

CÔNG NGHỆ WEB AN TOÀN

Bài 5.1 Xử lý nâng cao

1

JSON và XML

2

CSS, DOM và AJAX

3

jQuery và Bootstrap

4

Web Service

Mục tiêu bài học

1. Biết và áp dụng được một số kỹ thuật, công nghệ hiện đại trong phát triển ứng dụng, dịch vụ web (DOM, CSS, XML, JSON, AJAX, RESTful API)
2. Lựa chọn, sử dụng thư viện thích hợp (jQuery, Bootstrap) để thực hiện các xử lý trong phát triển ứng dụng web.

1

JSON và XML

2

CSS, DOM và AJAX

3

jQuery và Bootstrap

4

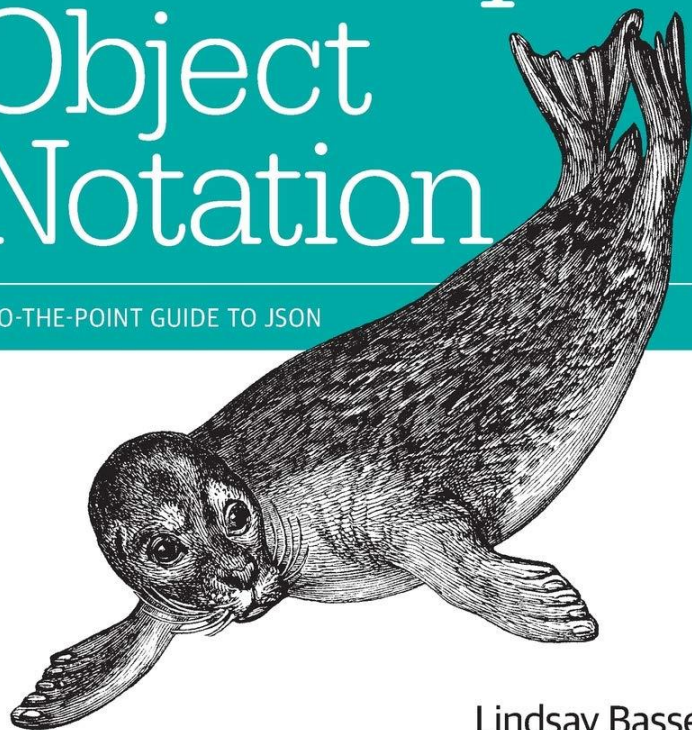
Web Service

Tài liệu tham khảo

O'REILLY®

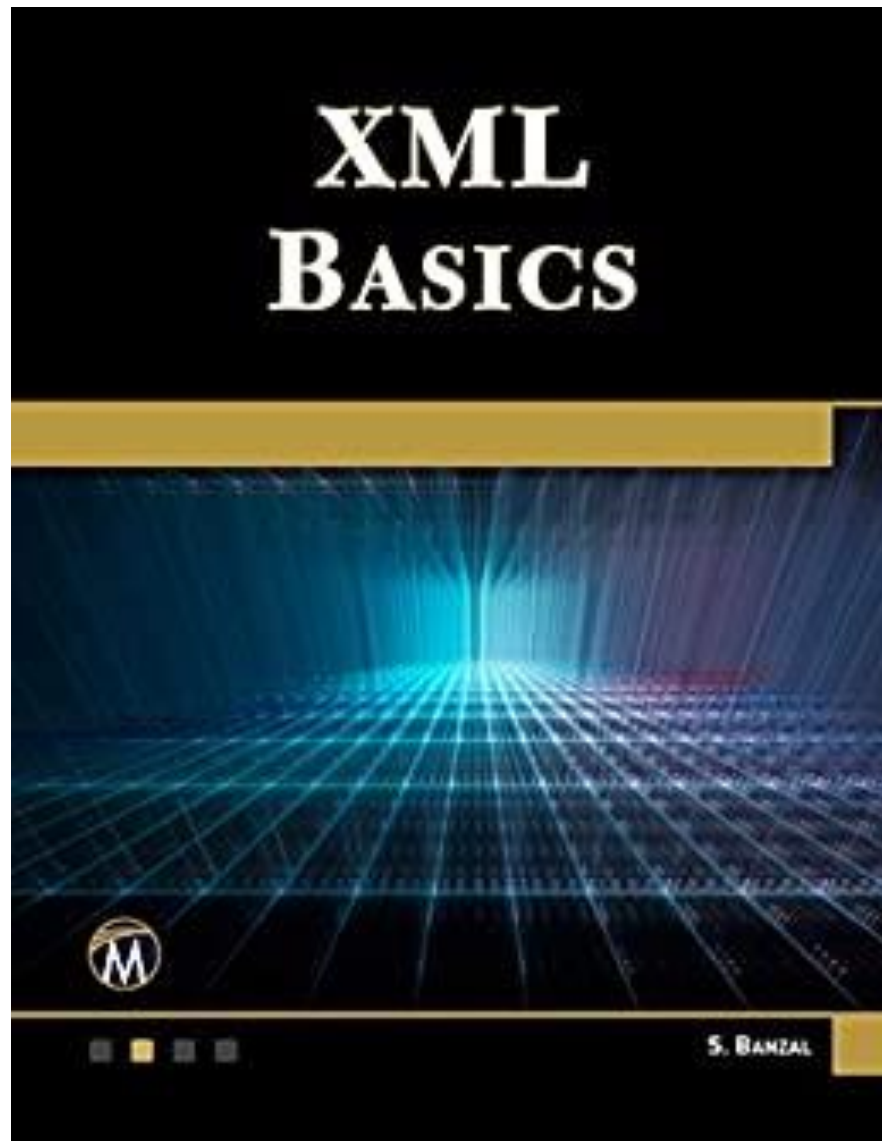
Introduction to JavaScript Object Notation

A TO-THE-POINT GUIDE TO JSON



Lindsay Bassett

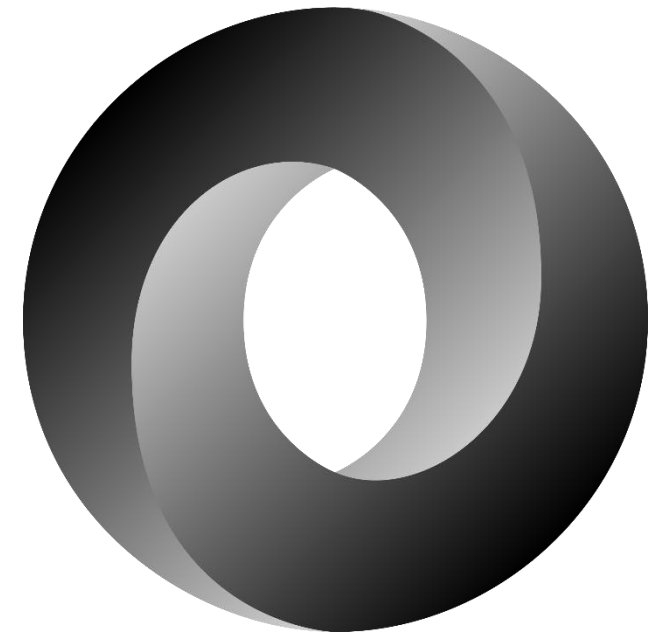
XML BASICS



Slide 5 of 68

JSON

- JSON = JavaScript Object Notation
- Là một chuẩn định dạng **dựa trên** cú pháp JavaScript Object Literals, sử dụng **văn bản** để biểu diễn dữ liệu có tính tương thích cao giữa các hệ thống khác nhau.
- JSON là độc lập với ngôn ngữ lập trình (cho dù trong tên gọi có "JavaScript")
- JSON chỉ có dữ liệu, không có phương thức (cho dù trong tên gọi có "Object")



Ví dụ: "course" object

```
{  
  "courseName" : "Công nghệ web an toàn",  
  "maxStudents" : 50,  
  "isFull" : false,  
  "location" : ["104-TA2", "305-TB3"],  
  "teacher" : {  
    "name" : "Trần Văn Bình",  
    "department" : "An toàn thông tin"  
  },  
  "prerequisite" : null  
}
```

Ví dụ: "course" object

```
{  
  "courseName" : "Công nghệ web an toàn",  
  "maxStudents" : 50,  
  "isFull" : false,  
  "location" : ["104-TA2", "305-TB3"],  
  "teacher" : {  
    "name" : "Trần Văn Bình",  
    "department" : "An toàn thông tin"  
  },  
  "prerequisite" : null  
}
```

Là OBJECT nên được đóng
bên trong cặp dấu ngoặc nhọn

Ví dụ: "course" object

```
{  
  "courseName" : "Công nghệ web an toàn",  
  "maxStudents" : 50,  
  "isFull" : false,  
  "location" : ["104-TA2", "305-TB3"],  
  "teacher" : {  
    "name" : "Trần Văn Bình",  
    "department"  
  },  
  "prerequisite"  
}
```

- Thuộc tính là một cặp NAME-VALUE
- Các thuộc tính phân tách bằng dấu phẩy
- NAME và VALUE phân tách bằng dấu 2 chấm
- NAME **bắt buộc** đặt trong cặp ngoặc kép

Ví dụ: "course" object

```
{  
  "courseName" : "Công nghệ web an toàn",  
  "maxStudents" : 50,  
  "isFull" : false,  
  "location" : ["104-TA2", "305-TB3"],  
  "teacher" : {  
    "name" : "Trần Văn Bình",  
    "department" : "An toàn thông tin"  
  },  
  "prerequisite" : null  
}
```

Các kiểu dữ liệu:

- string
- number
- boolean
- array
- object
- null

Ứng dụng của JSON

- JWT (Json Web Token)
- REST (REpresentational State Transfer) response
- Các giao thức cụ thể
 - OpenID Connect
 - Auth
 - ...
- Serialize các đối tượng
- Các ứng dụng khác

PHP JSON support

❑ Ví dụ `json_encode()`

```
$myObj = new stdClass();  
$myObj->name = "John";  
$myObj->age = 30;  
$myObj->city = "New York";  
$myJSON = json_encode($myObj);  
echo $myJSON;
```

❑ Kết quả

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



PHP JSON support

❑ Ví dụ `json_decode()`

```
$json = '{"name":"John", "age":30, "car":null}';  
$var = json_decode($json);  
var_dump($var);
```

❑ Kết quả

```
object(stdClass)#1 (3) {  
    ["name"]=> string(4) "John"  
    ["age"]=> int(30)  
    ["car"]=> NULL  
}
```



XML

- **XML = eXtensible Markup Language**
- Là ngôn ngữ đánh dấu để biểu diễn dữ liệu (không có thuộc tính định dạng)
- Các TAG là do người lập trình tự định nghĩa (HTML có tập TAG định trước)
- TAG phân biệt hoa/thường
- Để trao đổi dữ liệu giữa các ứng dụng
- Để tách biệt phần dữ liệu với phần định dạng của HTML.



Ví dụ XML

```
<?xml version='1.0' ?>
```

```
<course>
```

```
  <name>Công nghệ web an toàn</name>
```

```
  <maxStudents>50</maxStudents>
```

```
  <locations>
```

```
    <location type="lý thuyết">104-TA2</location>
```

```
    <location type="thực hành">305-TB3</location>
```

```
  </locations>
```

```
  <teacher>Trần Văn Bình</teacher>
```

```
</course>
```



Xử lý XML với PHP (1/2)

```
<?php
$xmlstr = <<<XMLSTR
<?xml version='1.0' ?>
<course>
  <name>Công nghệ web an toàn</name>
  <maxStudents>50</maxStudents>
  <locations>
    <location type="lý thuyết">104-TA2</location>
    <location type="thực hành">305-TB3</location>
  </locations>
  <teacher>Trần Văn Bình</teacher>
</course>
XMLSTR;
```



Xử lý XML với PHP (2/2)

```
<?php
```

```
$course = new SimpleXMLElement($xmlstr);
```

```
echo $course->teacher;
```



1

JSON và XML

2

CSS, DOM và AJAX

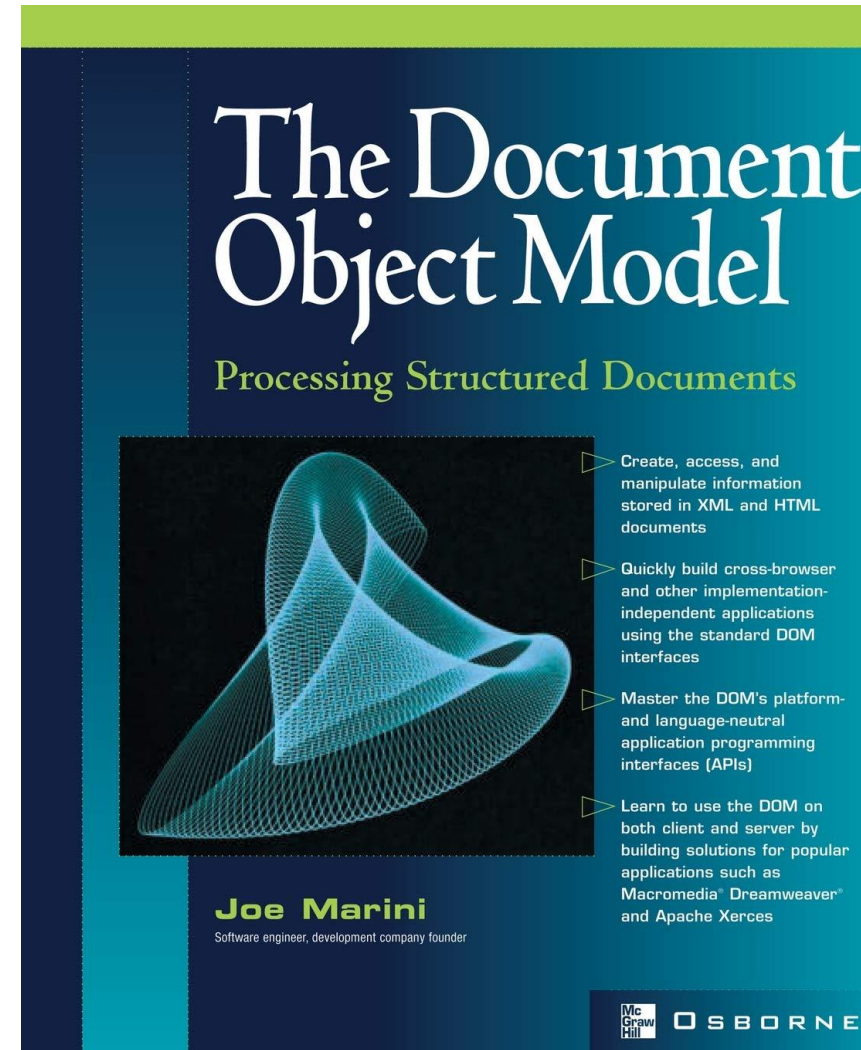
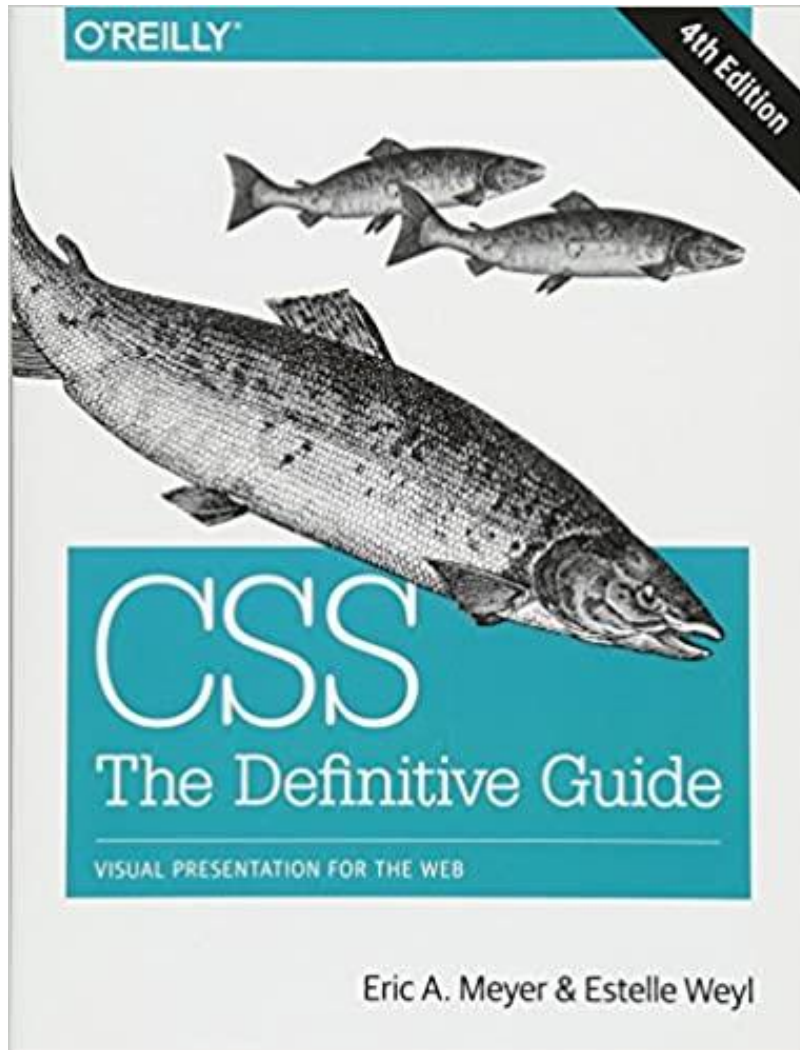
3

jQuery và Bootstrap

4

Web Service

Tài liệu tham khảo



CSS - Cascading Style Sheets

- **CSS** là một ngôn ngữ đánh dấu (markup language) để mô tả cách thức định dạng, hiển thị các thành phần của trang web
- Dùng bởi hầu hết website
- CSS, CSS2, CSS3...

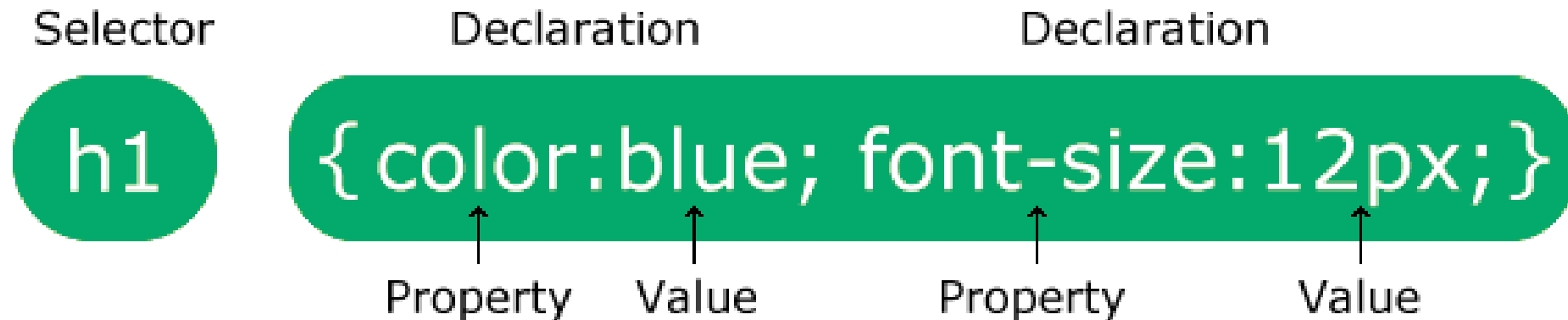


Cascading + Style Sheets

- **Style Sheet** = Tập hợp các quy tắc (rule) quy định cách trình duyệt hiển thị các phần tử HTML.
- **Cascading** = Giải quyết xung đột khi có nhiều quy tắc cùng áp dụng cho một phần tử. Cụ thể, quy tắc sau cùng trong cascade sẽ có mức ưu tiên cao nhất.

CSS Syntax

- **Một CSS rule gồm 2 phần:**
 - một Selector
 - một nhóm gồm 1 hoặc nhiều Declaration
- **Ví dụ:** "tất cả các thẻ <h1> trong trang được hiển thị với chữ màu xanh dương, cỡ chữ 12"



CSS Tutorial

CSS HOME

CSS Introduction

CSS Syntax

CSS Selectors

CSS How To

CSS Comments

CSS Colors

CSS Backgrounds

CSS Borders

CSS Margins

CSS Padding

CSS Height/Width

CSS Box Model

CSS Outline

CSS Text

CSS Fonts

CSS Icons

CSS Syntax

[< Previous](#)

A CSS rule consists of a se

CSS Syntax

Selector

h1

Decla

{ color

Property

The selector points to the H

Try it yourself

[https://www.w3schools.com/
Css/default.asp](https://www.w3schools.com/Css/default.asp)

Live Demo

https://www.w3schools.com/css/tryit.asp?filename=trycss_default

CSS Types

1. External Style Sheet
2. Internal Style Sheet
3. Inline Style



CSS Types

1. External Style Sheet
2. Internal Style Sheet
3. Inline Style

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

CSS Types

1. External Style Sheet
2. Internal Style Sheet
3. Inline S

```
<head>
  <style type="text/css">
    body {background-color: blue;}
    h1 {font-color: yellow; font-size: 16px;}
    p {margin-left: 5px;}
  </style>
</head>
```

CSS Types

1. External Style Sheet
2. Internal Style Sheet
3. Inline Style



```
<body style="background-color: blue;">  
  <h1 style="font-color: yellow; font-size: 16px;">Hello</h1>  
  <p style="margin-left: 5px;">  
    This is my first CSS page.  
  </p>  
</body>
```

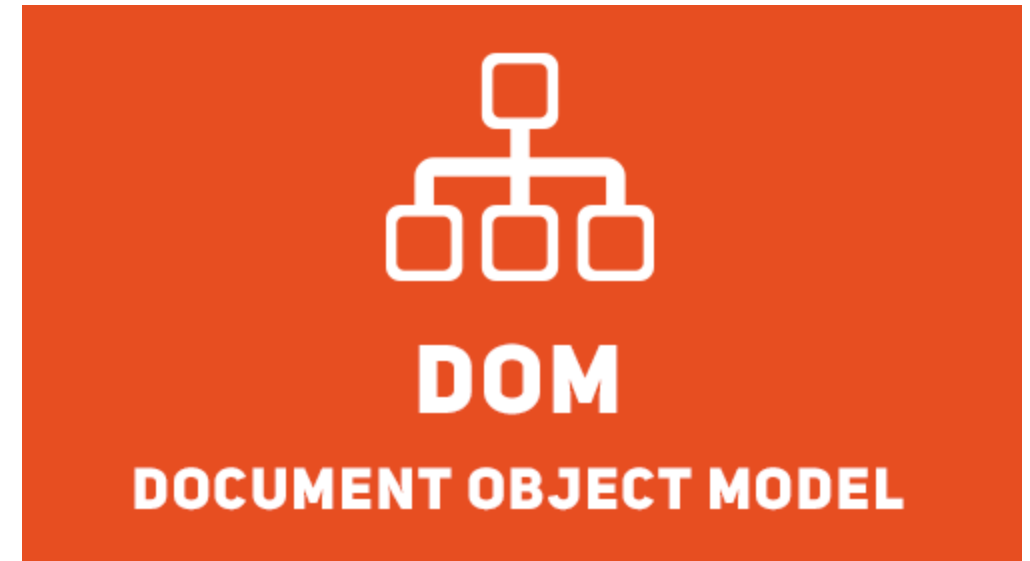
Khả năng của CSS

- Nhiều thuộc tính, hiệu ứng định dạng và trình bày
- Xây dựng 1 lần, dùng nhiều lần
- Chia sẻ, kế thừa (Bootstrap chẳng hạn)
- Tạo sự nhất quán về định dạng



DOM

- **DOM = Document Object Model**
- Các mô hình tài liệu
 - Linear model: một dãy các ký tự
 - Tree model: một cấu trúc cây gồm nhánh, nút
 - Object model: một đối tượng, và mỗi phần tài liệu cũng là một đối tượng

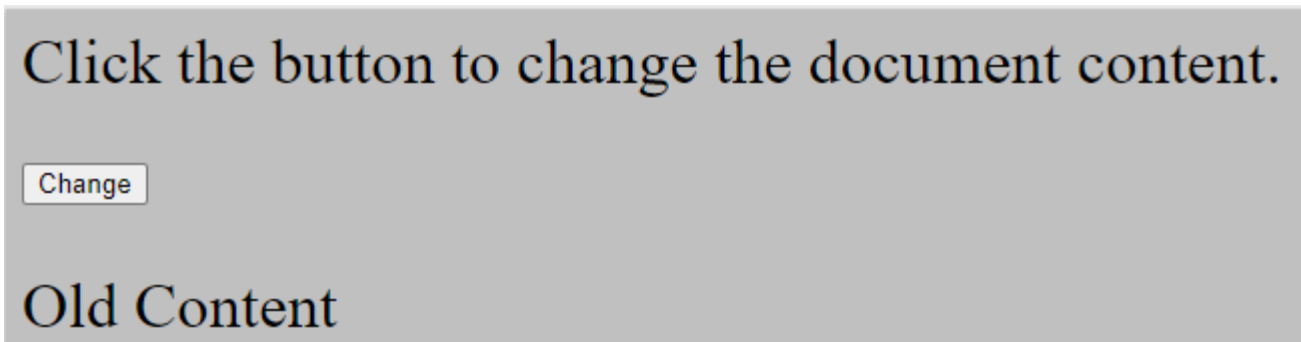


HTML Document Example

```
<html>
  <body>
    <p>Click the button to change the document content.</p>
    <button onclick="myFunction()">Change</button>
    <p id="demo">Old Content</p>
    <script>
      function myFunction() {
        document.getElementById("demo").innerHTML = "New New New....";
      }
    </script>
  </body>
</html>
```

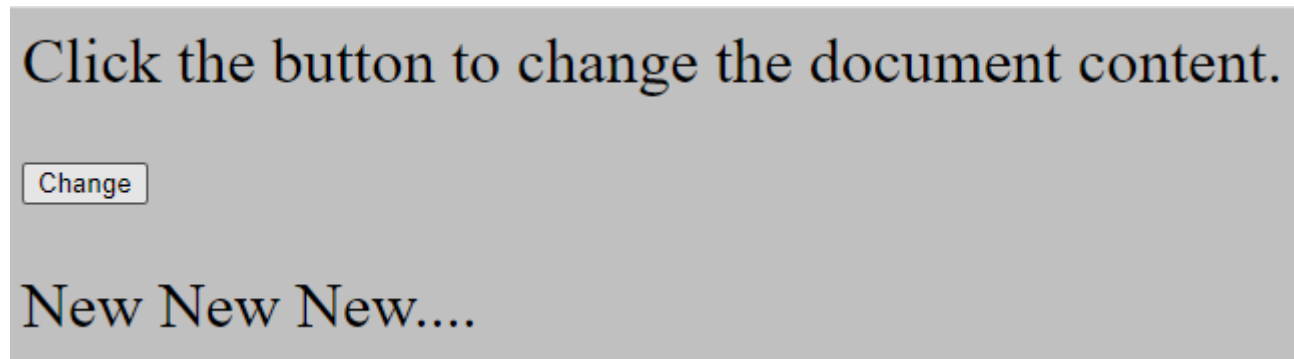
HTML Document Example

```
<html>
  <body>
    <p>Click the button to change the document content.</p>
    <button onclick="myFunction()">Change</button>
    <p id="demo">Old Content</p>
    <script>
      function myFunction() {
        document.getElementById("demo").innerHTML = "New New New....";
      }
    </script>
  </body>
</html>
```




HTML Document Example

```
<html>
  <body>
    <p>Click the button to change the document content.</p>
    <button onclick="myFunction()">Change</button>
    <p id="demo">Old Content</p>
    <script>
      function myFunction() {
        document.getElementById("demo").innerHTML = "New New New....";
      }
    </script>
  </body>
</html>
```



HTML Document Example

```
<html>
  <body>
    <p>Click the button
    <button onclick="m
    <p id="demo">Old (
    <script>
      function myFunc
      document.get
    }
  </script>
</body>
</html>
```



- Toàn bộ trang là một đối tượng kiểu **Document** (và có tên là "**document**")
- Properties: body, head, cookie, forms, images...
- Methods:
 - getElementById()
 - getElementsByTagName()
 - ...

HTML Document Example

```
<html>
```

```
<body>
```

```
<p>Click the button to change the document content.</p>
```

```
<button onclick="myFunction()">Change</button>
```

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

```
...
```

```
</body>
```

```
</html>
```



- Mỗi phần tử (kể cả body, head...) có kiểu **Element**
- Properties: children, firstChild, innerHTML...
- Methods:
 - remove()
 - getElementsByTagName()
 - onClick()...

DOM Object Types

- **Document** (đối tượng duy nhất có tên "document")
- **Element** (head, body, p, div, a, img, table...)
- **Attribute** (any HTML attribute)
- **Style**
- ...

DOM Object Selection Functions

- `getElementById()`
- `getElementsByTagName()`
- `getElementsByClassName()`
- `querySelector()`
- `querySelectorAll()`

Asynchronous JavaScript and XML

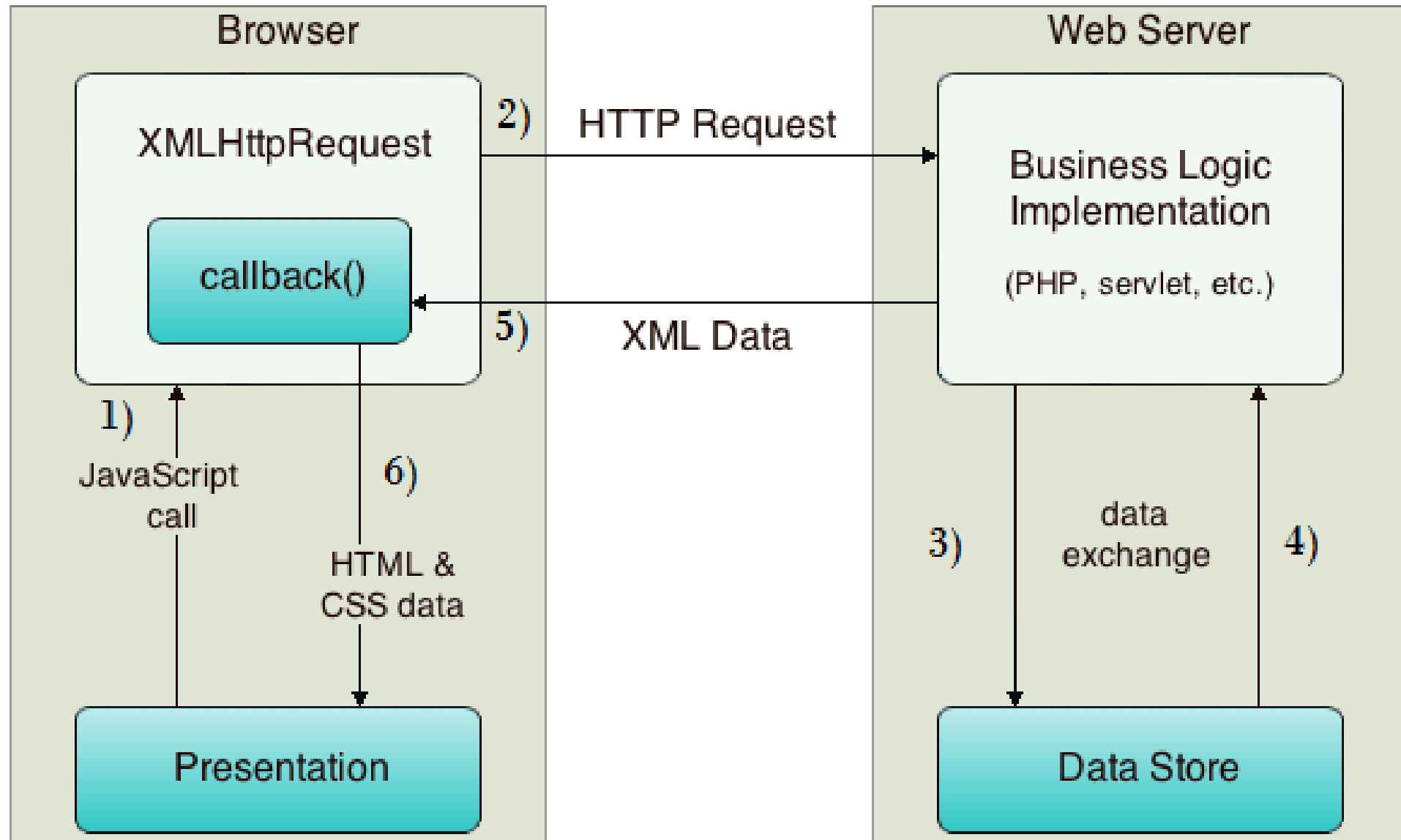
- **AJAX** (/ˈeɪdʒæks/) không phải là một công nghệ mà là một kỹ thuật tạo trang web ở phía client.
- Thay đổi nội dung mà không cần tải lại **toàn bộ** trang web.
- Được thực thi nhờ đối tượng **XMLHttpRequest**
- Ngày nay, JSON được dùng phổ biến hơn là XML (kết quả truy vấn)



Ví dụ AJAX

```
<div id="demo">
  <h1>The XMLHttpRequest Object</h1>
  <button type="button" onclick="loadDoc()">Change Content</button>
</div>
<script> function loadDoc() {
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200)
      document.getElementById("demo").innerHTML = this.responseText;
  };
  xhttp.open("GET", "ajax_info.txt", true);
  xhttp.send();
}</script>
```

AJAX



AJAX – Ưu điểm và hạn chế

- **Ưu điểm:**

- Tốc độ cao
- Tạo trang web có khả năng tương tác
- Giảm băng thông

- **Hạn chế:**

- Phức tạp trong thiết kế
- Người dùng không bookmark được trạng thái trang web





Sự khác nhau giữa ví dụ về AJAX với ví dụ về DOM trước đó?

1

JSON và XML

2

CSS, DOM và AJAX

3

jQuery và Bootstrap

4

Web Service





jQuery là gì?

- A lightweight, "write less, do more", JavaScript library.
- Much easier to use JavaScript.
- jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.
- jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.

2 cách sử dụng jQuery

❑ Tải và sử dụng bản sao của jQuery

```
<head>
```

```
<script src="jquery-3.6.0.min.js"></script>
```

```
</head>
```

❑ Sử dụng jQuery CDN

```
<head>
```

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
```

```
</head>
```

jQuery Syntax

- Cú pháp jQuery hướng đến **chọn** một (số) phần tử HTML và thực hiện một **hành động** trên (các) phần tử đó.
- **Basic syntax** is: **`$(selector).action()`**
 - **`$`** để gọi jQuery
 - **`(selector)`** để chọn (một số) phần tử HTML
 - **`action()`** để thực hiện thao tác trên (các) phần tử đã chọn
- **Example:**
 - **`$("p").hide()`** - hides all <p> elements.
 - **`$(".test").hide()`** - hides all elements with class="test".

jQuery Example

```
<head> <script>
```

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

```
        $("button").click(function(){
```

```
            $("#test").hide();
```

```
        });
```

```
    });
```

```
</script> </head>
```

```
<body>
```

```
    <p>This is a paragraph.</p>
```

```
    <p id="test">This is another paragraph.</p>
```

```
    <button>Click me</button>
```

```
</body>
```

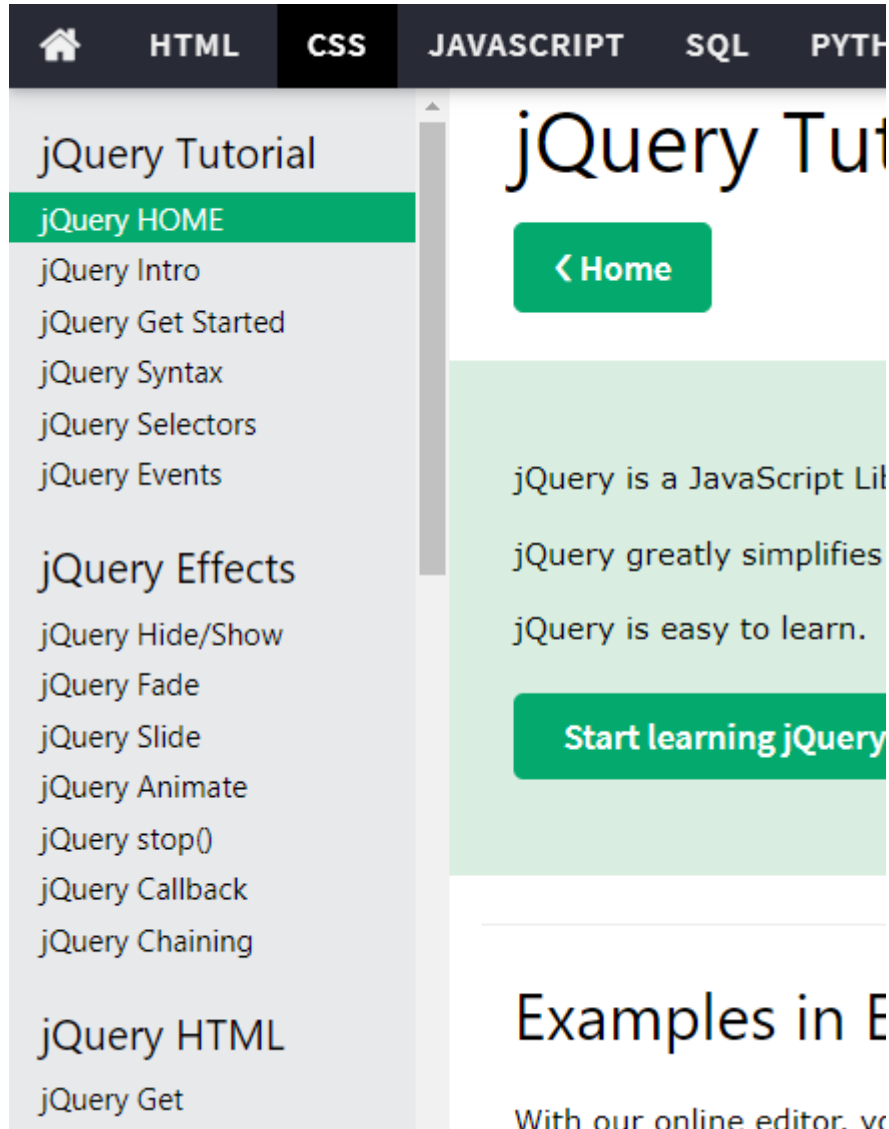
//Định nghĩa lại event "ready"

//Định nghĩa lại event "click"

Khả năng của jQuery

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities





Try It Yourself!

**[https://www.w3schools.com/
jquery/default.asp](https://www.w3schools.com/jquery/default.asp)**



Bootstrap

- **Bootstrap** là một CSS framework nguồn mở dùng cho phát triển các ứng dụng web có tính đáp ứng (responsive web), ưu tiên thiết bị di động (mobile-first).
- Khả năng của Bootstrap:
 - Quản lý lưới (grid)
 - Tạo các thành phần giao diện (menu, form, bảng...)
 - Chèn biểu tượng (glyphicons)
 -



Thiết lập để sử dụng Bootstrap

<!DOCTYPE html>

<html lang="en">

<head>

<title>Bootstrap Example</title>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

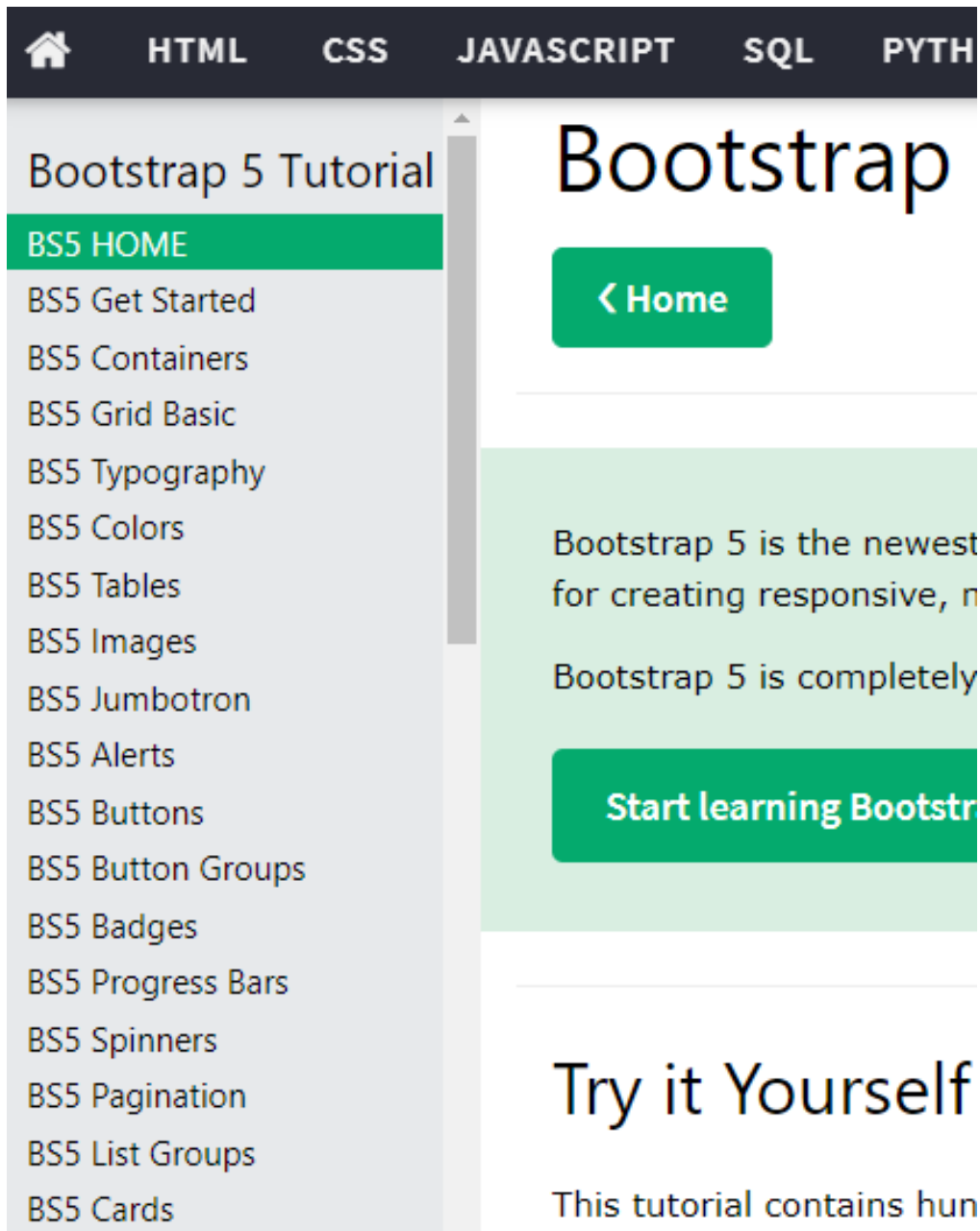
<link href="/bootstrap.min.css" rel="stylesheet">

<script src="/bootstrap.bundle.min.js"></script>

</head>

Bắt buộc

Nên sử dụng CDN hơn là tải về máy chủ web của mình



Try It Yourself!

[https://www.w3schools.com/
bootstrap5/index.php](https://www.w3schools.com/bootstrap5/index.php)



- **V4, 5 không còn hỗ trợ
glyphicons miễn phí như V3**
- **V5 không dùng jQuery như V3,4
(mà có thư viện JavaScript riêng)**

1

JSON và XML

2

CSS, DOM và AJAX

3

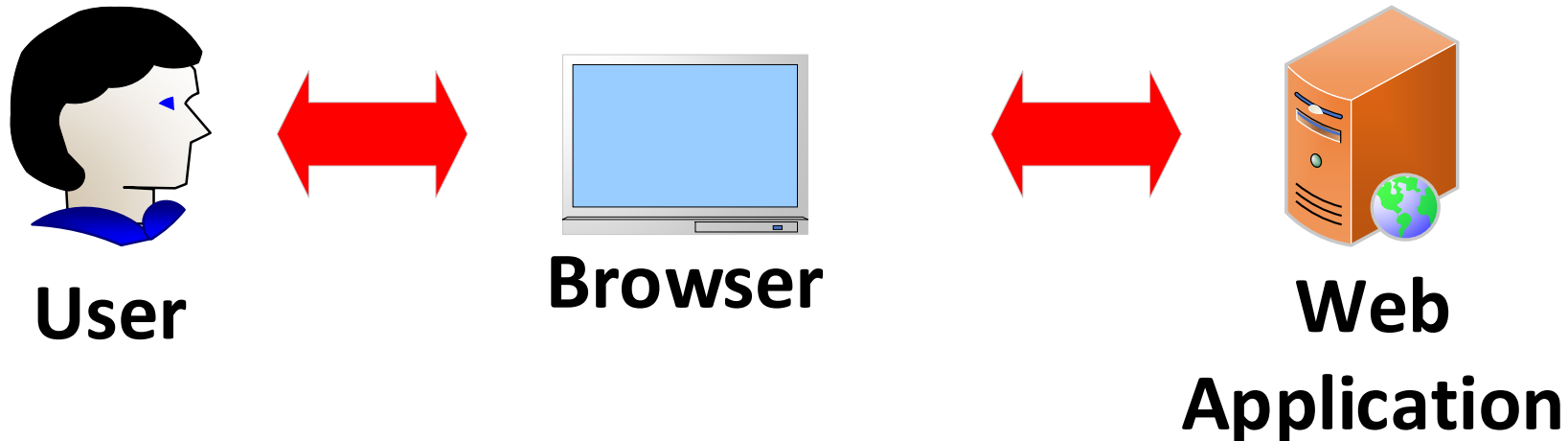
jQuery và Bootstrap

4

Web Service

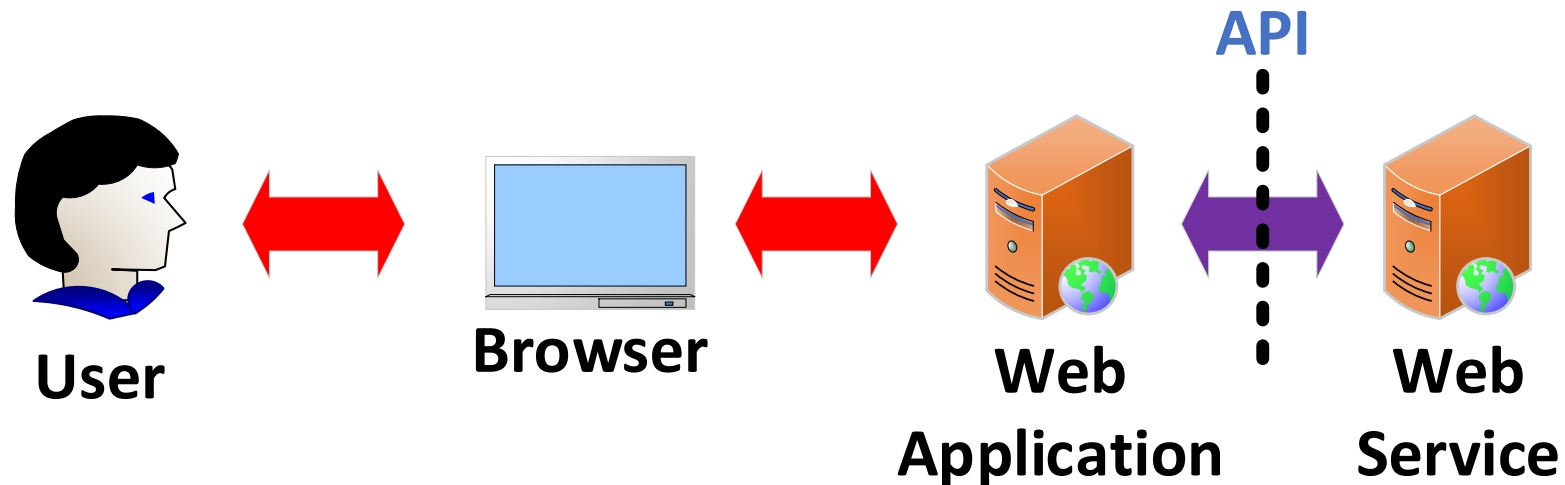
Web Application

- **Web Application:** là phần mềm thực hiện giao tiếp với người dùng thông qua một trình duyệt.



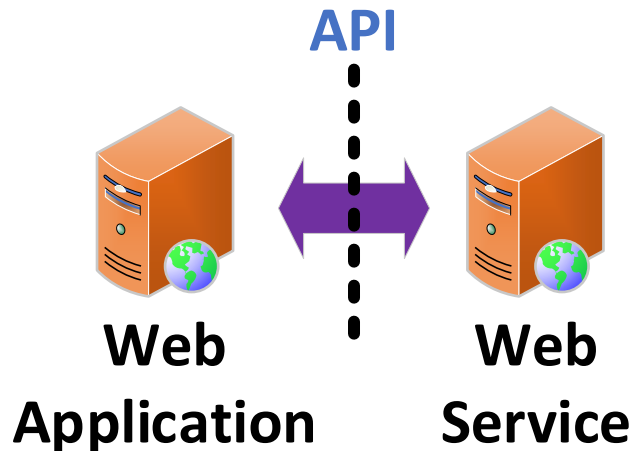
Web Service

- **Web Service:** là phần mềm thực hiện tương tác dữ liệu với phần mềm khác qua các giao thức web.
 - Không cung cấp giao diện người dùng
 - Tương tác qua các API (SOAP, RESTful)
 - Có thể là một phần (back-end) của một Web Application



SOAP vs REST

SOAP	REST
Simple Object Access Protocol	REpresentational State Transfer
Phức tạp	Đơn giản
Phù hợp với tương tác nhiều tham số, hoặc tham số có giá trị phức tạp	Phù hợp với tương tác ít tham số và tham số có giá trị đơn giản



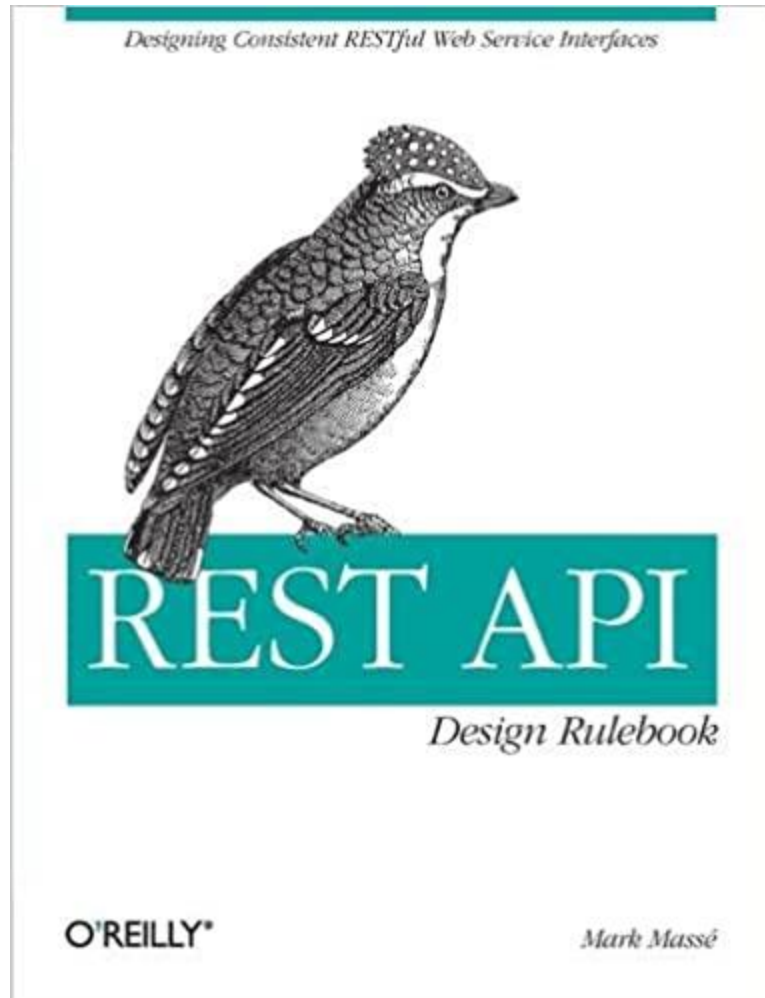
REST API

- REST API = RESTful API
- Hoàn toàn không trạng thái (stateless). Phía server không quản lý phiên như với Web Application.
 - Mỗi truy vấn đều phải kèm theo thông tin xác thực (JWT...)
- Dữ liệu trả về thường là JSON
 - nhưng có thể là kiểu khác: XML, plaintext, image...
- Chủ yếu sử dụng các phương thức HTTP: POST, GET, PUT, DELETE (mô hình CRUD)

Thiết kế REST API Endpoints

- **/api/v1/items**
 - GET: lấy danh sách các đối tượng
 - POST: thêm mới 1 đối tượng
- **/api/v1/items/<id>**
 - GET: lấy thông tin chi tiết về đối tượng <id>
 - PUT: cập nhật thông tin về đối tượng <id>
 - DELETE: xóa đối tượng <id>
- **KHÔNG sử dụng các URI như**
 - /api/v1/get-item-list
 - /api/v1/add-new-item/<id>

Thiết kế REST API Endpoints



REST API – Design Rulebook

Mark Masse

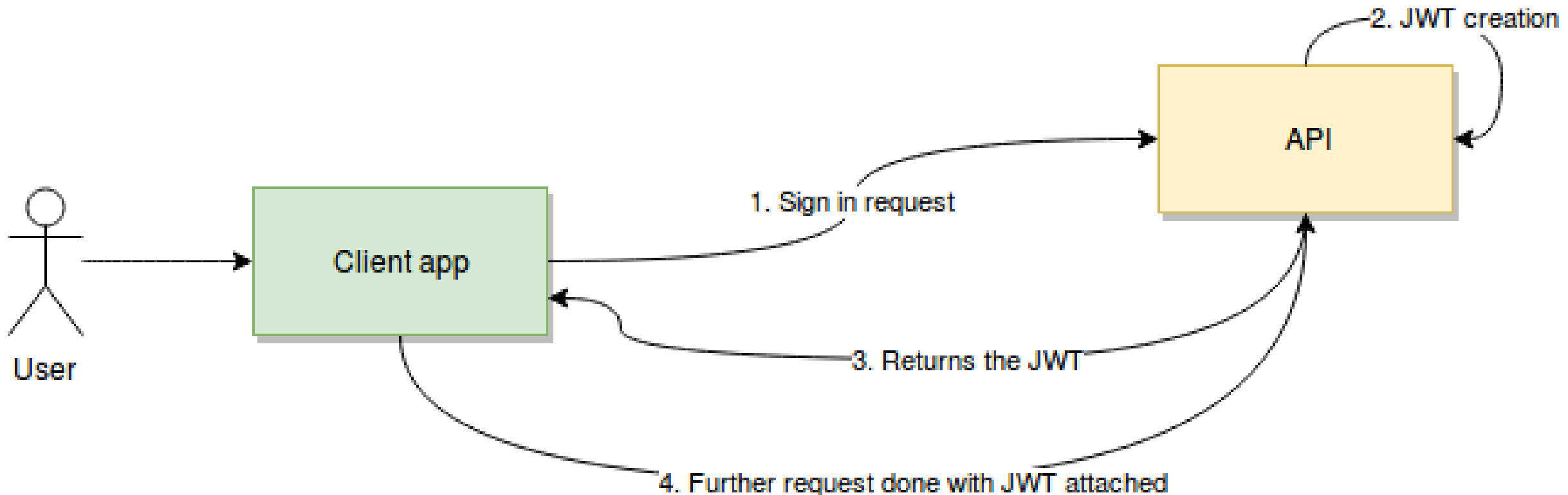
O'Reilly, 2012

Xác thực trong REST API

1. Client gọi API xác thực

Kết quả: Client nhận được token xác thực (JWT chẳng hạn)

2. Client gửi token xác thực kèm theo mỗi truy vấn



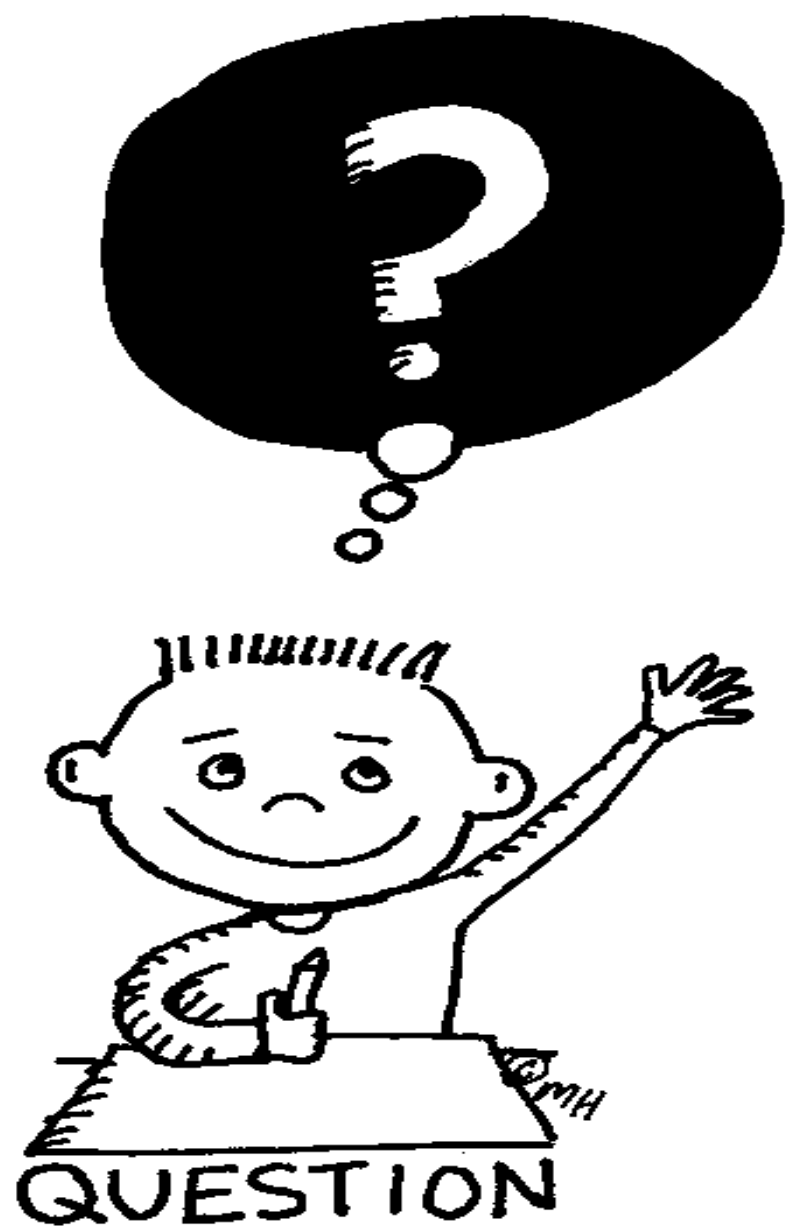
CSRF trong REST API

CSRF trong web application

- SESSIONID là một cookie, được **trình duyệt tự động gửi đi** theo truy vấn

CSRF trong REST API

- Token xác thực (JWT?) thường không được lưu ở dạng cookie nên CSRF sẽ thất bại.



1

JSON và XML

2

CSS, DOM và AJAX

3

jQuery và Bootstrap

4

Web Service

Bài tập về nhà

- Phát triển một REST API
- Xây dựng một ứng dụng web có sử dụng AJAX (jQuery) để lấy dữ liệu qua API ở trên.