contentType: 'application/json',
      data: JSON_DATA ,
      async: true,
      success: function(data) {
         var result = $.parseJSON(data);
         //code
      }
   });
```

//ประเภทข้อมูลที่ response
//ประเภทข้อมูลที่ request

# JSON

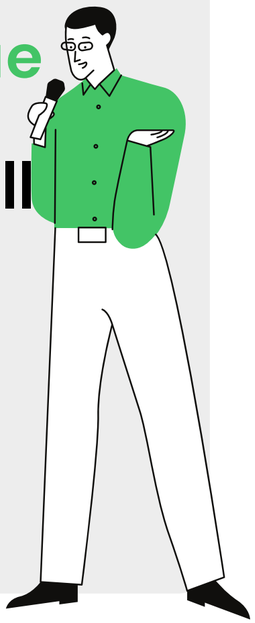
## คืออะไร ?

- ย่อมาจาก  JavaScript Object Notation
- เป็นรูปแบบสำหรับการแลกเปลี่ยนข้อมูลที่มีขนาดเล็ก

# JSON

## ส่วนประกอบของ JSON

- **Object** จะใช้เครื่องหมาย { } โดยจะมี { **members** } อยู่ในเครื่องหมาย
- **Members** จะประกอบด้วย **pair** หรือ **pair** , **members** อีกก็ได้
- **Pair** จะมีการประกาศตัวแปรและข้อมูล (**Key** : **Value**) ตามรูปแบบ **string** : **value**
- **Value** สามารถมีค่าเป็น **string**, **number**, **object**, **array**, **true**, **false** และ **null**
- **Array** จะใช้เครื่องหมาย [ ] โดยจะมี [ **elements** ] อยู่ในเครื่องหมาย
- **Elements** จะประกอบด้วย **value** หรือ **value** , **elements** อีกก็ได้



# JSON

## JSON Syntax

**JSON Data – A Name and a Value – (pair)**

"name" : "John“

JSON – Key ต้องเป็น **String** เท่านั้น

JavaScript – key อาจเป็น **String , number , identifier names**

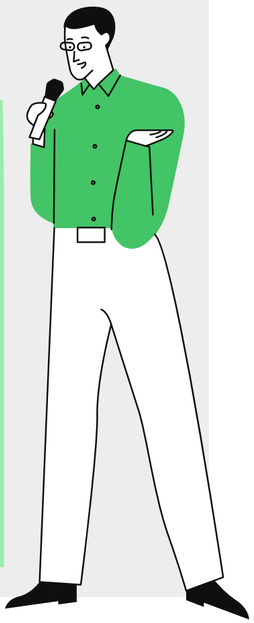
*JSON และ JavaScript Object มีรูปคล้ายคลึงกัน

**JSON**

{ "name" : "John" }

**JavaScript**

{ name : "John" }



# JSON

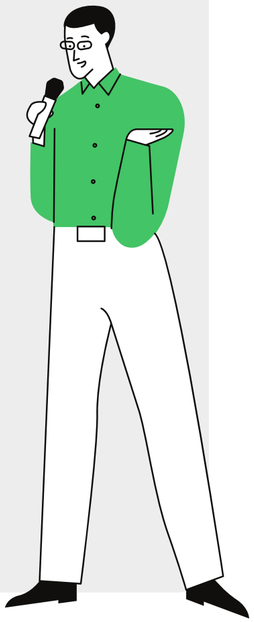
## JSON Syntax

**JSON Uses JavaScript Syntax**

```
var person = {
    name : "John",
    age : 31,
    city : "New York"
};
```

```
person.name
//return "John"
```

```
person["name"]
//return "John"
```



# JSON

## JSON Syntax

### JSON Strings

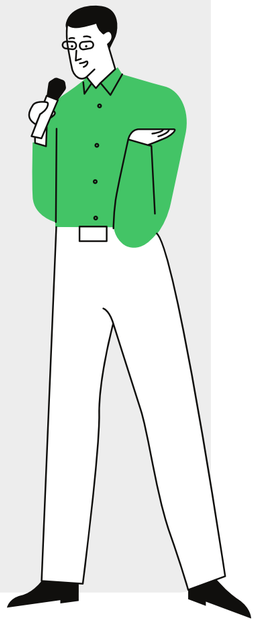
{ "name" : "John" }

### JSON Numbers

{ "age" : "30" }

### JSON Objects

{ "employee" : { "name":"John" , "age":30  , "city" : "New York"} }



# JSON

## JSON Syntax

**JSON Array**

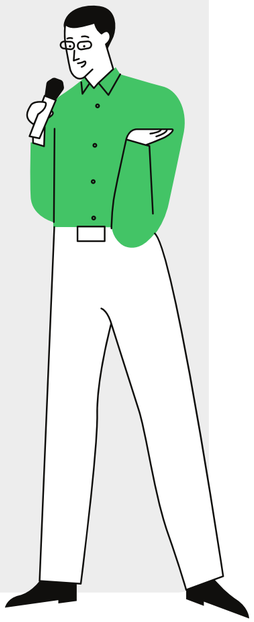
{ "employees" : ["John" , "Anna" , "Peter" ] }

**JSON Booleans**

{ "sale" : true }

**JSON null**

{ "middlename" : null }



# Parsing JSON

JSON

1

```
'{
    "name":"John",
    "age":30,
    "city":"New York"
}'
```

Convert

2

```
var obj = JSON.parse('{
    "name":"John",
    "age":30,
    "city":"New York"
}' );
```

JS Object

3

```
var obj = {
    name : "John",
    age : 30 ,
    city : "New York"
};
```

# Stringify a JavaScript Object

JSON

1

```
{
    name :"John",
    age : 30,
    city : "New York"
}
```

Convert

2

```
var strJson = JSON.stringify({
name :"John",
age : 30,
city : "New York"
});
```

JS Object

3

```
var strJson = '{
    "name" : "John",
    "age" : 30 ,
    "city" : "New York"
}';
```

# JSON

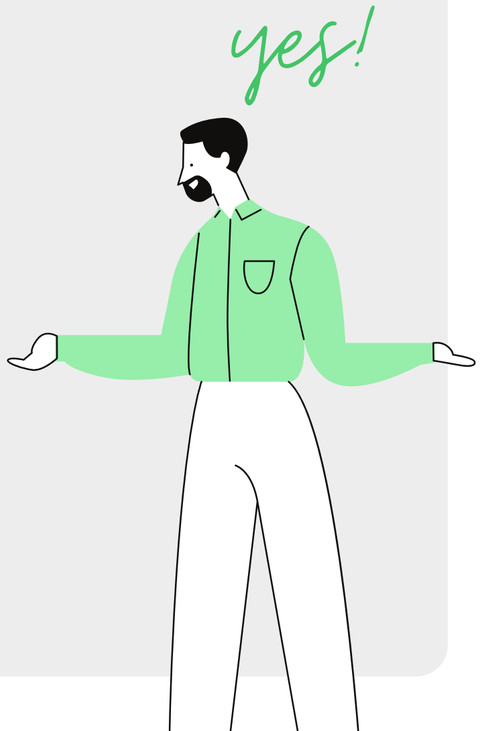
## เรียกใช้ค่าจาก Object

**{ "name":"John", "age":30, "car":null }**

**การรับค่า value จาก Object**

```
myObj = { "name":"John", "age":30, "car":null };
x = myObj.name;                                    //x = "John"

myObj = { "name":"John", "age":30, "car":null };
x = myObj["name"];                                 //x = "John"
```



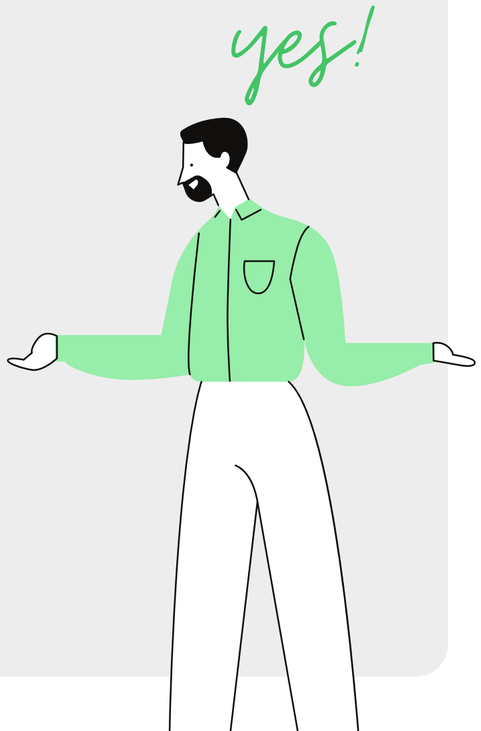
# JSON

## Looping an Object

**myObj = { "name":"John", "age":30, "car":null }**

```
for (x in myObj) {
      document.getElementById("demo").innerHTML += x;                  //Result : name age car
}

for (x in myObj) {
      document.getElementById("demo").innerHTML += myObj[x];       //Result : John 30 null
}
```



# JSON

## Nested JSON Objects

```
myObj = {
    "name" : "John",
     "age" : 30,
    "cars" : {
        "cars" : "Ford",
        "car2" : "BMW",
        "car3" : "Honda"
    }
}
```

### Get Value

```
x = myObj.cars.car2;              // x = "BMW"
x = myObj.cars["car2"];           // x = "BMW"
```

### Modify values

```
myObj.cars.car2 = "Mercedes";
myObj.cars["car2"] = "Mercedes";
```

### Delete value

```
delete myObj.cars.car2;
```

# Thank You