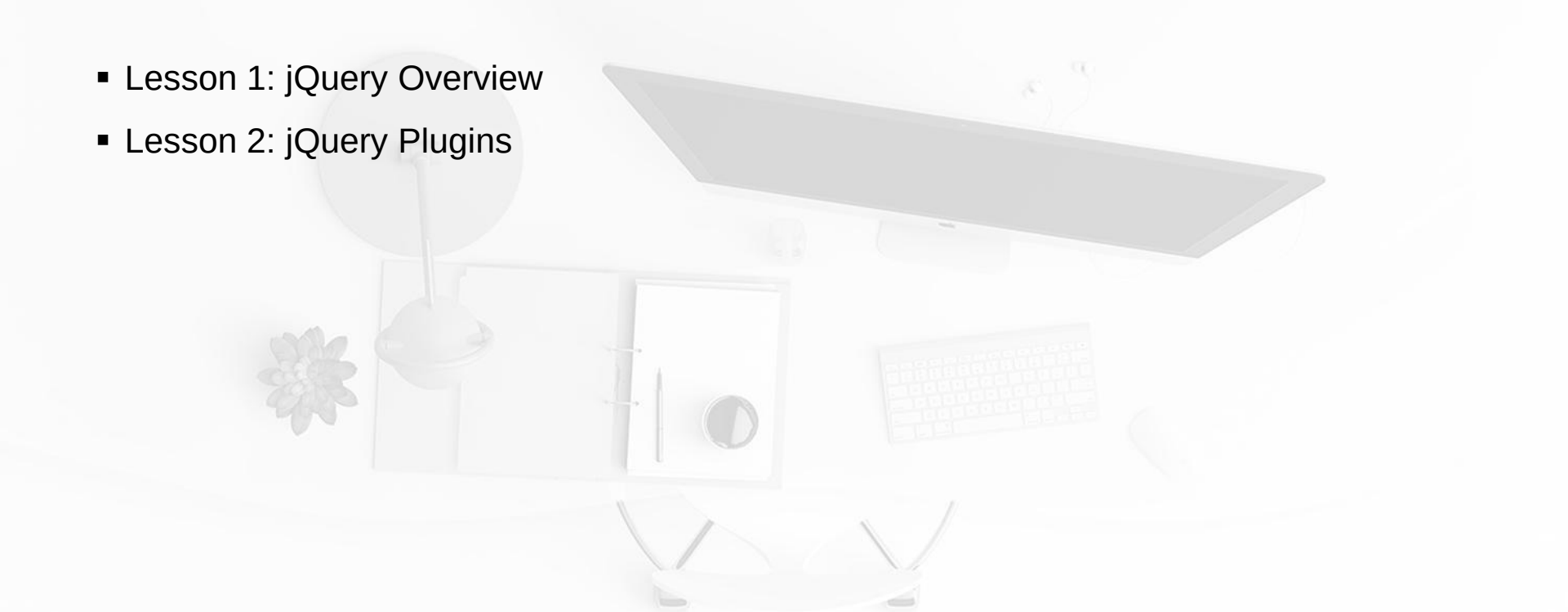
Front-end Development

Lecturer: Ung Văn Giàu
Email: giau.ung@eiu.edu.vn

Content

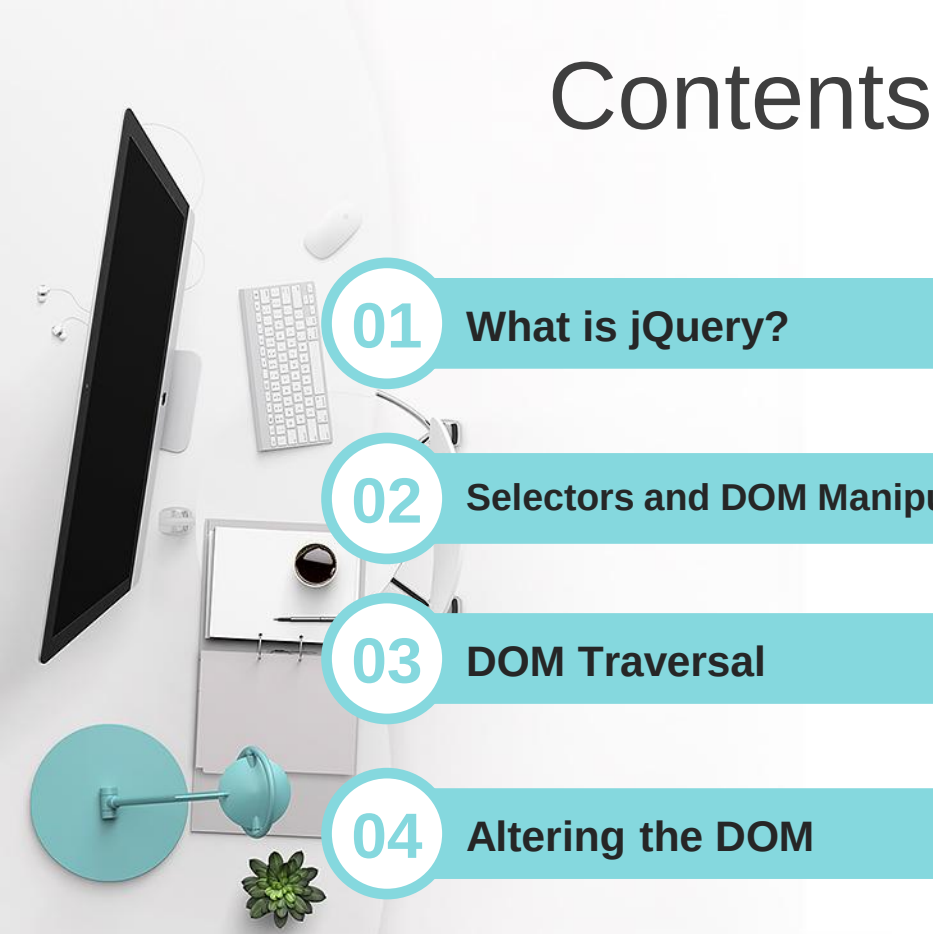


- Lesson 1: jQuery Overview
- Lesson 2: jQuery Plugins

jQuery Overview

Unleash the Power of jQuery

Contents



1. What is jQuery?

The world’s most popular JavaScript library

What is jQuery?

- jQuery is a cross-browser JavaScript library
  - Designed to simplify the client-side scripting of HTML
  - The most popular JavaScript library in use today
  - Free, open-source software

- jQuery’s syntax is designed to make it easier to
  - Navigate a document and select DOM elements
  - Create animations
  - Handle events

What is jQuery?

- jQuery also provides capabilities for developers to create plugins for
  - Low-level interaction and animation
  - Advanced effects and high-level, theme-able widgets
  - Creation of powerful and dynamic web pages

- Microsoft adopted jQuery within Visual Studio

  Used in Microsoft’s ASP.NET AJAX Framework and ASP.NET MVC Framework

Why jQuery is So Popular?

- Easy to learn

  Fluent programming style
- Easy to extend

  You create new jQuery plugins by creating new JavaScript functions
- Powerful DOM Selection

  Powered by CSS 3.0
- Lightweight
- Community Support

  Large community of developers and geeks

How to Add jQuery to a Web Site?

- Download jQuery files from

  http://www.jquery.com

- Self hosted
  - You can choose to self host the .js file
  - E.g. jquery-2.1.1.js or .min.js file

- Use it from CDN (content delivery network)

  Microsoft, jQuery, Google CDNs
  - http://code.jquery.com/jquery-2.1.1.min.js
  - http://ajax.microsoft.com/ajax/jquery/jquery-2.1.1.min.js

2. Selectors and DOM Manipulation

Selectors

- Selection of DOM elements in jQuery is much like as in pure JavaScript
- Selection of elements by using CSS selectors

$(selector)

- Like **querySelectorAll**

```
// By tag name
$("div")    // document.querySelectorAll("div");
// By class name
$(".menu-item") // document.querySelectorAll(".menu-item");
// By ID
$("#Search")
// By combination of selectors
$("ul.menu li:nth-child(2)")
```

Selection with jQuery

- Selecting items with jQuery
  - Almost always **returns a collection** of the items
  - Even if there is only one item
  - Can be stored in a variable or used right away
  - The usage of the elements is always the same, no matter whether a single or many elements

```
// select the item
$("#something").hide();
$(".widgets").fade(1);
```

- More at: http://learn.jquery.com/using-jquerycore/selecting-elements/

3. DOM Traversal

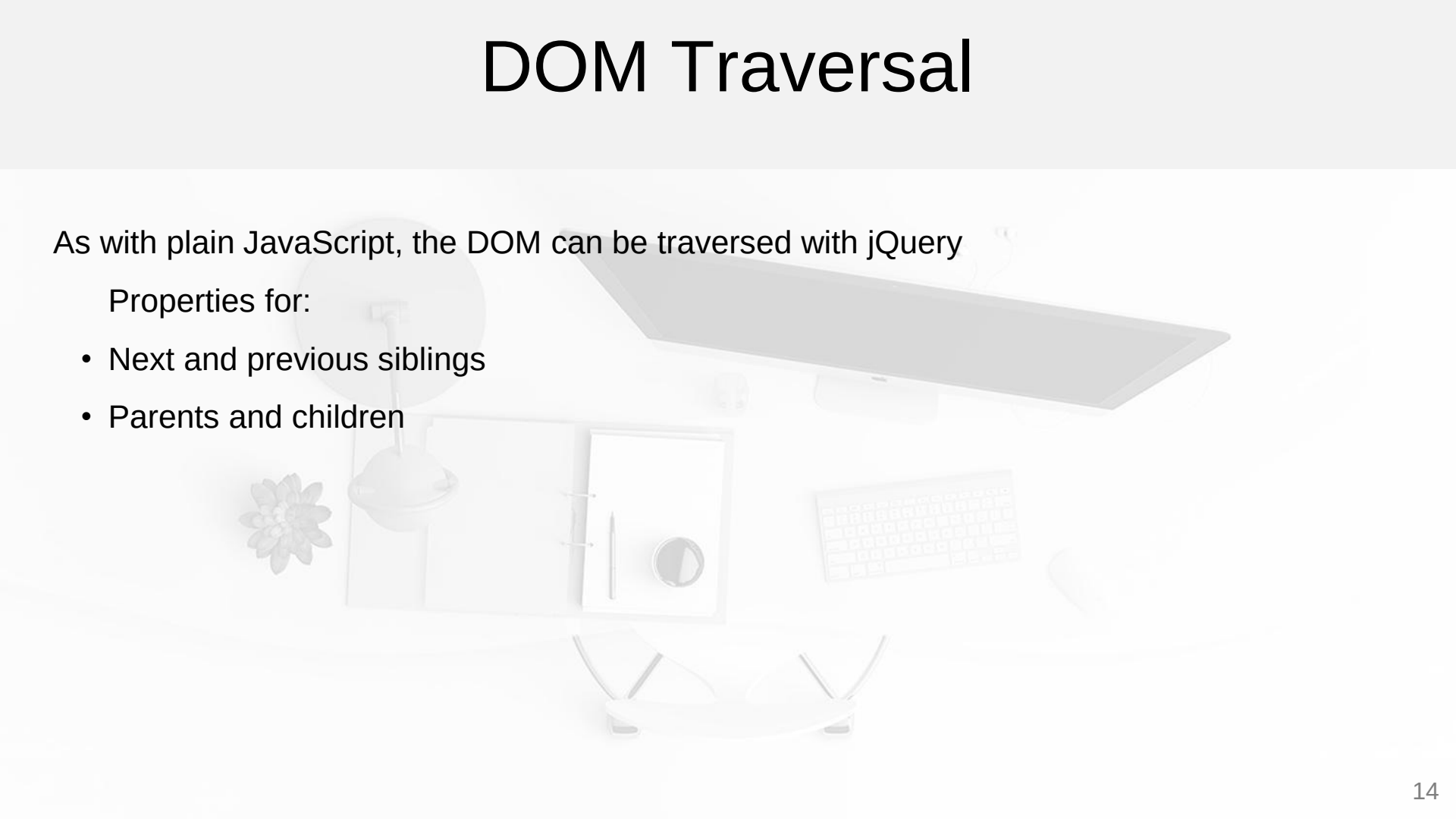
Traversing the nodes of the DOM

DOM Traversal

As with plain JavaScript, the DOM can be traversed with jQuery

Properties for:

- Next and previous siblings
- Parents and children



DOM Traversal

Next and Previous

- **jQuery.next(), jQuery.prev()**
  - Returns the next/prev sibling
  - Returns an HTML element

    Not a [text] node

```
<ul>
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
</ul>
```

```
var firstItem = $("li").first();
console.log(firstItem);
// Log "Item 1"
console.log(firstItem.next());
// Log "Item 2"
```

DOM Traversal

Parent

- **jQuery.parent()**

  Returns the parent of the element

- **jQuery.parents(selector)**

  Returns the first parent that matches the selector

```
<div id="wrapper">
    <ul id="items-list">
        <li>Item 1</li>
        <li>Item 2</li>
        <li class="special">Item 3</li>
        <li>Item 4</li>
    </ul>
</div>
```

```
var node = $(".special");
node.parent().attr("id");
// Log "items-list"
node.parents("div").attr("id");
// Log "wrapper"
node.parents("#wrapper").attr("id");
// Log "wrapper"
```

4. Altering the DOM

Adding and removing DOM elements

Creating Elements

Creating new elements is also easy

```
var divElement = $('<div>');
var anotherDivElement = $('<div />');
```

Adding Elements

Adding elements can be done on the fly

- jQuery.appendTo()/prependTo()
- jQuery.append()/prepend()

```
$('<ul><li>Hello</li></ul>').appendTo('body');
$("body").prepend("<h1>header</h1>");
```

Removing Elements

You can also remove elements from the DOM

Just as easy

- Before

```
<div>
    <p>Red</p>
    <p>Green</p>
</div>
```

```
// Removing elements
$('p').remove();
```

- After

```
<div>
</div>
```

5. jQuery Extended DOM Elements

jQuery Objects

Selected jQuery DOM elements are NOT pure DOM elements

- They are extended
- Have additional properties and methods
  - addClass(), removeClass(), toggleClass()
  - on(event, callback) for attaching events
  - animate(), fade(), etc.

```
// Parsing a regular DOM element to jQuery Element
var divJSObject = document.createElement("div");
var divjQueryObject = $(divJSObject);
```

Properties of jQuery Elements

- jQuery elements extend regular DOM elements

- Methods for altering the elements
  - jQuery.**css**(“color”, “#ccc”)

  - jQuery.**html**() returns the innerHTML
  - jQuery.**html**(content) sets the innerHTML

  - jQuery.**text**(content) sets the innerHTML, by escaping the content

6. jQuery Events

Cross-browser events

jQuery Events

jQuery has a convenient way for attaching and detaching events

- Works cross-browser
- Using methods **on()**

jQuery Events

.on()

.on(events [, selector] [, data], handler)

- **Description:**

  Attach an event handler function for one or more events to the selected elements

- **events**
  - Type: String
  - E.g.: “click” or “blur“
- **selector**
  - Type: String
  - A selector string to **filter the descendants of the selected elements** that trigger the event. If the selector is **null or omitted**, the event is always **triggered** when it reaches the **selected element**
  - E.g.: “li”

jQuery Events

.on()

.on(events [, selector] [, data], handler)

- **data**
  - Type: Anything
  - Data to be passed to the handler in **event.data** when an event is triggered
- **handler**
  - Type: Function
  - A function to execute when the event is triggered

jQuery Events

.on() Example

- **Direct event handler**

  If **selector** is **omitted or is null**, the event handler is referred to as direct or directly-bound

```
function onButtonClick(){
    $(".selected").removeClass("selected");
    $(this).addClass("selected");
}
$("a.button").on("click", onButtonClick);
```

jQuery Events

.on() Example

- **Delegated event handlers**

  have the advantage that they can process events from descendant elements that are added to the document later

- Optimize the event
  - Add it on the parent element
  - A bit different syntax

```
function onListItemClick(){
    $(".selected").removeClass("selected");
    $(this).addClass("selected");
}
$("ul").on("click", "li", onListItemClick);
```

jQuery Events

.on() Example

- Example 1:

```
$( "#dataTable tbody tr" ).on( "click", function() {
  console.log( $( this ).text() );
});
```

- Example 2:

```
$( "#dataTable tbody" ).on( "click", "tr", function() {
  console.log( $( this ).text() );
});
```

- The example 2 is much lower overhead than the example 1

jQuery Events

Event API

- **.change()**

  Bind an event handler to the “change” event, or trigger that event on an element.

- **.click()**

  Bind an event handler to the “click” event, or trigger that event on an elementA function to execute when the event is triggered

- **.keypress()**

  Bind an event handler to the “keypress” event, or trigger that event on an element

7. jQuery Chaining

Call after call, after call…

Query Chaining

- The chaining paradigm is as follows:
  - If a method should return result → Ok, return it
  - If a method should NOT return a result → return this

- jQuery implements this paradigm, so methods can be chained to one another:

```
$('<button />')
    .addClass('btn-success')
    .html('Click me for success')
    .on('click', onSuccessButtonClick)
    .appendTo(document.body);
```

8. jQuery AJAX

Creating HTTP requests with jQuery

jQuery AJAX

- AJAX stands for Asynchronous JavaScript and XML

  Meaning asynchronously get data from a remote place and render it dynamically
- jQuery provides some methods for AJAX
  - jQuery.**ajax**(options) – HTTP request with full control (headers, data, method, etc.)
  - jQuery.**get**(url) – HTTP GET request
  - jQuery.**post**(url) – HTTP POST request
  - jQuery(selector).**load**(url) – loads the contents from the url inside the selected node

jQuery.get()

jQuery.get(url [, data] [, success] [, dataType])

- **Description:**

  Load data from the server using a HTTP GET request

- **url:**
  - Type: String
  - A string containing the URL to which the request is sent
  - E.g.: https://domain.com/process-data.php
- **data:** optional
  - Type: PlainObject or String
  - A plain **object or string** that is sent to the server with the request
  - E.g. 1: {Parameter_1 : “value_1”, Parameter_2 : “value_2” }
  - E.g. 2 : Parameter_1=value_1&Parameter_2=value_2

jQuery.get()

jQuery.get(url [, data] [, success] [, dataType])

- **success:** optional
  - Type: Function(PlainObject data, String textStatus)
  - A callback function that is executed if the request succeeds
  - E.g.: function (data) { //your code }
- **dataType:** optional
  - Type: String (xml, json, script, text, html)
  - The type of data expected from the server
  - E.g.: “json”. **Note:** if you use dataType is json, you do not need to use JSON.pare() to convert a data string from server into a Javascript Object

jQuery.get()

Example

```
$.get(
    "http://example.com/get-data.php",
    {
        parameterName_1: "value_1",
        parameterName_2: "value_2"
    },
    function(data) {
        // Your code
    },
    "json"
);
```

jQuery.get()

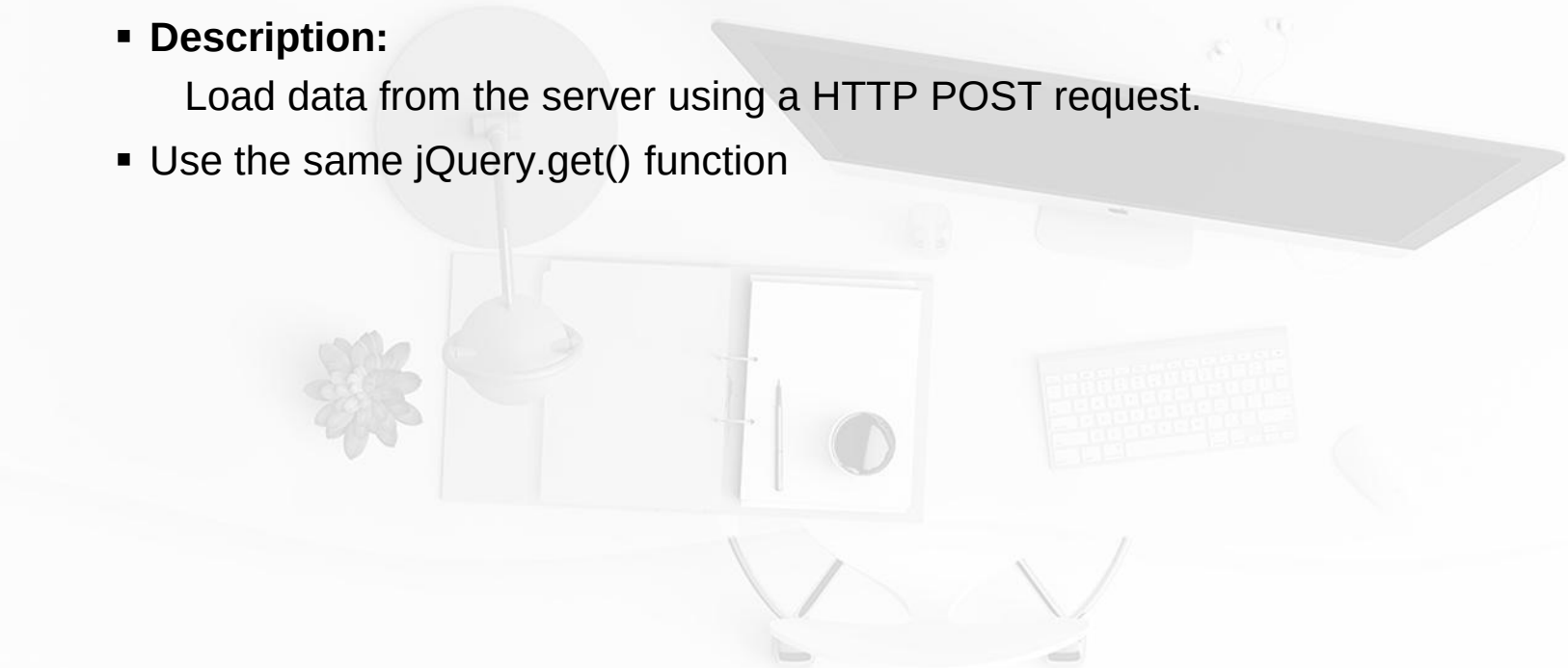
Example

```
$.get(
    "http://example.com/test.php",
    {
        name: "John", time: "2pm"
    },
    function( data ) {
        $( "body" )
            .append( "Name: " + data.name ) // John
            .append( "Time: " + data.time ); //  2pm
        }
    , "json"
);
```

jQuery.post()

- **Description:**

  Load data from the server using a HTTP POST request.
- Use the same jQuery.get() function



Exercises

1. Create a slider control using jQuery
   - The slider can have many slides
   - Only one slide is visible at a time
   - Each slide contains HTML code

     i.e. it can contain images, forms, divs, headers, links, etc.
   - Implement functionality for changing the visible slide after 5 seconds
   - Create buttons for next and previous slide

2. Using jQuery implement functionality to insert a DOM element before or after another element

Exercises

3. By given an array of students, generate a table that represents these students
   - Each student has first name, last name and grade
   - Use jQuery

4. Implement functionality to change the background color of a web page

   I.e. select a color from a color picker and set this color as the background color of the page

jQuery Plugins

Expanding the functionality of jQuery

jQuery Plugins

- A Plugin is just a method that extends the jQuery objects prototype

  Enabling all jQuery objects to use this method
- Once a plugin is imported, it is used as regular jQuery method

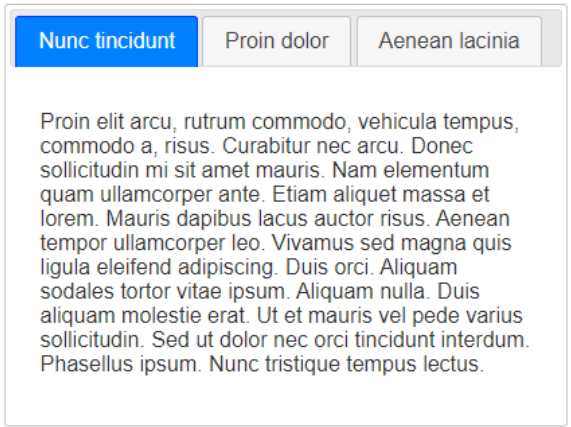
  Like **addClass**(), **fadeout**() and **hide**()

jQuery Plugins

- jQuery has many ready-to-use plugins
  A library jQueryUI for UI controls
- Plugins for UI

Tabs

```
$("#tabs-holder").tabs();
```



Arrangeable elements

```
$("#grid" ).sortable();
```


Summary

Summary

Source: https://api.jquery.com/

- $(“selector”).**click**(handler): Bind an event handler to the "click" JavaScript event, or trigger that event on an element.
- $(“selector”).**val**([value]): get/set the value
- $(“selector”).**addClass**(className): add the specified class(es)
- $(“selector”).**removeClass**(className): remove a single class, multiple classes
- $(“selector”).**remove**(): remove the set of matched elements from the DOM
- $(“selector”).**show**(): display the matched elements
- $(“selector”).**hide**(): hide the matched elements
- $(“selector”).**html**([value]): get/set the HTML contents

Summary

Source: https://api.jquery.com/

- $.**trim**(str): remove the whitespace from the beginning and end of a string
- $(“selector”).**attr**(attributeName [,value]): get / set the value of an attribute
- $(“selector”).**removeAttr**(attributeName): remove an attribute
- $(“selector”).**append**(content): insert content (HTML string, jQuery), specified by the parameter, to the end of the selector
- $(“selector”).**prepend**(content): insert content (HTML string, jQuery), specified by the parameter, to the beginning of the selector
- $(“selector”).**toggleClass**(className): add or remove one or more classes depending on either the class's presence or the value of the state argument.

$(document).ready()

- Code included inside $(document).ready() will only run once the page DOM is ready for JavaScript code to execute
- **Syntax:**

```
// A $( document ).ready() block.
$( document ).ready(function() {
    // Your code
});

// Shorthand for $( document ).ready()
$(function() {
    // Your code
});
```


Q&A