CSIS 3380
Advanced Web Programming with JavaScript & AJAX

AJAX and jQuery

Week 6

Revision

- Different types of events in the browser
- How to register an Event Handler to HTML element
- The Event object in event handler
  - event.target
  - event.preventDefault()
  - Event.stopPropagation()

REST

- REST stands for **RE**presentational **S**tate **T**ransfer
- It is a web standards based architecture and uses HTTP Protocol for data communication
- In REST architecture, a REST Server simply provides access to resources and the REST client accesses and presents the resources
- Each resource is identified by URIs/ Global IDs
- Many of the **REST** endpoints use JSON as the format to represent the resources

RESTful API

- A RESTful API explicitly takes advantage of HTTP methodologies
  - **GET** to retrieve a resource
  - **PUT** to change the state of or update a resource
  - **POST** to create that resource
  - **DELETE** to remove it
- Examples
  - GET /photos
  - PUT /photos/1234
  - POST /photos
  - DELETE /photos/1234

Examples of RESTful APIs

- http://universities.hipolabs.com/
- http://universities.hipolabs.com/search?country=canada&name=douglas
- https://github.com/toddmotto/public-apis

JSON

- JavaScript Object Notation (JSON) is a standard **text-based** format for storing and exchanging data
- A JSON can be stored in its own file, which is basically just a text file with an extension of .json
- JSON is the most commonly format for the response from a RESTful API
- Analogy
  - When you access a **webpage URL**, it returns **HTML**
  - When you invoke a **RESTful API**, it returns **JSON**

Example of a JSON

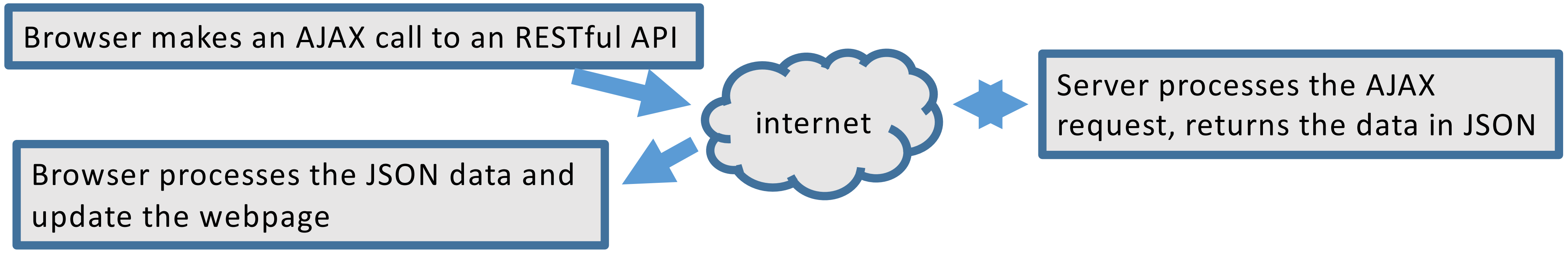
```
{
  "id": 654197,
  "name": "John Smith",
  "email": "john.smith@example.com",
  "address": {
    "street": "1861 Main St",
    "city": "Vancouver",
    "province": "BC"
  }
}
```

```
[
  {
    "id": 654197,
    "name": "John Smith",
    "email": "john.smith@example.com",
    "address": {
      "street": "1861 Main St",
      "city": "Vancouver",
      "province": "BC"
    }
  },
  {
    "id": 326721,
    "name": "Ryan Wong",
    "email": "ryan.wong@example.com",
    "address": {
      "street": "273 Pleasant Road",
      "city": "Vancouver",
      "province": "BC"
    }
  }
]
```

Note the similarity between JSON and javascript objects

AJAX

- **A**synchronous **J**avaScript **A**nd **X**ML
- Note most APIs nowadays use JSON rather than XML
- AJAX allows web pages to be updated **asynchronously** by exchanging data with a web server behind the scenes
- This means that it is possible to update parts of a web page, without reloading the whole page


Browser makes an AJAX call to an RESTful API
internet
Server processes the AJAX request, returns the data in JSON
Browser processes the JSON data and update the webpage

Javascript libraries

- A JavaScript library is a library of pre-written JavaScript which allows for easier development of JavaScript-based applications
- When you include a Javascript library on your website, you essentially enhance the javascript capability of your site
- We will look at jQuery in this course

jQuery

- jQuery is a javascript library that makes writing common javascript tasks easier
- Example:

Compare this:

```
var demo = document.getElementById(”#demo");
demo.innerHTML = "Hello, World!";
```

with this:

```
$("#demo").html("Hello, World!");
```

Adding jQuery to your website

- You can download jQuery from its site: http://jquery.com/download/ and link the downloaded local file in your HTML
- Or you can link to a jQuery CDN directly in your HTML

```
<script src="my_downloaded_jquery.js”></script>
```

OR

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.js"></script>
```

- After jQuery is added to your website, jQuery will be represented by the dollar sign (**$**)
- Wait for the DOM to be loaded before start using jQuery

```
$(document).ready(function() {
   // all custom jQuery will go here
   $("#demo").html("Hello, World!");
});
```

- The above javascript needs to be **after** the script tag that link to jQuery

How to use jQuery

- Selectors are how we tell jQuery which elements we want to work on
- To access a selector, use the jQuery symbol $, followed by parentheses ()

With native addEventListener:

```
<!DOCTYPE html>
<html>
<body>
<div>
 <button id="myBtn">Via event listener</button>
</div>
<script>
 function toBlue(event) {
  event.target.style.backgroundColor = 'blue';
 }
 var myBtn = document.getElementById('#myBtn');
 myBtn.addEventListener('click', toBlue);
</script>
</body>
</html>
```

With jQuery:

```
<!DOCTYPE html>
<html>
<body>
<div>
 <button id="myBtn">Via event listener</button>
</div>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.js"></script>
<script>
 $(document).ready(function() {
  function toBlue(event) {
   event.target.style.backgroundColor = 'blue';
  }
  $('#myBtn').on('click', toBlue);
 });
</script>
</body>
</html>
```

jQuery events

- The list of jQuery events: https://api.jquery.com/category/events/
- A brief overview of some of the most commonly used event methods.
  - click()- **Click:** executes on a single mouse click.
  - hover() - **Hover:** executes when the mouse is hovered over an element. mouseenter() and mouseleave() apply only to the mouse entering or leaving an element, respectively.
  - submit() - **Submit:** executes when a form is submitted.
  - scroll() - **Scroll:** executes when the screen is scrolled.
  - keydown() - **Keydown:** executes when you press down on a key on the keyboard
- You can use the jQuery **.on()** method to attach an **event handler** for an **event** to the **selected HTML elements**
- Likewise, you can use the **.off()** method to remove an event handler from the selected elements

jQuery Effects

- jQuery enables you to add animations to the page easily
- You can control speed of toggle by passing in argument ”fast” or “slow”
- Try changing toggle with different effect methods:
  - fadeToggle
  - slideToggle
- Complete effect methods list: http://api.jquery.com/category/effects/

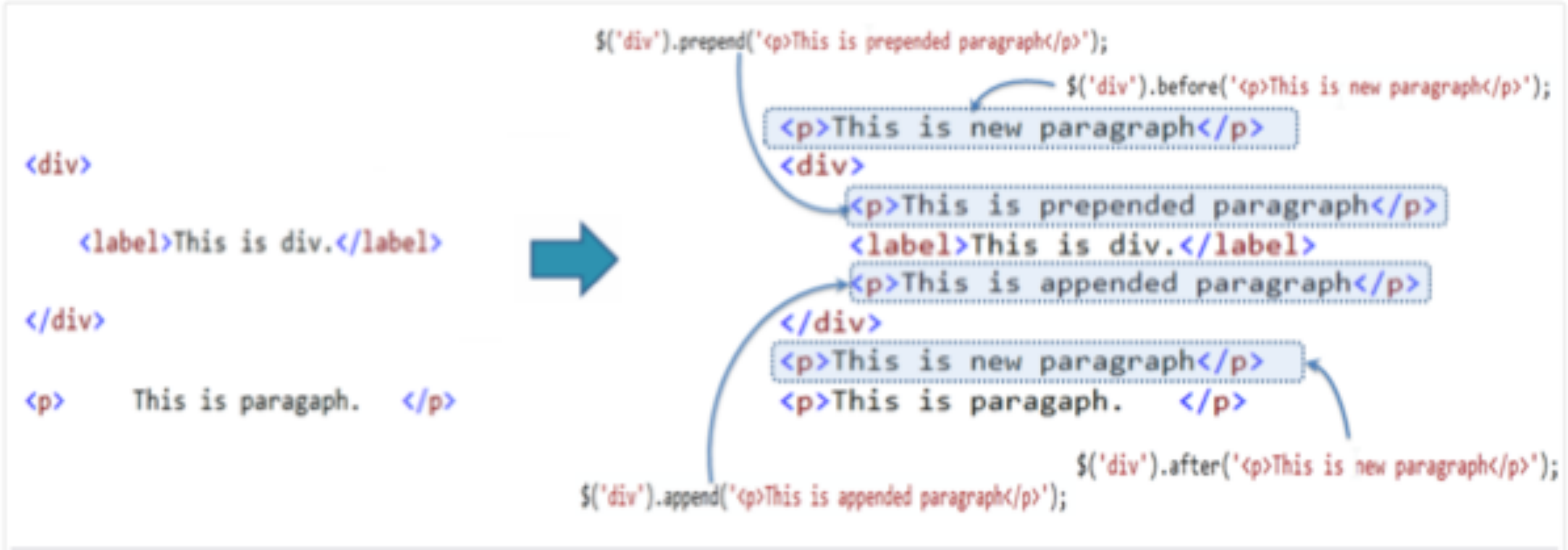
```
<html>
 <head>
  <title>jQuery Demo</title>
 </head>
 <body>
  <button id="clickme">Open</button>
  <div>
   <img src="douglas_logo.png" style="display: none">
  </div>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.js">
  </script>
  <script>
   $(document).ready(function() {
    $("#clickme").on('click', function() {
      $("div img").toggle();
    });
   });
  </script>
</body>
</html>
```

jQuery DOM manipulation

- http://api.jquery.com/category/manipulation/

Method	Description
append()	Inserts content to the end of element(s) which is specified by a selector.
before()	Inserts content (new or existing DOM elements) before an element(s) which is specified by a selector.
after()	Inserts content (new or existing DOM elements) after an element(s) which is specified by a selector.
prepend()	Insert content at the beginning of an element(s) specified by a selector.
remove()	Removes element(s) from DOM which is specified by selector.
replaceAll()	Replace target element(s) with specified element.
wrap()	Wrap an HTML structure around each element which is specified by selector.

jQuery DOM manipulation


$('div').prepend('<p>This is prepended paragraph</p>');
$('div').before('<p>This is new paragraph</p>');
$('div').append('<p>This is appended paragraph</p>');
$('div').after('<p>This is new paragraph</p>');

```
<div>

    <label>This is div.</label>

</div>

<p>     This is paragaph.    </p>
```

```
<p>This is new paragraph</p>
<div>
    <p>This is prepended paragraph</p>
    <label>This is div.</label>
    <p>This is appended paragraph</p>
</div>
<p>This is new paragraph</p>
<p>This is paragaph.    </p>
```

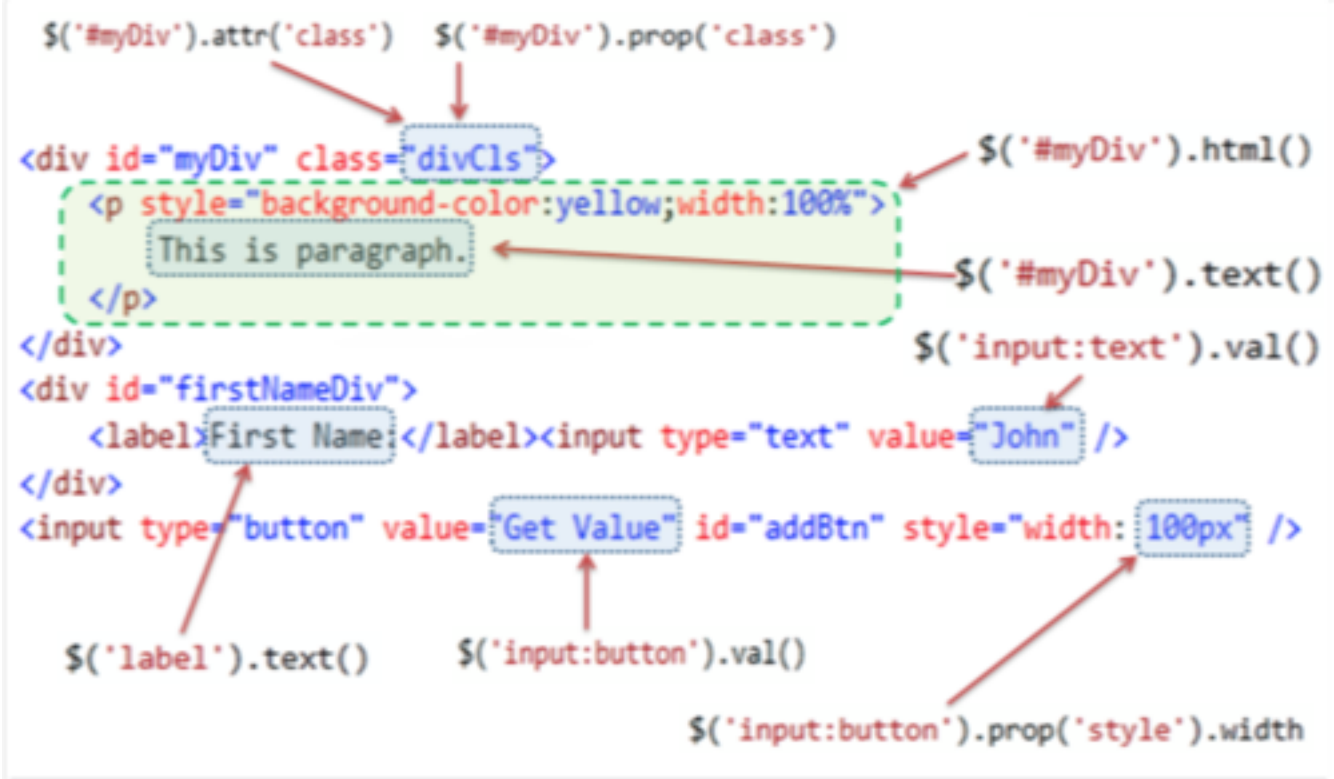
DOM Manipulation Methods

jQuery Attribute manipulation

- http://api.jquery.com/category/attributes/

jQuery Method	Description
attr()	Get or set the value of specified attribute of the target element(s).
prop()	Get or set the value of specified property of the target element(s).
html()	Get or set html content to the specified target element(s).
text()	Get or set text for the specified target element(s).
val()	Get or set value property of the specified target element.

jQuery Attribute manipulation



```
$('#myDiv').attr('class')   $('#myDiv').prop('class')

<div id="myDiv" class="divCls">                              $('#myDiv').html()
    <p style="background-color:yellow;width:100%">
        This is paragraph.                                   $('#myDiv').text()
    </p>
</div>                                                       $('input:text').val()
<div id="firstNameDiv">
    <label>First Name:</label><input type="text" value="John" />
</div>
<input type="button" value="Get Value" id="addBtn" style="width: 100px" />

   $('label').text()        $('input:button').val()

                                  $('input:button').prop('style').width
```

jQuery methods to access element's values

jQuery AJAX

- jQuery provides an easy way to make AJAX requests and handle response accordingly
- https://api.jquery.com/category/ajax/
- Example of a **GET** request :

```
<script>
    $(document).ready(function() {
      $.ajax({
        type: 'GET',
        url: 'https://randomuser.me/api/',
        dataType: 'json',
        success: function(data) {
          console.log(data);
        }
      });
    });
  </script>
```

More examples

```
<html>
 <head>
  <title>jQuery Demo</title>
 </head>
 <body>
  <h2>Users</h2>
  <ul id="users"></ul>
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.js">
</script>
  <script src="main.js"></script>
</body>
</html>
```

main.js

```
$(document).ready(function() {
 var $users = $('#users');

 $.ajax({
  type: 'GET',
  url: 'https://randomuser.me/api/?nat=us&results=10',
  dataType: 'json',
  success: function(data) {
   console.log(data);
   data.results.forEach(function(user) {
    var name = user.name.first +' '+ user.name.last;
    $users.append('<li>'+name+'</li>');
   });
  }
 });
});
```

Example of an AJAX POST request

```
$(document).ready(function() {
  var student = { name: 'John Smith', age: 21 };

  $.ajax({
    url: "https://reqres.in/api/users",
    type: "POST",
    data: student,
    success: function(response){
      console.log(response);
    }
  });
});
```

Note user is not really created on server but it is faked as if the data is created on the server

Lab

- Download **lab6-story.html** and **lab6-story.css** from Blackboard
- The objective of this lab exercise to implement the following behavior in **lab6-story.html** :
  - When **lab6-story.html** is loaded initially, the 2 buttons for style selection should hidden
  - Clicking the ”**Style Picker**” should toggle the 2 buttons between display and hidden
  - When the buttons are displayed, the “**default**” button will be selected by default
  - The selected button should be **bold** so that user knows which button is currently selected
  - Clicking the “**print**” button should make the font-size of the content 1.5 bigger
  - Use the CSS declaration in lab6.css to help you

Lab6

- Download **lab6-jobs.html** and **lab6-jobs-server.zip** from blackboard
- Unzip **lab6-jobs-server.zip** and start the server in the CLI with the command: **node app.js**
- Use jQuery to make an API call to http://localhost:3000?job=javascript
- The **job** query parameter in the API should be changed according to the selected job
- Get the **first** job returned in the array and rendered it on **lab6-jobs.html**