FPT Education

FPT POLYTECHNIC

cdio™

Conceive Design Implement Operate

JAVASCRIPT NÂNG CAO

Thư viện Jquery (Phần 2)

THỰC HỌC – THỰC NGHIỆP

http://www.poly.edu.vn

Mục tiêu

- Giải thích và sử dụng các api Traversing của jQuery
- Sử dụng được jQuery Ajax để gửi request
- Sử dụng được các thư viện trên nền tảng jQuery

NỘI DUNG

- Các API xác định DOM (DOM Traversing) của jQuery
  - jQuery Ancestors
  - jQuery Descendants
  - jQuery Siblings
  - jQuery Filtering
- Sử dụng jQuery ajax
  - $.ajax()
  - $.post()
  - $.get()
- Sử dụng thư viện trên nền tảng jQuery

NHẮC LẠI BÀI CŨ

- 📖Giải thích được jQuery là gì?
- 📖Cài đặt thư viện jQuery.
- 📖Giải thích và sử dụng được các Selector trong jQuery.
- 📖Sử dụng các api sự kiện trong jQuery
- 📖Sử dụng được các api attributes trong jQuery

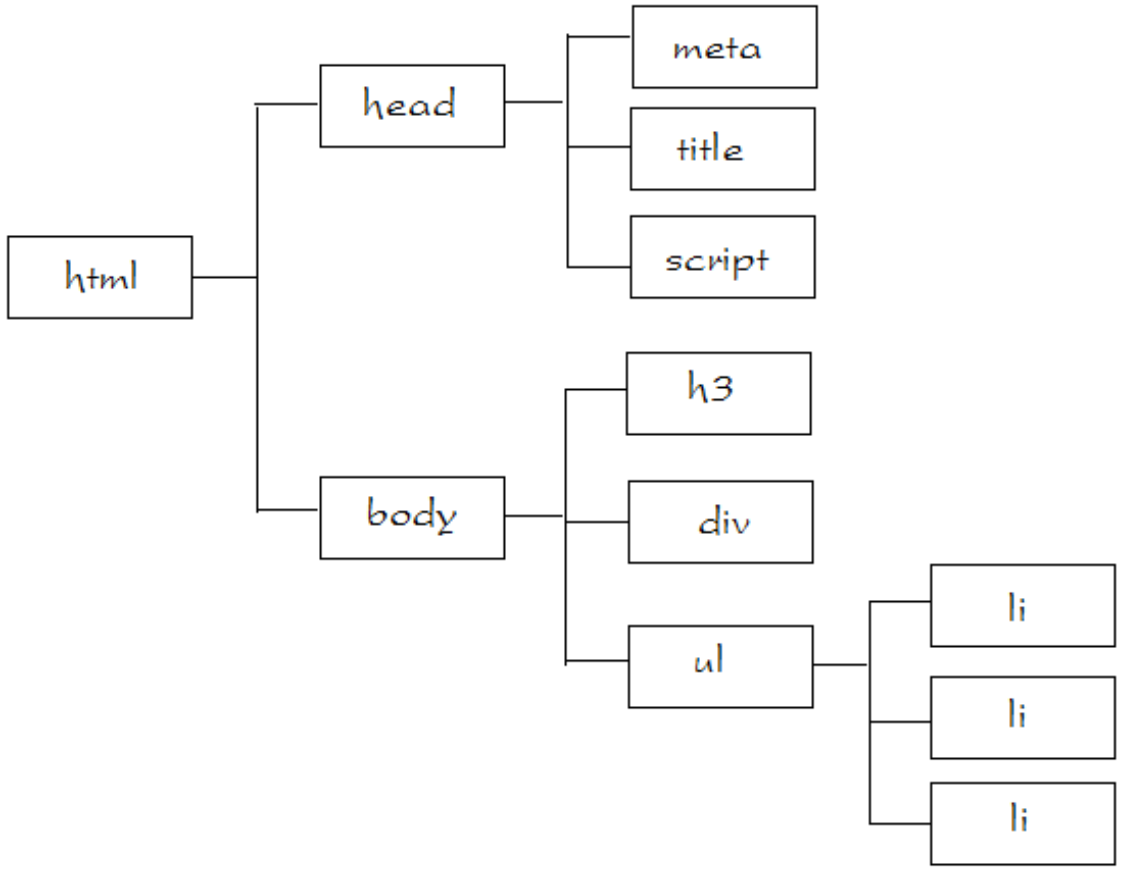
PHẦN 1:
JQUERY DOM TRAVERSING

- **jQuery DOM Traversing**: được sử dụng để tìm kiếm (hoặc lựa chọn) các phần tử HTML dựa trên quan hệ của nó với các phần tử khác. Bắt đầu tại các vị trí (các phần tử) được chọn và di chuyển cho tới khi bắt gặp các phần tử mà ta mong muốn sử dụng.

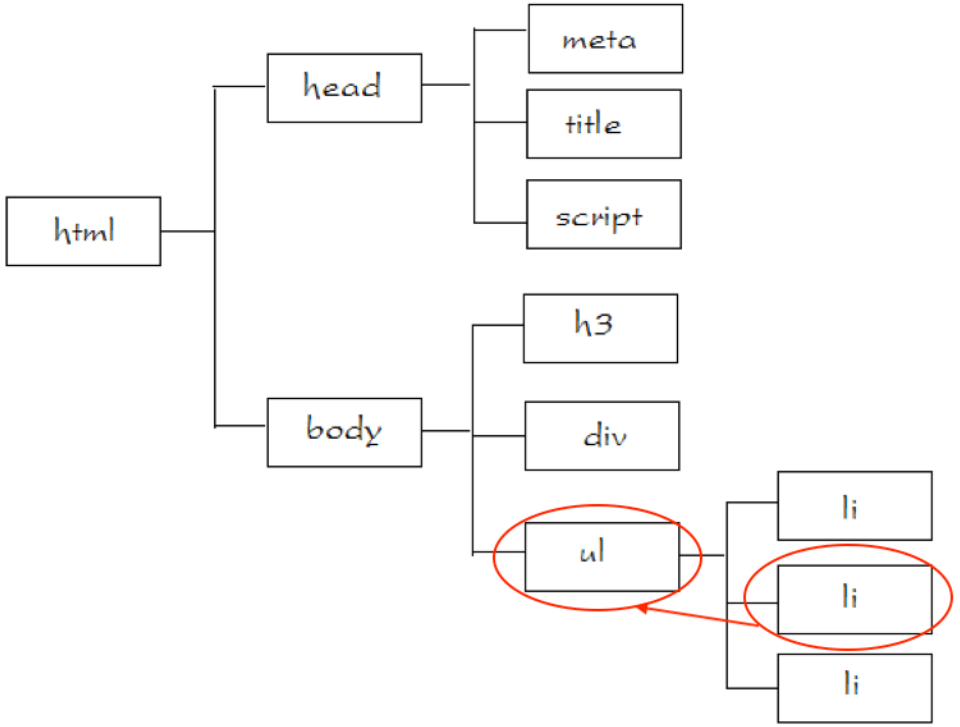
Ví dụ 4: cấu trúc dạng cây của 1 trang html

```
<html>
<head>
    <meta charset="utf-8">
    <title>jQuery Traversing</title>

    <script type="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

</head>
<body>

    <h3>jQuery Traversing</h3>

    <div>

        <ul>
            <li>Java</li>
            <li>.Net</li>
            <li>PHP</li>
        </ul>

    </div>

</body>
</html>
```

- Ví dụ 4: cấu trúc dạng cây của 1 trang html



- Trong trường hợp chúng ta đang sử dụng selector để nhận sự kiện từ thẻ li bất kỳ nhưng lại muốn tác động đến thẻ bên cạnh nó hay các thẻ cha, ông,... của nó thì ta sẽ cần phải sử dụng đến jQuery Traversing để xác định được các thẻ này.



JQUERY TRAVERSING – ANCESTORS (PHẦN TỬ TỔ TIÊN)

- Một phần tử tổ tiên (ancestor element) là phần tử cha, ông, v..v.. của một phần tử khác.
- Sử dụng **jQuery** bạn có thể di chuyển lên (traverse up) các nút cao hơn của cây **DOM** để tìm kiếm các phần tử **ancestors** của một phần tử đang được xảy ra sự kiện hoặc được chọn bởi selector.

jQuery Traversing – Ancestors (phần tử tổ tiên)

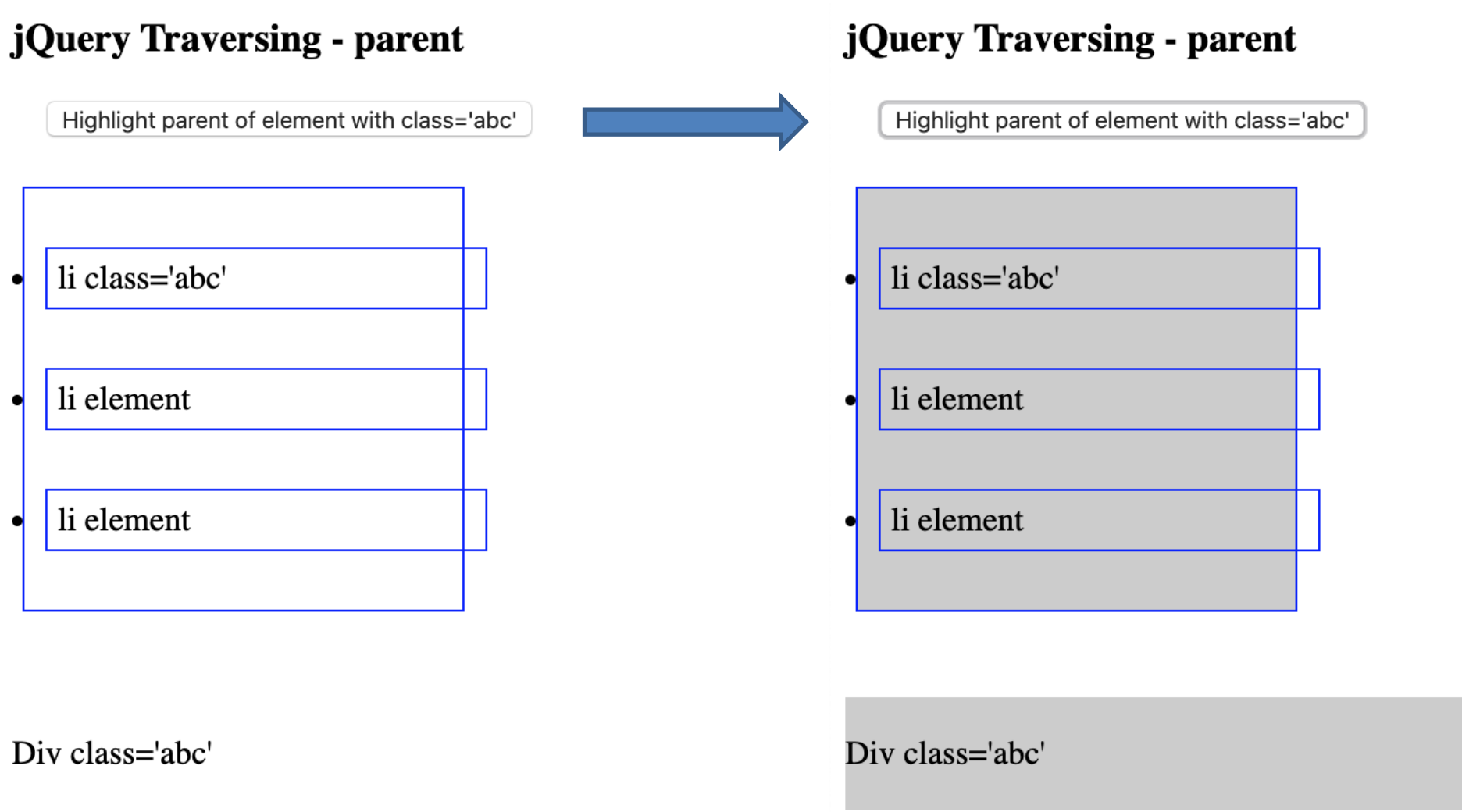
- **.parent()**: trả về phần tử cha trực tiếp của phần tử đang chọn.
- Ví dụ: tìm tất cả các phần tử cha của phần tử có ***class='abc'***.

```
<h3>jQuery Traversing - parent </h3>
<button type="button" onclick="highlightParent();">
    Highlight parent of element with class='abc'
</button>

<div>

    <ul>
        <li class='abc'>li class='abc' </li>
        <li>li element</li>
        <li>li element</li>
    </ul>

</div>

<div>
    <div class='abc'>Div class='abc'</div>
</div>
```

```
<style>
    div, ul, li {
        width: 200px;
        padding: 5px;
        margin: 5px;
        border: 1px solid blue;
    }
    .abc {
        border: 1px solid red;
    }
</style>
```

```
<script>

    function highlightParent()  {

        $('.abc').parent().css("background","#ccc");
    }
</script>
```

JQUERY TRAVERSING – ANCESTORS (PHẦN TỬ TỔ TIÊN)



JQUERY TRAVERSING – ANCESTORS (PHẦN TỬ TỔ TIÊN)

- **.parents(selector)**: trả về các phần tử tổ tiên (ancestor) của phần tử đang được chọn mà **thỏa mãn giá trị selector**, phương thức này sẽ quét qua tất cả các phần tử tổ tiên của nó cho tới thẻ gốc (<html>).
- Ví dụ tìm kiếm tất cả các phần tử ** ** là tổ tiên của phần tử có ***class= 'abc'***.

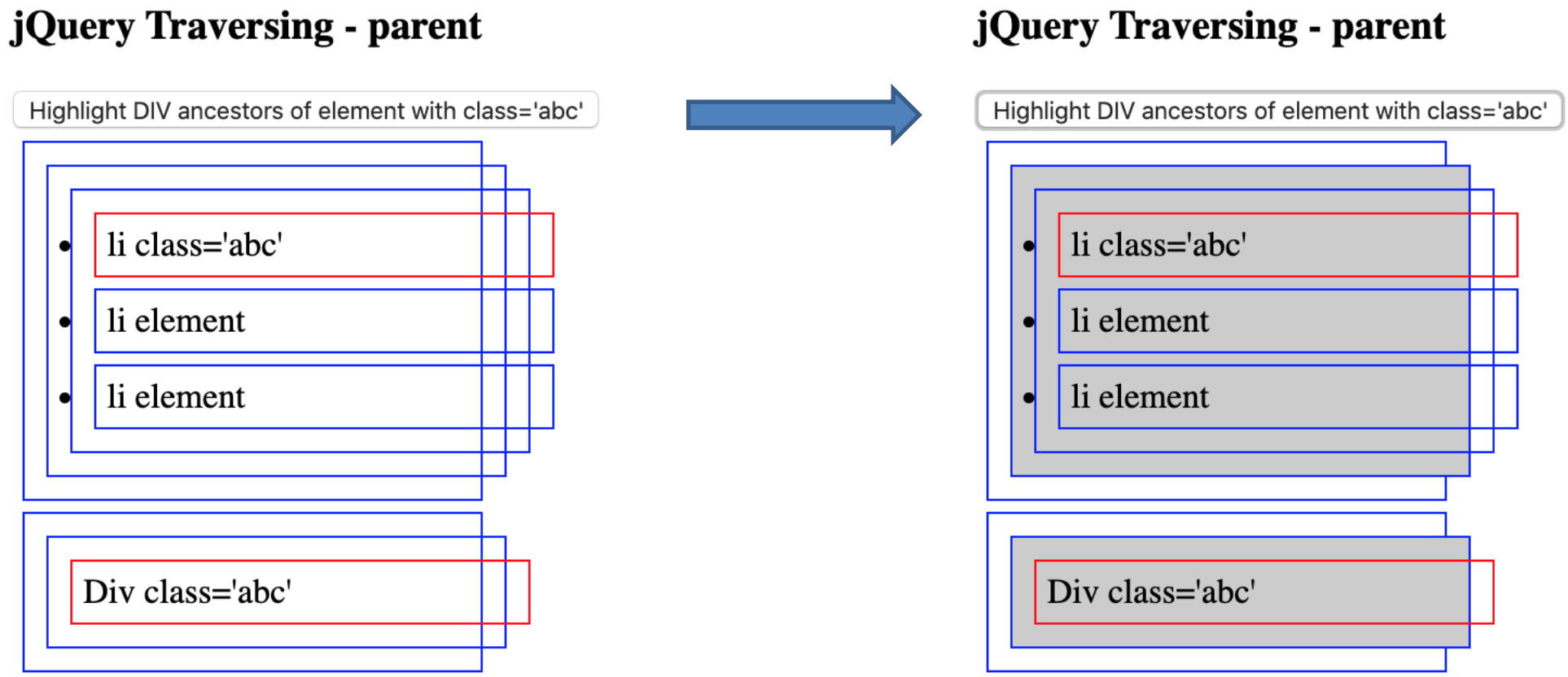
jQuery Traversing – Ancestors (phần tử tổ tiên)

```
<h3>jQuery Traversing - parent </h3>
<button type="button" onclick="highlightParent();">
    Highlight DIV ancestors of element with class='abc'
</button>
<div class="first">
    <div class="second">
        <ul>
            <li class='abc'>li class='abc' </li>

            <li>li element</li>

            <li>li element</li>
        </ul>
    </div>
</div>
<div class="first">
    <div class="second">

        <div class='abc'>Div class='abc'</div>
    </div>
</div>
```

```
<style>
    div,
    ul,
    li {
        width: 200px;
        padding: 5px;
        margin: 5px;
        border: 1px solid blue;
    }

    .abc {
        border: 1px solid red;
    }
</style>
```

```
<script>
    function highlightParent()  {
        $('.abc').parents('div.second').css("background", "#ccc");
    }
</script>
```

jQuery Traversing – Ancestors (phần tử tổ tiên)



JQUERY TRAVERSING – ANCESTORS (PHẦN TỬ TỔ TIÊN)

- **parentsUntil():** trả về tất cả các phần tử tổ tiên nằm giữa 2 phần tử gồm phần tử đang chọn và phần tử xác định trong tham số
- Ví dụ tìm các element tổ tiên của **<h2>** cho tới khi bắt gặp phần tử ** **:

jQuery Traversing – Ancestors (phần tử tổ tiên)

```
<style>
    div,
    ul,
    li {
        width: 200px;
        padding: 5px;
        margin: 5px;
        border: 1px solid blue;
    }

    .abc {
        border: 1px solid red;
    }
</style>
```

```
<h3>jQuery Traversing - parentsUtil</h3>
<p>$('h2').parentsUntil('div').css("border","2px solid red");</p>

<button type="button" onclick="highlightParentsUntil();">
    Highlight ancestors elements between H2 and DIV
</button>

<div> div element
    <ul> ul element

        <li>
            li element
            <h2>h2 element</h2>
        </li>
        <li>li element</li>
    </ul>
</div>
```

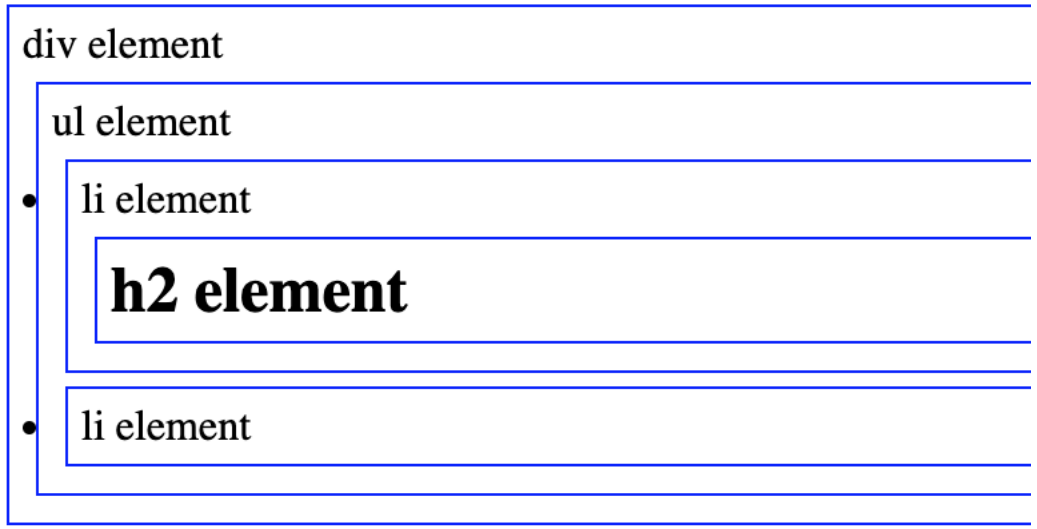
```
<script>
    function highlightParentsUntil()  {
        $('h2').parentsUntil('div').css("border", "2px solid red");
    }
</script>
```

jQuery Traversing – Ancestors (phần tử tổ tiên)

jQuery Traversing - parentsUtil

$('h2').parentsUntil('div').css("border","2px solid red");

Highlight ancestors elements between H2 and DIV



jQuery Traversing - parentsUtil

$('h2').parentsUntil('div').css("border","2px solid red");

Highlight ancestors elements between H2 and DIV

jQuery Traversing – Descendants (phần tử hậu duệ)

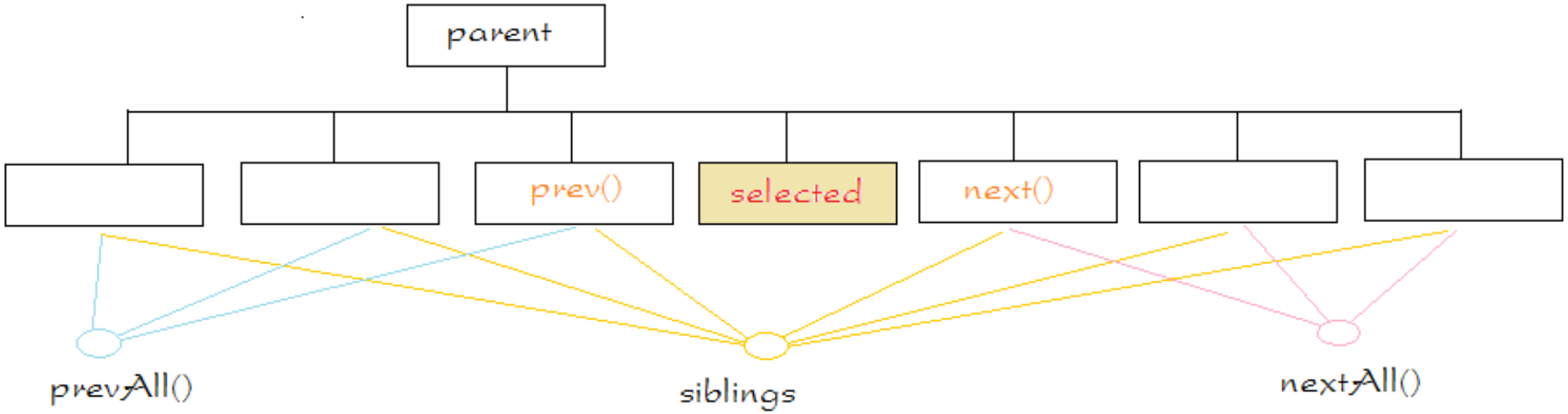
- Ngược lại với Ancestors, **jQuery Descendants** giúp chung ta có thể tìm các thẻ hậu duệ của thẻ được chọn. Chính là các thẻ con, cháu, chắt,...
- Có 2 phương thức của **jQuery Descendants:**
  - **.children([selector])**: trả về tất cả các phần tử con trực tiếp (giới hạn trong 1 cấp cha-con | không tìm kiếm ở cấp cháu) của phần tử đang chọn. Trường hợp có tham số là selector thì sẽ trả về các thành phần con thỏa mãn selector
  - **.find([selector])**: trả về tất cả các phần tử con (không giới hạn về cấp, miễn là nằm trong) của phần tử đang chọn. Trường hợp có tham số là selector thì sẽ trả về các thành phần con thỏa mãn selector

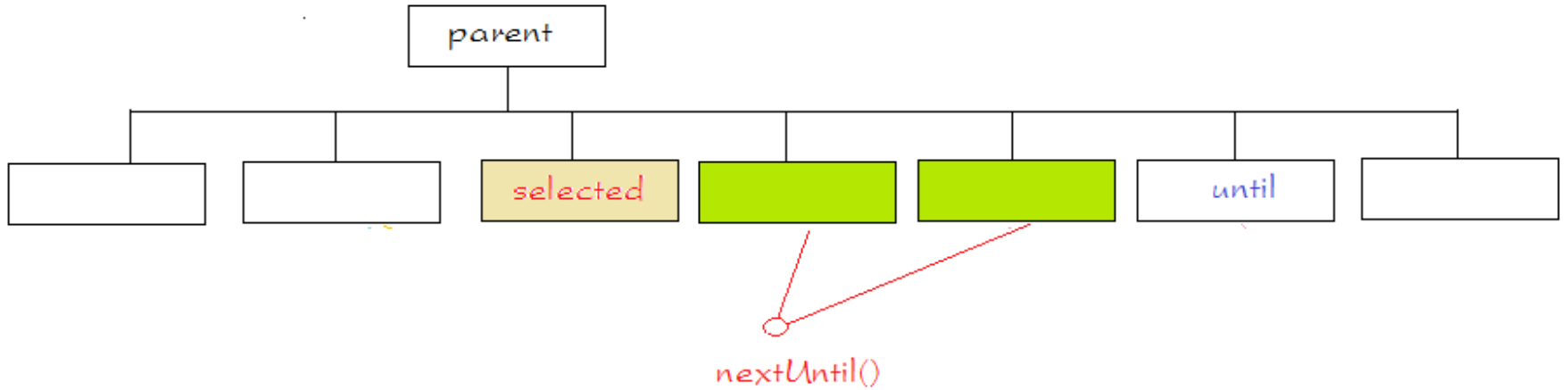
jQuery Traversing – Siblings (phần tử anh em)

- Tìm kiếm các phần tử anh em, cùng phần tử cha với phần tử hiện tại.
- Các phương thức thuộc loại siblings:
  - siblings()
  - next()
  - nextAll()
  - nextUntil()
  - prev()
  - prevAll()
  - prevUntil()

jQuery Traversing – Siblings (phần tử anh em)

- Minh họa:





jQuery Traversing – Siblings (phần tử anh em)

Giải thích ý nghĩa của các phương thức siblings

Phương thức	Mô tả	Ví dụ
siblings([selector])	Phương thứcsiblings()trả về tất cả các phần tử anh em của phần tử được chọn.	$('h2').siblings()
next([selector])	Phương thứcnext()trả về phần tử anh em kế tiếp của phần tử được chọn.	$('h2').next()
prev([selector])	Phương thức prev() trả về phần tử anh em ngay trước phần tử được chọn.	$('h2').prev()
nextAll([selector])	Phương thứcnextAll()trả về tất cả các phần tử anh em kế tiếp của phần tử được chọn.	$('h2').nextAll()
prevAll([selector])	Phương thứcprevAll()trả về tất cả cac phần tử anh em đứng trước phần tử được chọn.	$('h2').prevAll()
nextUntil(selector)	Phương thứcnextUntil()trả về các phần tử kế tiếp phần tử được chọn, cho tới khi gặp một phần tử phù hợp điều kiện của tham số.	$('h2').nextUntil('h4.blue')
prevUntil(selector)	Phương thứcprevUntil()trả về các phần tử phía trước của phần tử đang được chọn, cho tới khi gặp một phần tử phù hợp điều kiện của tham số.	$('h2').prevUntil('h1')

Các phương thức tìm kiếm chung trong jQuery

Phương thức	Mô tả	Ví dụ:
eq(index)	Phương thức eq() trả về một phần tử ứng với chỉ số trong tham số trong tập các phần tử được chọn.	$('p').eq(1)
filter(selector)	Phương thức filter() cần bạn chỉ rõ tiêu chí trong tham số. Các phần tử không khớp với tiêu chí sẽ bị loại bỏ khỏi tập hợp đang chọn, và trả về các phần tử phù hợp.	$('p').filter('.abc')
filter(fn)	Loại bỏ tất cả các phần tử trong tập hợp được chọn mà không phù hợp với hàm được chỉ định (Trong tham số), trả về các phần tử còn lại.	
first()	Phương thức first() trả về phần tử đầu tiên trong danh sách được lựa chọn.	$('p').first()
has()	Xây dựng một đối tượng jQuery mới từ một tập hợp con của các thành phần phù hợp.	$('li').has('p');
is(selector)	Kiểm tra các phần tử đang được lựa chọn có phù hợp với một biểu thức trong tham số không, nếu ít nhất một phần tử phù hợp phương thức trả về true.	$('.test').is(':first-child');
last()	Phương thức last() trả về phần tử cuối cùng trong danh sách được lựa chọn.	$('p').last()
map(callback)	Chuyển tập hợp các phần tử trong một đối tượng jQuery thành một tập hợp mảng (có thể là không chứa phần tử nào).	$('input').map(function(){return this id});
not(selector)	Phương thức not() trả về tất cả các phần tử không khớp với tiêu chí trong tham số.	$('input[type="checkbox"]').not(":checked");
slice(start,[end])	Trả về một tập con của tập hợp đang chọn.	$('p').slice(1, 4) $('p').slice(2)

PHẦN 2:
JQUERY CÁC PHƯƠNG THỨC CỦA SỰ KIỆN

CÁC PHƯƠNG THỨC CỦA SỰ KIỆN

- Trong jQuery ngoài việc cung cấp các phương thức sự kiện còn hỗ trợ các phương thức tác động đến việc trình duyệt có quyết định thực thi các sự kiện đó hay không (ngăn chặn hay cho phép).

Phương thức	Mô tả
preventDefault()	Ngăn chặn trình duyệt thực hiện các hành động mặc định.
isDefaultPrevented()	Trả về true nếu đã gọi event.preventDefault().
stopPropagation()	Ngăn chặn sự lan truyền sự kiện này tới các phần tử cha, các phần tử cha sẽ không nhận biết được sự kiện này, hoặc kích hoạt các sự kiện của nó.
isPropagationStopped()	Trả về true nếu event.stopPropagation() đã từng được gọi trong sự kiện này.
stopImmediatePropagation(	Ngăn chặn các phần còn lại của các bộ xử lý đang được thực thi.
isImmediatePropagationSto	Trả về true nếu event.stopImmediatePropagation() đã từng được gọi trong sự kiện này.

CÁC PHƯƠNG THỨC CỦA SỰ KIỆN

- Ví dụ khi bạn nhấn chuột vào 1 thẻ anchor (thẻ <a>) thì trình duyệt sẽ tự động reload lại theo đường dẫn mới. Bạn có thể sử dụng ***preventDefault()*** để ngăn chặn hành động mặc định này của trình duyệt.

```
<a href="http://google.com" title="">Click link (không hoạt động)</a>
<script>
    $("a").click(function(event) {
        event.preventDefault();
        alert("a element not working!!");
    });
</script>
```

PHẦN 3:
JQUERY AJAX

- jQuery cũng có phương thức hỗ trợ gửi http request từ client thông qua phương thức **.ajax()** giống như hàm **fetch()** trong javascript thuần đã học ở những bài trước.
- Bài toán bất đồng bộ được **.ajax()** giải quyết bằng cách sử dụng các hàm callback được jQuery xây dựng trong phương thức.

- Ví dụ: gửi request lên https://reqres.in/api/product để lấy dữ liệu danh sách sản phẩm

```
<script
    src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script>

    $(document).ready(function(){
        $.ajax({
            url: "https://reqres.in/api/product",
            method: "get",
            responseType: "json",
            success: function(data){
                console.log(data);
            }
        })
    });

</script>
```

Dữ liệu trả về


```
Object
  data: Array(6)
    0: {id: 1, name: "cerulean", year: 2000, color: "#98B2D1", pantone_value: "15-4020"}
    1: {id: 2, name: "fuchsia rose", year: 2001, color: "#C74375", pantone_value: "17-2031"}
    2: {id: 3, name: "true red", year: 2002, color: "#BF1932", pantone_value: "19-1664"}
    3: {id: 4, name: "aqua sky", year: 2003, color: "#7BC4C4", pantone_value: "14-4811"}
    4: {id: 5, name: "tigerlily", year: 2004, color: "#E2583E", pantone_value: "17-1456"}
    5: {id: 6, name: "blue turquoise", year: 2005, color: "#53B0AE", pantone_value: "15-5217"}
    length: 6
    __proto__: Array(0)
  page: 1
  per_page: 6
  total: 12
  total_pages: 2
  __proto__: Object
```

JQUERY AJAX – CÚ PHÁP – THIẾT LẬP THUỘC TÍNH

- Cú pháp: **$.ajax({object})**
- Các thuộc tính của **object**:
  - url: đường dẫn sẽ nhận request từ ajax
  - method: phương thức gửi dữ liệu (GET|POST|PUT|DELETE|...)
  - data: JSON object – dữ liệu gửi lên server
  - dataType: ép kiểu dữ liệu trả về
    - "xml"
    - "html"
    - "script"
    - "json"
    - "text"

- Các thuộc tính của **object**:
  - **headers**: new Header({object config}) – thiết lập cấu hình cho header của request
  - **beforeSend**: callback – thực thi 1 hàm trước khi gửi request (ví dụ enable phần loading của trang để ngăn người dùng tương tác)
  - **success**: callback – phương thức nhận dữ liệu từ server trả về, tham số trong hàm callback là dữ liệu nhận được từ server sau khi đã ép kiểu từ thuộc tính **datatype**
  - **error**: callback - Một hàm sẽ được gọi khi request fails.
  - **complete**: Một hàm được thực thi khi request kết thúc (sau khi hàm gọi lại success và error được thực thi).
  - **async**: Thiết lập giá trị false để thực hiện một request đồng bộ.
  - **accepts**: Nội dung được gửi trong request header giúp server biết được kiểu response server sẽ chấp nhận khi trả về.
  - **...**

jQuery Ajax – phương thức GET và POST

- **$.post()** có tác dụng lấy dữ liệu từ server bằng phương thức **HTTP POST REQUEST**, vì thế chúng ta có thể sử dụng cách này hoặc cách $.ajax() với method POST để viết Ajax.
- **Cú pháp**: $.post(url [, data] [, success] [, dataType]);

jQuery Ajax – phương thức GET và POST

- **$.get()** có tác dụng lấy dữ liệu từ server bằng phương thức **HTTP GET REQUEST**, vì thế chúng ta có thể sử dụng cách này hoặc cách $.ajax() với method GET để viết Ajax.
- **Cú pháp**: $.get(url [, data] [, success] [, dataType]);

jQuery Ajax – phương thức get và post

- Ví dụ: sử dụng $.post() để tạo mới sản phẩm với https://reqres.in/api/product

```
<script
    src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script>

    $(document).ready(function(){
        $.post({
            url: "https://reqres.in/api/product",
            method: "post",
            data: {
                name: "máy tính asus ROG 205",
                year: "2019",
                color: "grey",
                pantone_value: "23-6220"
            },
            responseType: "json",
            success: function(data){
                console.log(data);
            }
        })
    });

</script>
```

- Ví dụ: sử dụng $.post() để tạo mới sản phẩm với https://reqres.in/api/product - Kết quả trả về 1 sản phẩm mới với 1 id được sinh tự động


```
{name: "máy tính asus ROG 205", year: "2019", color: "grey", pantone_value: "23-6220", id: "849", …}
  color: "grey"
  createdAt: "2019-11-27T08:08:17.148Z"
  id: "849"
  name: "máy tính asus ROG 205"
  pantone_value: "23-6220"
  year: "2019"
  __proto__: Object
```

jQuery Ajax – Tổng kết

- Với jQuery ajax ($.ajax(), $.get(), $.post()) chúng ta có 3 phương án để gửi request từ client lên server:
  - fetch(): sử dụng với javascript thuần – không cần cài thêm thư viện
  - jQuery ajax – cần cài đặt thư viện jQuery – cú pháp ngắn gọn, sử dụng callback để giải quyết bất đồng bộ
  - Thư viện axios – cần cài đặt thêm thư viện – cú pháp ngắn gọn, thuận tiện tích hợp cho các dự án sử dụng frontend framework – sử dụng được cả cho trình duyệt và nodejs
- Tất cả các phương án đều có kết quả như nhau, tuy nhiên tùy theo trường hợp và các tính huống cụ thể của dự án chúng ta sẽ có thể lựa chọn cách tốt nhất để sử dụng.

PHẦN 4:
CÁC THƯ VIỆN PHÁT TRIỂN TRÊN NỀN JQUERY

CÁC THƯ VIỆN PHÁT TRIỂN TRÊN NỀN JQUERY

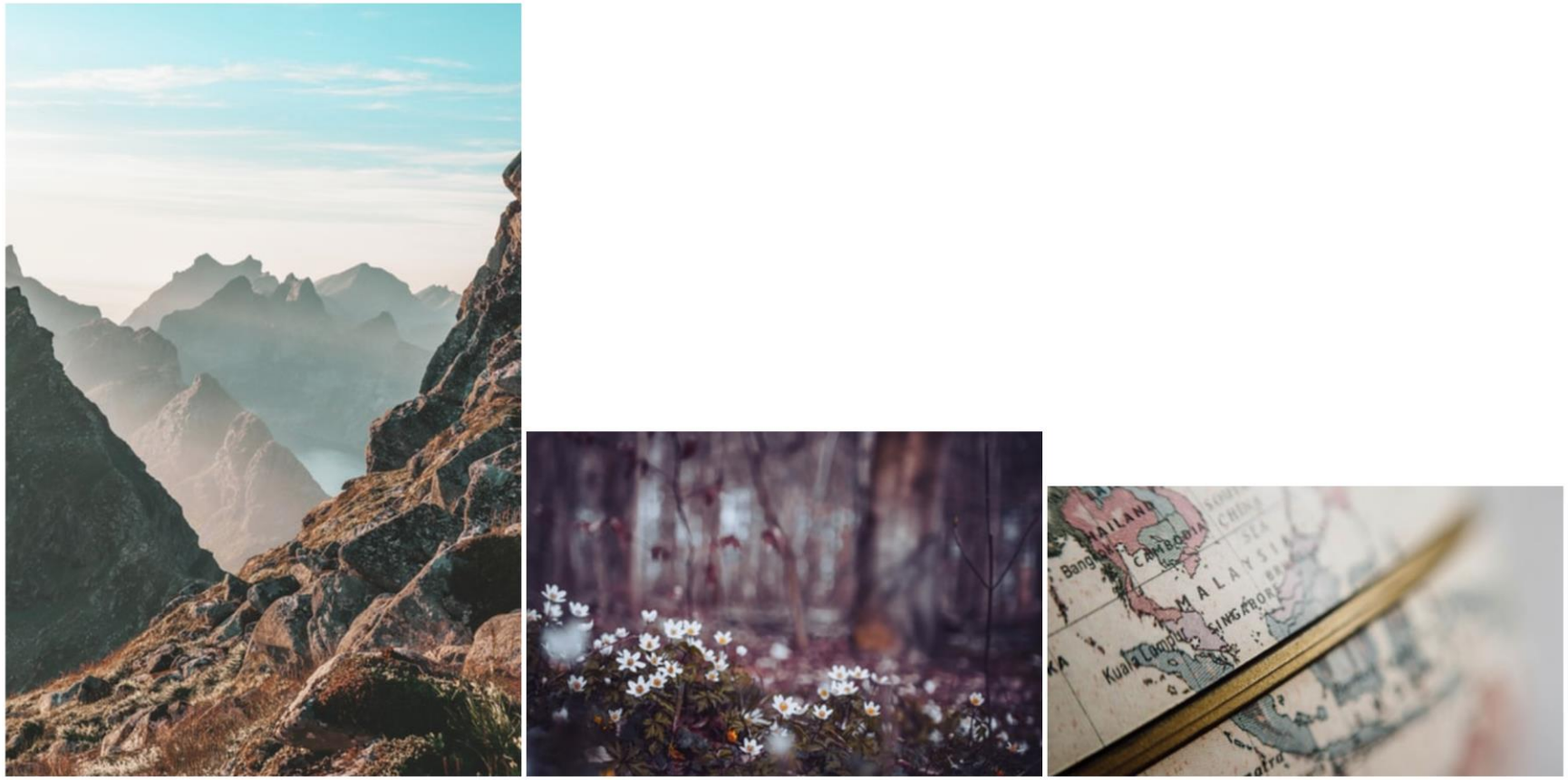
- Với sự tiện ích của jQuery trong việc xử lý các DOM trên màn hình, jQuery là cơ sở cho rất nhiều các thư viện tiện ích khác:
  - JQuery lightSlider - http://sachinchoolur.github.io/lightslider/
  - jQuery Validation - https://jqueryvalidation.org/
  - Fancybox - http://fancyapps.com/fancybox/3/
  - Alertify js - http://fabien-d.github.io/alertify.js/
  - jQuery File Upload - http://blueimp.github.io/jQuery-File-Upload/
  - Select2 - https://select2.org/
  - ...

CÁC THƯ VIỆN PHÁT TRIỂN TRÊN NỀN JQUERY

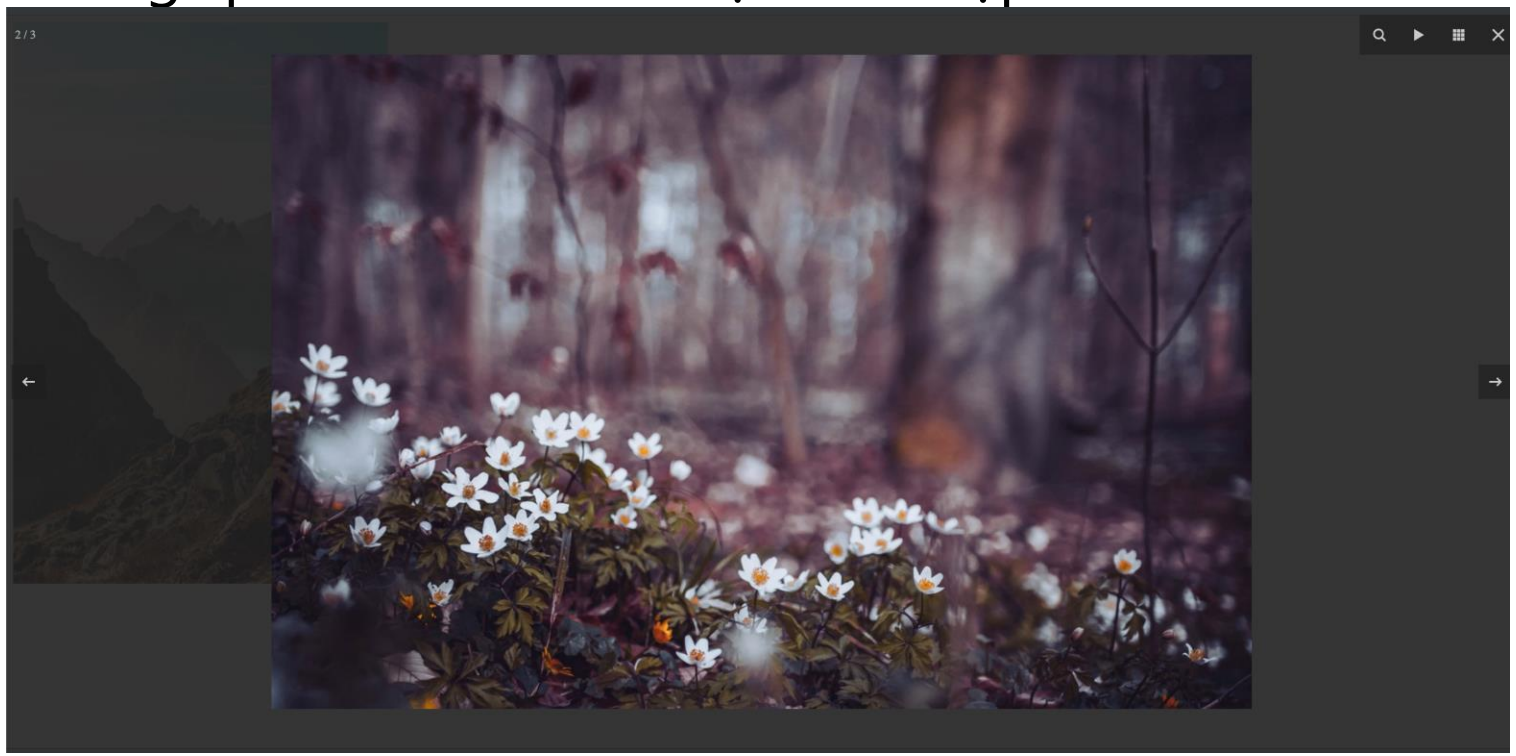
- Demo sử dung fancybox để show gallery ảnh
  - Set up code – nhúng thư viện jquery trước, sau đó đến thư viện fancybox
  - Viết các thẻ hiển thị ảnh (sinh viên tự viết style)

```
<script src="https://cdn.jsdelivr.net/npm/jquery@3.4.1/dist/jquery.min.js"></script>

<link rel="stylesheet" href="https://cdn.jsdelivr.net/gh/fancyapps/fancybox@3.5.7/dist/
jquery.fancybox.min.css" />
<script src="https://cdn.jsdelivr.net/gh/fancyapps/fancybox@3.5.7/dist/jquery.fancybox.min.js"></
script>

<a data-fancybox="gallery" href="./big-1.jpeg"><img src="./small-1.jpeg"></a>
<a data-fancybox="gallery" href="./big-2.jpeg"><img src="./small-2.jpeg"></a>
<a data-fancybox="gallery" href="./big-3.jpeg"><img src="./small-3.jpeg"></a>
```

- Demo sử dung fancybox để show gallery ảnh
  - Giao diện khi hiển thị mặc định



- Demo sử dung fancybox để show gallery ảnh
  - Giao diện khi click vào một ảnh
  - Fancybox sẽ hiển thị gallery ảnh sinh động và có thể tùy chỉnh lựa chọn các ảnh thông qua các button được thiết lập sẵn



TỔNG KẾT

- ☑ Giải thích và sử dụng các api Traversing của jQuery
- ☑ Sử dụng được jQuery Ajax để gửi request
- ☑ Sử dụng được các thư viện trên nền tảng jQuery


thank you!
TIN HỌC VĂN PHÒNG QUỐC TẾ THEO CHUẨN ICDL
Syllabus 5 - Office 2007
LUẬT KINH TẾ
LUẬT KINH TẾ