BKFA CN-Web Sheet

XML

<?xml version="1.0" encoding="utf-8"?>

<!DOCTYPE dssk [

<!ELEMENT dssk (sk)+ >

<!ATTLIST dssk ten CDATA #IMPLIED>

<!ELEMENT sk (dsvdv, kq) >

<!ATTLIST sk ten CDATA #REQUIRED>

<!ELEMENT dsvdv (vdv)+ >

<!ELEMENT vdv (vitri)? >

<!ATTLIST vdv ten CDATA #REQUIRED>

<!ATTLIST vdv quocgia CDATA #REQUIRED>

<!ATTLIST vdv ten CDATA #REQUIRED>

<!ELEMENT kq (nhat, nhi, ba) >

<!ELEMENT nhat (vdv)\* >

<!ELEMENT nhi (vdv)\* >

<!ELEMENT ba (vdv)\* >

]>

<dssk>

<sk ten="dienkinh">

<dsvdv>

<vdv ten="ulsanbolt" quocgia="VN">

</vdv>

</dsvdv>

<kq>

<nhat>

</nhat>

<nhi>

</nhi>

<ba>

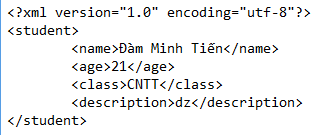
</ba>

</kq>

</sk>

</dssk>

## 1. Giới thiệu XML



Văn bản có cấu trúc XML cho phép biểu diễn thông tin về các đối tượng trong thực tế. XML dùng để phục vụ cho việc mô tả dữ liệu (thông tin lưu trữ bao gồm những gì, lưu trữ ra sao) để các hệ thống khác nhau có thể đọc và sử dụng những thông tin này một cách thuận tiện. Các thẻ tag ủa XML thường không được định nghĩa trước mà chúng được tạo ra theo quy ước của người theo những quy ước riêng. XML sử dụng các khai báo kiểu dữ liệu DTD (Document Type Definition) hay lược đồ Schema để mô tả dữ liệu.

### **Định chuẩn XML**

Có ba kiểu tài liệu XML là:

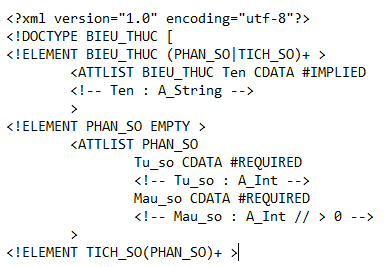
* Well-formed Document: tài liệu XML chuẩn cú pháp
* DTD - Constrained Document: Tạo XML có khai bá DTD (Document type definition) để mô tả cấu trúc dữ liệu trong XML
* XML-Schema - Constrained Document: Tạo XML có sử dụng lược đồ Schema để kiểm tra tính hợp lệ của XML.

## 2. Well-formed XML Document

Các quy ước:

* Có duy nhất một phần tử thuộc cấp cao nhất trong tài liệu, còn gọi là nút gốc
* Mỗi một thẻ mở đều phải có thẻ đóng và tên thẻ là phân biệt hoa thường
* Các thẻ khi đóng phải theo đúng trình tự (mở sau đóng trước)
* Tên thẻ không nên có khoảng trống, không nên bắt đầu bằng “xml”
* Các thuộc tính (attributes) của một thẻ luôn luôn tồn tại theo cặp quy ước <tên> = “giá trị”; không nên đặt tên thuộc tính trùng nhau và giá trị của thuộc tính phải đặt trong cặp dấu nháy kép hoặc nháy đơn. Tên của thuộc tính sẽ theo quy luật như tên thẻ.
* Các thẻ (tag) trong XML có thể lồng nhau.

## 3. DTD (Document Type Definition)



Ví dụ đặc tả một biểu thức phân số

DTD định nghĩa các thành phần - thuộc tính và cấu trúc của tài liệu XML.

Các phần trong DTD:

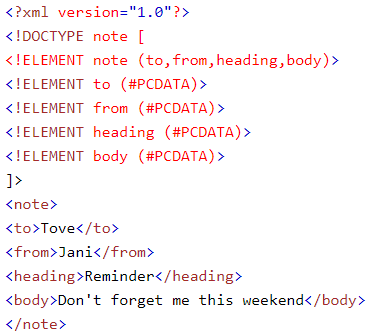
* Elements: các thẻ chính của tài liệu XML
* Attributes: thuộc tính của elements
* Entities: các ký tự có ý nghĩa đặc biệt trong tài liệu XML, ví dụ như “<“ đổi thành &lt;
* PCDATA: các đoạn text sẽ được parser chuyển đổi thành dữ liệu
* CDATA: các đoạn text sẽ được parser bỏ qua

Cú pháp đặc tả chung cấu trúc của 1 thẻ

<!ELEMENT Ten\_the Bieu\_thuc\_dac\_ta\_cau\_truc>

Bieu\_thuc sẽ chứa các thẻ con trong Ten\_the. Với A, B là 2 thẻ con của thẻ X:

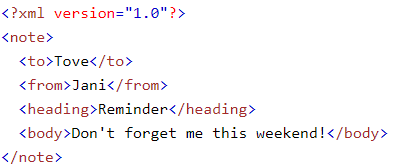
A,B A, B sắp xếp tuần tự A đến B A\* A có thể lặp lại ít nhất 0 lần (>=0) B+ B có thể lặp lại ít nhất 1 làn (>=1)) A? A có thể xuất hiện 0 hoặc 1 lần (0 hoặc 1) A|B Có thể sử dụng A hoặc B



Ví dụ trên W3S

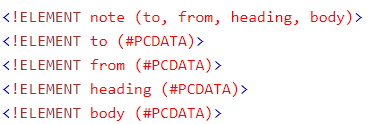
## 4. XML Schema (XSD)

XML Schema hay XML Schema Definition miêu tả cấu trúc tài liệu XML 1 cách chi tiết hơn. Ví dụ 1 tài liệuXML



XML well-formed

DTD File:



DTD

XSD File:



XML Schema File

XML Schema có 3 loại kiểu chính:

* Kiểu định nghĩa sẵn (Bultin Type)
* Kiểu đơn giản (simpleType)
* Kiểu phức hợp (complexType)

### **Kiểu định nghĩa sẵn:**

Được sử dụng khi:

* Thẻ không có thuộc tính
* Thẻ không chứa thẻ khác và có miền giá trị thích hợp với kiểu

Các loại kiểu: string, int, integer, float, double, boolean, date, month, ID, binary

Cú pháp

<xs:element name="Ten\_the" type="Ten\_kieu\_co\_so" .../><xs:attribute name="Ten\_thuoc\_tinh" type="Ten\_kieu\_co\_so" .../>

### **Kiểu đơn giản:**

Được sử dụng khi:

* Thẻ không có thuộc tính
* Thẻ không chứa thẻ khác và có miền giá trị là tập con của miền giá trị của 1 kiểu cơ sở nào đó

Cú pháp:

<xs:simpleType name="Ten\_kieu"><xs:restriction base="Ten\_kieu\_co\_so"> Giới hạn trên miền giá trị</xs:restriction>

</xs:simpleType>

Khai báo cận dưới: minInclusive (cho phép sử dụng biên), minExclusive (không sử dụng biên) Khai báo cận trên: maxInclusive (cho phép sử dụng biên), maxExclusive (không sử dụng biên)

Ngoài ra còn có kiểu liệt kê:

<xs:simpleType name="HOC\_LUC"><xs:restriction base="string"> <xs:enumeration value="Xuất xắc"/> <xs:enumeration value="Giỏi"/></xs:restriction>

</xs:simpleType>

### **Kiểu phức hợp:**

Được sử dụng khi:

* Thẻ có thuộc tính
* Thẻ chứa thẻ khác

Các dạng

* Tuần tự
* Tùy chọn
* Lặp

Tuần tự:

<xs:complexType name="Ten\_kieu"> <xs:sequence>

Thanh\_phan\_1

Thanh\_phan\_2

...

</xs:sequence>

...

</xs:complexType>

Tùy chọn:

<xs:complexType name="Ten\_kieu"> <xs:choice>

Thanh\_phan\_1

Thanh\_phan\_2

...

</xs:choice>

...

</xs:complexType>

Lặp:

<xs:complexType name="Ten\_kieu"> <xs:sequence> ...

<xs:element name="Ten\_the\_con"

type="Kieu\_the\_con" minOccurs="So\_lan\_lap\_min"

maxOccurs="So\_lan\_lap\_max"

/>

...

</xs:sequence>

...

</xs:complexType>

Thuộc tính của thẻ:

<xs:attribute name="Ten\_thuoc\_tinh" type="Kieu\_thuoc\_tinh"

Tu\_khoa\_1="Gia\_tri\_1" Tu\_khoa\_2="Gia\_tri\_2"/>

Một số tính chất thông dụng:

* default: giá trị mặc định
* fixed: giá trị cố định
* use: tùy chọn sử dụng (có hay không)

MVC

[StudentExampleController.php](https://github.com/damminhtien/PHP-MVC/blob/master/controller/StudentExampleController.php)

|  |
| --- |
| <?php |
|  | include\_once("../model/StudentExampleModel.php"); |
|  |  |
|  | class StudentController { |
|  | public $student; |
|  |  |
|  | public function \_\_construct(){ |
|  | $this->student = new Student(); |
|  | } |
|  |  |
|  | public function invoke(){ |
|  | if (!isset($\_GET["student"])){ |
|  | // if not get specify student -> get all |
|  | $students = $this->student->getAll(); |
|  | include "../view/student/studentlist.php"; |
|  | } |
|  | else{ |
|  | // if get specify student |
|  | $student = $this->student->find($\_GET["student"]); |
|  | include "../view/student/viewstudent.php"; |
|  | } |
|  | } |
|  | } |
|  | ?>  Model Database |
| <?php |
|  | // lớp kết nối và truy xuất cơ sở dữ liệu |
|  | class Database { |
|  | public static $hostname = "localhost"; |
|  | public static $username = "root"; |
|  | public static $password = ""; |
|  | public static $dbname = null; |
|  | protected $conn = null; |
|  |  |
|  | public function \_\_contruct(){ |
|  | $this->connect(); |
|  | echo "Set default database!"; |
|  | } |
|  |  |
|  | public function connect(){ |
|  | $this->conn = mysqli\_connect(static::$hostname, static::$username, static::$password, static::$dbname) or die ('Connect Error!'); |
|  | mysqli\_query($this->conn, "SET NAMES 'utf8'"); |
|  | } |
|  |  |
|  | public function disconnect(){ |
|  | if($this->conn) |
|  | mysqli\_close($this->conn); |
|  | } |
|  |  |
|  | public static function query($sql){ |
|  | try{ |
|  | $result = mysqli\_query($this->conn, $sql) or die ('Query Error!'); |
|  | return $result; |
|  | catch(mysqli\_sql\_exception $e){ |
|  | return false; |
|  | } |
|  | } |
|  | } |
|  | ?> |

Model Model

|  |
| --- |
| <?php |
|  | // lớp chung cho từng bảng dữ liệu |
|  | class Model extends Database{ |
|  | public $tableName; |
|  | public $properties = array(); |
|  | public $primaryKey; |
|  |  |
|  | public function \_\_contruct($tableName){ |
|  | parent::\_\_construct(); |
|  | $this->tableName = $tableName; |
|  | } |
|  |  |
|  | public getAll(){ |
|  | $sql = "SELECT \* FROM {$tableName}"; |
|  | $result = seft::query($sql); |
|  | return fetch($result); |
|  | } |
|  |  |
|  | public function find($pKeyValue){ |
|  | $pKey = isset($primaryKey) ? $primaryKey : "id"; |
|  | $sql = "SELECT \* FROM {$tableName} WHERE {$pKey} = {$pKeyValue}"; |
|  | $result = seft::query($sql); |
|  | return fetch($result)[0]; |
|  | } |
|  |  |
|  | public function findBy($key, $value){ |
|  | $sql = "SELECT \* FROM {$tableName} WHERE {$key} = {$value}"; |
|  | $result = seft::query($sql); |
|  | return fetch($result); |
|  | } |
|  |  |
|  | public function findLike($key, $value){ |
|  | $sql = "SELECT \* FROM {$tableName} WHERE {$key} = %{$value}%"; |
|  | $result = seft::query($sql); |
|  | return fetch($result); |
|  | } |
|  |  |
|  | public function delete($key, $value){ |
|  | $sql = "DELETE FROM {$tableName} WHERE {$key} = {$value}"; |
|  | result = seft::query($sql); |
|  | if($result) |
|  | echo "Delete successful!"; |
|  | else |
|  | echo "Delete failed!"; |
|  | } |
|  |  |
|  | public function fetch($data){ |
|  | $result = array(); |
|  | if ($data->num\_rows > 0) { |
|  | // output data of each row |
|  | while($row = $data->fetch\_assoc()) { |
|  | $result[count(result)] = $row; |
|  | } |
|  | } else { |
|  | echo "0 results"; |
|  | } |
|  | return $result; |
|  | } |
|  | } |
|  | ?> |

**StudentExampleModel.php**

|  |
| --- |
| <?php |
|  | //examples class |
|  | class Student extends Model{ |
|  | public $id; |
|  | public $name; |
|  | public $class; |
|  |  |
|  | public function \_\_construct(){ |
|  | parent::\_\_contruct(); |
|  | echo "Create object without define properties!"; |
|  | } |
|  |  |
|  | public function \_\_construct($id, $name, $class){ |
|  | $this->id = $id; |
|  | $this->name = $name; |
|  | $this->class = $class; |
|  | $this->primaryKey = isset($primaryKey) ? $primaryKey : "id"; |
|  | $this->tableName = $tableName || get\_class($this); |
|  | } |
|  |  |
|  | public function save(){ |
|  | $this->primaryKey = isset($primaryKey) ? $primaryKey : "id"; |
|  | $this->tableName = isset($tableName) ? $tableName : get\_class($this); |
|  | $sql = "SELECT \* FROM {$tableName} WHERE {$primaryKey} = $this->$primaryKey"; |
|  | $result = seft::query($sql); |
|  | if ($result->num\_rows > 0) { |
|  | $sql = "UPDATE {$tableName} SET id = '{$id}', name = '{$name}', class = '{$class}' WHERE {$primaryKey} = $this->$primaryKey"; |
|  |  |
|  | } else { |
|  | $sql = "INSERT INTO {$tableName} VALUES ('{$id}', '{$name}', '{$class}')"; |
|  | } |
|  | $result = seft::query($sql); |
|  | if($result) |
|  | echo "Save successful!"; |
|  | else |
|  | echo "Save failed"; |
|  | } |
|  | } |
|  | ?> |

[studentlist.php](https://github.com/damminhtien/PHP-MVC/blob/master/views/student/studentlist.php)

|  |
| --- |
| <html> |
|  | <head></head> |
|  | <body> |
|  | <table> |
|  | <tbody><tr><td>Title</td><td>Author</td><td>Description</td></tr></tbody> |
|  | <?php |
|  | foreach ($students as $id => $student) |
|  | { |
|  | echo '<tr><td><a href="index.php?student='.$student->id.'">'.$student->id.'</a></td><td>'.$student->name.'</td><td>'.$student->class.'</td></tr>'; |
|  | } |
|  | ?> |
|  | </table> |
|  | </body> |
|  | </html |

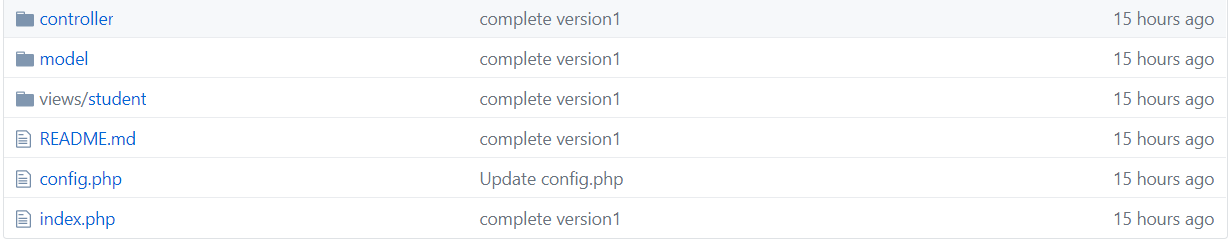
[viewstudent.php](https://github.com/damminhtien/PHP-MVC/blob/master/views/student/viewstudent.php)

|  |  |
| --- | --- |
| <html> |  |
|  |  | <head></head> |
|  |  | <body> |
|  |  | <?php |
|  |  | echo "Id: {$student->id} <br/>"; |
|  |  | echo "Name: {$student->name} <br/>"; |
|  |  | echo "Class: {$student->class} <br/>"; |
|  |  | ?> |
|  |  | </body> |
|  |  | </html> |

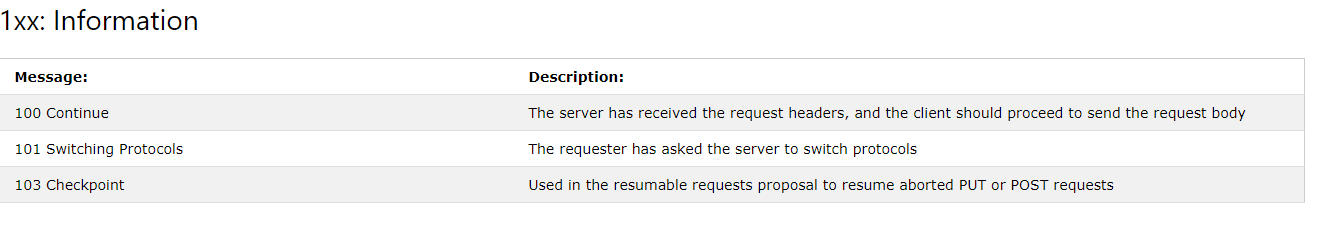
**config.php**

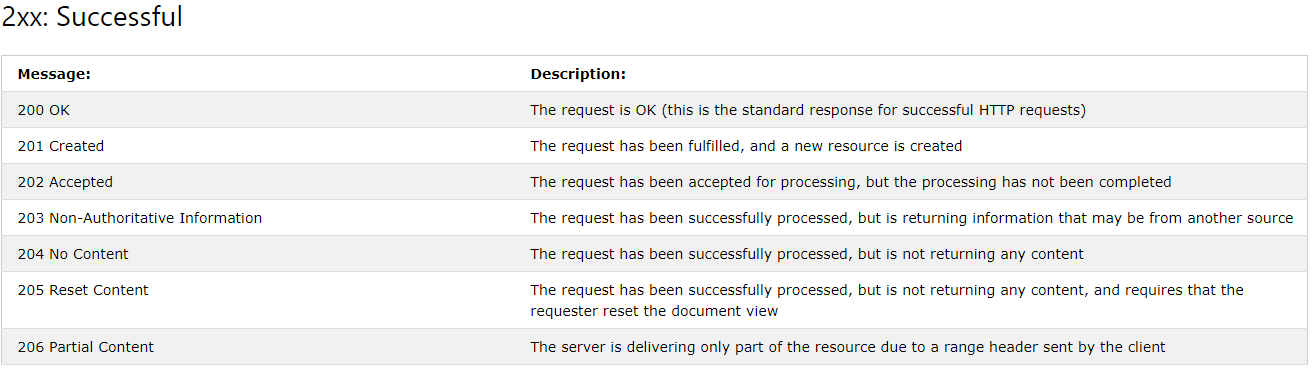
|  |
| --- |
| <?php |
|  | // config Database |
|  | const $APP\_NAME = "PHP-MVC"; |
|  |  |
|  | const $AUTHOR = "DMT" |
|  |  |
|  | const $HOST\_NAME = "localhost"; |
|  |  |
|  | const $USER\_NAME = "root"; |
|  |  |
|  | const $PASSWORD = ""; |
|  |  |
|  | const $DBNAME = null; |
|  | ?>  ---------------------------------------------------------------------------------    **index.php** |
|  |  |
| <?php |
|  | include\_once("config.php"); |
|  | include\_once("model/Database.php"); |
|  | include\_once("controller/StudentExampleController.php"); |
|  |  |
|  | Database::$hostname = $HOST\_NAME; |
|  | Database::$username = $USER\_NAME; |
|  | Database::$password = $PASSWORD; |
|  | Database::$dbname = $DBNAME; |
|  |  |
|  | $studentController = new StudentController(); |
|  | $studentController->invoke(); |
|  | ?> |

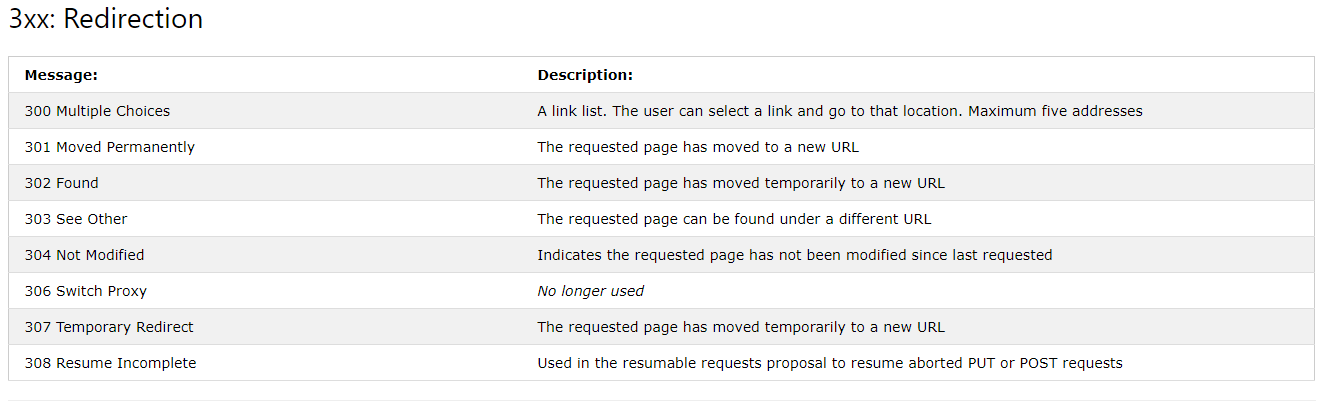
----------------------------------------------------------------------------------------------------------------------------------



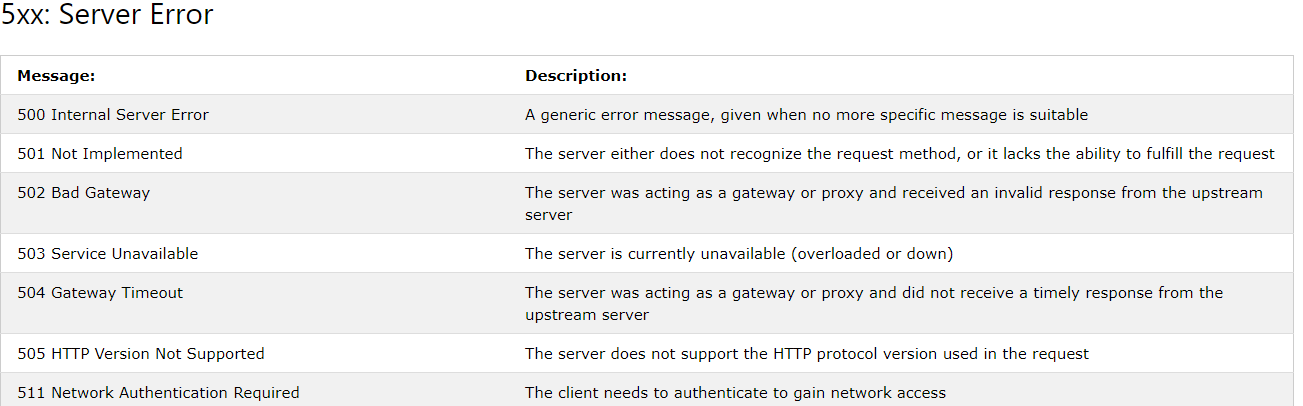
HTML











CSS

Background Image

body {  
    background-image: url("paper.gif");  
}

Hình ảnh được lặp lại chỉ theo chiều ngang

body {  
    background-image: url("gradient\_bg.png");  
    background-repeat: repeat-x; //dọc-> background-repeat: repeat-y;  
}

Không lặp

body {  
    background-image: url("img\_tree.png");  
    background-repeat: no-repeat;  
}

Chỉ định vị trí với background-position

body {  
    background-image: url("img\_tree.png");  
    background-repeat: no-repeat;  
    background-position: right top;  
}

Xác định hình ảnh cố định, sử dụng thuộc tính background-attachment

body {  
    background-image: url("img\_tree.png");  
    background-repeat: no-repeat;  
    background-position: right top;  
    background-attachment: fixed;  
}

Viết gọn trong một thuộc tính background

body {  
    background: #ffffff url("img\_tree.png") no-repeat right top;  
}

Border

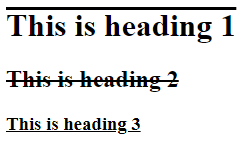


Khung viền tròn

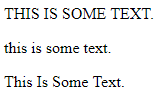
 border-radius: 5px;

Xóa gạch chân thẻ a

a {  
    text-decoration: none;  
}

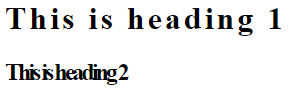


h1 {  
    text-decoration: overline;  
}  
  
h2 {  
    text-decoration: line-through;  
}  
  
h3 {  
    text-decoration: underline;  
}



p.uppercase {  
    text-transform: uppercase;  
}  
  
p.lowercase {  
    text-transform: lowercase;  
}  
  
p.capitalize {  
    text-transform: capitalize;  
}

Xác định khoảng cách



h1 {  
    letter-spacing: 3px;  
}  
  
h2 {  
    letter-spacing: -3px;  
}

p {  
    direction: rtl;  
}

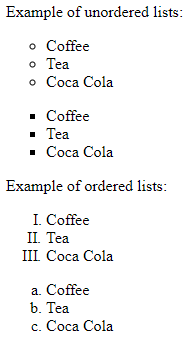
Đổ bóng text

h1 {  
    text-shadow: 3px 2px red; //ngang dọc màu  
}

Định dạng thẻ a

a:link {  
    text-decoration: none;  
}  
  
a:visited {  
    text-decoration: none;  
}  
  
a:hover {  
    text-decoration: underline;  
}  
  
a:active {  
    text-decoration: underline;  
}

Đánh mục danh sách



ul.a {  
    list-style-type: circle;  
}  
  
ul.b {  
    list-style-type: square;  
}  
  
ol.c {  
    list-style-type: upper-roman;  
}  
  
ol.d {  
    list-style-type: lower-alpha;  
}

Dùng ảnh

ul {  
    list-style-image: url('sqpurple.gif');  
}

Định dạng lại viền bảng

table {  
    border-collapse: collapse; // bỏ viền  
}  
  
table, th, td {  
    border: 1px solid black; // viền mới  
}

Bảng sọc vằn

tr:nth-child(even) {background-color: #f2f2f2;}

Vị trĩ tĩnh

div.static {  
    position: static;  
    border: 3px solid #73AD21;  
}

Vị trí tương đối

div.relative {  
    position: relative;  
    left: 30px;  
    border: 3px solid #73AD21;  
}

Vị trí cố định

div.fixed {  
    position: fixed;  
    bottom: 0;  
    right: 0;  
    width: 300px;  
    border: 3px solid #73AD21;  
}

Vị trí tuyệt đối so với thẻ cha, nếu ko có thẻ cha thì nó sẽ lấy phần thân html và di chuyển cùng với việc cuộn trang

div.absolute {  
    position: absolute;  
    top: 80px;  
    right: 0;  
    width: 200px;  
    height: 100px;  
    border: 3px solid #73AD21;  
}

Vị trí dính (Dính vào top hoặc bên phải, … của trang web. Vd như fixed menu trên top)

div.sticky {  
    position: -webkit-sticky; /\* Safari \*/  
    position: sticky;  
    top: 0;  
    background-color: green;  
    border: 2px solid #4CAF50;  
}

So với display: block, sự khác biệt chính là display: inline-blockkhông thêm một ngắt dòng sau phần tử, do đó phần tử có thể ngồi bên cạnh các phần tử khác.

Căn chình ảnh vào giữa

img {  
    display: block;  
    margin-left: auto;  
    margin-right: auto;  
    width: 40%;  
}

Lấy phần tử liền kề bên dưới thì sử dụng +

div + p {  
    background-color: yellow;  
}

Lấy phần tử đầu:

p:first-child {  
    color: blue;  
}

Lấy chữ cái đầu:

p::first-letter {  
    color: #ff0000;  
    font-size: xx-large;  
}

Trước phần tử

h1::before { // sau ::after  
    content: url(smiley.gif);  
}

Hiệu ứng ảnh mờ

img {  
    opacity: 0.5;  
    filter: alpha(opacity=50); /\* For IE8 and earlier \*/  
}

Tạo menu dọc 1 cấp

ul {  
    list-style-type: none;  
    margin: 0;  
    padding: 0;  
    width: 200px;  
    background-color: #f1f1f1;  
}  
  
li a {  
    display: block;  
    color: #000;  
    padding: 8px 16px;  
    text-decoration: none;  
}  
  
/\* Change the link color on hover \*/  
li a:hover {  
    background-color: #555;  
    color: white;  
}

Tạo menu ngang một cấp

ul {  
    list-style-type: none;  
    margin: 0;  
    padding: 0;  
    overflow: hidden;  
    background-color: #333;  
}  
  
li {  
    float: left;  
}  
  
li a {  
    display: block;  
    color: white;  
    text-align: center;  
    padding: 14px 16px;  
    text-decoration: none;  
}  
  
/\* Change the link color to #111 (black) on hover \*/  
li a:hover {  
    background-color: #111;  
}

Chuyển dọc thành ngang

@media screen and (max-width: 600px){

li {  
    float: none;  
}  
  
}

Tạo menu ngang 2 cấp

<!DOCTYPE html>

<html>

<head>

<style>

ul {

list-style-type: none;

margin: 0;

padding: 0;

overflow: hidden;

background-color: #333;

}

li {

float: left;

}

li a, .dropbtn {

display: inline-block;

color: white;

text-align: center;

padding: 14px 16px;

text-decoration: none;

}

li a:hover, .dropdown:hover .dropbtn {

background-color: red;

}

li.dropdown {

display: inline-block;

}

.dropdown-content {

display: none;

position: absolute;

background-color: #f9f9f9;

min-width: 160px;

box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);

z-index: 1;

}

.dropdown-content a {

color: black;

padding: 12px 16px;

text-decoration: none;

display: block;

text-align: left;

}

.dropdown-content a:hover {background-color: #f1f1f1}

.dropdown:hover .dropdown-content {

display: block;

}

</style>

</head>

<body>

<ul>

<li><a href="#home">Home</a></li>

<li><a href="#news">News</a></li>

<li class="dropdown">

<a href="javascript:void(0)" class="dropbtn">Dropdown</a>

<div class="dropdown-content">

<a href="#">Link 1</a>

<a href="#">Link 2</a>

<a href="#">Link 3</a>

</div>

</li>

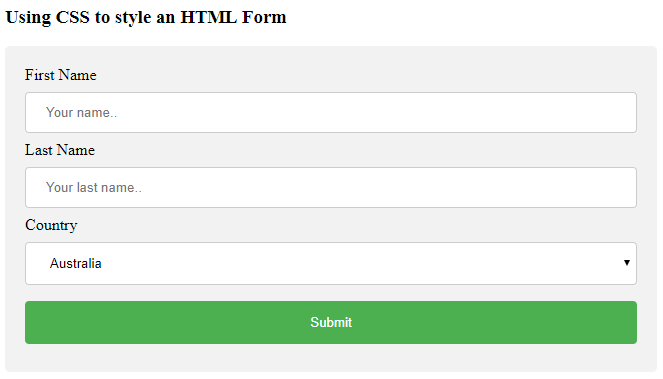
</ul>

<h3>Dropdown Menu inside a Navigation Bar</h3>

<p>Hover over the "Dropdown" link to see the dropdown menu.</p>

</body>

</html>



<!DOCTYPE html>

<html>

<style>

input[type=text], select {

width: 100%;

padding: 12px 20px;

margin: 8px 0;

display: inline-block;

border: 1px solid #ccc;

border-radius: 4px;

box-sizing: border-box;

}

input[type=submit] {

width: 100%;

background-color: #4CAF50;

color: white;

padding: 14px 20px;

margin: 8px 0;

border: none;

border-radius: 4px;

cursor: pointer;

}

input[type=submit]:hover {

background-color: #45a049;

}

div {

border-radius: 5px;

background-color: #f2f2f2;

padding: 20px;

}

</style>

<body>

<h3>Using CSS to style an HTML Form</h3>

<div>

<form action="/action\_page.php">

<label for="fname">First Name</label>

<input type="text" id="fname" name="firstname" placeholder="Your name..">

<label for="lname">Last Name</label>

<input type="text" id="lname" name="lastname" placeholder="Your last name..">

<label for="country">Country</label>

<select id="country" name="country">

<option value="australia">Australia</option>

<option value="canada">Canada</option>

<option value="usa">USA</option>

</select>

<input type="submit" value="Submit">

</form>

</div>

</body>

</html>

Tạo bóng cho thẻ

div {  
    box-shadow: 10px 10px;  
}

chuyển tiếp cho cả thuộc tính chiều rộng và chiều cao, với thời lượng là 2 giây cho chiều rộng và 4 giây cho chiều cao:

div {  
    -webkit-transition: width 2s, height 4s; /\* Safari \*/  
    transition: width 2s, height 4s;  
}

Tạo Tooltip

<style>  
/\* Tooltip container \*/  
.tooltip {  
    position: relative;  
    display: inline-block;  
    border-bottom: 1px dotted black; /\* If you want dots under the hoverable text \*/  
}  
  
/\* Tooltip text \*/  
.tooltip .tooltiptext {  
    visibility: hidden;  
    width: 120px;  
    background-color: black;  
    color: #fff;  
    text-align: center;  
    padding: 5px 0;  
    border-radius: 6px;  
   
    /\* Position the tooltip text - see examples below! \*/  
    position: absolute;  
    z-index: 1;  
}  
  
/\* Show the tooltip text when you mouse over the tooltip container \*/  
.tooltip:hover .tooltiptext {  
    visibility: visible;  
}  
</style>  
  
<div class="tooltip">Hover over me  
  <span class="tooltiptext">Tooltip text</span>  
</div>

Phân trang

<!DOCTYPE html>

<html>

<head>

<style>

.pagination {

display: inline-block;

}

.pagination a {

color: black;

float: left;

padding: 8px 16px;

text-decoration: none;

}

.pagination a.active {

background-color: #4CAF50;

color: white;

border-radius: 5px;

}

.pagination a:hover:not(.active) {

background-color: #ddd;

border-radius: 5px;

}

</style>

</head>

<body>

<h2>Rounded Active and Hover Buttons</h2>

<div class="pagination">

<a href="#">&laquo;</a>

<a href="#">1</a>

<a href="#" class="active">2</a>

<a href="#">3</a>

<a href="#">4</a>

<a href="#">5</a>

<a href="#">6</a>

<a href="#">&raquo;</a>

</div>

</body>

</html>

Back To Top

<!DOCTYPE html>

<html>

<head>

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

<style>

body {

font-family: Arial, Helvetica, sans-serif;

font-size: 20px;

}

#myBtn {

display: none;

position: fixed;

bottom: 20px;

right: 30px;

z-index: 99;

font-size: 18px;

border: none;

outline: none;

background-color: red;

color: white;

cursor: pointer;

padding: 15px;

border-radius: 4px;

}

#myBtn:hover {

background-color: #555;

}

</style>

</head>

<body>

<button onclick="topFunction()" id="myBtn" title="Go to top">Top</button>

<div style="background-color:black;color:white;padding:30px">Scroll Down</div>

<div style="background-color:lightgrey;padding:30px 30px 2500px">This example demonstrates how to create a "scroll to top" button that becomes visible when the user starts to scroll the page.</div>

<script>

// When the user scrolls down 20px from the top of the document, show the button

window.onscroll = function() {scrollFunction()};

function scrollFunction() {

if (document.body.scrollTop > 20 || document.documentElement.scrollTop > 20) {

document.getElementById("myBtn").style.display = "block";

} else {

document.getElementById("myBtn").style.display = "none";

}

}

// When the user clicks on the button, scroll to the top of the document

function topFunction() {

document.body.scrollTop = 0;

document.documentElement.scrollTop = 0;

}

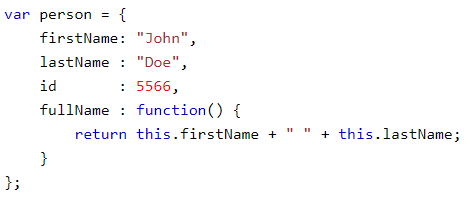
</script>

</body>

</html>

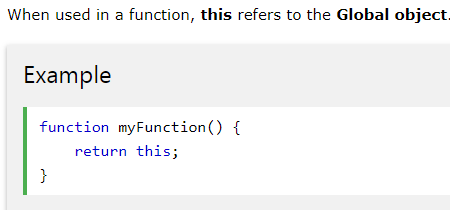
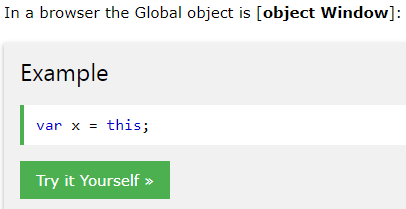
JavaScript

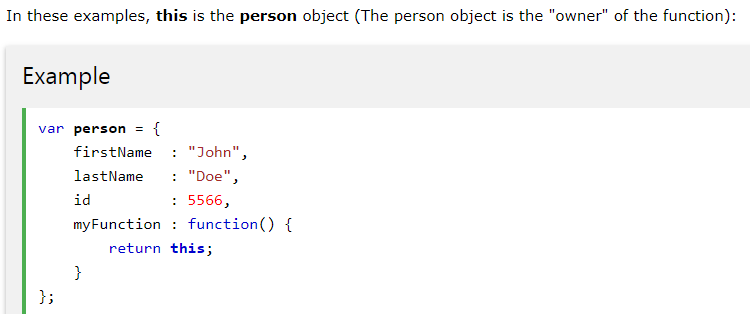
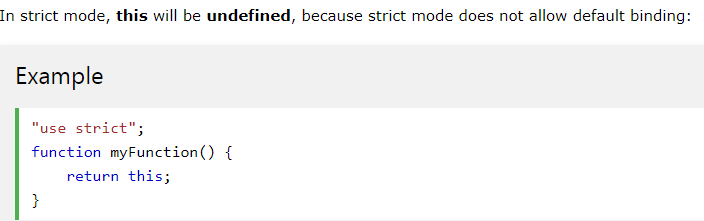
# 1. Object trong JS



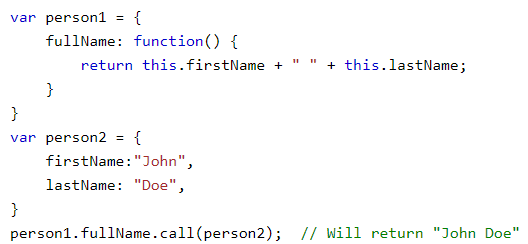
## 1.1 Từ khóa this:

Default Binding:

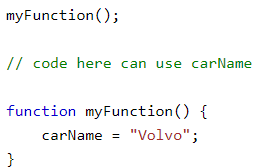




Hàm call() và apply() để gọi hàm cùng tên của đối tượng khác.



# 2. Phạm vi biến

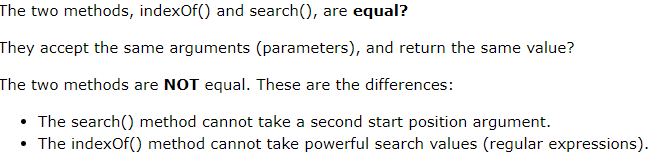
- Biến toàn cục tự định nghĩa:

# 3. Các phương thức string

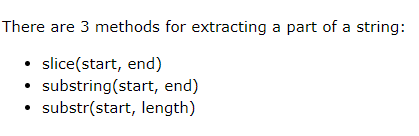
Hàm indexOf(str): trả về vị trí đầu tiên của str trong chuỗi.

Hàm lastIndexOf(str): trả về vị trí cuối cùng của str trong chuỗi.

Hàm search(str): trả về vị trí của str trong chuỗi.



Các hàm cut string:

end: vị trí của phần tử cuối + 1

Hàm replace(str1, str2): thay chuỗi str1 = chuỗi str2 trong str ban đầu.

Hàm str.toUpperCase(), str.toLowerCase()

Hàm str.concat(str1, str2)  
Hàm charAt(i), charCodeAt()

Hàm split(char) cắt chuỗi theo ký tự char vào mảng

var str = "Hello";

var arr = str.split("");

* [‘H’, ‘e’, ‘l’, ‘l’, ‘o’]

# 4. NaN

var x = 100 / "Apple";  // x will be NaN (Not a Number)

var x = 100 / "10";     // x will be 10

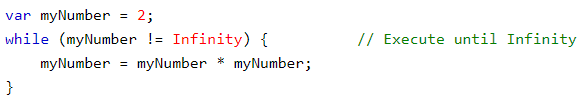
var x = 100 / "Apple";  
isNaN(x); // true

var x = NaN;  
var y = 5;  
var z = x + y;         // z will be NaN

var x = NaN;  
var y = "5";  
var z = x + y;         // z will be NaN5

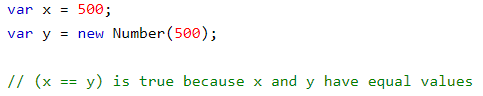
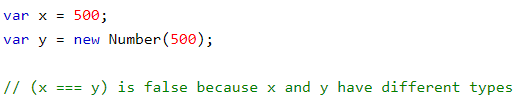
typeof NaN;            // returns "number"

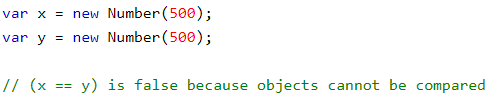
# 5. Infinity



# 6. Number class



# 7. Array Method

Hàm toString đưa 1 arr thành chuỗi, các phần tử phân cách nhau bởi dấu ,

Hàm join(char) nối arr thành chuỗi, phân cách nhau bởi char

pop() bỏ phần tử cuỗi cùng, push() thêm vào cuối.

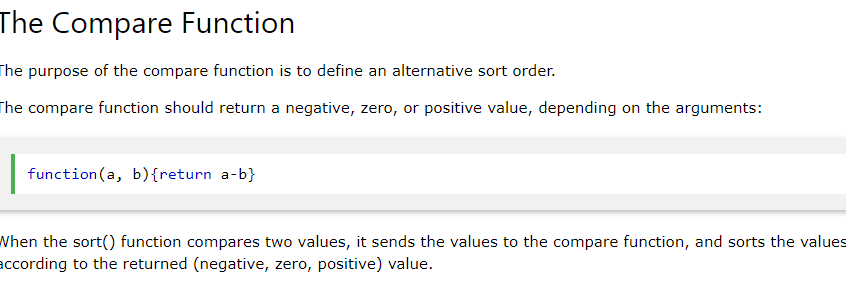
shift() thêm phần tử vào đầu, unshift() xóa phần tử ở đầu.

splice(x,y,str1,str2) thêm str1, str2 vào vị trí x trong mảng và bỏ y phần tử ở vị trí x.

arr1.concat(arr2) nối arr2 vào arr1.

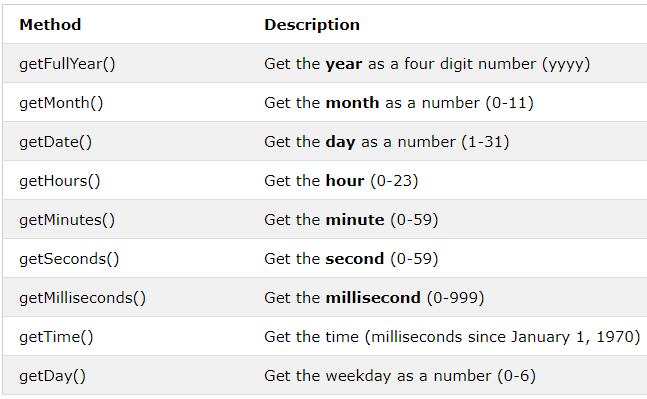
arr1.slice(x,y) cắt arr1 từ x đến y vào arr nguồn.

sort() sắp xếp arr asc, reverser() sắp xếp arr desc.



# 8. Date

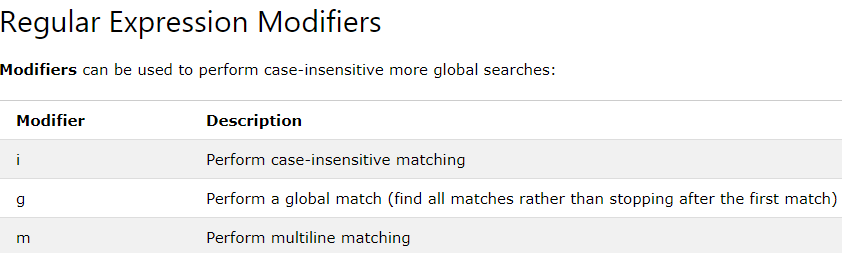
var d = new Date(); lấy time hiện tại

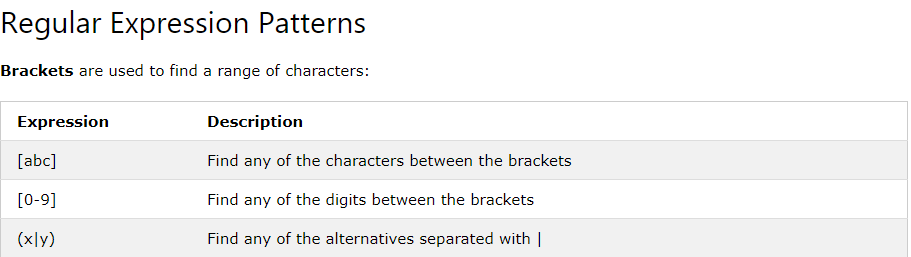
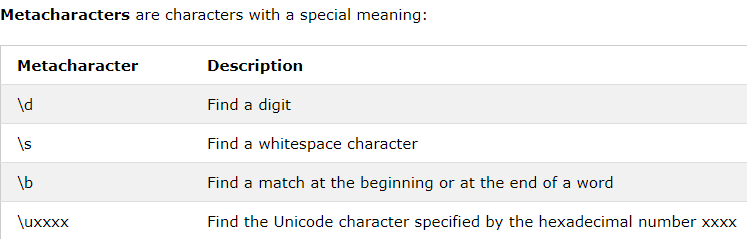


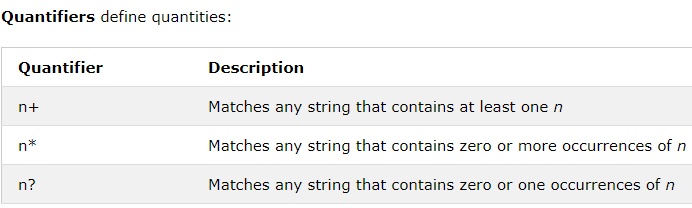
# 9. Math

Math.random() trả về 1 số từ 0->1

# 10. Regexp

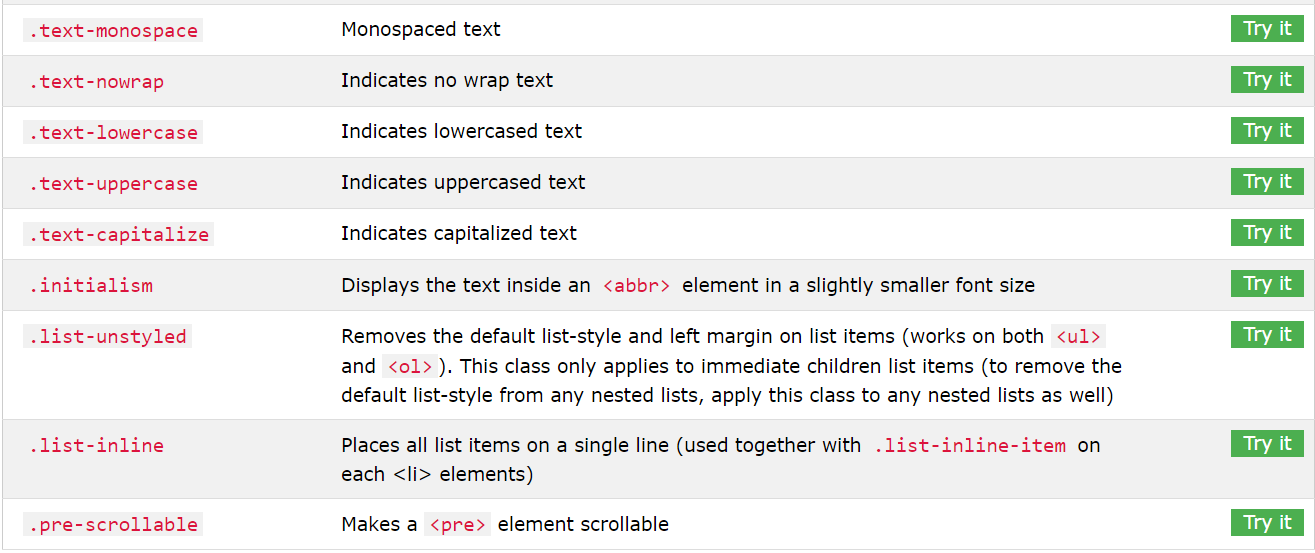




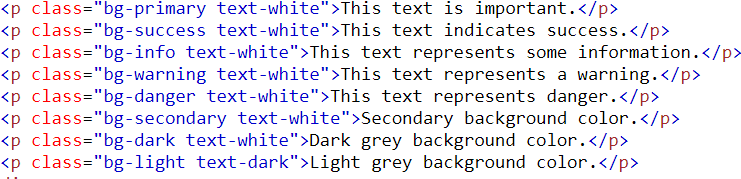


Bootstrap



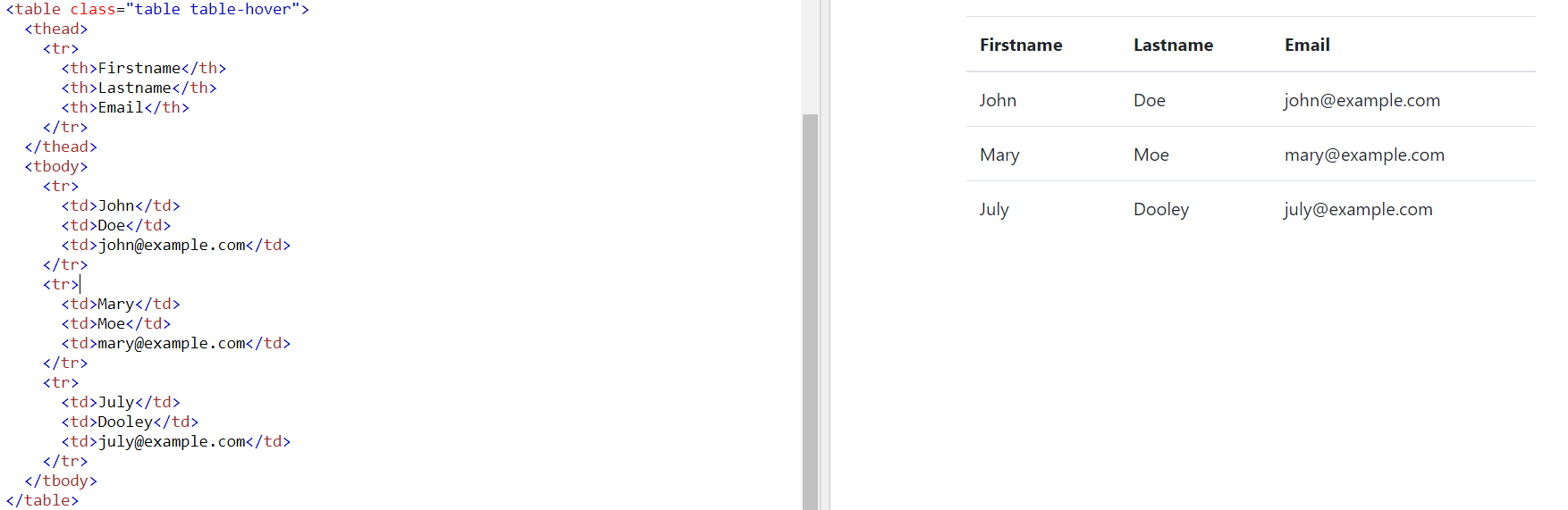


Background Colors





## Table



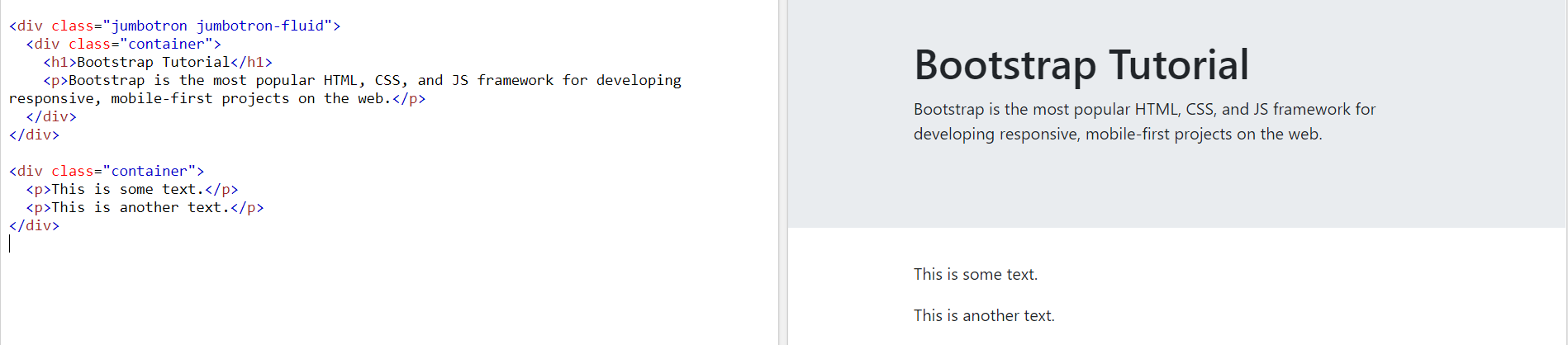
# **Bootstrap 4**Images



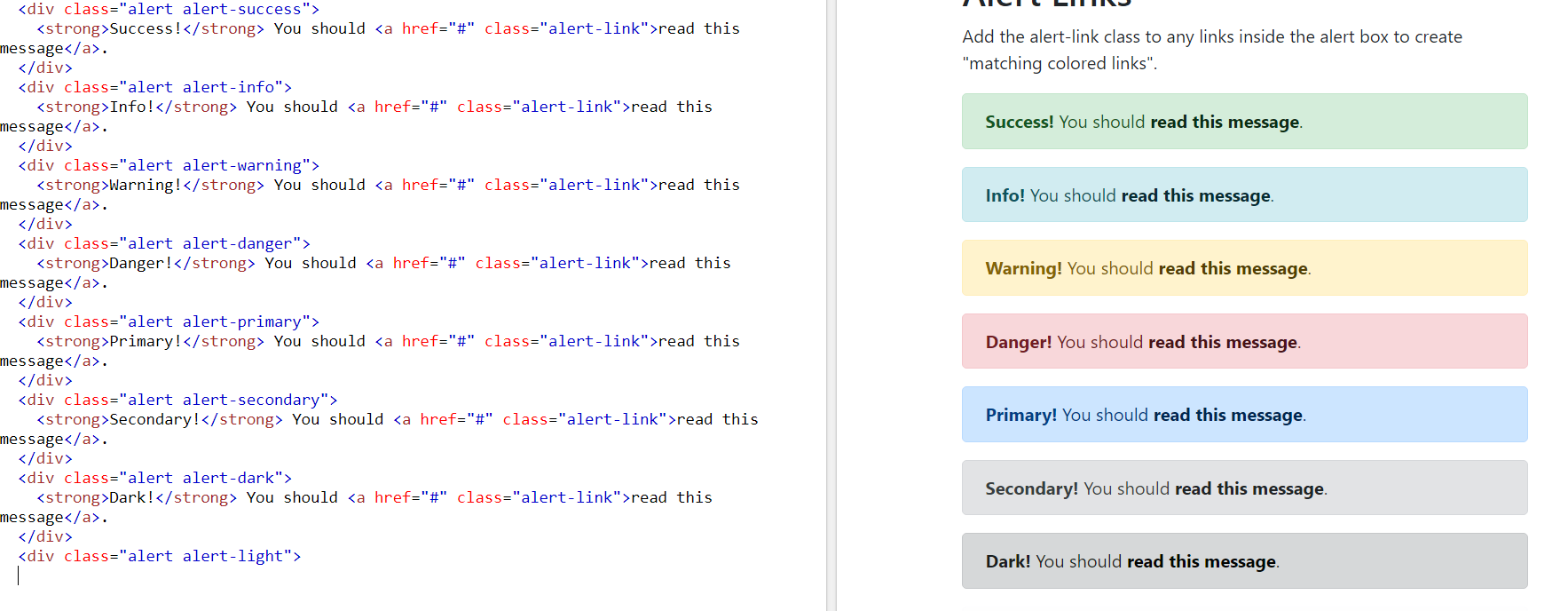


<img src="paris.jpg" class="mx-auto d-block" style="width:50%"> :: căn giữ

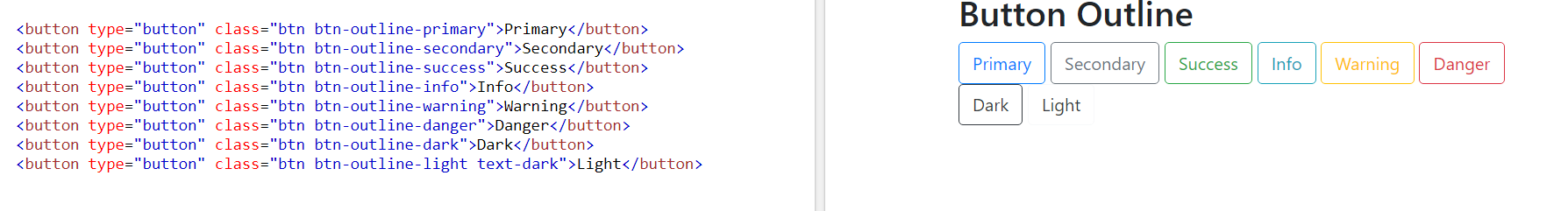
# **Bootstrap 4**Jumbotron

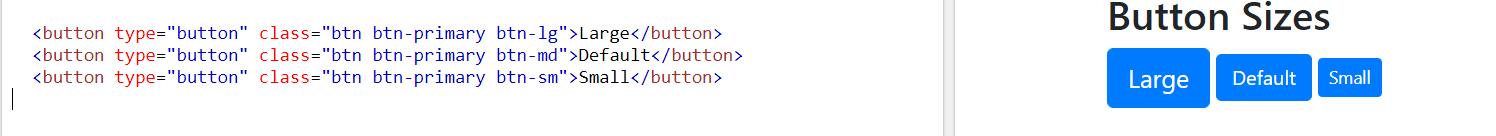


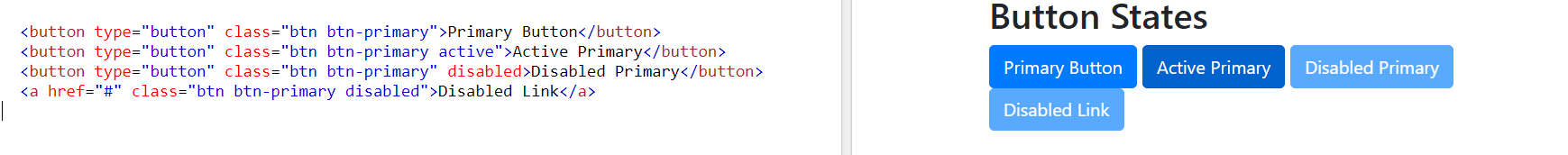
## Bootstrap 4 Alerts



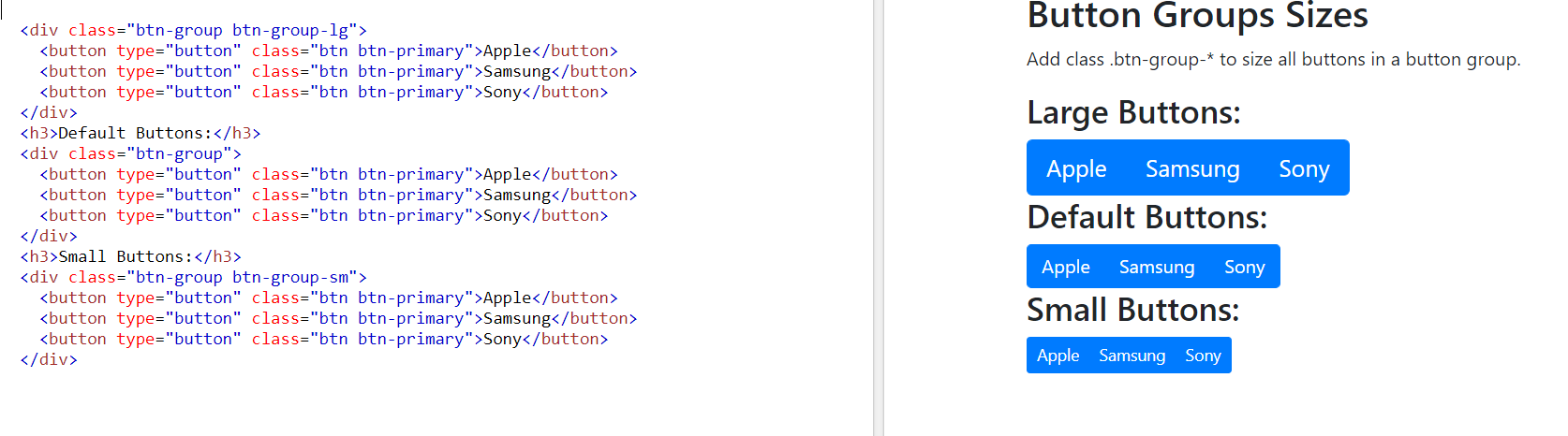
# **Bootstrap 4**Buttons

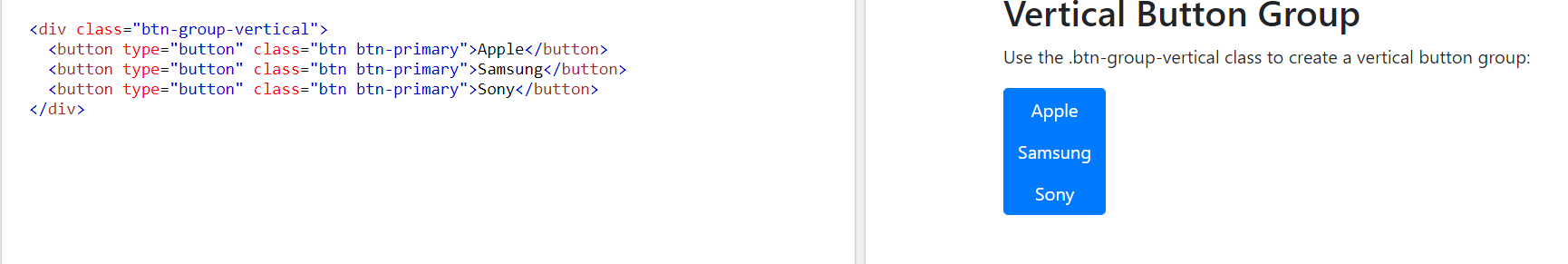


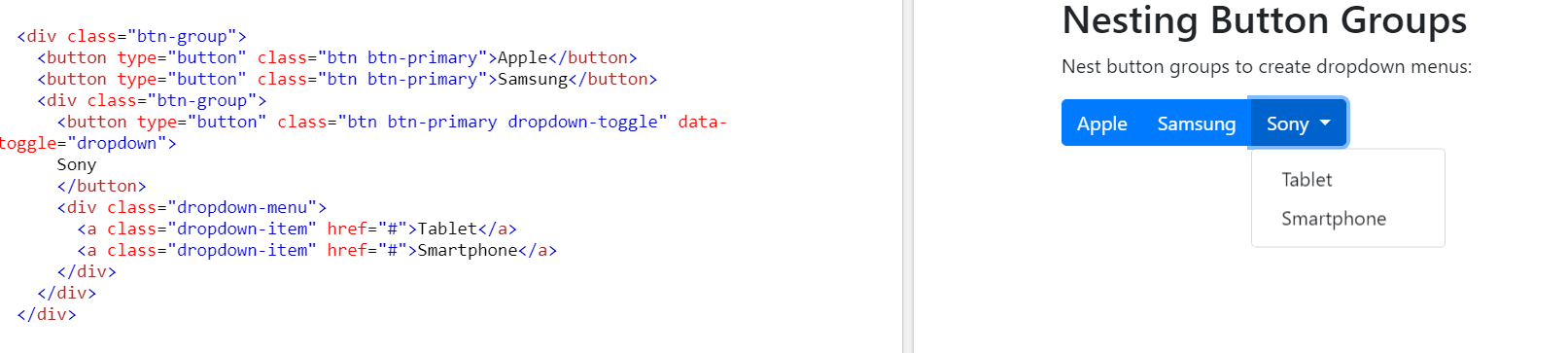






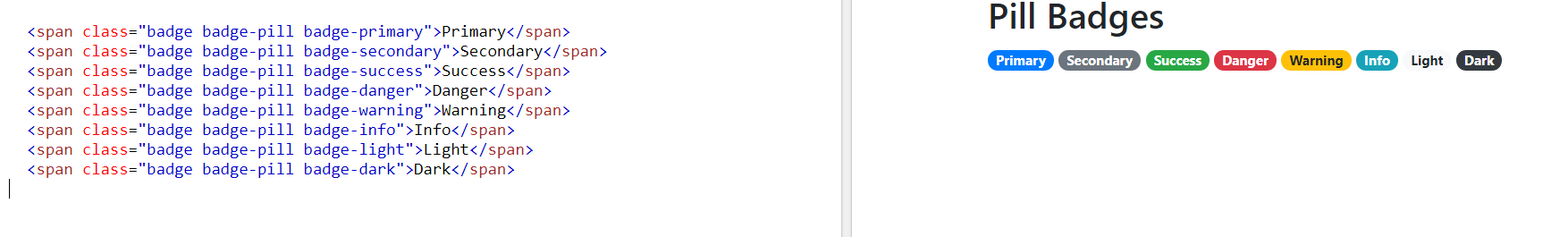






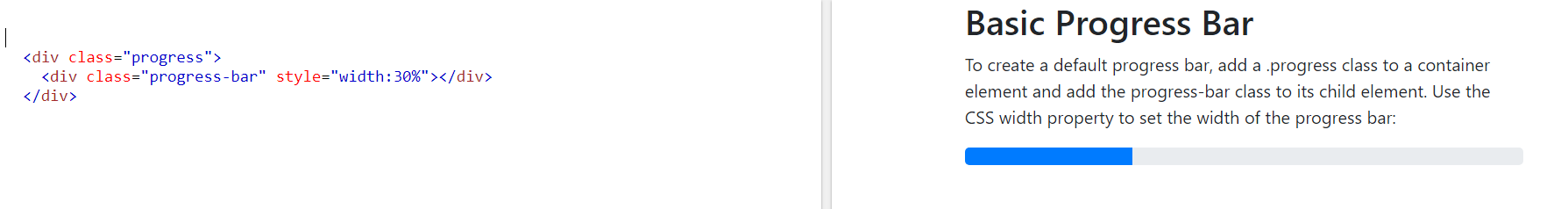
# **Bootstrap 4**Badges



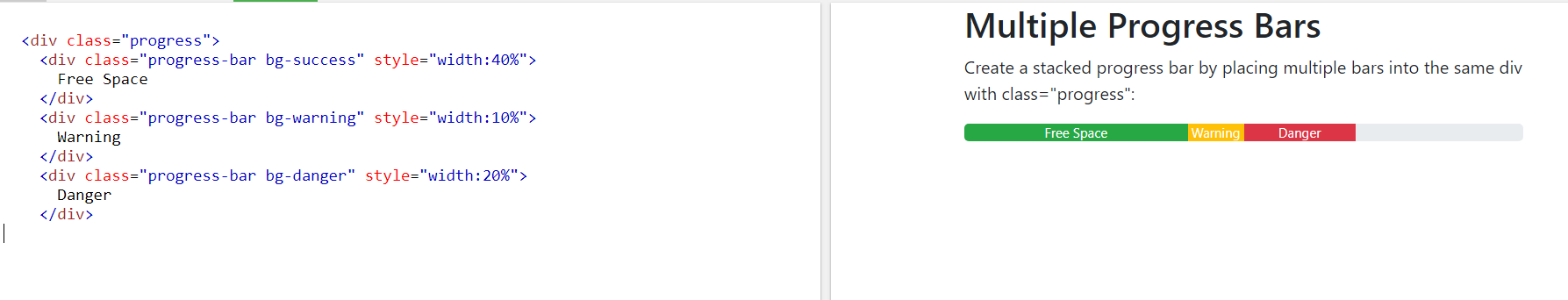




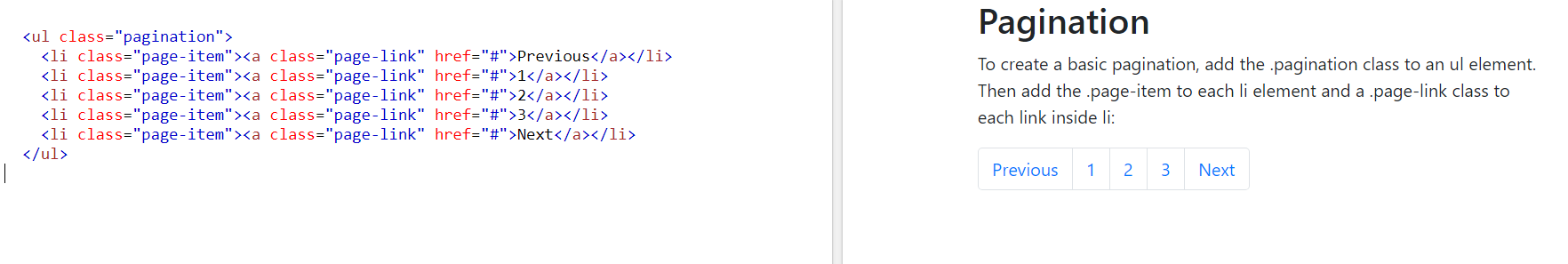
## Basic Progress Bar

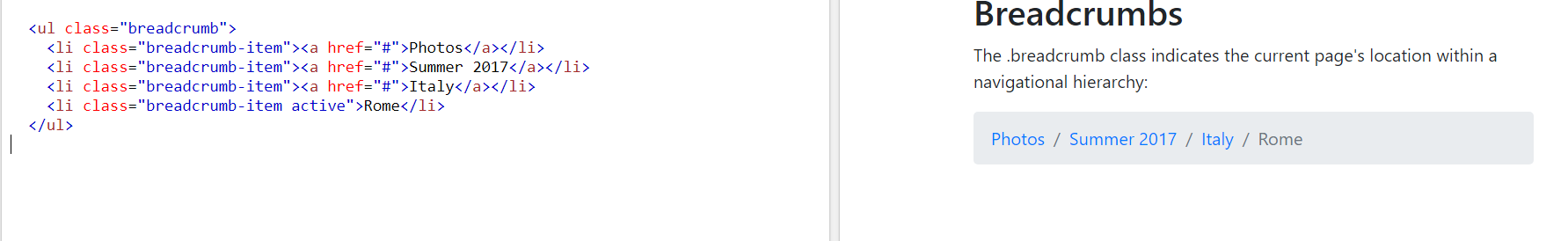






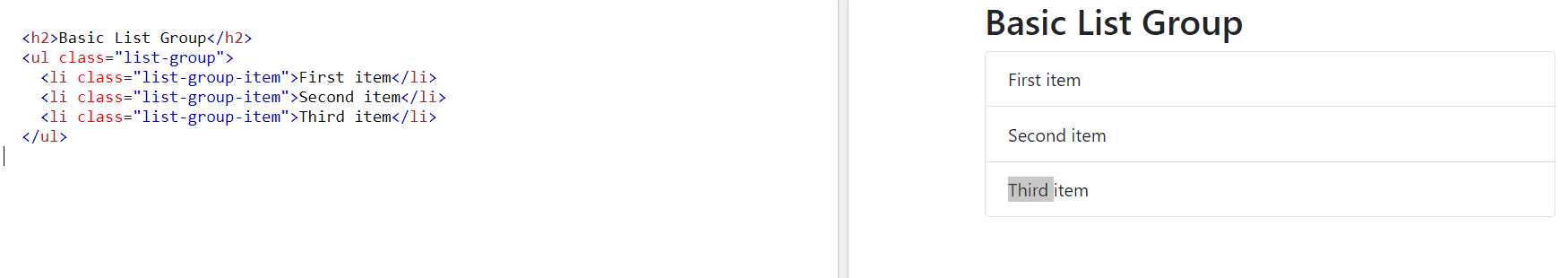
# **Bootstrap 4**Pagination

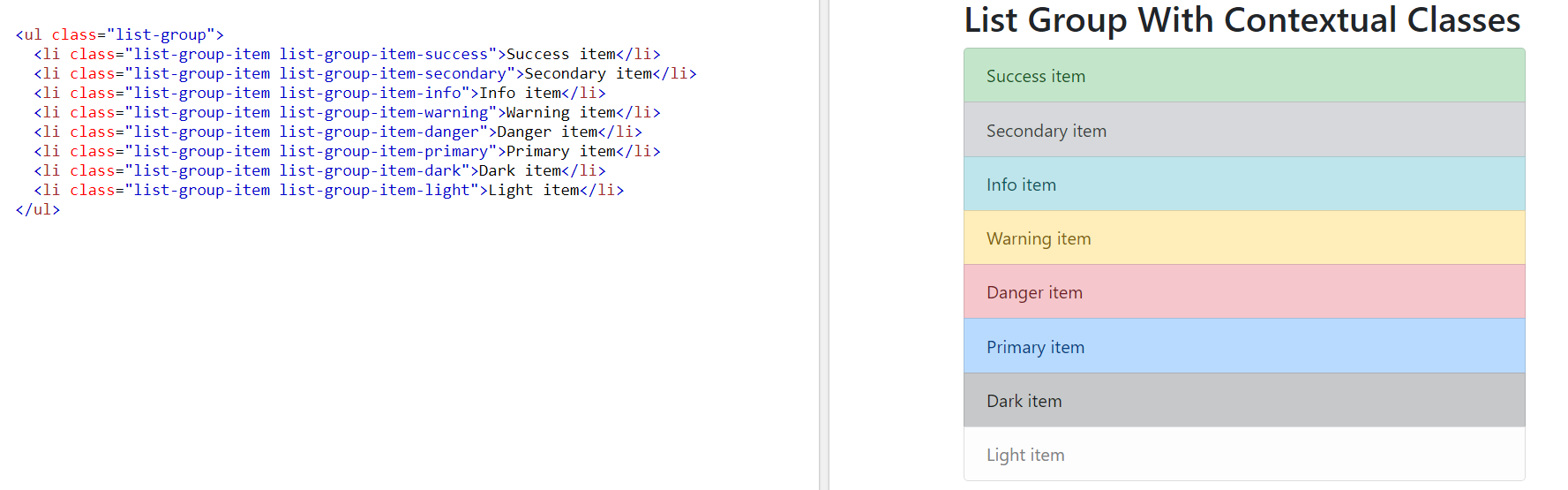






# **Bootstrap 4**List Groups



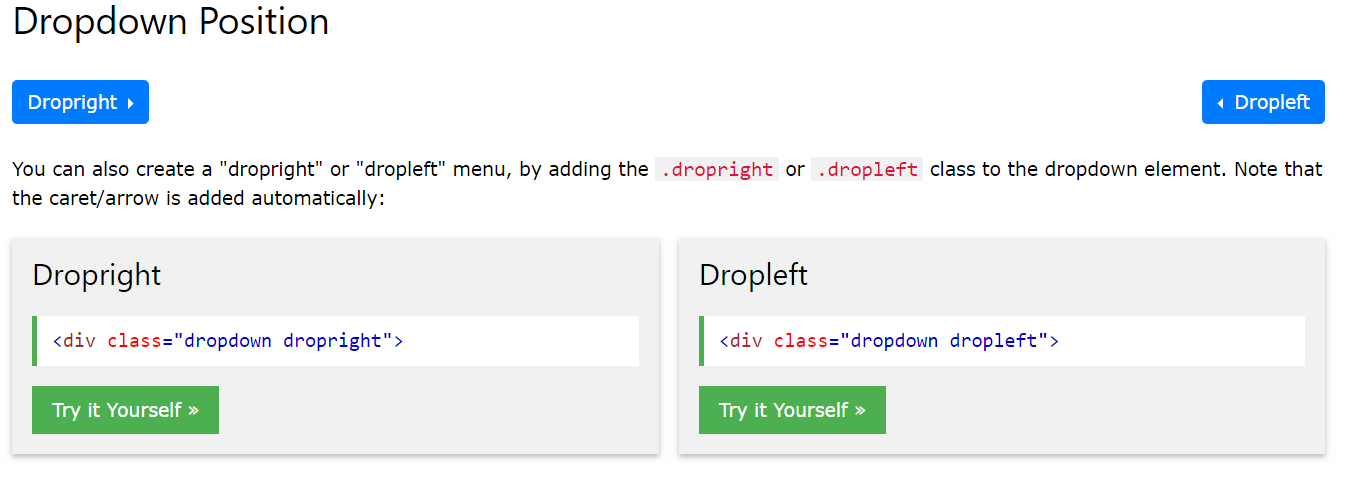


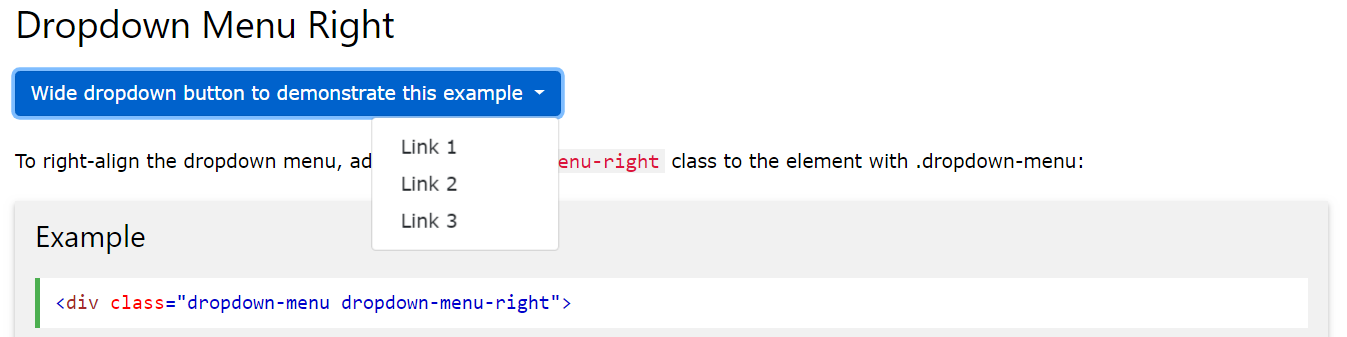
# **Bootstrap 4**Dropdowns

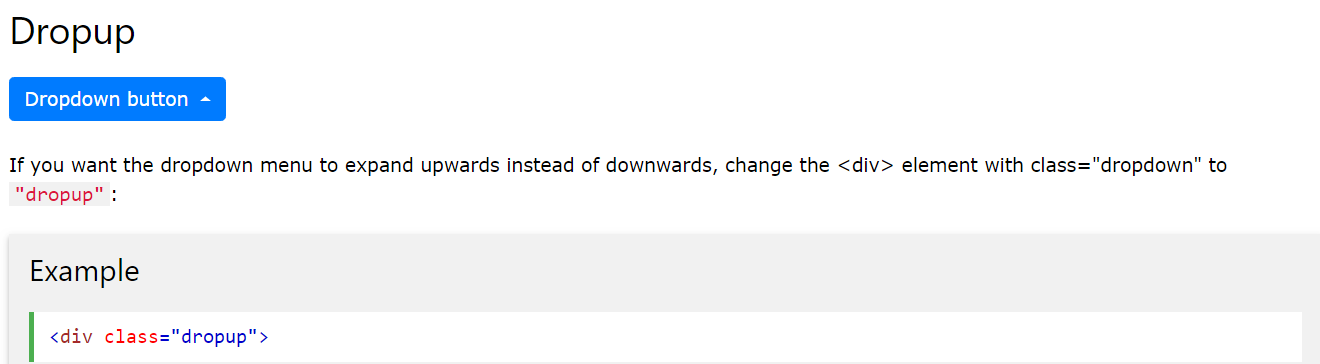


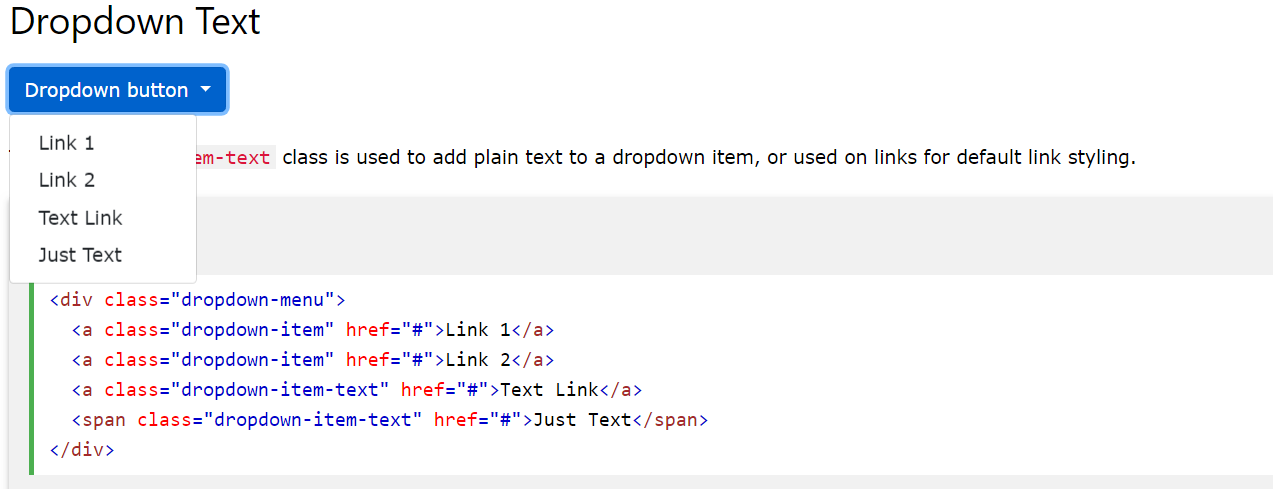


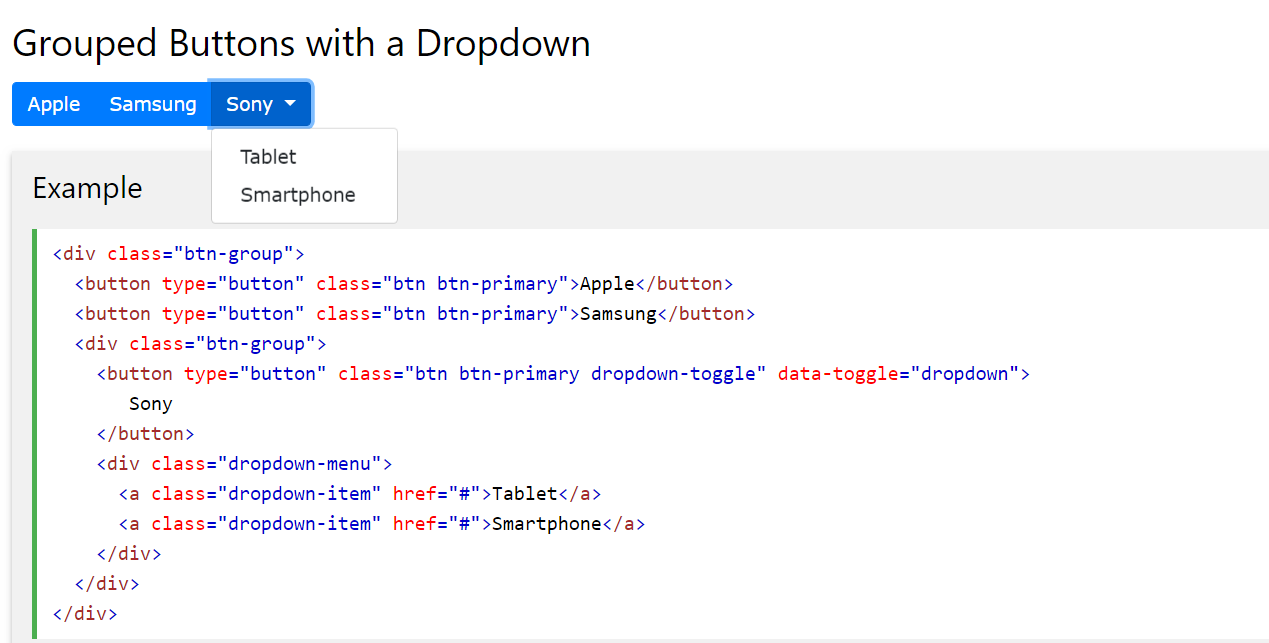




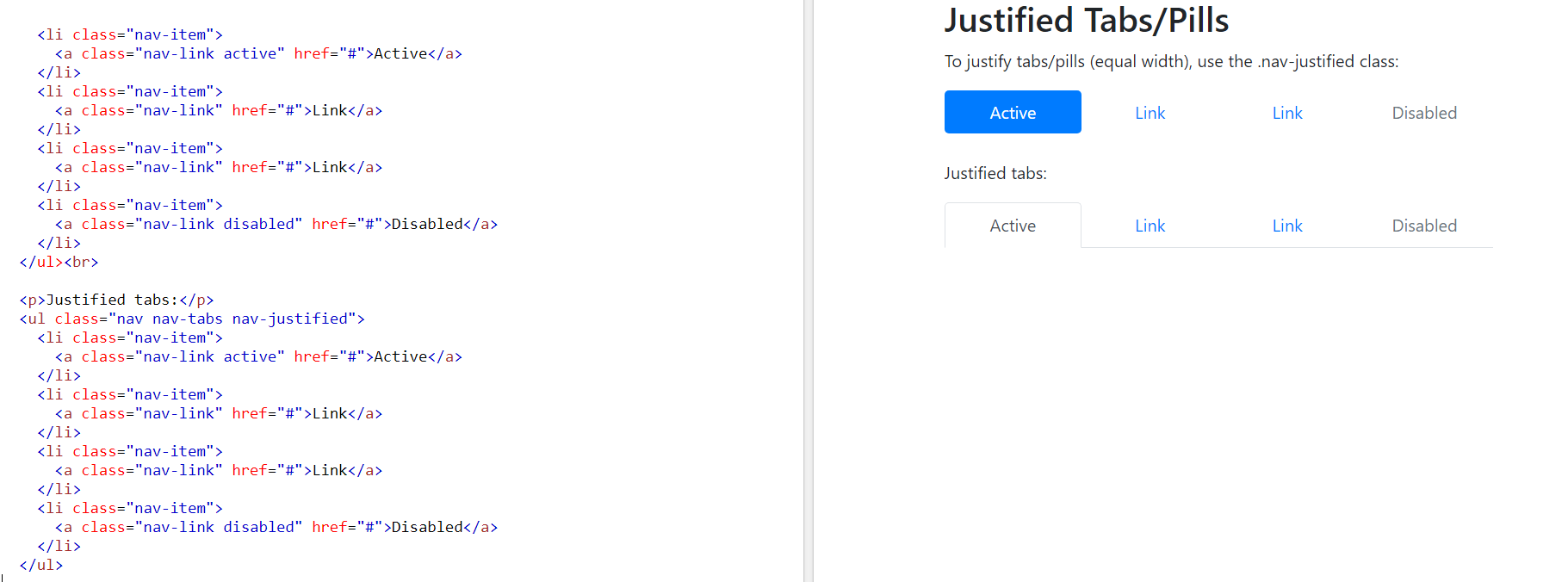








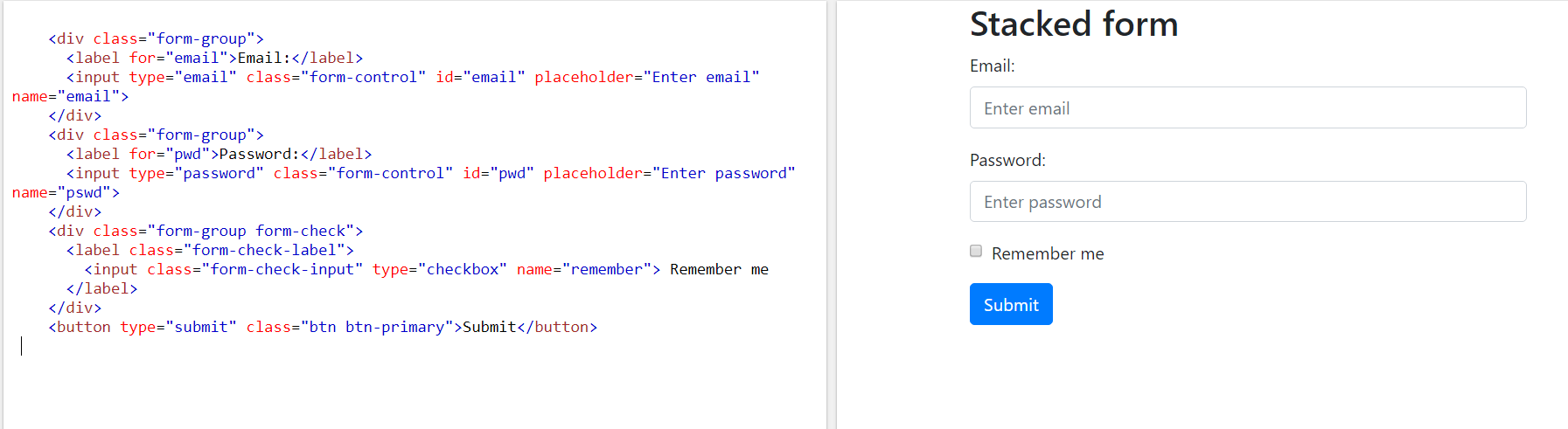
# **Bootstrap 4**Navs



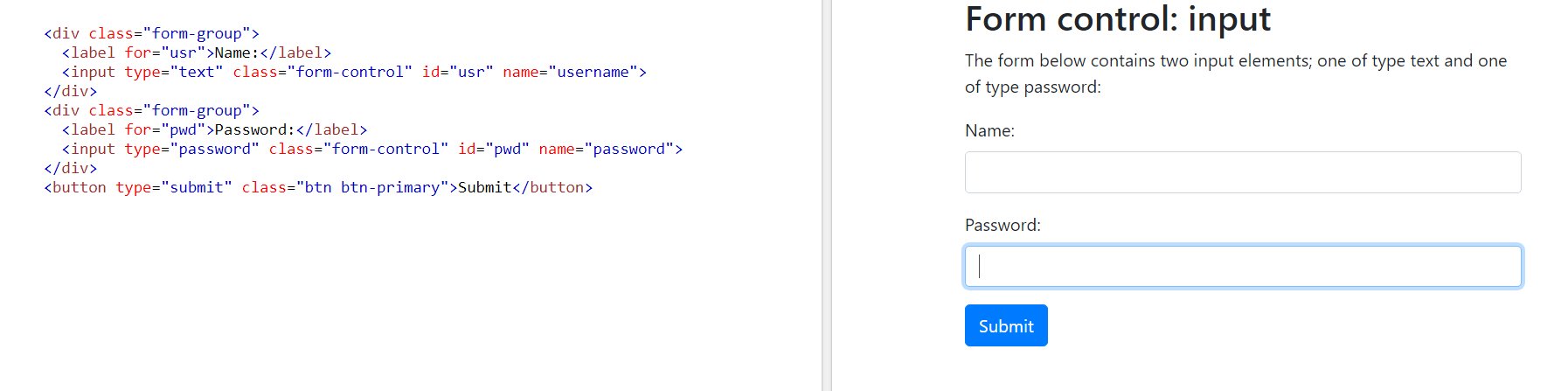
# **Bootstrap 4**Navigation Bar



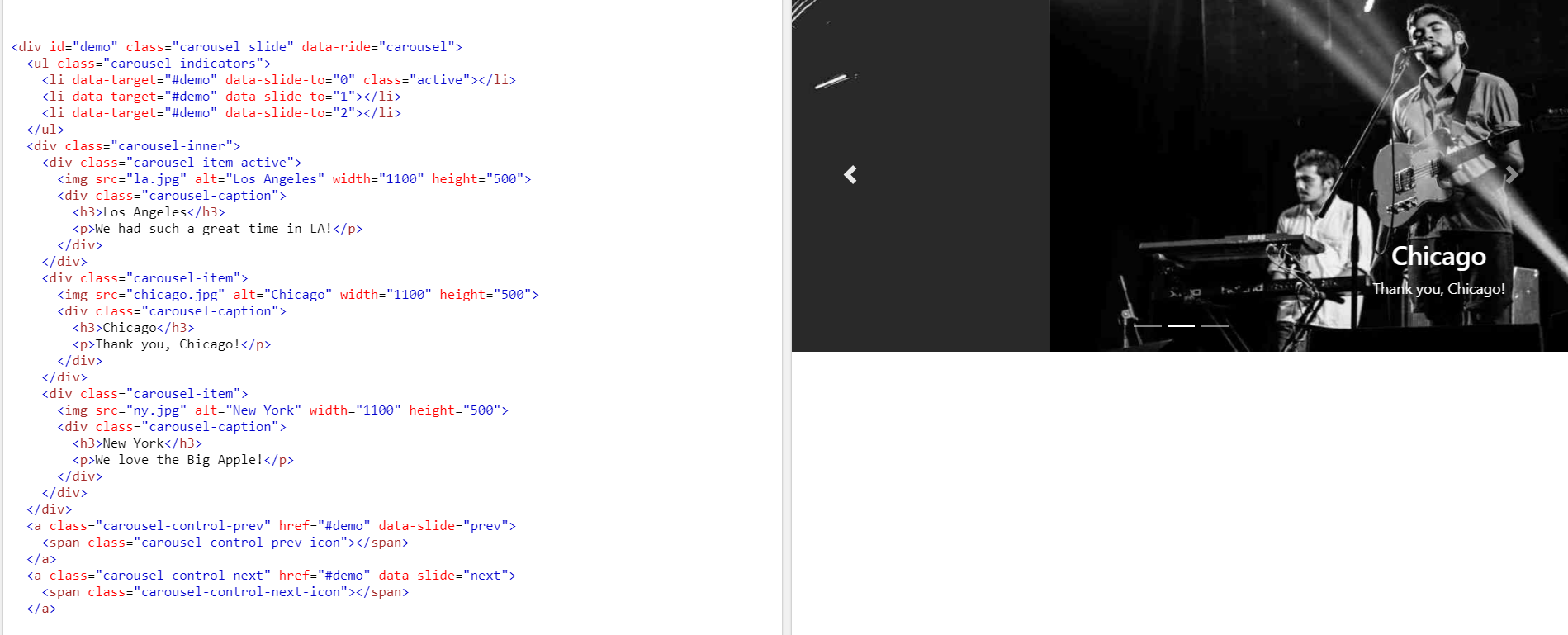
# **Bootstrap 4**Forms

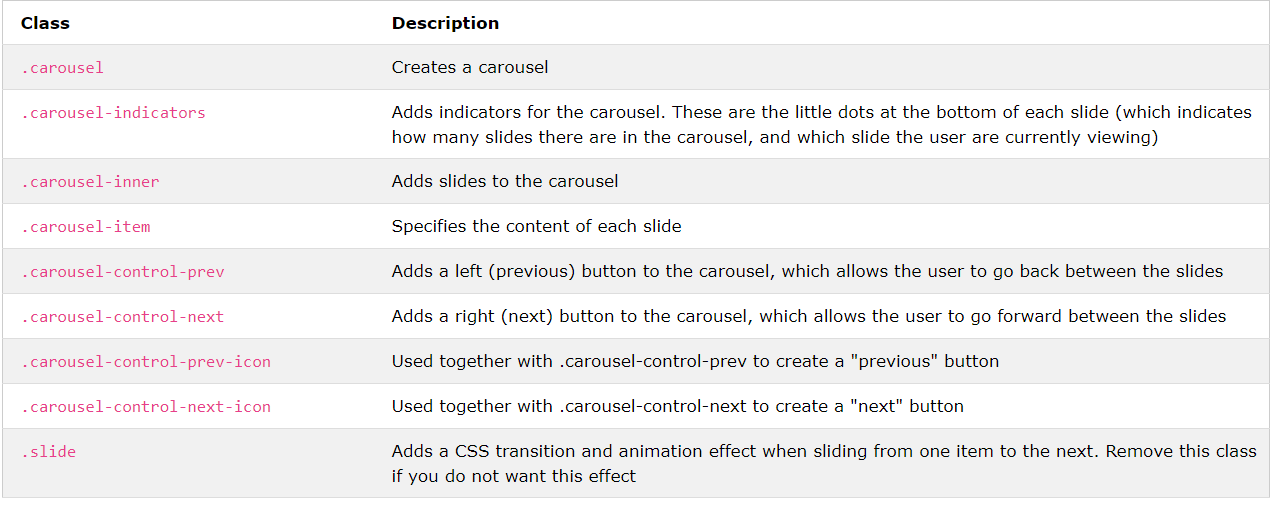


# **Bootstrap 4**Form Inputs

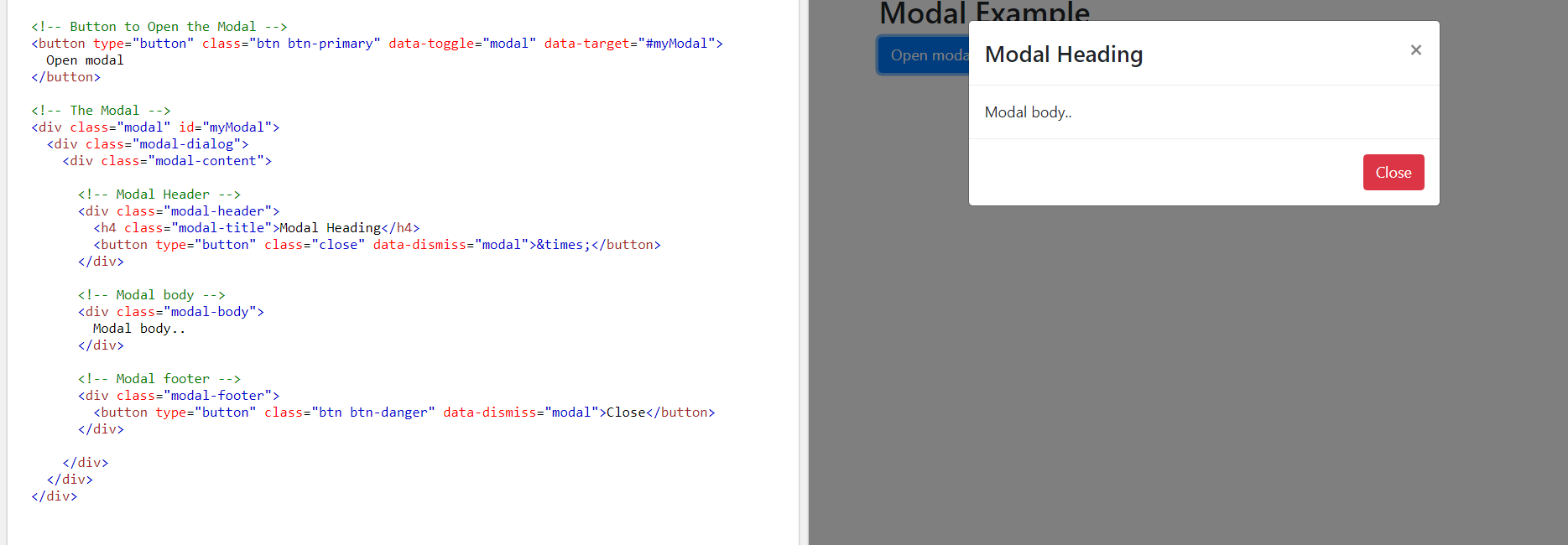


# **Bootstrap 4**Carousel

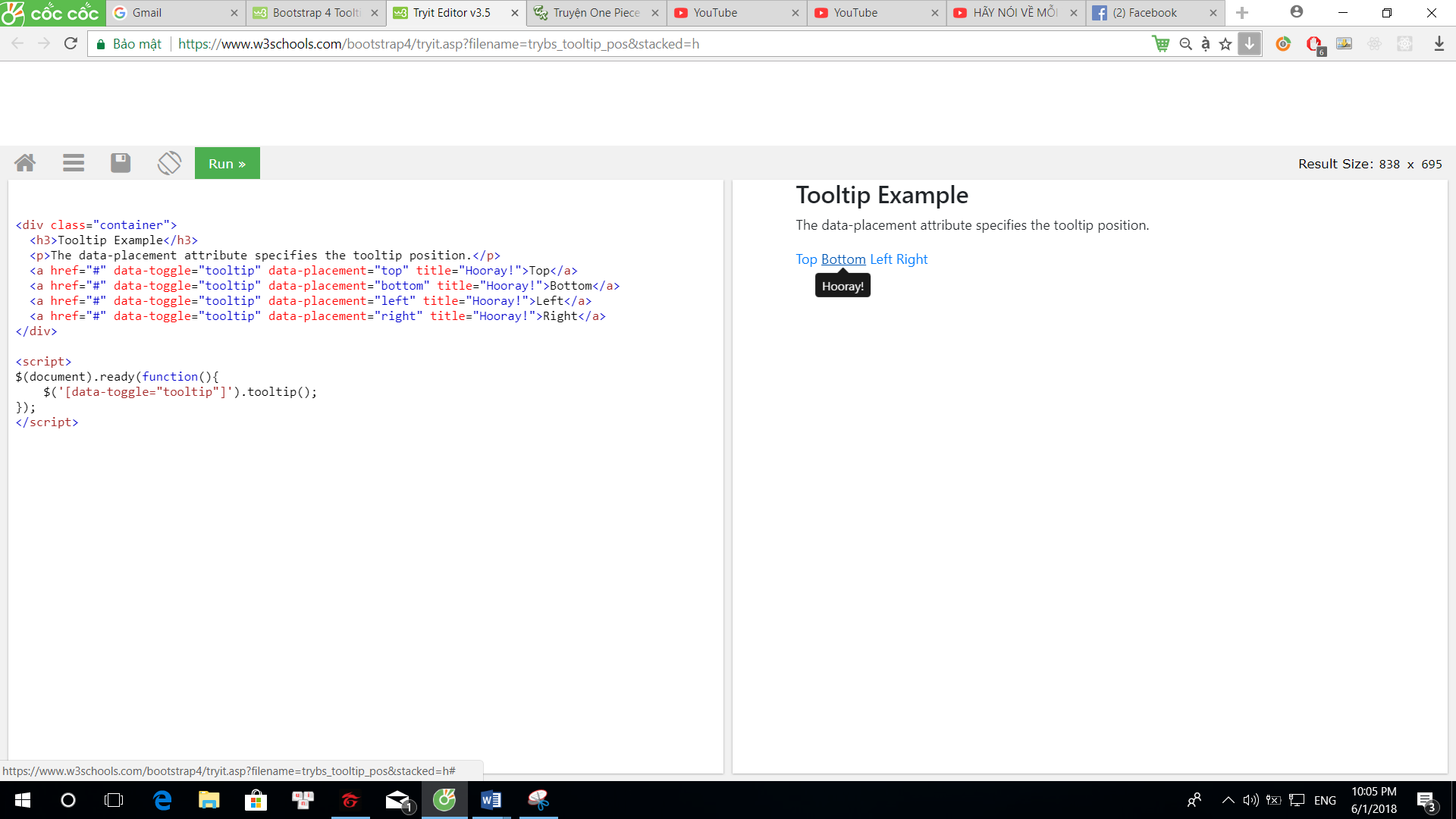




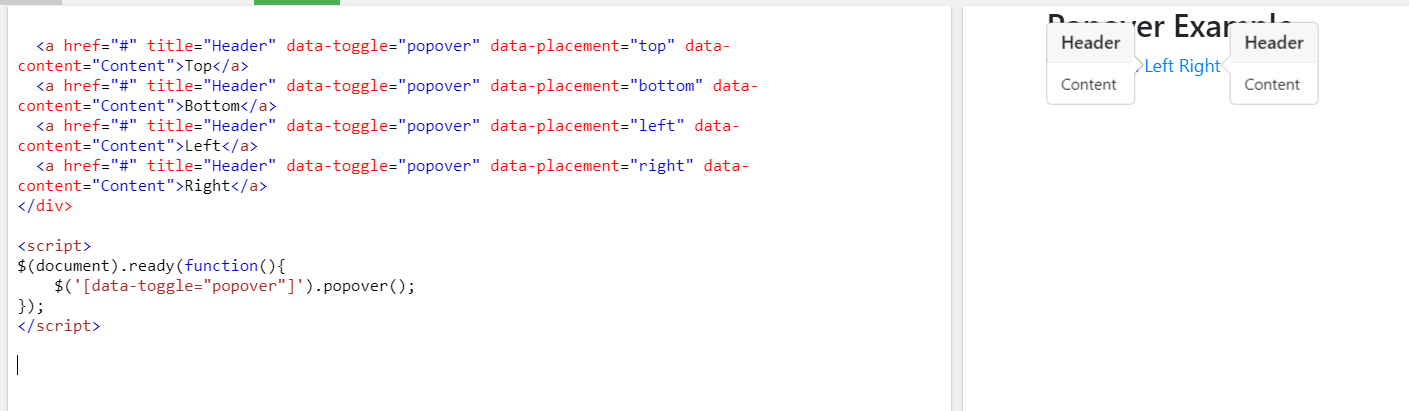
# **Bootstrap 4**Modal



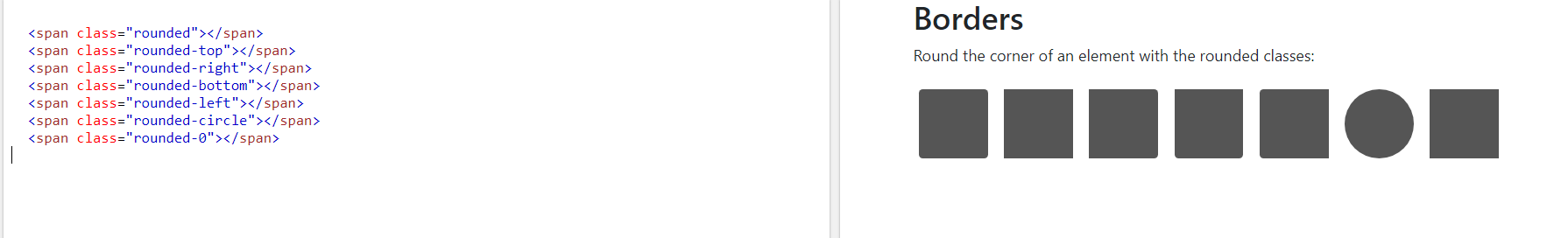
# **Bootstrap 4**Tooltip



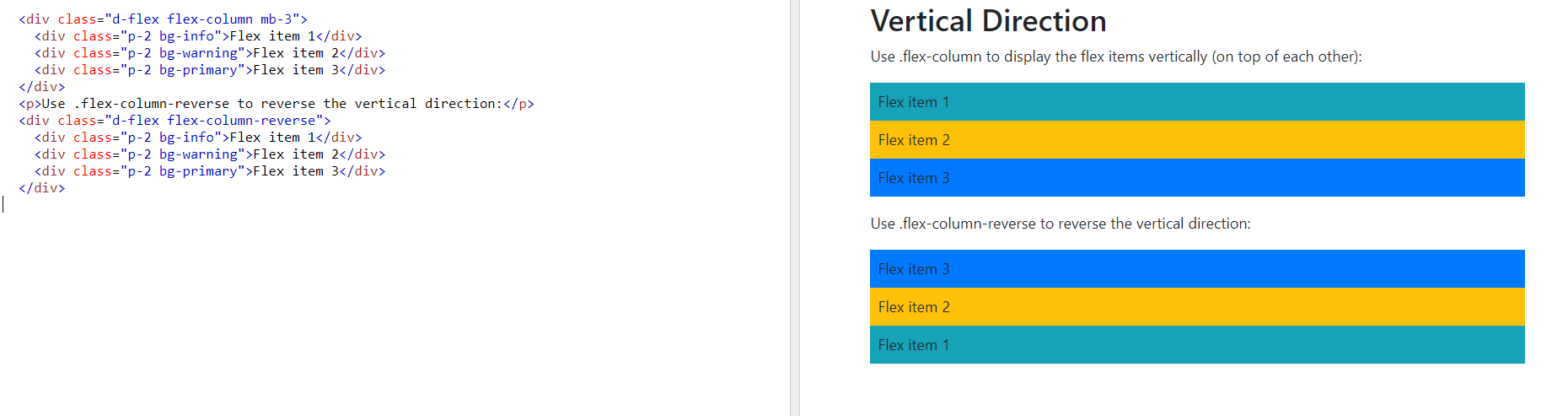
# **Bootstrap 4**Popover

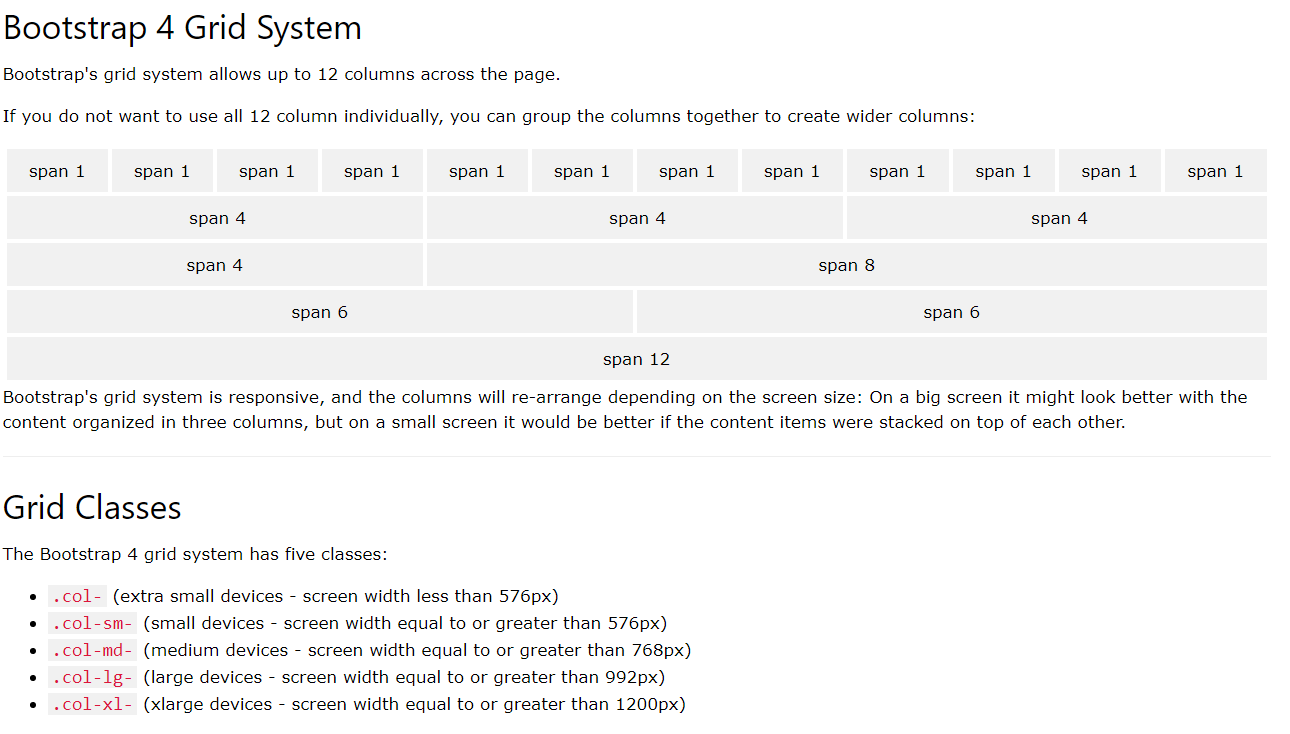


# **Bootstrap 4**Utilities

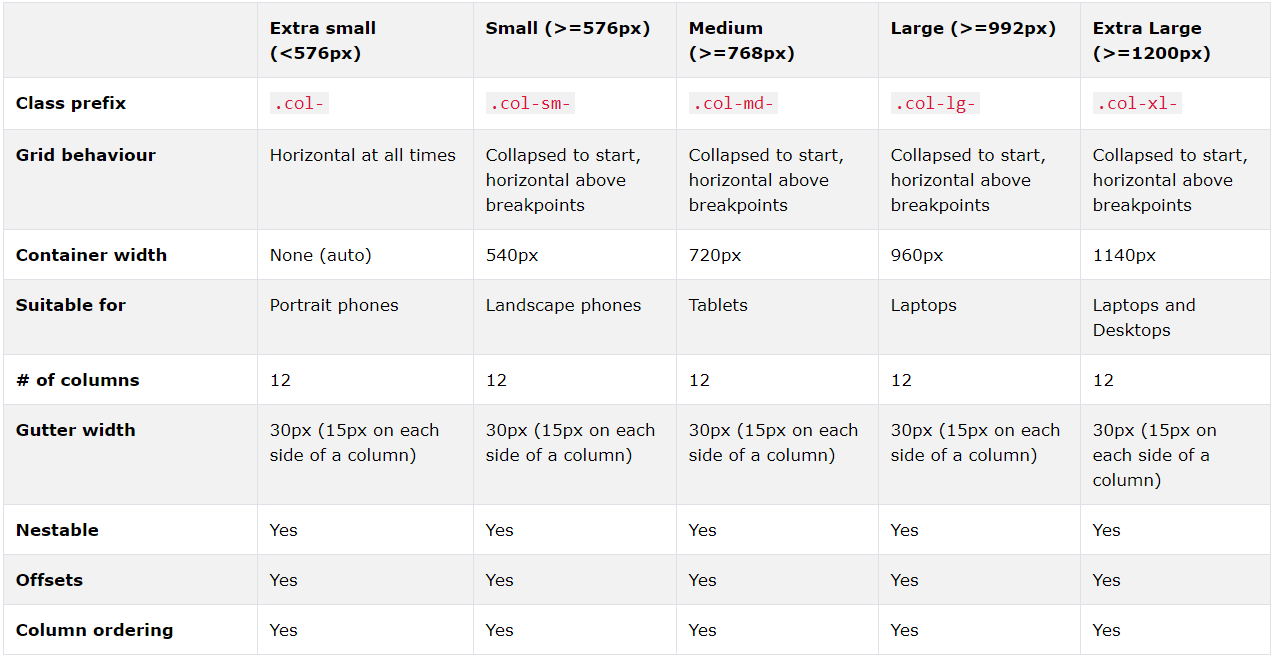


# **Bootstrap 4**Flex

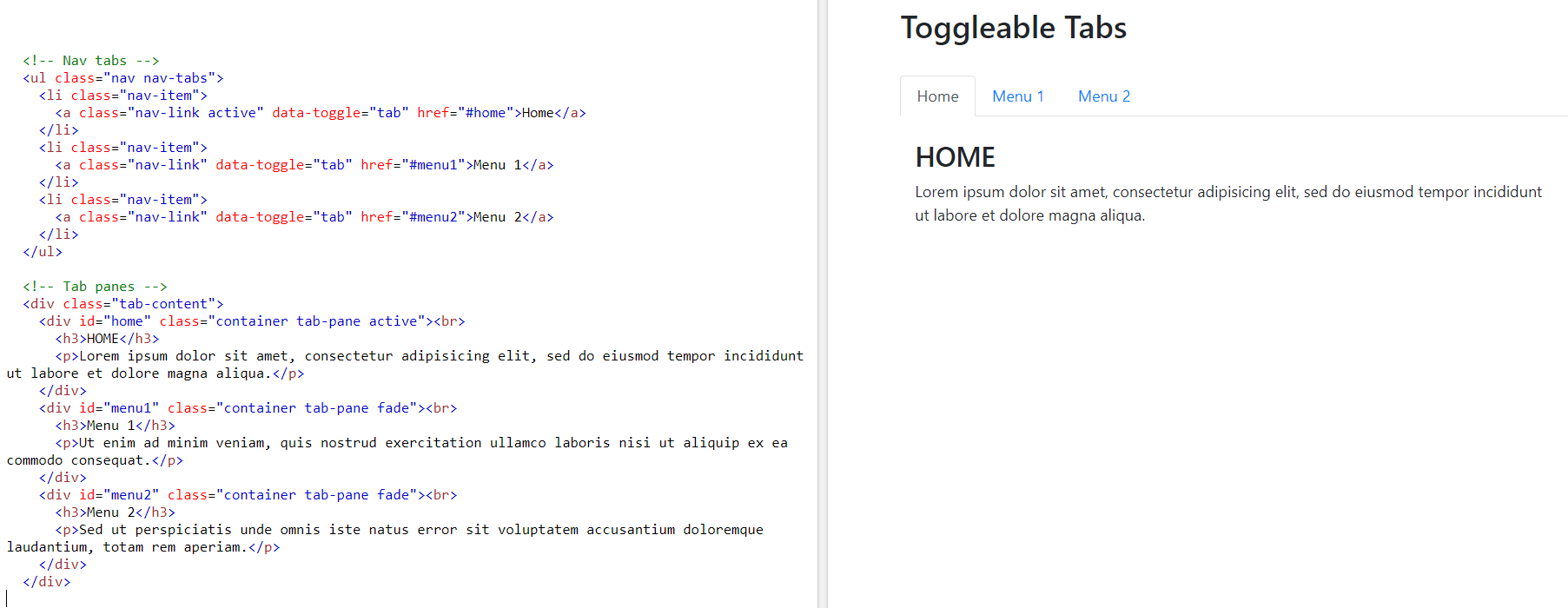




-------------------------------------------------------------------------------------------------------------------------------



# **Bootstrap**JS Tab



AJAX

AJAX - Asynchronous JavaScript And XML.

## Ajax PHP

**Tạo file index.php để xử lý ajax**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title></title>**

**<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">**

**<script language="javascript">**

**function load\_ajax(){**

**alert('Clicked');**

**}**

**</script>**

**</head>**

**<body>**

**<div id="result">**

**Nội dung ajax sẽ được load ở đây**

**</div>**

**<input type="button" name="clickme" id="clickme" onclick="load\_ajax()" value="Click Me"/>**

**</body>**

**</html>**

Trong đó tôi đã tạo một thẻ div dùng để chứa nội dung **load ajax**, một button trong đó có sự kiện **onclick**, sự kiện này sẽ gọi đến hàm **load\_ajax**.

### Tạo file trả về nội dung ajax

**<?php**

**echo 'Đây là nội dung trả về';**

**?>**

### Viết code gọi ajax lấy nội dung từ file result

 Viết hàm xủ lý **gọi ajax** để lấy nội dung từ file result.php và gán nội dung đó vào thẻ div như khai báo ở file index.php trên

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title></title>**

**<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">**

**<script language="javascript">**

**function load\_ajax()**

**{**

**// Tạo một biến lưu trữ đối tượng XML HTTP. Đối tượng này**

**// tùy thuộc vào trình duyệt browser ta sử dụng nên phải kiểm**

**// tra như bước bên dưới**

**var xmlhttp;**

**// Nếu trình duyệt là  IE7+, Firefox, Chrome, Opera, Safari**

**if (window.XMLHttpRequest)**

**{**

**xmlhttp = new XMLHttpRequest();**

**}**

**// Nếu trình duyệt là IE6, IE5**

**else**

**{**

**xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");**

**}**

**// Khởi tạo một hàm gửi ajax**

**xmlhttp.onreadystatechange = function()**

**{**

**// Nếu đối tượng XML HTTP trả về với hai thông số bên dưới thì mọi chuyện**

**// coi như thành công**

**if (xmlhttp.readyState == 4 && xmlhttp.status == 200)**

**{**

**// Sau khi thành công tiến hành thay đổi nội dung của thẻ div, nội dung**

**// ở đây chính là**

**document.getElementById("result").innerHTML = xmlhttp.responseText;**

**}**

**};**

**// Khai báo với phương thức GET, và url chính là file result.php**

**xmlhttp.open("GET", "result.php", true);**

**// Cuối cùng là Gửi ajax, sau khi gọi hàm send thì function vừa tạo ở**

**// trên (onreadystatechange) sẽ được chạy**

**xmlhttp.send();**

**}**

**</script>**

**</head>**

**<body>**

**<div id="result">**

**Nội dung ajax sẽ được load ở đây**

**</div>**

**<input type="button" name="clickme" id="clickme" onclick="load\_ajax()" value="Click Me"/>**

**</body>**

**</html>**

## Ajax JS

### XMLHttpRequest

Để tạo một đối tượng XMLHttpRequest:

var xhttp = new XMLHttpRequest();

Các phương thức phổ biến của lớp XMLHttpRequest:

* **new XMLHttpRequest()**: khởi tạo một đối tượng XMLHttpRequest
* abort(): hủy http request hiện tại
* getAllResponseHeaders(): trả về thông tin header http request
* **open(*method, url, async, user, psw*)**: tạo 1 http request, với các tham số: method - GET hoặc POST, url - vị trí lưu file, async - true không đồng bộ false đồng bộ, user - tên người dùng, psw - mật khẩu.
* **send()**: gửi request lên server (thường dùng với phương thức GET)
* **send(string)**: gửi request lên server (thường dùng với phương thức POST)

**Các thuộc tính của lớp XMLHttpRequest:**

* onreadystatechange: định nghĩa một hàm sẽ được gọi khi thuộc tính readyState thay đổi
* **readyState** : trạng thái của HTTP request. 0: request chưa được khởi tạo, 1: request đã khởi tạo, 2: request đã nhận được bởi server, 3: request đang trong thời gian xử lý, 4: request đã hoàn thành và dữ liệu cần đã được trả về.
* responseText: lưu dữ liệu trả về dưới dạng chuỗi
* responseXML: lưu dữ liệu trả về dưới dạng XML
* **status** : trạng thái của HTTP request - dạng số (ví dụ 200 , 403, 404 )
* statusText: trạng thái của HTTP request - dạng text (ví dụ "OK", "Forbidden", "Not Found" )

**Ví dụ:**

Get XML file:

function loadDoc() {

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange = function() {

if (this.readyState == 4 && this.status == 200) {

myFunction(this);

}

};

xhttp.open("GET", "cd\_catalog.xml", true);

xhttp.send();

}

function myFunction(xml) {

var i;

var xmlDoc = xml.responseXML;

var table="<tr><th>Artist</th><th>Title</th></tr>";

var x = xmlDoc.getElementsByTagName("CD");

for (i = 0; i <x.length; i++) {

table += "<tr><td>" +

x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +

"</td><td>" +

x[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +

"</td></tr>";

}

document.getElementById("demo").innerHTML = table;

}

**Post dữ liệu:**

**xhttp.open("POST", "ajax\_test.asp", true);**

**xhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");**

**xhttp.send("fname=Henry&lname=Ford");**