Web Application Development

Javascript (Frontend Development 3)

Instructor: Thanh Binh Nguyen

February 1st, 2020

S3Lab
Smart Software System Laboratory


“A successful website does three things:
It attracts the right kinds of visitors.
Guides them to the main services or product you offer.
Collect Contact details for future ongoing relation.”
– Mohamed Saad

JavaScript

Overview

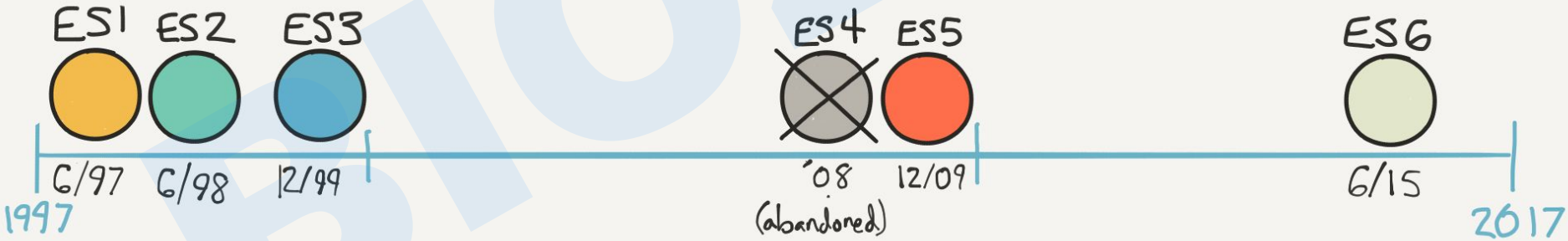
No Relate to Java

Loosely typed

- A programming language of HTML and the Web based Application.
- **ECMAScript** is the name of the international standard that defines javascript
- Developed by Technical Committee 39 (TC-39) of Ecma International
- Issued as a Ecma-262 and ISO/IEC 16262
- Not part of W3C

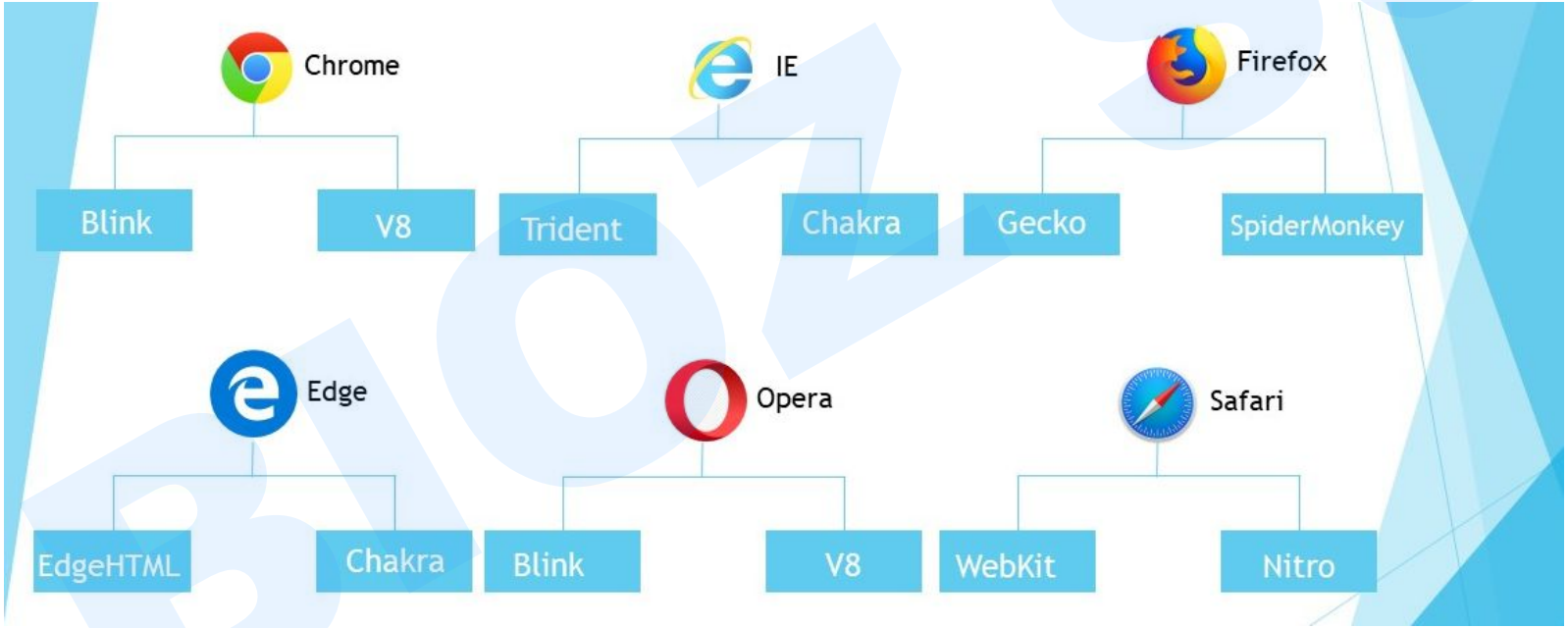
Script Language

Dynamic Programming Language



JavaScript

Rendering & Javascript Engines



JavaScript

11 things you can build with JS

- Websites -> any modern web site is running javascript on some level.
- Web Applications -> Google Map, ...
- Presentations -> Web-based slide: slides.com, Reveal JS
- Server Applications -> Node JS based Server
- Web Servers -> NodeJS + EXpressJS
- Games -> browser-based games: working on HTML5 canvas
- Art -> canvas element, 3D shapes rendering, ...
- Smartwatch Apps -> Pebble has created pebble.js
- Mobile Apps -> Web responsive, React Native, PWA (Progressive Web App), ...
- Desktop Apps -> electronjs.org
- Flying Robots -> using nodejs

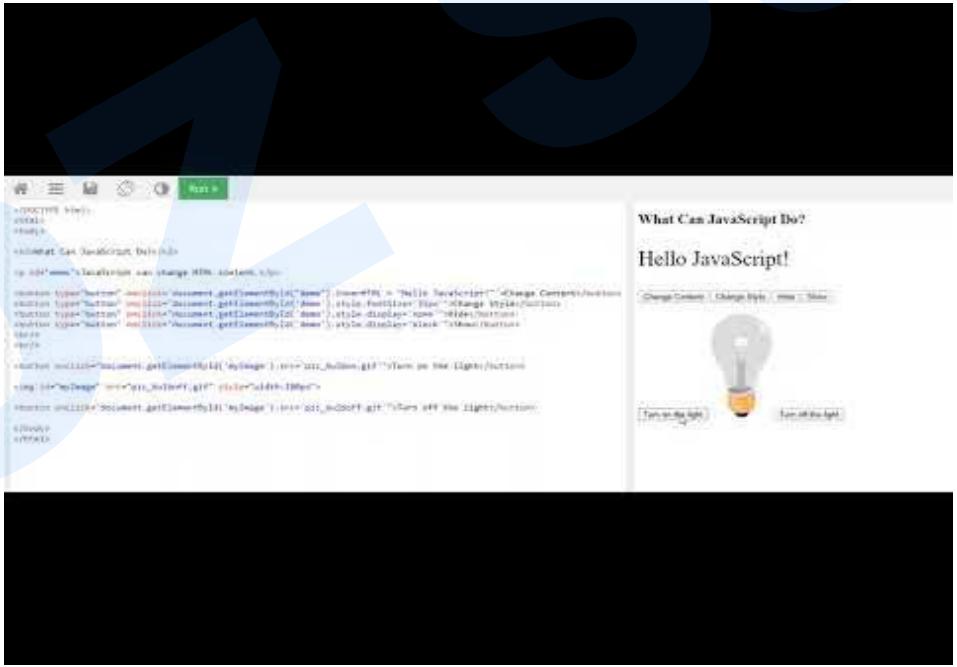
JavaScript

JS can do: 5 core things to HTML

- Change HTML content
- Change HTML attributes.
- Change CSS style
- Hide HTML elements
- Show hidden HTML elements

Document Object Model

On the fly



JavaScript

JS can do: the others

```
<!--[if lt IE 9]>
    <script src="html5shim.js"></script>
<![endif]-->
```

- React to user input
- Interact to the browser
- Request or submit the content and information from / to the server.
- Test for browser's individual features and capabilities
- Fill in gaps where a browser's built-in functionality falls short, or add some of the features found in newer browsers to older browsers (**shims** or **polyfills**).
- ...

JavaScript

How to insert

Embedded & External

- In **<head>**
- In **<body>**
- In an external file:
  - **<script src="myScript.js"></script>**
- In an external url:
  - **<script src="https://www.w3schools.com/js/myScript.js"></script>**
- In an external folder:
  - **<script src="/js/myScript.js"></script>**

```
<!DOCTYPE html>
<html>
<head>
<script>
  function myFunction0() {
    document.getElementById("demo").innerHTML = "Paragraph changed 0.";
  }
</script>
</head>
<body>
  <h2>JavaScript in Head</h2>

  <p id="demo">A Paragraph.</p>

  <button type="button" onclick="myFunction0()">Change 0</button>
  <button type="button" onclick="myFunction1()">Change 1</button>

  <script>
    function myFunction1() {
      document.getElementById("demo").innerHTML = "Paragraph changed 1.";
    }
  </script>
</body>
</html>
```

In Head

In Body

JavaScript

The way to "display" the data

- Writing into an HTML element, using **innerHTML**.
- Writing into the HTML output using **document.write()**.
- Writing into an alert box, using **window.alert()**.
- Writing into the browser console, using **console.log()**.

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

<h2>Hello javascript</h2>

<button onclick="window.print()">Print this page</button>

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

<script>
  document.getElementById("demo").innerHTML = "display into HTML element";
  document.write("display by document");
  window.alert("display by alert box");
  console.log("this is debug log");
</script>

</body>
</html>
```

Chrome: Ctr - Shift - I
Using Developer Tool to see

JavaScript

The basics in syntax

- Javascript is case-sensitive: var myLove; # var mylove;
- A script is made up of a series of **statements**.
- Using a semicolon **";"** to end of the command.
- **Comments**: using **//** for single line, **/* */** for multiple-line.
- **Variables**: an information container. Variables name may contain **letters**, **digits** and **underscores**. Must start with a **letter** or an **underscore**.
- **Data Types**: **undefined** (variable without initialized), **null** (no inherent value), **numbers** (3, 4, 5.6), **string** ("abc"), **boolean** (true | false), **array** ([a, b, c])

JavaScript

The basics in syntax: Comparison Operator

- == Is equal to
- != Is not equal to
- === Is identical to (equal to and of the same data type)
- !== Is not identical to
- > Is greater than
- >= Is greater than or equal to
- < Is less than
- <= Is less than or equal to

JavaScript

The basics in syntax: Mathematical Operator

- + add
- - subtract
- * multiply
- / divide
- ++ increases the value of a number by 1
- -- decreases the value of a number by 1
- % modulus (Division Remainder)
- ** Exponentiation

JavaScript

The basics in syntax: Logical and Type Operators

Operator	Description
&&	logical and
||	logical or
!	logical not

Operator	Description
typeof	Returns the type of a variable
instanceof	Returns true if an object is an instance of an object type

JavaScript

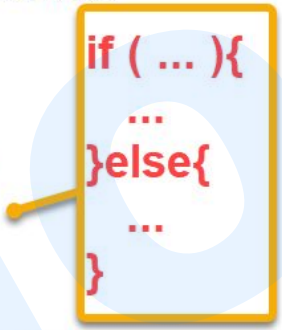
The basics in syntax: Bitwise Operators

Operator	Description	Example	Same as	Result	Decimal
&	AND	5 & 1	0101 & 0001	0001	1
|	OR	5 | 1	0101 | 0001	0101	5
~	NOT	~ 5	~0101	1010	10
^	XOR	5 ^ 1	0101 ^ 0001	0100	4
<<	Zero fill left shift	5 << 1	0101 << 1	1010	10
>>	Signed right shift	5 >> 1	0101 >> 1	0010	2
>>>	Zero fill right shift	5 >>> 1	0101 >>> 1	0010	2

JavaScript

The basics in syntax: If/else and switch case

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

<p>Click the button to get a time-based greeting:</p>

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

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

<script>
function myFunction() {
  var greeting;
  var time = new Date().getHours();
  if (time < 10) {
    greeting = "Good morning";
  } else if (time < 20) {
    greeting = "Good day";
  } else {
    greeting = "Good evening";
  }
  document.getElementById("demo").innerHTML = greeting;
}
</script>

</body>
</html>
```




```
switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

JavaScript

The basics in syntax: loops

```
var cars = ["BMW", "Volvo", "Saab", "Ford"];
var i = 0;
var text = "";

while (cars[i]) {
  text += cars[i] + "<br>";
  i++;
}
```

While

```
do {
  text += "The number is " + i;
  i++;
}
while (i < 10);
```

Do / While

```
for (i = 0; i < 5; i++) {
  text += "The number is " + i + "<br>";
}
```

For

```
var cars = ['BMW', 'Volvo', 'Mini'];
var x;

for (x of cars) {
  document.write(x + "<br >");
}
```

For / Of

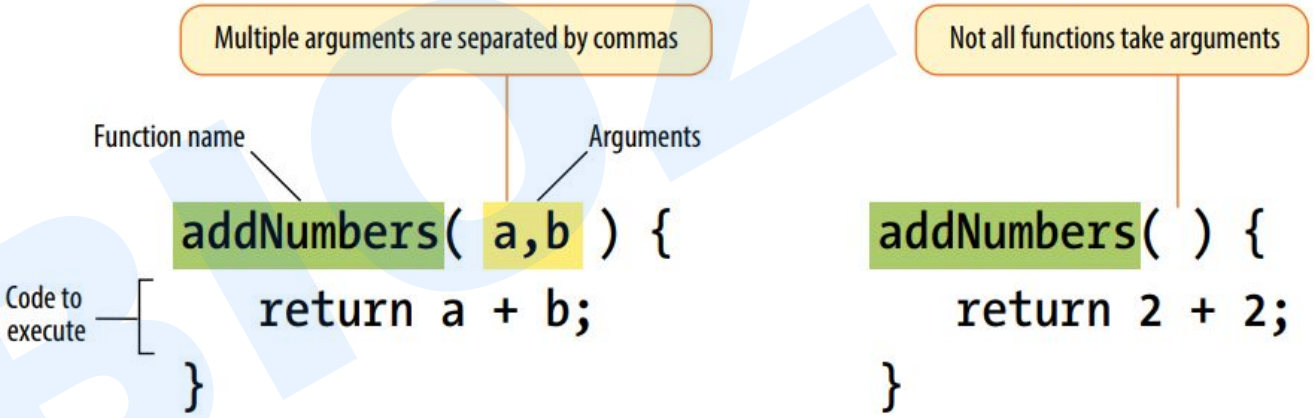
```
var person = {fname:"John", lname:"Doe", age:25};

var text = "";
var x;
for (x in person) {
  text += person[x];
}
```

For / In

JavaScript

The basics in syntax: functions

- **Native functions:** Date(), parseInt("123"), alert(""), confirm(""), prompt(""), ...
- **Custom functions**: defined by user



JavaScript

The basics in syntax: variable scope -> var, let, const

- **Globally scoped:**
  - Can be used by any scripts on page, defined outside of function with **var**. The variable is also a member of **window** object.
  - In Node js -> **global namespace object**: **global.abc = “”;**
- **Locally scoped**:
  - Defined inside of function with **var**.
  - Variable was defined by using **let** and **const**. Just have meaning in a block of code, that’s anything between two curly braces **{}**.

JavaScript

The basics in syntax: Browser or window Object

Property/method	Description
`event`	Represents the state of an event
`history`	Contains the URLs the user has visited within a browser window
`location`	Gives read/write access to the URI in the address bar
`status`	Sets or returns the text in the status bar of the window
`alert()`	Displays an alert box with a specified message and an OK button
`close()`	Closes the current window
`confirm()`	Displays a dialog box with a specified message and an OK and a Cancel button
`focus()`	Sets focus on the current window

JavaScript

The basics in syntax: Browser or window Object

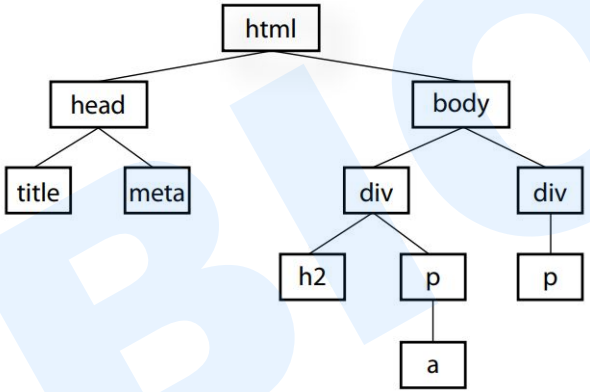
- **Handle the event by 3 ways:**
  - HTML attribute:
    - <body onclick=”my();”>
  - As a method:
    - Window.onclick = my**;**
  - **addEventListener:**
    - window.addEventListener(“click”, my);
    - window.addEventListener(“Click”, function(e){});

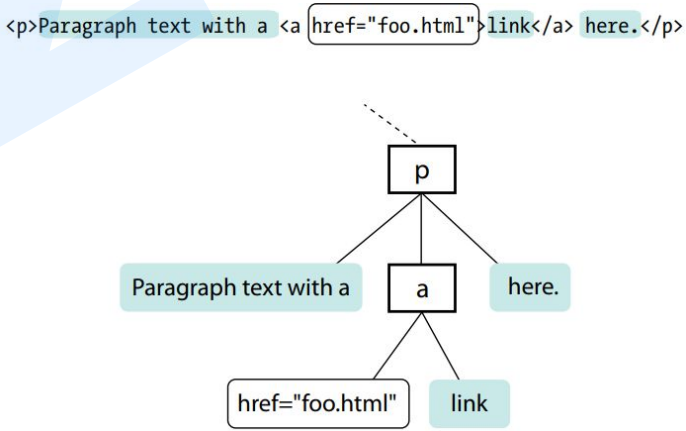
Event handler	Event description
onblur	An element loses focus
onchange	The content of a form field changes
onclick	The mouse clicks an object
onerror	An error occurs when the document or an image loads
onfocus	An element gets focus
onkeydown	A key on the keyboard is pressed
onkeypress	A key on the keyboard is pressed or held down
onkeyup	A key on the keyboard is released
onload	A page or an image is finished loading
onmousedown	A mouse button is pressed
onmousemove	The mouse is moved
onmouseout	The mouse is moved off an element
onmouseover	The mouse is moved over an element
onmouseup	A mouse button is released
onsubmit	The submit button is clicked in a form

JavaScript

The basics in syntax: DOM - a collection of nodes

- A programming interface (an API) for HTML and XML pages. DOM serves as a map to all the elements on a page. We can use it to find elements by their names or attributes, then add, modify, or delete elements and their content.





JavaScript

The basics in syntax: DOM - Accessing DOM nodes

- var paragraphs = document.**getElementsByTagName**("p");
- var photo = document.**getElementById**("lead-photo");
  - photo.**setAttribute**("src", "lespaul.jpg");
  - photo.**style**.backgroundColor = "#f58220";
- var firstColumn = document.**getElementsByClassName**("column-a");
  - firstColumn.**innerHTML** = " This is our intro text ";
- var sidebarPara = document.**querySelectorAll**(".sidebar p");

JavaScript

The basics in syntax: DOM - Adding and Removing elements

- Our paragraph text
  var ourDiv = document.getElementById("our-div");
  var ourParam = document.getElementById("our-paragraph");

  var newParagraph = document.**createElement**("p");
  var copy = document.**createTextNode**("Hello, world!");
  newParagraph.**appendChild**(copy);
  ourDiv.**appendChild**(newParagraph);

  ourDiv.**insertBefore**(newParagraph, ourParam);
  ourDiv.**replaceChild**(newParagraph, ourParam);
  parentDiv.**removeChild**(ourParam);

JavaScript

ES5 # ES6

- Block scope with **let**, **const**
- Arrow Function
  - var x = function (x, y) { return x * y } -> ES5
  - const x = (x, y) **=>** x * y **or** => { return x * y } -> ES6
- Classes: a type of function
  - Class car {

```
        constructor(brand){
            This.brand = brand;
        }
}
```

JavaScript

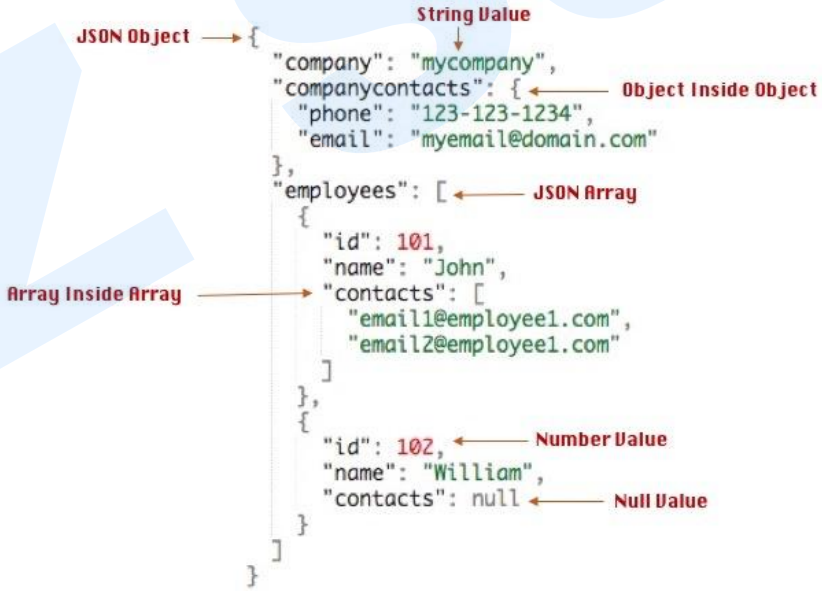
ES5 # ES6

- Default parameter values:
  - function myFunction(x, **y = 10**) {
    return x + y;
    }
    myFunction(5); // will return 15
- Array.find(), Array.findIndex()
  - var numbers = [4, 9, 16, 25, 29];
    function myFunction(value, index, array) {
    return value > 18;
    }
  - var first = numbers.**find**(myFunction);
  - var first = numbers.**findIndex**(myFunction);
- Exponentiation **

JavaScript

JavaScript Object Notation

- JSON is a format for storing and transporting data
- JSON is often used when data is sent from a server to a client. Ex. Restful API
- JSON is a lightweight data interchange format
- JSON is a language independent.
- JSON is “self-describing” and easy to understand.



JavaScript

JavaScript Object Notation

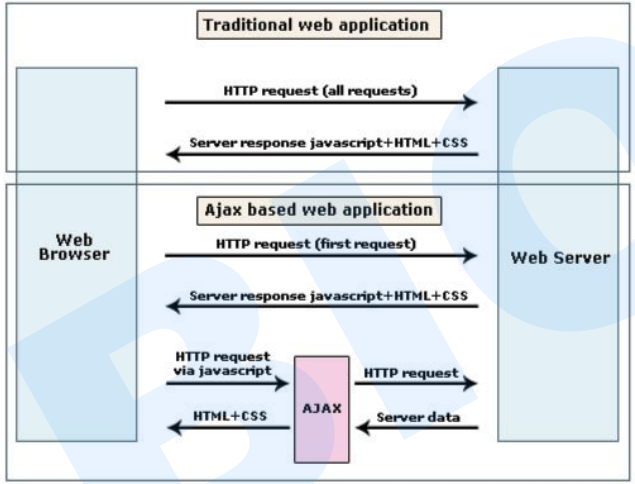
```
const jsonObj = {
    id: 1,
    name: 'Hamburger',
    icon: '🍔',
    type: 'Food'
};

// convert JSON object to string
const jsonStr = JSON.stringify(jsonObj);

console.log(jsonStr);
// '{"id":1,"name":"Hamburger","icon":"🍔","type":"Food"}'

//convert string back to JSON object
console.log(JSON.parse(jsonStr));
// {id: 1, name: "Hamburger", icon: "🍔", type: "Food"}
```

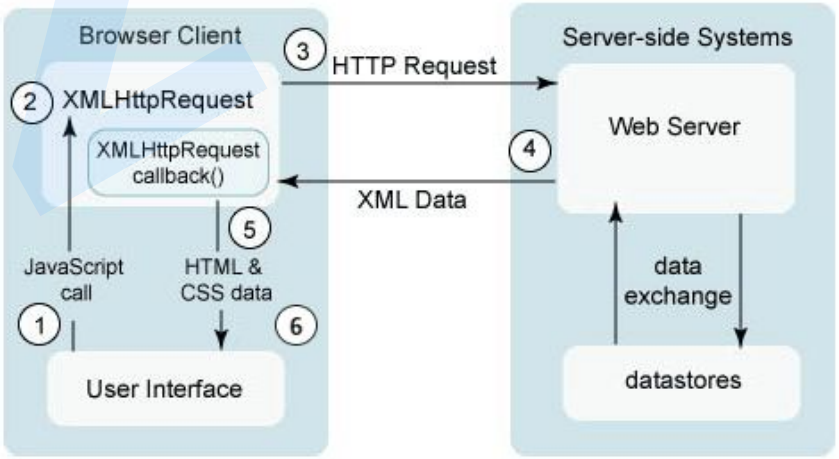
https://attacomsian.com/blog/json-parse-stringify

JavaScript

Asynchronous JavaScript And XML

- **AJAX**: a combination of **HTML**, **CSS**, the **DOM**, and Javascript, include **XMLHttpRequest** object that allows data to be transferred asynchronously. Ajax may use **XML** for data, but it has become more common to use **JSON**. https://www.w3schools.com/js/js_ajax_intro.asp





JavaScript

Library - JQuery

- Written in 2005 by John Resig
- jQuery is a fast, small, and feature-rich JavaScript library
- It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

JavaScript

Library - JQuery: how to use?

- Download from https://jquery.com/download/ and put it to your web application.
- Using the online CDN:
  - <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
- Start the coding:
  - Work with DOM simply by CSS selector.
    - var paragraph = document.getElementById("status"); -> Normal JS
    - var paragraph = $("#status"); -> JQuery
    - $("button.continue").html("Next Step...")
  - var hiddenBox = $("#banner-message");
    $("#button-container button").on("click", function(event) {
        hiddenBox.show();
    });

JavaScript

Library - JQuery: how to use?

```
$.ajax({
  url: "/api/getWeather",
  data: {
    zipcode: 97201
  },
  success: function( result ) {
    $( "#weather-temp" ).html( "<strong>" + result + "</strong> degrees" );
  }
});
```

Q & A

Cảm ơn đã theo dõi

Hy vọng cùng nhau đi đến thành công.