



THỰC HỌC – THỰC NGHIỆP

LẬP TRÌNH JAVASCRIPT

Dynamic Element Manipulation using
DOM

HỆ THỐNG BÀI CŨ

- Giới thiệu về mô hình Document Object Model
- Giới thiệu về HTML DOM
- Cấu trúc DOM
- Thuộc tính của node
- Phương thức của node
 - Truy cập đến node
 - Thêm node
 - Xóa node
- **This keyword và DOM (chú ý)**
- Đối phó với các trình duyệt khác nhau

MỤC TIÊU BÀI HỌC

- Thao tác với element (phần tử) style.
- Thao tác với attribute.
- Interactive content & event listeners (onload, mouse, onchange, onblur,...)



PHẦN I

THAO TÁC VỚI ELEMENT STYLE, ATTRIBUTES

THAO TÁC VỚI KIỂU CỦA PHẦN TỬ

- Có thể sử dụng Javascript để **thay đổi style** cho trang web
- Javascript cung cấp **đối tượng style** cho mỗi **element** của trang web để thay đổi style của trang web
- Sử dụng thuộc tính style của mỗi element để truy cập đến đối tượng style

```
var hTieuDe = document.getElementById("hTieuDe");  
hTieuDe.style.fontFamily = "arial";
```

- Đối tượng style trong Javascript có các thuộc tính tương ứng với các thuộc tính của CSS
 - **Chú ý:** Với các thuộc tính CSS có dấu gạch ngang thì sẽ được bỏ dấu gạch ngang và viết hoa chữ cái của từ sau dấu gạch ngang (`font-family > fontFamily`)

THAO TÁC VỚI KIỂU CỦA PHẦN TỬ

- Có thể thiết lập style cho các thành phần bằng
 - ID
 - Thẻ
 - Class
- Thiết lập style bằng ID. Các bước thực hiện
 - Bước 1: Dùng ID để truy cập đến các element
 - Bước 2: Sử dụng thuộc tính style để thiết lập style cho element đó

```
var hTieuDe = document.getElementById("hTieuDe");  
hTieuDe.style.fontFamily = "arial";
```

DEMO

Ví dụ `element.style.display = "none|block"`

```
<html>
  <head>
    <title>Hi</title>
  </head>
  <body>
    <script>
      function toggleDisplay(){
        let p = document.getElementById("magic");
        if(p.style.display === "none") {
          p.style.display = "block";
        } else {
          p.style.display = "none";
        }
      }
    </script>
    <p id="magic">I might disappear and appear.
    <button onclick="toggleDisplay()">Magic!</button>
  </body>
</html>
```

Thiết lập style cho một nhóm phần tử

- Vấn đề nảy sinh:

- Muốn thay đổi style cho nhiều phần tử

```
<div id="red"></div>
<div id="orange"></div>
<div id="yellow"></div>
<div id="green"></div>
<div id="blue"></div>
<div id="indigo"></div>
<div id="violet"></div>
<button onclick="rainbowify()">Make me a rainbow</button>
```

- Giải pháp
- => Sử dụng phương thức `getElementsByName` hoặc `getElementsByTagName` để lấy nhóm các phần tử
- => Sử dụng vòng lặp `for` để duyệt qua các phần tử (xem demo)

DEMO

- Ví dụ `element.style.backgroundColor = "color"`

```
<script>
function rainbowify(){
    let divs = document.getElementsByTagName("div");
    for(let i = 0; i < divs.length; i++) {
        divs[i].style.backgroundColor = divs[i].id;
    }
}
</script>

<div id="red"></div>
<div id="orange"></div>
<div id="yellow"></div>
<div id="green"></div>
<div id="blue"></div>
<div id="indigo"></div>
<div id="violet"></div>
<button onclick="rainbowify()">Make me a rainbow</button>
```



THAY ĐỔI CLASS CỦA PHẦN TỬ

- Thêm class cho phần tử
- Di chuyển class khỏi phần tử
- Toggle class

THAY ĐỔI CLASS CỦA PHẦN TỬ

- Thêm class cho phần tử: `classList.add(<class name>)`

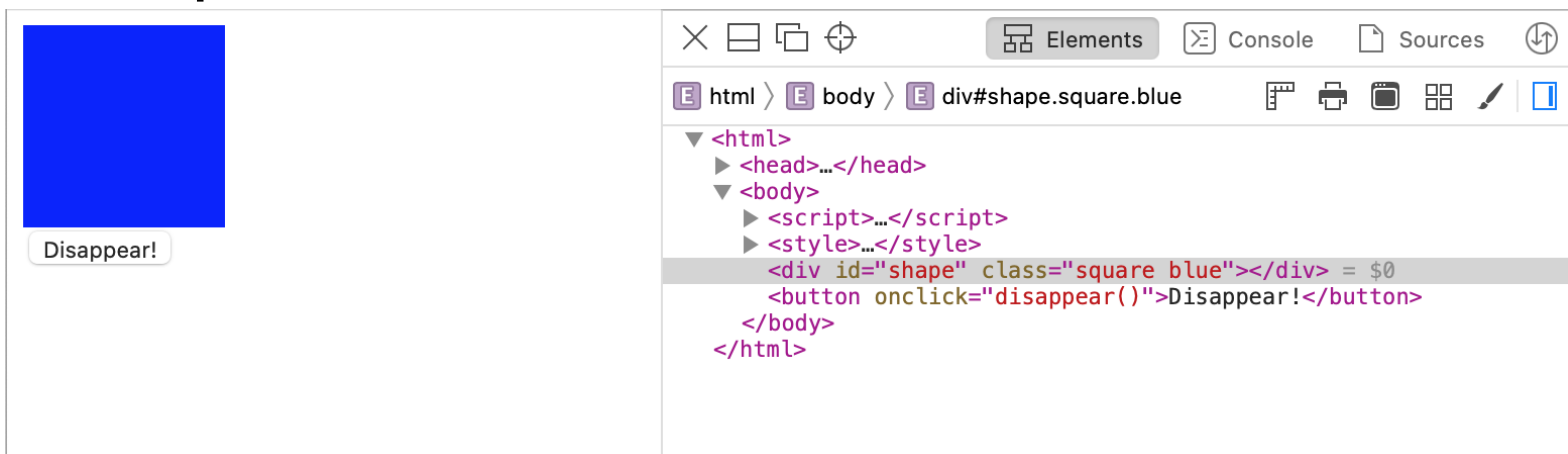
```
<script>
function disappear(){
  document.getElementById("shape").classList.add("hide");
}
</script>
```

```
<div id="shape" class="square blue"></div>
<button onclick="disappear()">Disappear!</button>
```

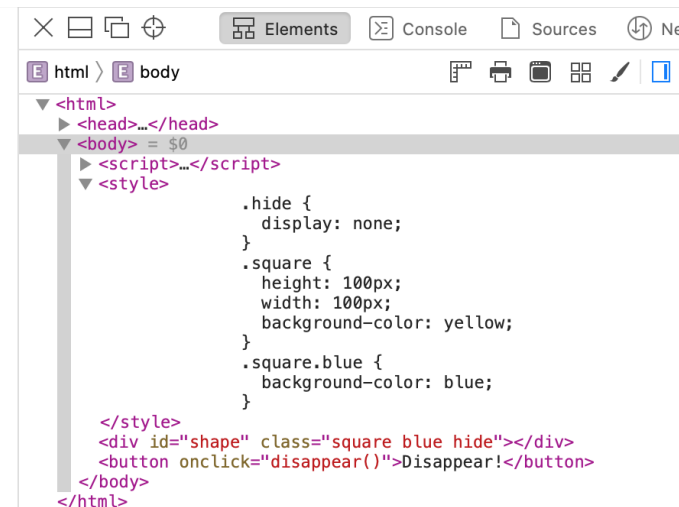
```
<style>
  .hide {
    display: none;
  }
  .square {
    height: 100px;
    width: 100px;
    background-color: yellow;
  }
  .square.blue {
    background-color: blue;
  }
</style>
```

THAY ĐỔI CLASS CỦA PHẦN TỬ

- Thêm class cho phần tử: `classList.add(<class name>)`

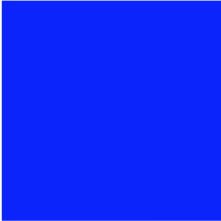


Disappear!

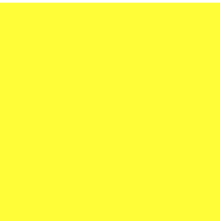


THAY ĐỔI CLASS CỦA PHẦN TỬ

- Di chuyển (remove) class khỏi phần tử: `classList.remove(<class name>)`



Disappear!



Change!

Elements Console Sources

html > body > div#shape.square.blue

```
<html>
  <head>...</head>
  <body>
    <script>...</script>
    <style>...</style>
    <div id="shape" class="square blue"></div> = $0
    <button onclick="disappear()">Disappear!</button>
  </body>
</html>
```


Elements Console Sources

html > body > div#shape.square.blue

```
<html>
  <head>...</head>
  <body>
    <script>...</script>
    <style>...</style>
    <div id="shape" class="square"></div> = $0
    <button onclick="change()">Change!</button>
  </body>
</html>
```

THAY ĐỔI CLASS CỦA PHẦN TỬ

- Toggling class: `classList.toggle(<class name>)`



Disappear!

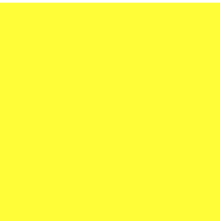
✕ □ □ ⊕ Elements Console Sources ↻

html > body > div#shape.square.blue

```

<html>
  <head>...</head>
  <body>
    <script>...</script>
    <style>...</style>
    <div id="shape" class="square blue"></div> = $0
    <button onclick="disappear()">Disappear!</button>
  </body>
</html>

```



Change!

✕ □ □ ⊕ Elements Console Sources ↻

html > body > div#shape.square.blue

```

<html>
  <head>...</head>
  <body>
    <script>...</script>
    <style>...</style>
    <div id="shape" class="square"></div> = $0
    <button onclick="change()">Change!</button>
  </body>
</html>

```

THAO TÁC THUỘC TÍNH

- Dùng phương thức **setAttribute** của node để thiết lập thuộc tính cho chính node đó.
- Ví dụ

```
<a id="friend" class="fancy boxed"
href="https://www.google.com">Ask my friend
here.</a>
```

Thuộc tính của ví dụ trên gồm: id, class, href

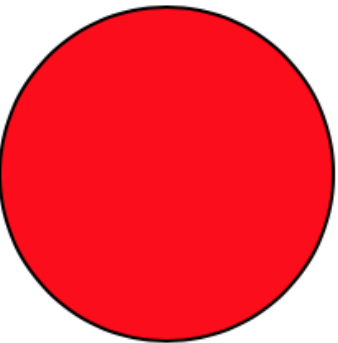
THAO TÁC THUỘC TÍNH



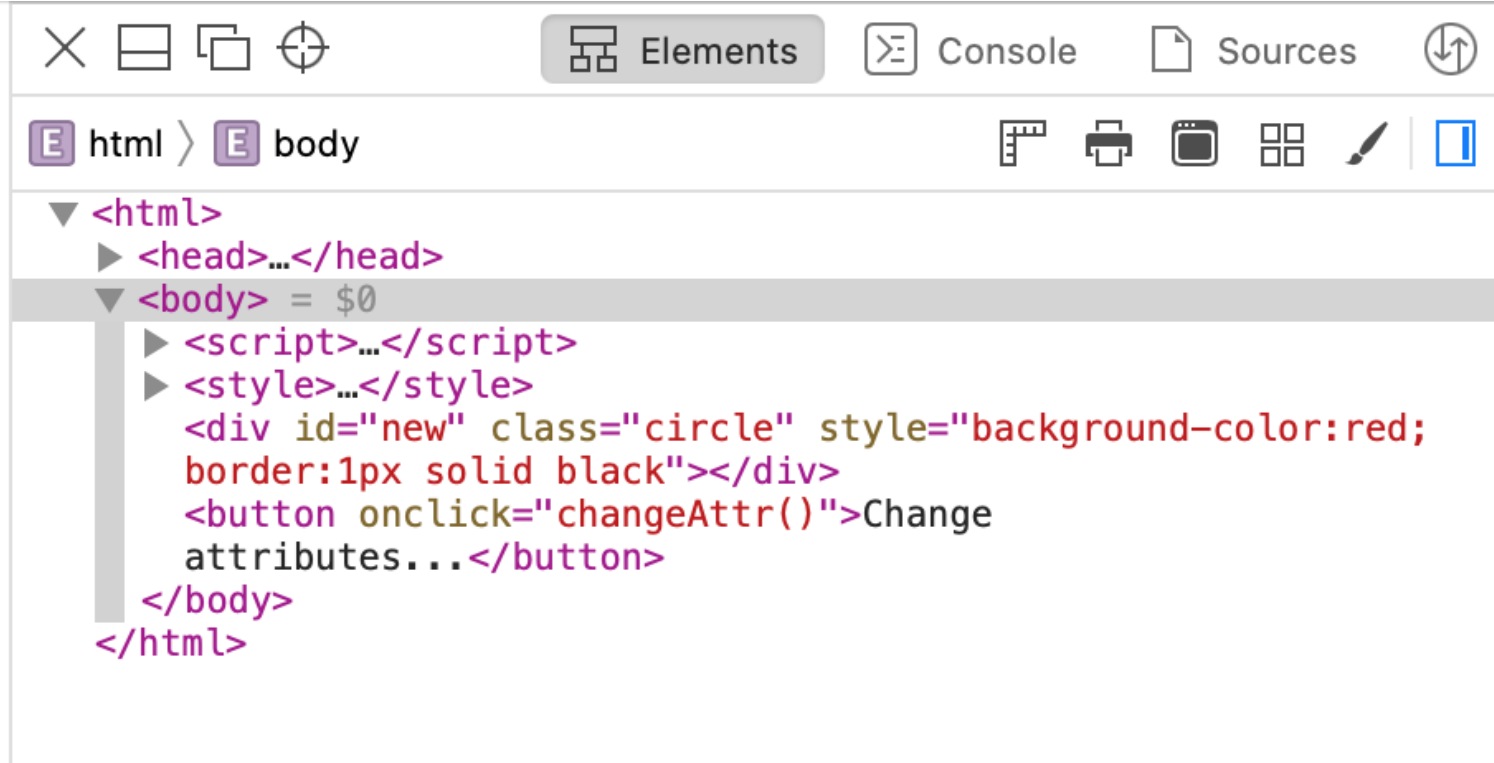
Change attributes...

```
✕ □ □ □
Elements Console Sources ↻
html > body
<html>
  <head>...</head>
  <body> = $0
    <script>
      function changeAttr(){
        let el = document.getElementById("shape");
        el.setAttribute("style", "background-
        color:red;border:1px solid black");
        el.setAttribute("id", "new");
        el.setAttribute("class", "circle");
      }
    </script>
    <style>
      div {
        height: 100px;
        width: 100px;
        background-color: yellow;
      }
      .circle {
        border-radius: 50%;
      }
    </style>
    <div id="shape" class="square"></div>
    <button onclick="changeAttr()">Change
    attributes...</button>
  </body>
</html>
```


THAO TÁC THUỘC TÍNH



Change attributes...



The screenshot shows a web browser's developer tools interface. The 'Elements' panel is active, displaying the DOM tree. The tree structure is as follows:

- `<html>`
 - `<head>...</head>`
 - `<body> = $0`
 - `<script>...</script>`
 - `<style>...</style>`
 - `<div id="new" class="circle" style="background-color:red; border:1px solid black"></div>`
 - `<button onclick="changeAttr()">Change attributes...</button>`

The code for the `<body>` element is expanded, showing the following HTML structure:

```
<html>  
  <head>...</head>  
  <body> = $0  
    <script>...</script>  
    <style>...</style>  
    <div id="new" class="circle" style="background-color:red;  
      border:1px solid black"></div>  
    <button onclick="changeAttr()">Change  
      attributes...</button>  
  </body>  
</html>
```



PHẦN II

INTERACTIVE CONTENT & EVENT LISTENERS

INTERACTIVE CONTENT

(Nội dung tương tác)

- Nhắc lại (bài 3): ***Một phần tử chỉ có 1 trình xử lý sự kiện (event handler) làm thuộc tính. Nếu phần tử có xử lý sự kiện onclick thì không thể có onmouseover***
=> Sử dụng event listeners có thể thêm nhiều sự kiện vào phần tử
- Quy trình gồm 2 bước:
 - Chọn element mà bạn muốn thêm 1 sự kiện (event)
 - Sử dụng `addEventListener("event", function)`

INTERACTIVE CONTENT

(Nội dung tương tác)

- Interactive content là nội dung phản hồi lại các hành động của người dùng
- Interactive content có thể thực hiện được bằng cách thay đổi DOM dựa vào tương tác người dùng. Ví dụ tương tác người dùng:
 - Nhập nội dung vào input field
 - Nhấp chuột vào bất kỳ đâu trên trang web
 - Di chuyển chuột qua phần tử
 - ...
- Chỉ định sự kiện
 - Chỉ định sự kiện bằng HTML
 - Chỉ định sự kiện bằng Javascript
 - Chỉ định sự kiện với event listeners

INTERACTIVE CONTENT

(Nội dung tương tác)

- Chỉ định sự kiện bằng HTML

```
<p id="unique" onclick="magic()">Click here for magic!</p>
```

- Chỉ định sự kiện bằng Javascript

```
document.getElementById("unique").onclick = function() {  
    magic();  
};
```

- Chỉ định sự kiện với event listeners

```
document.getElementById("unique").addEventListener("click",  
    magic);
```

INTERACTIVE CONTENT

(Nội dung tương tác)

- **Onload event** (sự kiện onload) sẽ được kích hoạt sau khi một phần tử nhất định được tải

- Cú pháp

```
window.onload = function() {  
    // whatever needs to happen after the page  
    loads goes here  
}
```

```
<body onload="function()">
```

- Load event sẽ kích hoạt khi kết thúc quá trình tải tài liệu (document)
=> Tất cả các object trong document đều trong DOM > có thể sử dụng `addEventListener()` trên bất kỳ element nào để xử lý bất kỳ sự kiện nào

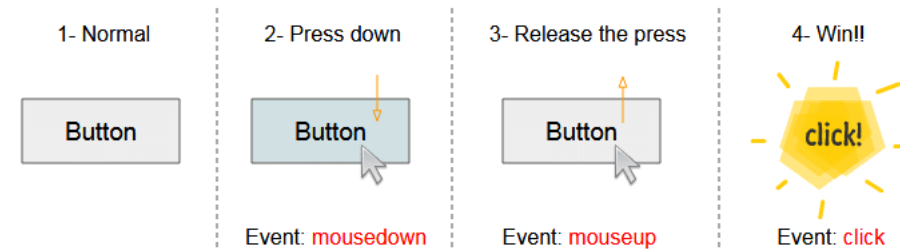
DEMO

```
<script>
  window.onload = function() {
    document.getElementById("square").addEventListener("click", changeColor);
    document.getElementById("square").addEventListener("click", unique);
  }
  function changeColor(){
    window.alert("Hello");
    let red = Math.floor(Math.random() * 256);
    let green = Math.floor(Math.random() * 256);
    let blue = Math.floor(Math.random() * 256);
    this.style.backgroundColor = `rgb(${red}, ${green}, ${blue})`;
  }
</script>
<body>
  <div class="shape" id="square" style="width:100px;height:100px;background-color: grey;">
    Click for magic
  </div>
  <p class="unique">Click here for magic!</p>
  <script>...
</script>
</body>
```

INTERACTIVE CONTENT

(Nội dung tương tác)

- Mouse event handlers



Sự kiện	Mô tả
onclick	Nhấp đôi chuột
onmousedown	Click chuột lên phần tử nhưng không thả/buông chuột (released)
onmouseup	Click chuột lên đầu phần tử
onmouseenter	Khi chuột di chuyển vào một phần tử
onmouseleave	Khi chuột di chuyển ra khỏi một phần tử và tất cả các phần tử con của nó

INTERACTIVE CONTENT

(Nội dung tương tác)

- **Mouse event handlers**

Sự kiện	Mô tả
onmousemove	Khi chuột di chuyển trên 1 phần tử
onmouseout	Khi chuột rời khỏi một phần tử (riêng lẻ)
onmouseover	Khi chuột di chuyển qua 1 phần tử

Tham khảo: Phân biệt onmousemove, onmouseout và onmouseover

<https://javascript.info/mousemove-mouseover-mouseout-mouseenter-mouseleave>

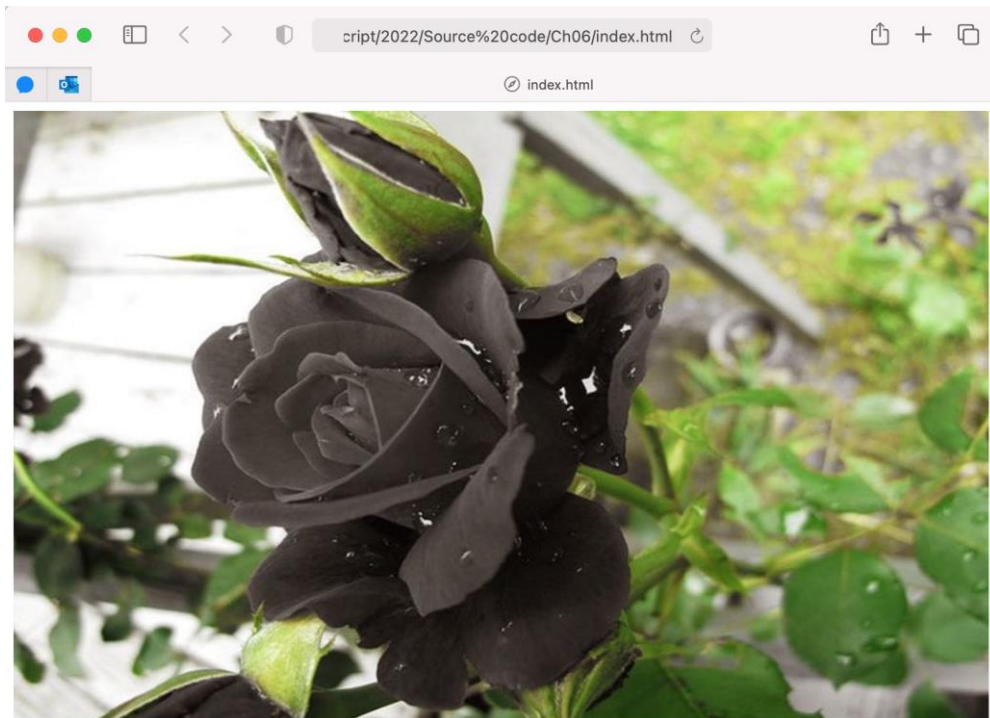
DEMO

- Mouse event handlers

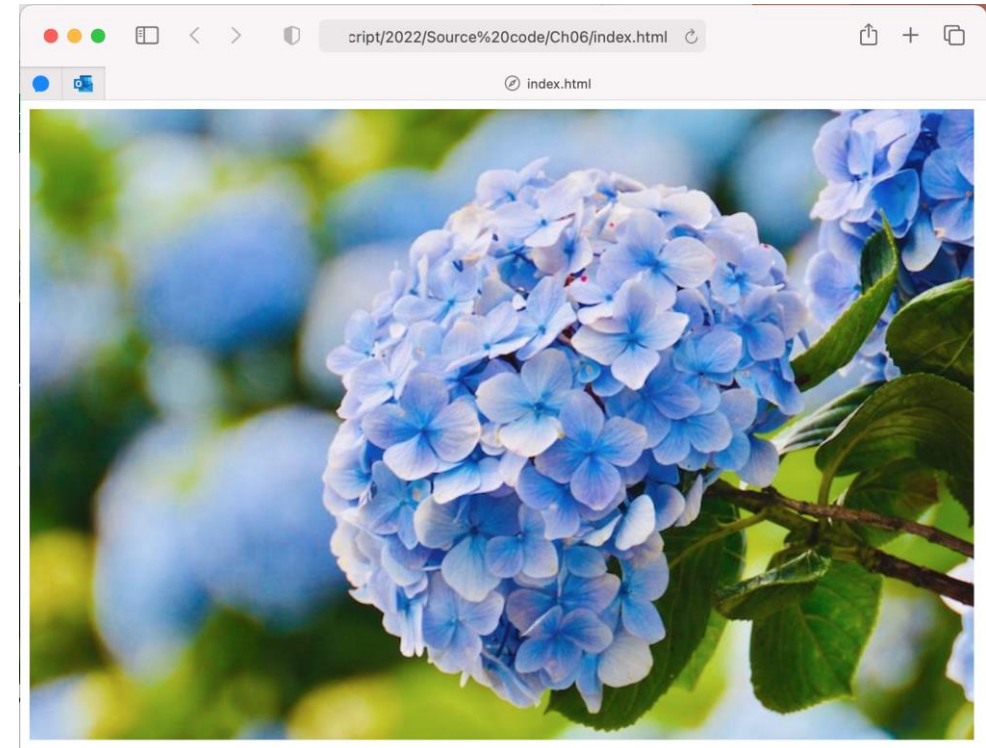
```
<html>
  <head>
    <title>Hi</title>
  </head>
  <script> ...
</script>
  <body>
    <div id="divvy" onmouseover="changeColor()"
      style="width: 100px; height: 100px; background-color: pink;">
    <script>
      function changeColor() {
        document.getElementById("divvy").style.backgroundColor = "blue";
      }
    </script>
  </body>
</html>
```

IMAGE ROLLOVER

- Image Rollover là hiệu ứng thay đổi ảnh khi di chuyển chuột lên ảnh



Ảnh hiện lên khi chưa
chạy ứng dụng



Ảnh hiện lên khi di/rê
chuột lên ảnh

TẠO HIỆU ỨNG ROLLOVER

- Thực hiện hiệu ứng này bằng cách xử lý sự kiện onMouseOver và onMouseOut cho thẻ img
 - onMouseOver: sự kiện được kích hoạt khi người dùng di/rê chuột lên ảnh
 - onMouseOut: sự kiện được kích hoạt khi người dùng di/rê chuột ra ngoài ảnh

```
<html>
  <head>
    <script type="text/javascript">
      function onMouseOverEvent() { document.img_hoa.src = "hoaover.jpg"; }
      function onMouseOutEvent() { document.img_hoa.src = "hoaout.jpg"; }
    </script>
  </head>
  <body>
    
  </body>
</html>
```

Truy cập đến phần tử
bằng attribute name

Vấn đề nảy sinh khi thực hiện Rollover

- Vấn đề: Lần đầu tiên di chuột lên ảnh, sẽ mất một thời gian ảnh mới được load mặc dù trang web đã được mở từ lâu
- Nguyên nhân: Khi người dùng di chuột lên, ảnh mới được load
- Giải pháp: load trước ảnh
 - Kỹ thuật:
 - Tạo đối tượng image cho mỗi ảnh muốn load trước
 - Gán đường dẫn của ảnh cho thuộc tính src của đối tượng ảnh đó
 - Giải thích:
 - Khi gặp lệnh `imageObject.src="link_cua_anh"` thì ảnh được load ngầm bên dưới, máy tính tiếp tục thực hiện các lệnh sau lệnh này
 - => Giải pháp này không làm cho việc load trang web chậm đi

DEMO LOAD TRƯỚC ẢNH

```
<html>
  <head>
    <script type="text/javascript">
      var hoaover = new Image();
      var hoaout = new Image();
      function loadAnh() {
        hoaover.src = "hoaover.jpg";
        hoaout.src = "hoaout.jpg";
      }
      function onMouseOverEvent() { document.img_hoa.src = hoaover.src; }
      function onMouseOutEvent() { document.img_hoa.src = hoaout.src; }
    </script>
  </head>
  <body onload="loadAnh();">
    
  </body>
</html>
```


SLIDESHOW

- Slide Show là gì
 - Là hiệu ứng ảnh được hiện ra thay thế cho ảnh trước đó
 - Có thể có thanh điều khiển cho người dùng



Thanh điều khiển

<< Previous Play > Stop Next >>

Các bước làm slideshow

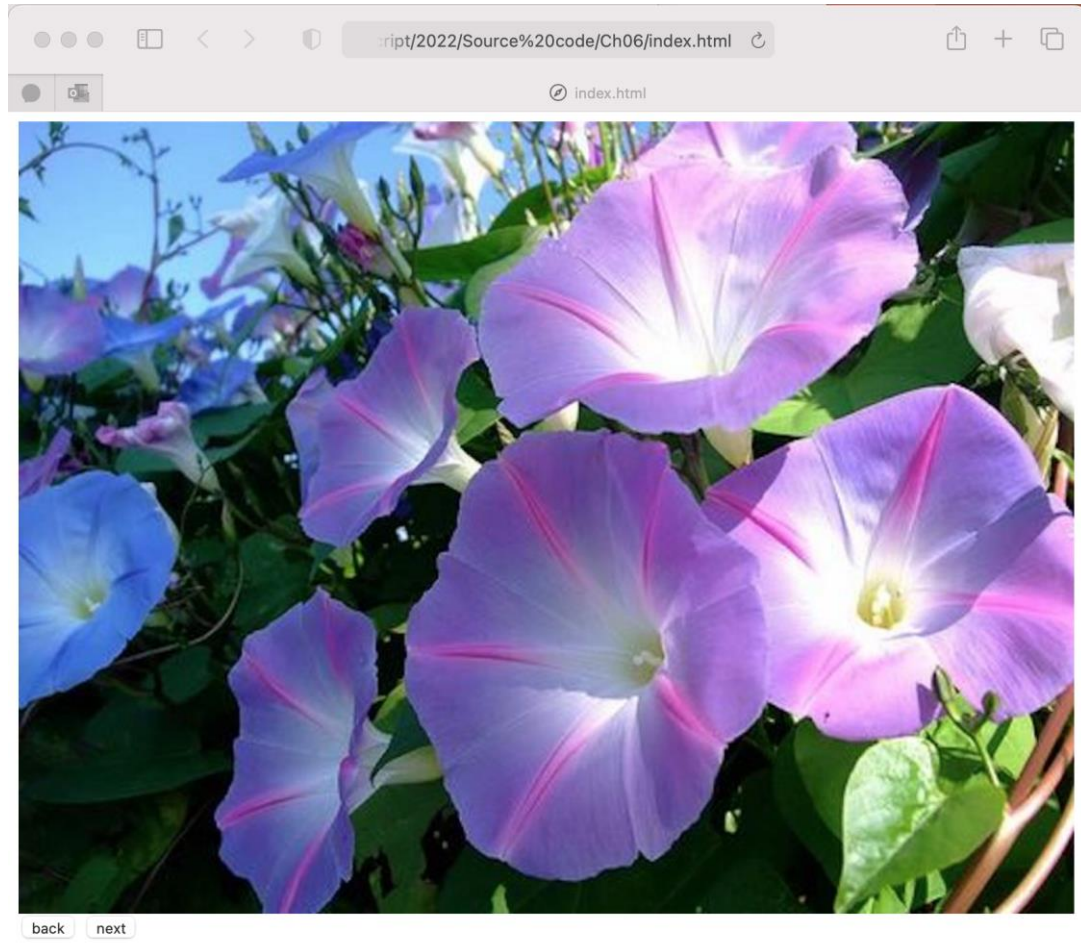
- Các bước làm slideshow
 - Load trước tất cả các ảnh
 - Xử lý sự kiện cho button Next
 - Xử lý sự kiện cho button Back

```
<body onload = "loadAnh();">  
      
    <input type="button" value="back" onclick="back();" />  
    <input type="button" value = "next" onclick="next();"/>  
</body>
```

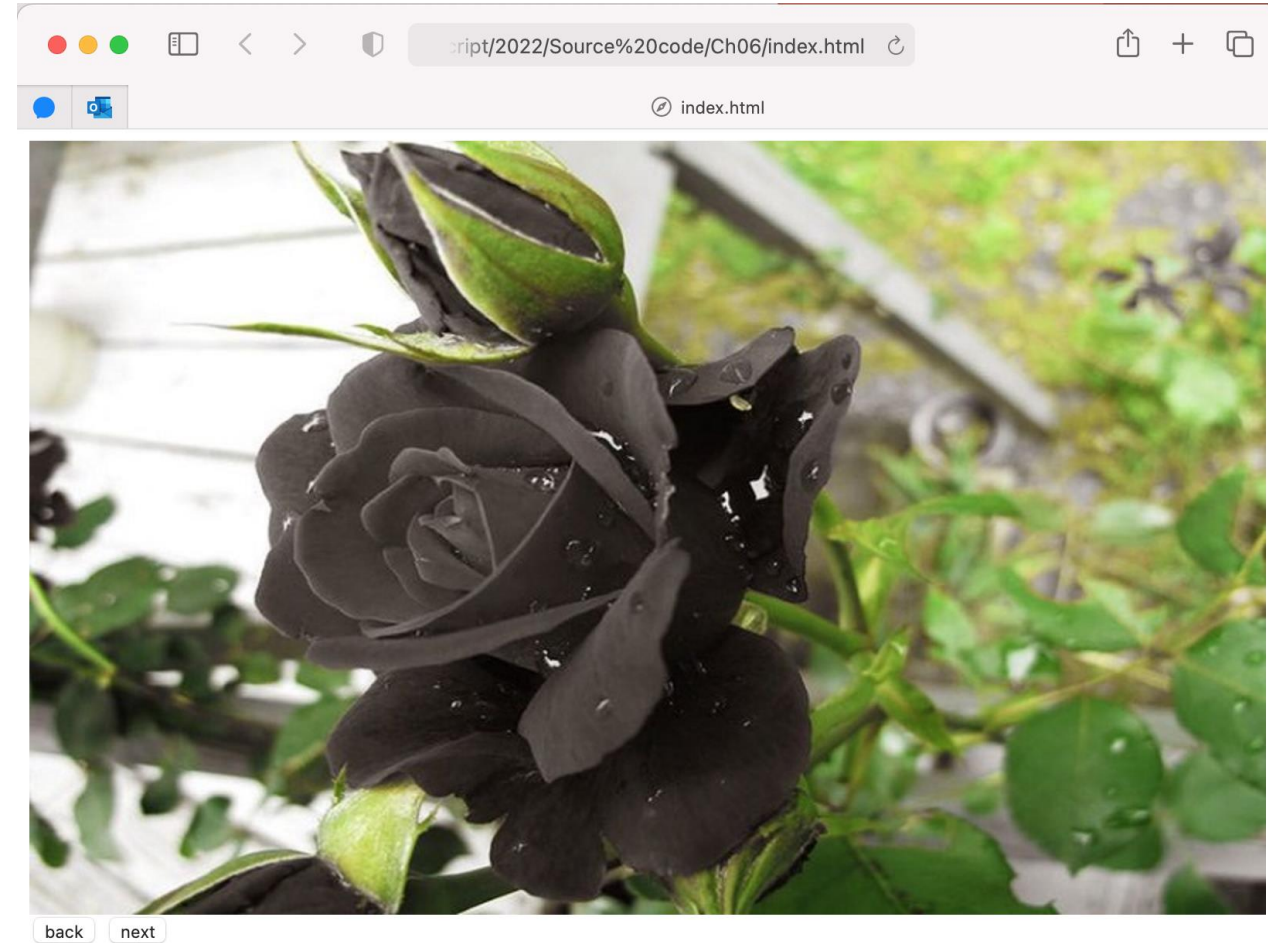

Demo làm slideshow

```
var anhAr = [];  
var currentIndex=0;  
function loadAnh() {  
    for (var i = 0; i < 6; i++) {  
        anhAr[i] = new Image();  
        anhAr[i].src = "anh"+i+".jpg";  
    }  
}  
function next() {  
    if (currentIndex < 4) {  
        currentIndex++;  
        document.getElementById("hoa").src = anhAr[currentIndex].src;  
    }  
}  
function back() {  
    if (currentIndex > 0) {  
        currentIndex--;  
        document.getElementById("hoa").src = anhAr[currentIndex].src;  
    }  
}
```

Kết quả demo



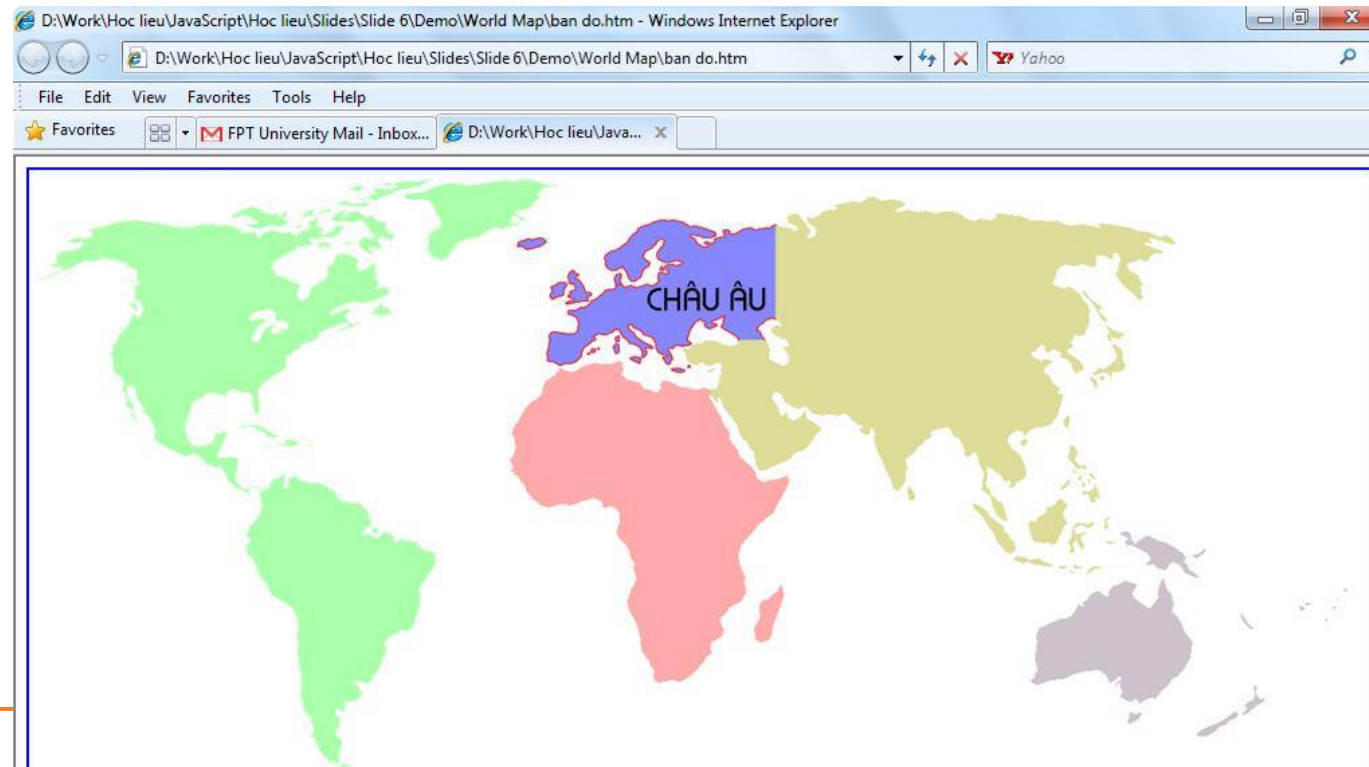
Ảnh mới load



Ảnh được thay đổi sau khi nhấn next

BẢN ĐỒ ẢNH

- Bản đồ ảnh là ảnh có những vùng cụ thể được định nghĩa để thực hiện một hành động nào đó khi người dùng tác động lên
 - Cho người dùng chọn khu vực (đất nước) mà người truy cập cư trú
 - Sử dụng như menu cho người dùng chọn mặt hàng cần mua



BẢN ĐỒ ẢNH

- HTML cung cấp thẻ map để tạo bản đồ ảnh
- Thẻ map đi liền sau thẻ img, thuộc tính usemap của thẻ img có giá trị bằng giá trị thuộc tính name của thẻ map tương ứng
- Thẻ area trong thẻ map để chỉ ra các vùng trên bản đồ. Vùng trên bản đồ được chỉ ra bằng thuộc tính coords

Thuộc tính usemap của ảnh có giá trị bằng giá trị của thuộc tính name của thẻ map

```
<p></p>
<map name="bando">
  <area shape="poly" coords="" href="#" onmouseover="document.getElementById('bando').src='bando.chaumy.jpg'"/>
  <area shape="poly" coords="" href="#" onmouseover="document.getElementById('bando').src='bando.chauphi.jpg'"/>
  <area shape="poly" coords="" href="#" onmouseover="document.getElementById('bando').src='bando.chauau.jpg'"/>
  <area shape="poly" coords="" href="#" onmouseover="document.getElementById('bando').src='bando.chaua.jpg';"/>
  <area shape="poly" coords="" href="#" onmouseover="document.getElementById('bando').src='bando.chauuc.jpg';"/>
</map>
```

Tổng kết bài học

- Javascript cung cấp đối tượng style trong mỗi element để thay đổi style cho các element.
- Javascript cung cấp cách để chỉ định sự kiện cho element
 - Chỉ định sự kiện bằng HTML
 - Chỉ định sự kiện bằng Javascript
 - Chỉ định sự kiện với event listeners
- Mouse event handlers gồm:
 - Onbclick, onmousedown, onmouseup, onmouseenter, onmouseleave
 - Onmousemove, onmouseout, onmouseover
- Có thể dùng javascript kết hợp mouse event để tạo hiệu ứng rollover, slideshow

*Thank
you!*