Introduction to JavaScript: Part 3

Introduction to Internet and Web

부산대학교 정보·의생명공학대학
정보컴퓨터공학부

Contents

- ❖ jQuery
- ❖ AJAX
- ❖ JSON

JQUERY

What is jQuery

- jQuery is a lightweight, "write less, do more", JavaScript library.
- The purpose of jQuery is to make it much easier to use JavaScript on your website.
- jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.
- The jQuery library contains the following features:
  - HTML/DOM manipulation
  - CSS manipulation
  - HTML event methods
  - Effects and animations
  - AJAX
  - Utilities

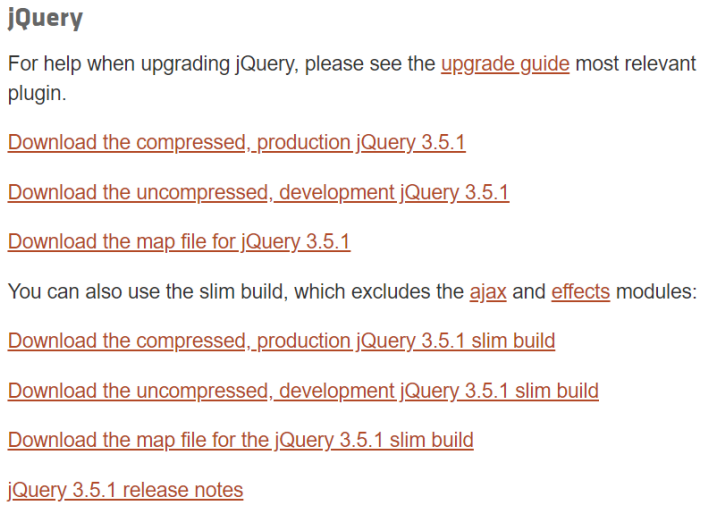
Adding jQuery to Your Web Pages

❖ Download the jQuery library from jQuery.com

- Production version - this is for your live website because it has been minified and compressed
- Development version - this is for testing and development (uncompressed and readable code)
- https://jquery.com/download/

❖ Include jQuery from a CDN, like Google

- Content Delivery Network (CDN)



jQuery

For help when upgrading jQuery, please see the upgrade guide most relevant plugin.

Download the compressed, production jQuery 3.5.1

Download the uncompressed, development jQuery 3.5.1

Download the map file for jQuery 3.5.1

You can also use the slim build, which excludes the ajax and effects modules:

Download the compressed, production jQuery 3.5.1 slim build

Download the uncompressed, development jQuery 3.5.1 slim build

Download the map file for the jQuery 3.5.1 slim build

jQuery 3.5.1 release notes

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
</head>
```

jQuery Syntax

❖ Basic syntax is: $(selector).action()

- A $ sign to define/access jQuery
- A (selector) to "query (or find)" HTML elements
- A jQuery action() to be performed on the element(s)
- Example)
  - $(this).hide(), $("p").hide(), $(".text").hide()

❖ This is to prevent any jQuery code from running before the document is finished loading (is ready).

```
$(document).ready(function(){

  // jQuery methods go here...

});
```

jQuery Selector

❖ It's based on the existing CSS Selectors

Syntax	Description
$("*")	Selects all elements
$(this)	Selects the current HTML element
$("p.intro")	Selects all <p> elements with class="intro"
$("p:first")	Selects the first <p> element
$("ul li:first")	Selects the first element of the first
$("ul li:first-child")	Selects the first element of every
$("[href]")	Selects all elements with an href attribute
$("a[target='_blank']")	Selects all <a> elements with a target attribute value equal to "_blank"
$("a[target!='_blank']")	Selects all <a> elements with a target attribute value NOT equal to "_blank"
$(":button")	Selects all <button> elements and <input> elements of type="button"
$("tr:even")	Selects all even <tr> elements
$("tr:odd")	Selects all odd <tr> elements

jQuery Events

❖ An event represents the precise moment when something happens.

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("p").click(function(){
    $(this).hide();
  });
});
</script>
</head>
<body>

<p>If you click on me, I will disappear.</p>
<p>Click me away!</p>
<p>Click me too!</p>

</body>
</html>
```

If you click on me, I will disappear.

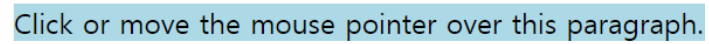
Click me away!

Click me too!

Mouse Events	Keyboard Events	Form Events	Document/Window Events
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

jQuery Events

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("p").on({
    mouseenter: function(){
      $(this).css("background-color", "lightgray");
    },
    mouseleave: function(){
      $(this).css("background-color", "lightblue");
    },
    click: function(){
      $(this).css("background-color", "yellow");
    }
  });
});
</script>
</head>
<body>

<p>Click or move the mouse pointer over this paragraph.</p>

</body>
</html>
```



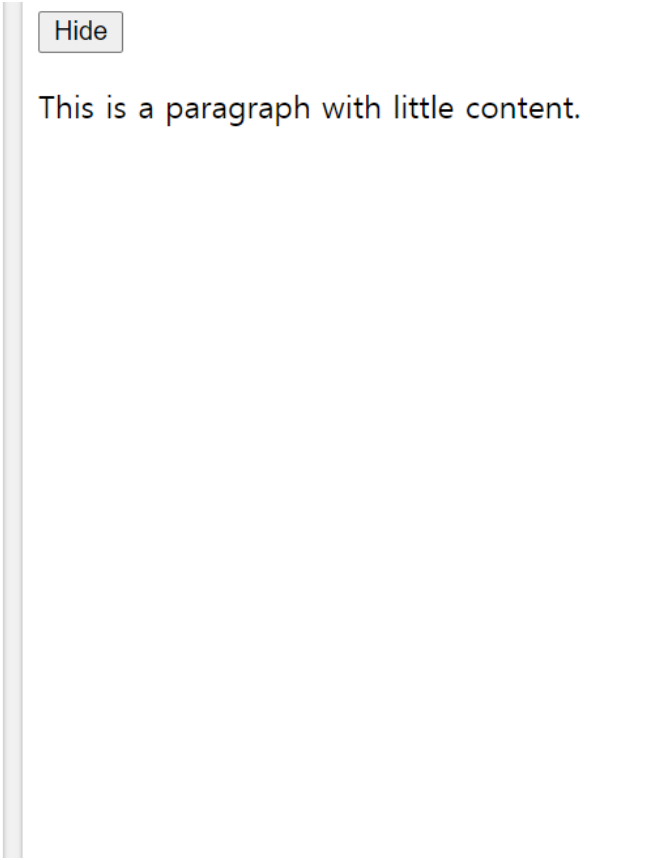
jQuery Effects

- hide() : hide HTML objects
- show(): show HTML objects
- toggle(): toggle between hiding and showing an element
- Syntax
  - $(selector).hide(speed,callback);
  - $(selector).show(speed,callback);
  - $(*selector*).toggle(*speed,callback*);

```
$("button").click(function(){
  $("p").hide(1000);
});
```

jQuery Effects

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide("slow", function(){
      alert("The paragraph is now hidden");
    });
  });
});
</script>
</head>
<body>

<button>Hide</button>

<p>This is a paragraph with little content.</p>

</body>
</html>
```



jQuery HTML

❖ Get/Set content

- text()/text("value"): Sets or returns the text content of selected elements
- html()/html("value"): Sets or returns the content of selected elements (including HTML markup)
- val()/val("value"): Sets or returns the value of form fields

❖ Get/Set attributes

- attr("propertyname")/attr("propertyname", "value"): get/set/change attribute values

❖ Callback function

- The callback function has two parameters: the index of the current element in the list of elements selected and the original (old) (value/attribute value).
- Then return the string you wish to use as the new (value/attribute value) from the function.

jQuery HTML example

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("#btn1").click(function(){
    $("#test1").text(function(i, origText){
      return "Old text: " + origText + " New text: Hello world! (index: "
+ i + ")";
    });
  });

  $("#btn2").click(function(){
    $("#test2").html(function(i, origText){
      return "Old html: " + origText + " New html: Hello <b>world!</b>
(index: " + i + ")";
    });
  });
});
</script>
</head>
<body>

<p id="test1">This is a <b>bold</b> paragraph.</p>
<p id="test2">This is another <b>bold</b> paragraph.</p>

<button id="btn1">Show Old/New Text</button>
<button id="btn2">Show Old/New HTML</button>

</body>
</html>
```

This is a **bold** paragraph.

This is another **bold** paragraph.

Show Old/New Text | Show Old/New HTML

Old text: This is a bold paragraph. New text: Hello world! (index: 0)

This is another **bold** paragraph.

Show Old/New Text | Show Old/New HTML

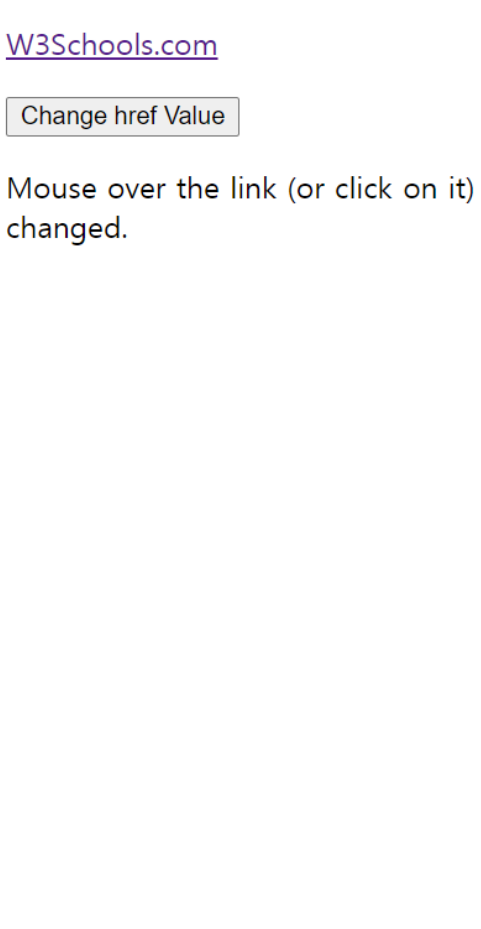
Old text: This is a bold paragraph. New text: Hello world! (index: 0)

Old html: This is another **bold** paragraph. New html: Hello **world!** (index: 0)

Show Old/New Text | Show Old/New HTML

jQuery HTML example

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("#w3s").attr("href", function(i, origValue){
      return origValue + "/jquery/";
    });
  });
});
</script>
</head>
<body>

<p><a href="https://www.w3schools.com" id="w3s">W3Schools.com</a></p>

<button>Change href Value</button>

<p>Mouse over the link (or click on it) to see that the value of the href
attribute has changed.</p>

</body>
</html>
```



jQuery CSS

- **The css() method sets and returns one or more style properties for the selected elements**
- **Syntax**
  - css(“propertyname”);
  - css(“propertyname”, “value”);
  - css({“propertyname1”:“value1”, “propertyname2”:“value2”, …});

JavaScript vs jQuery

❖ jQuery

- var myElement = $("#id01");
- myElement.text("Hello Sweden!");
- $("#id").remove();

❖ JavaScript

- var myElement = document.getElementById("id01");
- myElement.textContent = "Hello Sweden!";
- element.parentNode.removeChild(element);

❖ **More details**

- https://www.w3schools.com/js/js_jquery_selectors.asp

AJAX

What is AJAX?

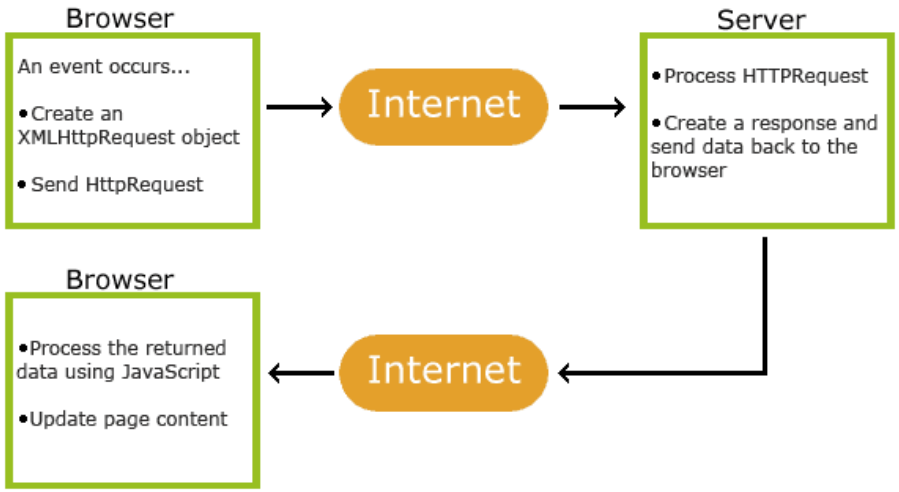
❖ AJAX = Asynchronous JavaScript And XML.

- AJAX is a misleading name. AJAX applications might use XML to transport data, but it is equally common to transport data as plain text or JSON text.

❖ AJAX is not a programming language.

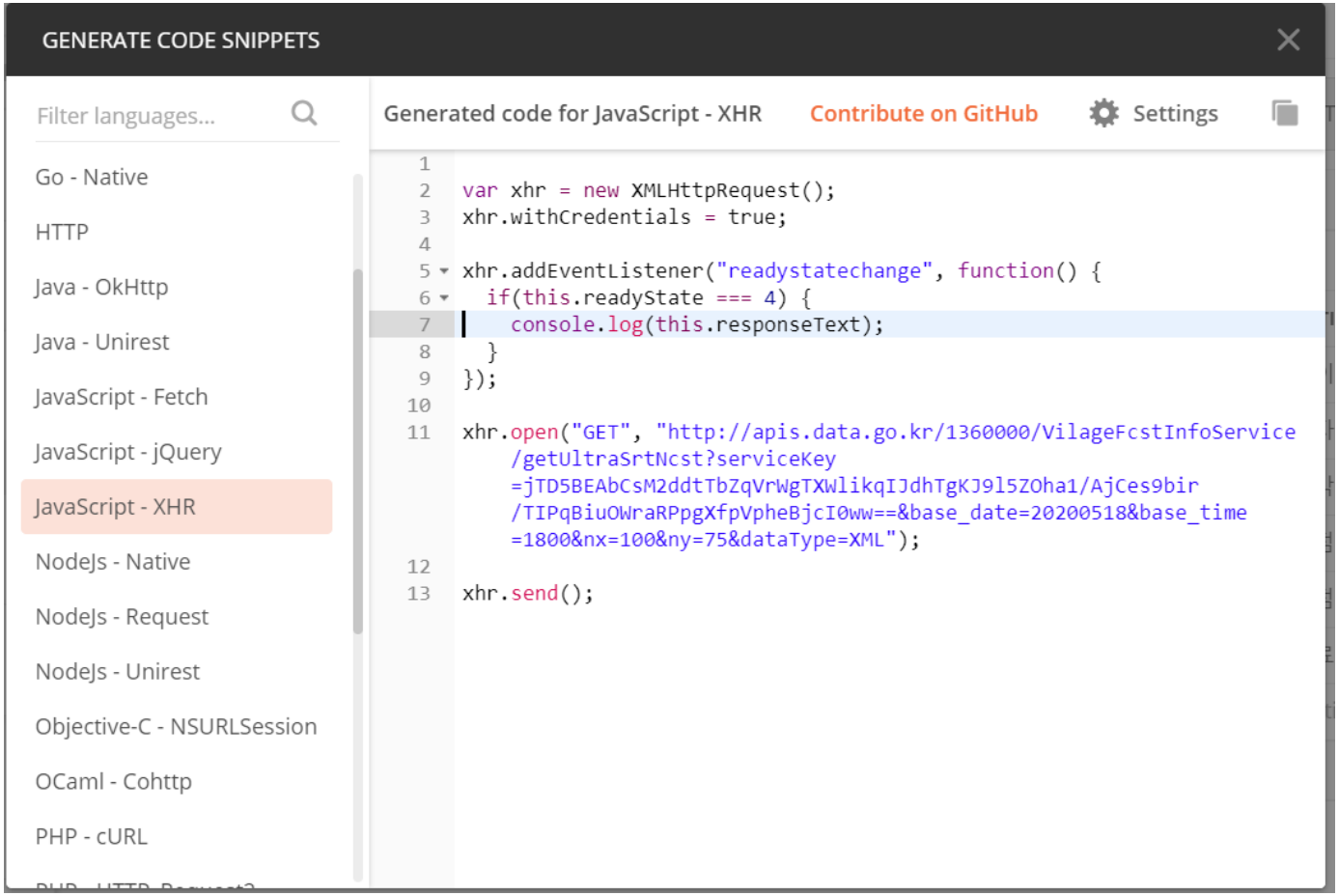
❖ AJAX just uses a combination of:

- A browser built-in XMLHttpRequest object (to request data from a web sever)
- JavaScript and HTML DOM (to display or use the data)



AJAX introduction

- ❖ Read data from a web server - after the page has loaded
- ❖ Update a web page without reloading the page
- ❖ Send data to a web server - in the background



XMLHttpRequest Object

- Can be used to exchange data with a web server behind the scenes. This means that it is possible to update parts of a web page, withouth reloading the whole page.

- XMLHttpRequest Object Methods

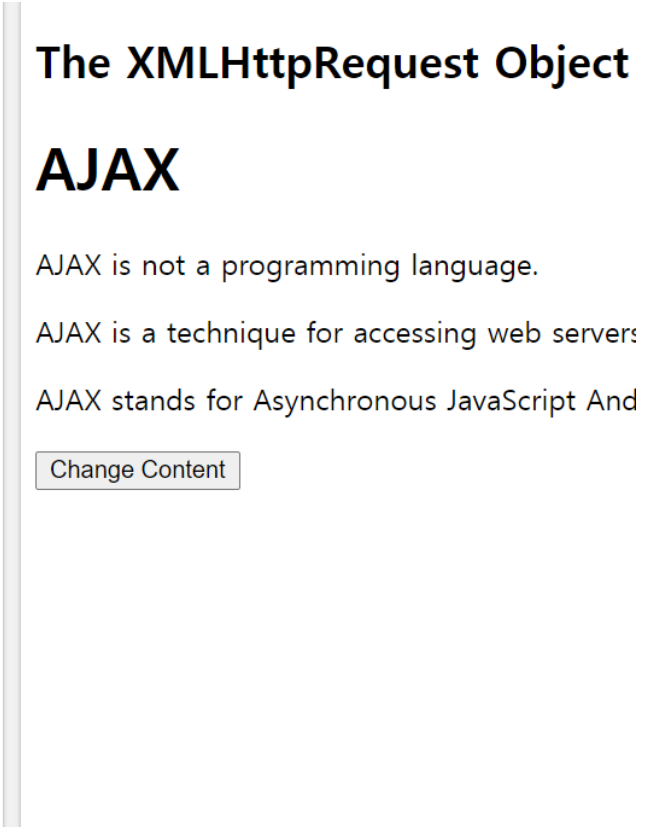
Method	Description
new XMLHttpRequest()	Creates a new XMLHttpRequest object
abort()	Cancels the current request
getAllResponseHeaders()	Returns header information
getResponseHeader()	Returns specific header information
open(*method, url, async, user, psw*)	Specifies the request *method*: the request type GET or POST *url*: the file location *async*: true (asynchronous) or false (synchronous) *user*: optional user name *psw*: optional password
send()	Sends the request to the server Used for GET requests
send(*string*)	Sends the request to the server. Used for POST requests
setRequestHeader()	Adds a label/value pair to the header to be sent

XMLHttpRequest Object

Property	Description
onreadystatechange	Defines a function to be called when the readyState property changes
readyState	Holds the status of the XMLHttpRequest. 0: request not initialized 1: server connection established 2: request received 3: processing request 4: request finished and response is ready
responseText	Returns the response data as a string
responseXML	Returns the response data as XML data
status	Returns the status-number of a request 200: "OK" 403: "Forbidden" 404: "Not Found" For a complete list go to the Http Messages Reference
statusText	Returns the status-text (e.g. "OK" or "Not Found")

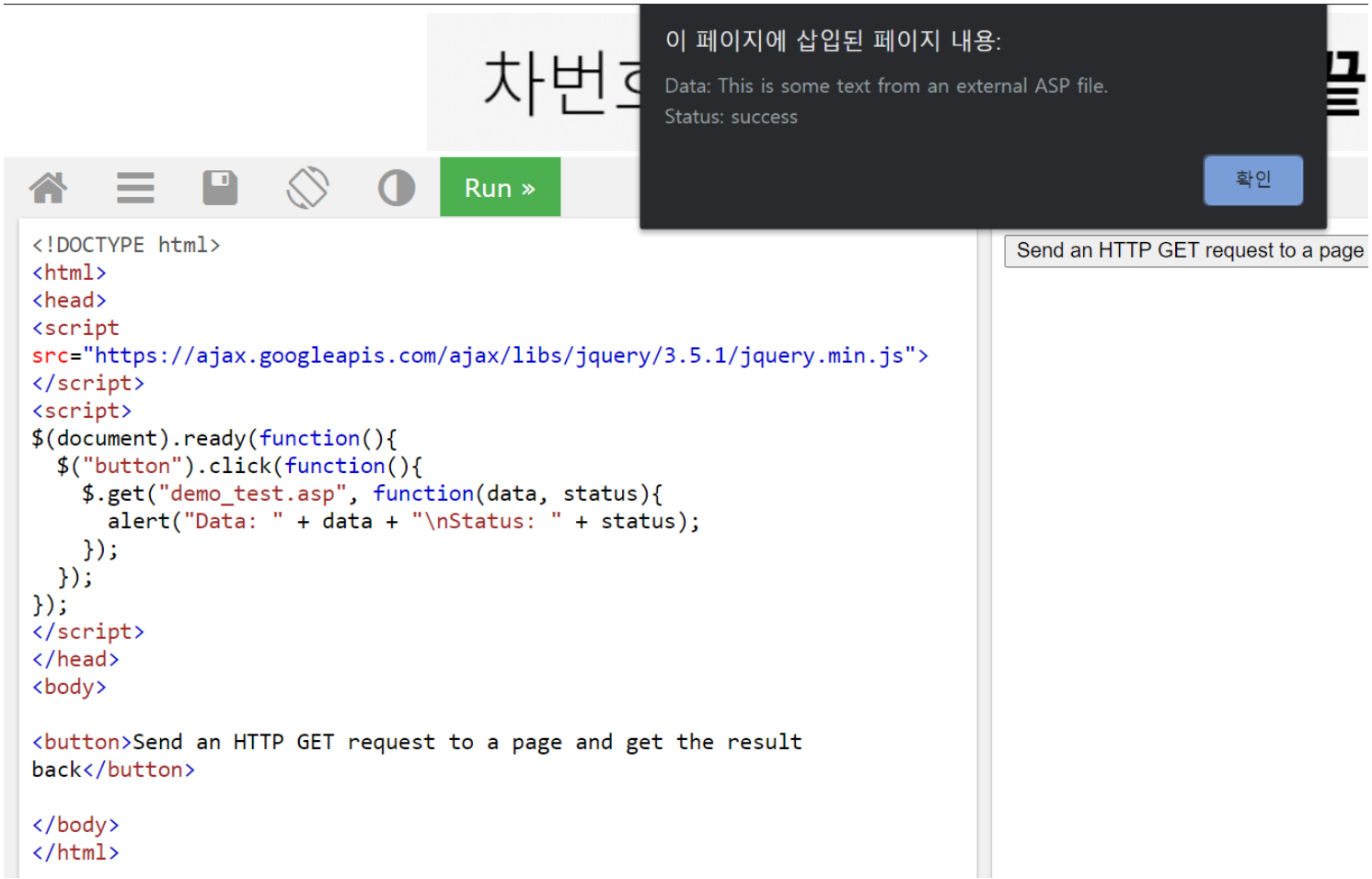
AJAX example

```
<!DOCTYPE html>
<html>
<body>

<h2>The XMLHttpRequest Object</h2>

<p id="demo">Let AJAX change this text.</p>

<button type="button" onclick="loadDoc()">Change Content</button>

<script>
function loadDoc() {
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("demo").innerHTML = this.responseText;
    }
  };
  xhttp.open("GET", "ajax_info.txt", true);
  xhttp.send();
}
</script>

</body>
</html>
```



jQuery and AJAX

- The jQuery get() and post() methods are used to request data from the server with an HTTP GET or POST request.




```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $.get("demo_test.asp", function(data, status){
      alert("Data: " + data + "\nStatus: " + status);
    });
  });
});
</script>
</head>
<body>

<button>Send an HTTP GET request to a page and get the result
back</button>

</body>
</html>
```

JSON

What is JSON?

- JSON stands for **J**ava**S**cript **O**bject **N**otation
- JSON is a lightweight data-interchange format
- JSON is "self-describing" and easy to understand
- JSON is language independent
  - JSON uses JavaScript syntax, but the JSON format is text only. Text can be read and used as a data format by any programming language.
- JavaScript has a built in function to convert a string, written in JSON format, into native JavaScript objects:
  - JSON.parse()

When JSON is used

❖ Exchanging data between a browser and a server

❖ Sending data & receiving data

```
var myObj = {name: "John", age: 31, city: "New York"};
var myJSON = JSON.stringify(myObj);
window.location = "demo_json.php?x=" + myJSON;
```

```
var myJSON = '{"name":"John", "age":31, "city":"New York"}';
var myObj = JSON.parse(myJSON);
document.getElementById("demo").innerHTML = myObj.name;
```

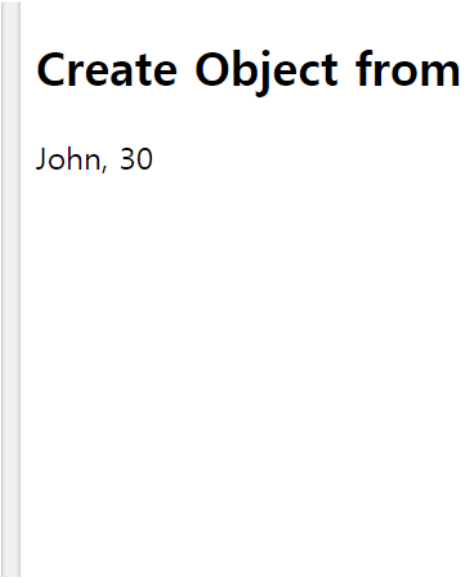
❖ Storing data in local storage

```
// Storing data:
myObj = {name: "John", age: 31, city: "New York"};
myJSON = JSON.stringify(myObj);
localStorage.setItem("testJSON", myJSON);

// Retrieving data:
text = localStorage.getItem("testJSON");
obj = JSON.parse(text);
document.getElementById("demo").innerHTML = obj.name;
```

JSON parse

❖ Use the JavaScript function JSON.parse() to covert text into a JavaScript object

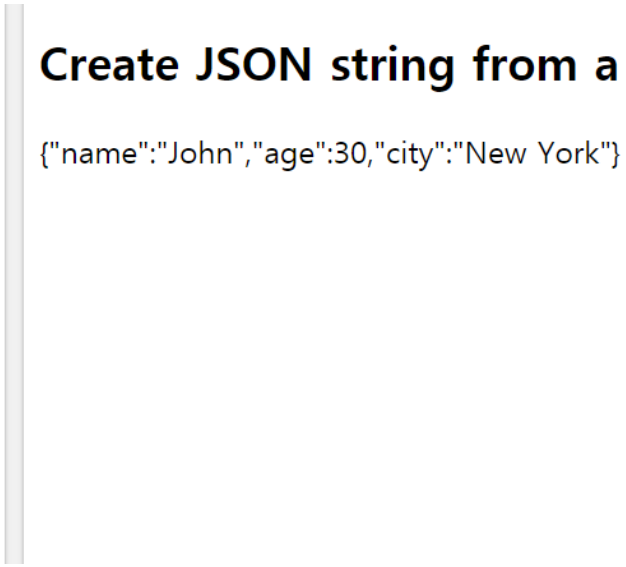
```
<!DOCTYPE html>
<html>
<body>

<h2>Create Object from JSON String</h2>

<p id="demo"></p>

<script>
var txt = '{"name":"John", "age":30, "city":"New York"}'
var obj = JSON.parse(txt);
document.getElementById("demo").innerHTML = obj.name + ", " + obj.age;
</script>

</body>
</html>
```



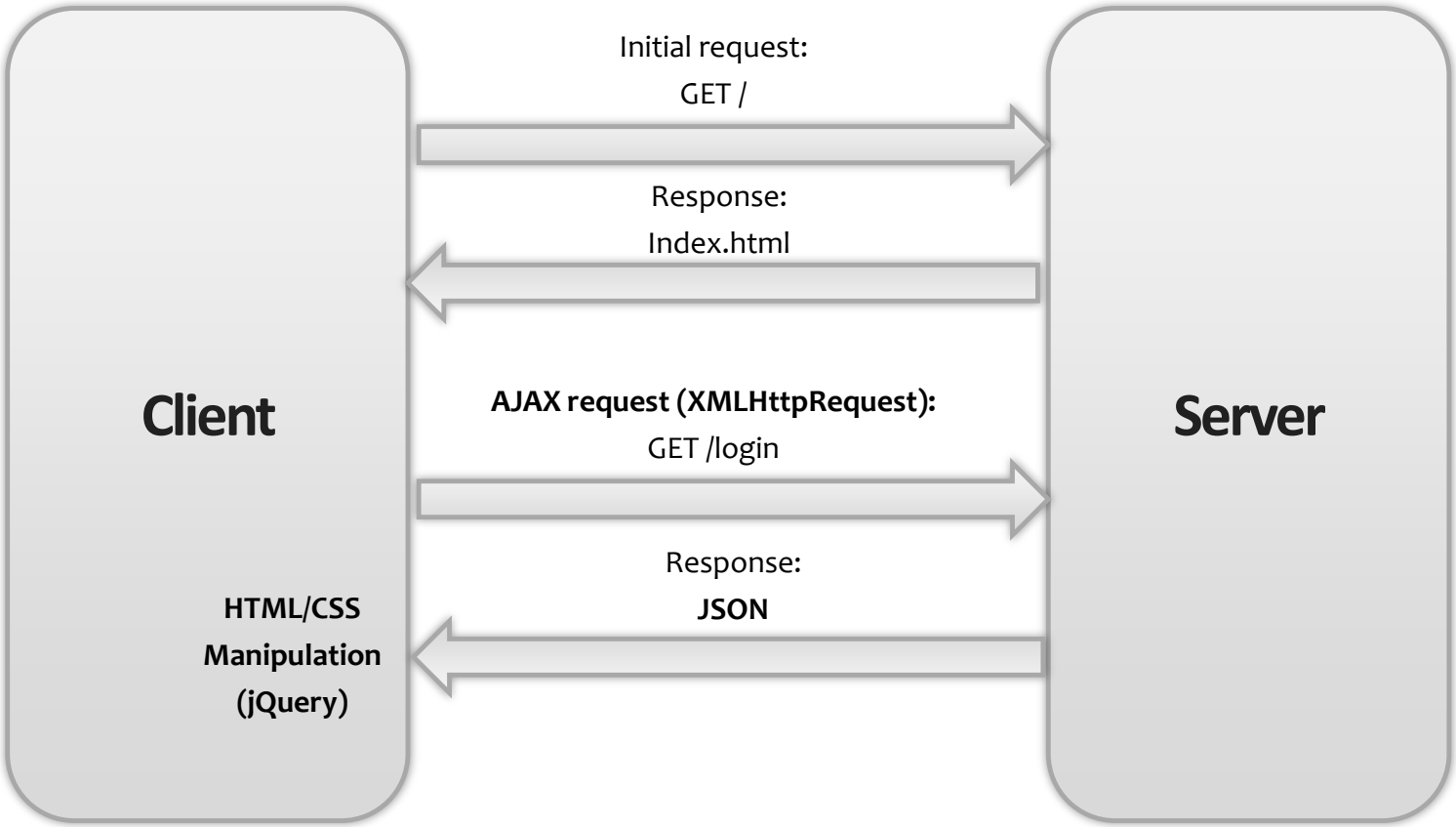
JSON Stringify

- When sending data to a web server, the data has to be a string.
- Convert a JavaScript object into a string with JSON.stringify

```
<!DOCTYPE html>
<html>
<body>

<h2>Create JSON string from a JavaScript object.</h2>

<p id="demo"></p>

<script>
var obj = { name: "John", age: 30, city: "New York" };
var myJSON = JSON.stringify(obj);
document.getElementById("demo").innerHTML = myJSON;
</script>

</body>
</html>
```



요 약


Client
Server
Initial request:
GET /
Response:
Index.html
AJAX request (XMLHttpRequest):
GET /login
Response:
JSON
HTML/CSS
Manipulation
(jQuery)