jQuery - Javascript Library

Computers don’t understand JavaScript — browsers do.

BOARD

jQuery is not a framework

$ is simply an alias for jQuery

- $ **John Resig** in early **2005**
- $ jQuery is a fast, lightweight.
- $ Feature-rich JavaScript library that is based on the principle "write less, do more".
- $ Ajax based application.

Obtain the latest version of jQuery https://jquery.com/

Installing jQuery is as easy as placing it within your web application and using the HTML <script> tag to include it in your pages, like this:

```
<script type="text/javascript" src="scripts/jquery-1.4.js"></script>
```

https://code.jquery.com/
CDN

```
<script src="https://code.jquery.com/jquery-3.6.0.min.js"
integrity="sha256-/xUj+3OJU5yExlq6GSYGSHk7tPXikynS7ogEvDej/m4
=" crossorigin="anonymous"></script>
```

What you can do with jQuery

- ➔ select elements to perform manipulation.
- ➔ manipulate DOM elements and their attributes.
- ➔ implement Ajax to enable asynchronous data exchange between client and server.
- ➔ traverse all around the DOM tree to locate any element.
- ➔ perform multiple actions on an element with a single line of code.
- ➔ create complex CSS animation with fewer lines of code.

How to use jQuery inside my javascript

```
$(selector).action()
```

Will call jQuery to select selector elements

Assign it to event

```
$(document).ready(function(){
  alert("Hello World!");
  $("#blackBox").hide();
  });
```

```
window.addEventListener("load", () => {
  alert("Hello World!");

document.getElementById("blackBox").styl
e.display = "none";
  });
```

```
window.onload = function() {
  alert("Hello World!");
  document.getElementById("blackBox").style.display =
"none";
  };
```


Practice

jQuery Selectors

```
$(selector).click(function (e) {
      // write your code


   });
```

```
$("p") → Element Selector
$("#idname") → #id Selector
$(".className") → .class Selector
```

jQuery – Attributes

```
$("input[name='first_name']").val();
$("input[name='last_name']").val();
$("input[name='email']").val();
$("input[name='phone']").val();
```

Get Content

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

jQuery Add Elements

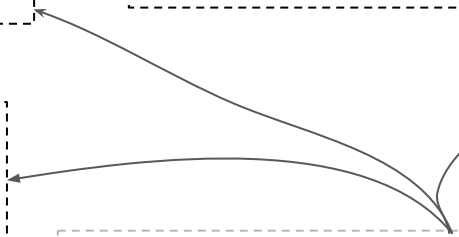
- append() - Inserts content at the end of the selected elements
- prepend() - Inserts content at the beginning of the selected elements
- after() - Inserts content after the selected elements
- before() - Inserts content before the selected elements

jQuery Manipulating CSS

- addClass() - Adds one or more classes to the selected elements
- removeClass() - Removes one or more classes from the selected elements
- toggleClass() - Toggles between adding/removing classes from the selected elements
- css() - Sets or returns the style attribute

jQuery – Remove Elements

- remove() - Removes the selected element (and its child elements)
- empty() - Removes the child elements from the selected element



DOM Manipulation


Practice

jQuery Selector Patterns

Category

```
Find Element

     $('div')                Finds all <div> elements
     $('p'),$('div')         Finds all the <p>, <div> elements
```

```
Finds all the Element

    $(*)         Finds all the elements
    $('div p')   Finds the elements that`s the descendent of div
    $('div >p')  Finds the p tag that is a child of the div tag
```

jQuery provides a number of ways to select a specific DOM element(s)

jQuery Selector Patterns

Category

Find by CSS class name

Finds all the div element that has the class name as firstclass.

```
$('div.firstclass') $('.firstclass')
```

Finds all the elements whose class name is either first class or second class

```
$('.firstclass,.secondclass')
```

Finding the child elements

Finds all p tag elements that are the first child of the element

```
$('p:first-child')
```

Selects the p element tag that is the first type of the parent.

```
$('p:first-of-type')
```

Selects all the p tag elements that is the fourth child of the parent.

```
$('p:nth-child(4)')
```

Selects the p tag elements which is the second last of the parent element.

```
$('p:nth-last-child(2)')
```

Selects the p element which has the last child of their parent.

```
$("p:last-child")
```


Practice

jQuery Effects

The jQuery library provides several techniques for adding animation to a web page. These include simple, standard animations that are frequently used, and the ability to craft sophisticated custom effects.

These effects takes parameters such as:

1. **Speed** – This defines how fast or slow the animation effect should be. Either it can be mentioned as “Slow” , “Normal”, “Fast” or in milliseconds (1000,2000)
2. **Callback function** – It is a custom function that can be applied to add in extra effects as needed or if needed. This function is completely optional.

Code Snippets:

```
$("#demo").hide();        // sets to display: none
$("#demo").show(200);     // shows hidden element with animation (speed)
$("#demo").toggle();      // toggle between show and hide

$( "#element" ).hide( "slow", function() {
// hide with callback function
console.log( "Animation complete." );
});
```

Basics
.hide()
.show()
.toggle()

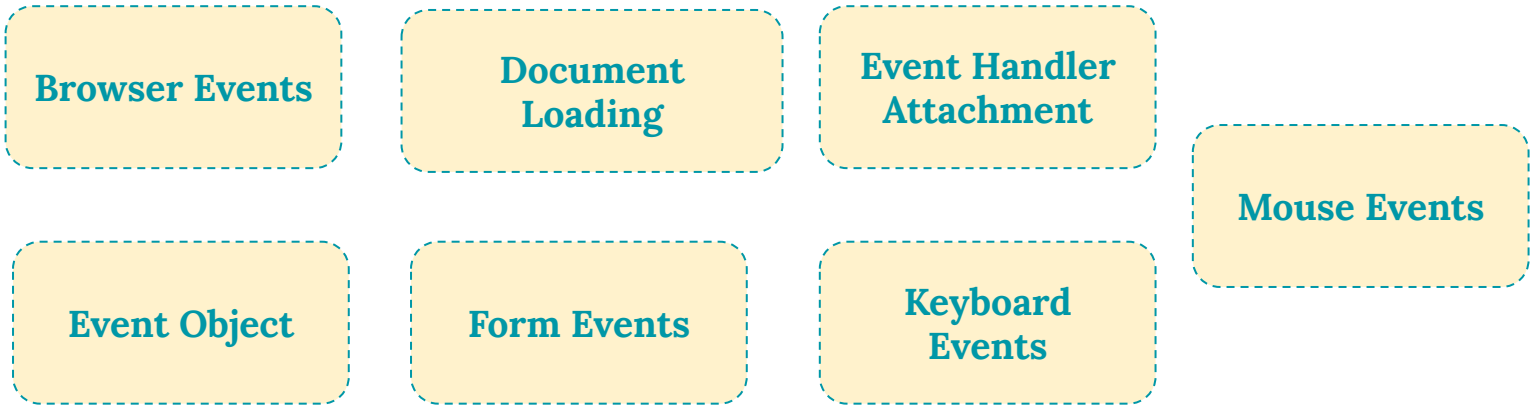
Custom
.animate()
.clearQueue()
.delay()
.dequeue()
jQuery.dequeue()
.finish()
jQuery.fx.interval
jQuery.fx.off
jQuery.speed
.queue()
jQuery.queue()
.stop()

Fading
.fadeIn()
.fadeOut()
.fadeTo()
.fadeToggle()

Sliding
.slideDown()
.slideToggle()
.slideUp()

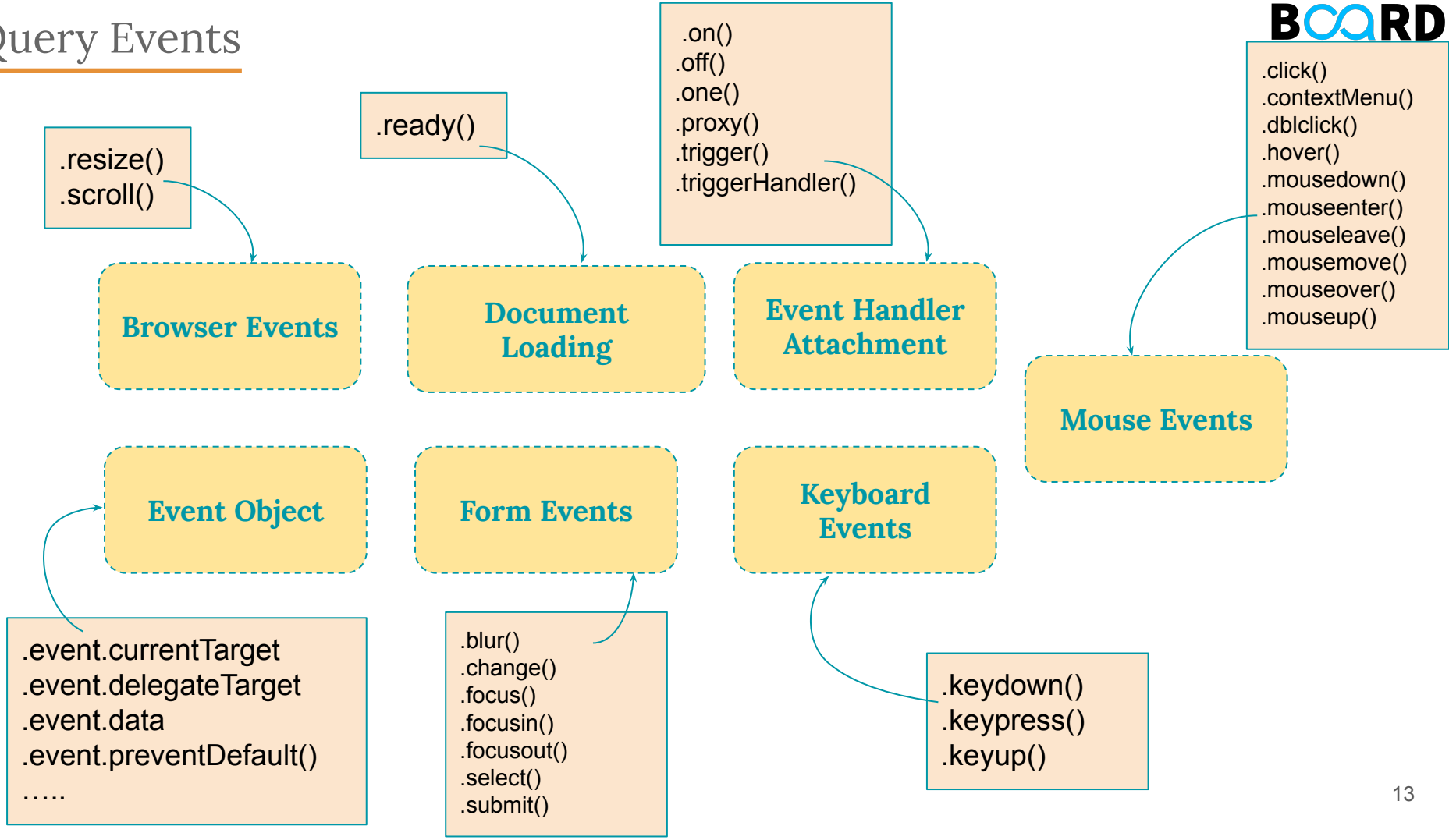
jQuery Events

These methods are used to register behaviors to take effect when the user interacts with the browser, and to further manipulate those registered behaviors.



Get jQuery Syntax : https://j11y.io/jquery/#v=1.11.2&fn=

jQuery Events




Practice

jQuery Exercises

Part 1

Build a list page where users can add and rate examples from a category (e.g. movies, albums, or sports). This page should include the following:

1. A form, where you can add something to the list and rate it.
2. A table of all of the things you've added.
3. A delete button for each row of the table that lets you remove elements from the list.

Note: All DOM manipulation should be done using jQuery - don't use any vanilla JavaScript properties or methods to interact with the DOM!

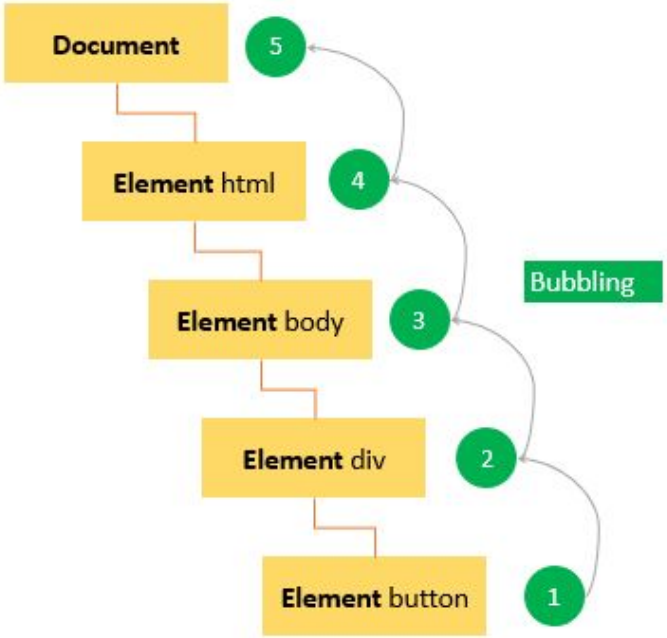
jQuery Exercises

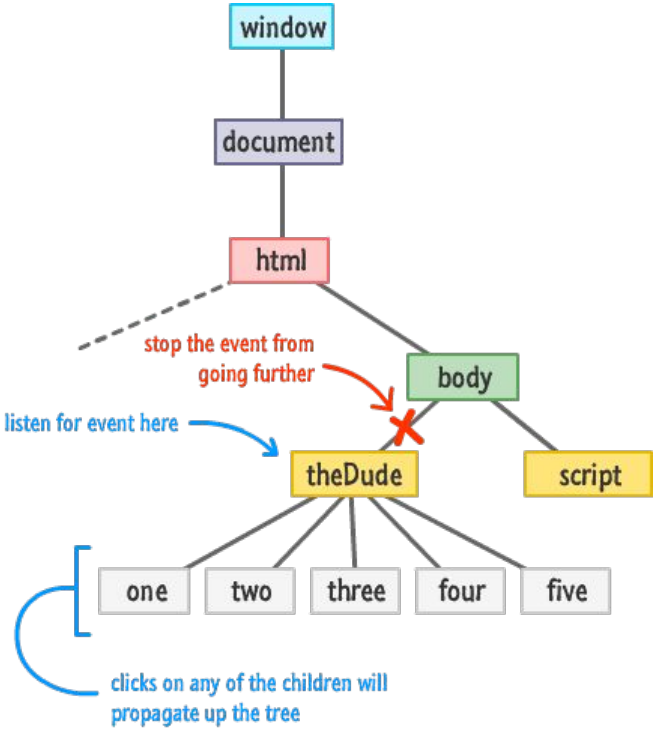
Your Turn

Now that you have learned a bit of jQuery, it's time to refactor an application from vanilla JS to use jQuery!

You can use an application that you have built previously, like the Todo application.

Event Bubbling or Event propagation





More details : https://learn.jquery.com/events/event-delegation/

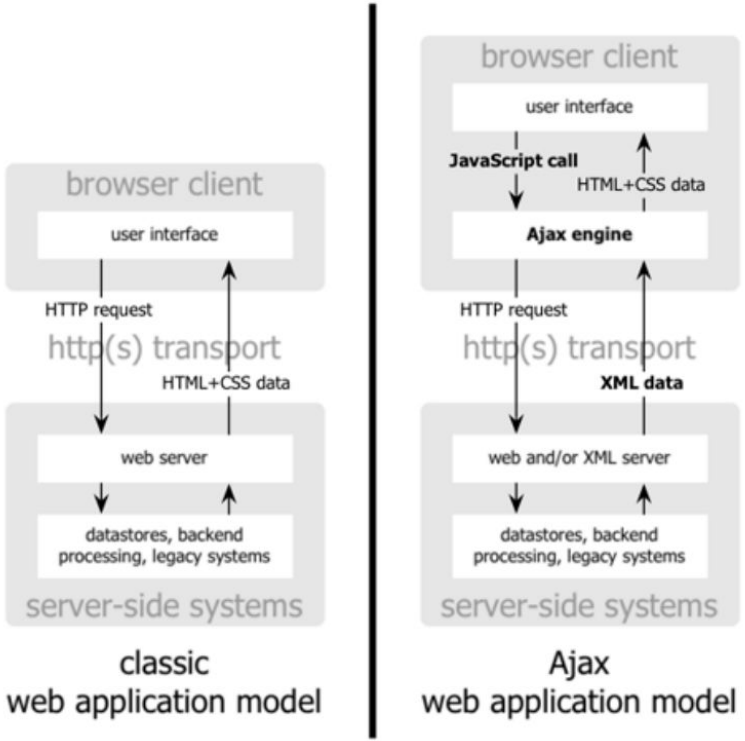
Ajax - Asynchronous JavaScript and XML.

AJAX is the most viable Rich Internet Application (RIA) technology.

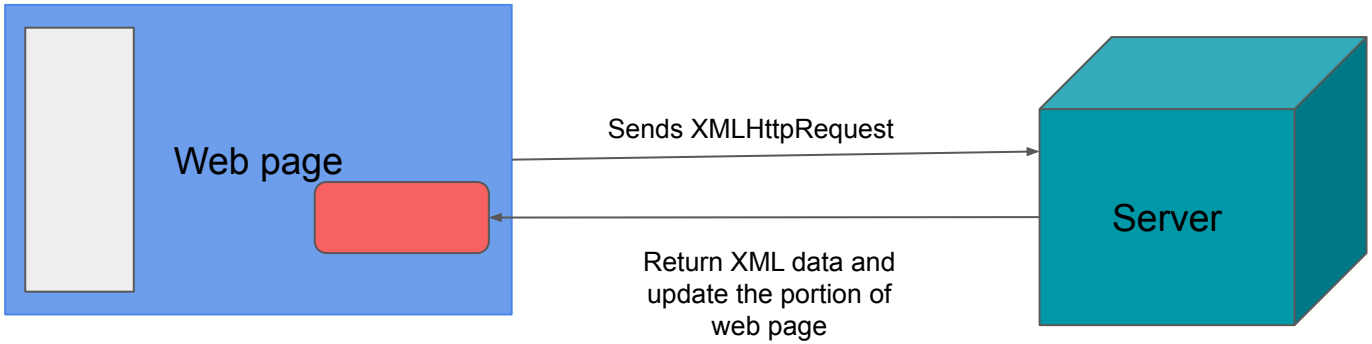
JavaScript includes features of sending asynchronous http request using XMLHttpRequest object.

Ajax is about using this ability of JavaScript to send asynchronous http request and get the xml data as a response (also in other formats) and update the part of a web page (using JavaScript) without reloading or refreshing entire web page.

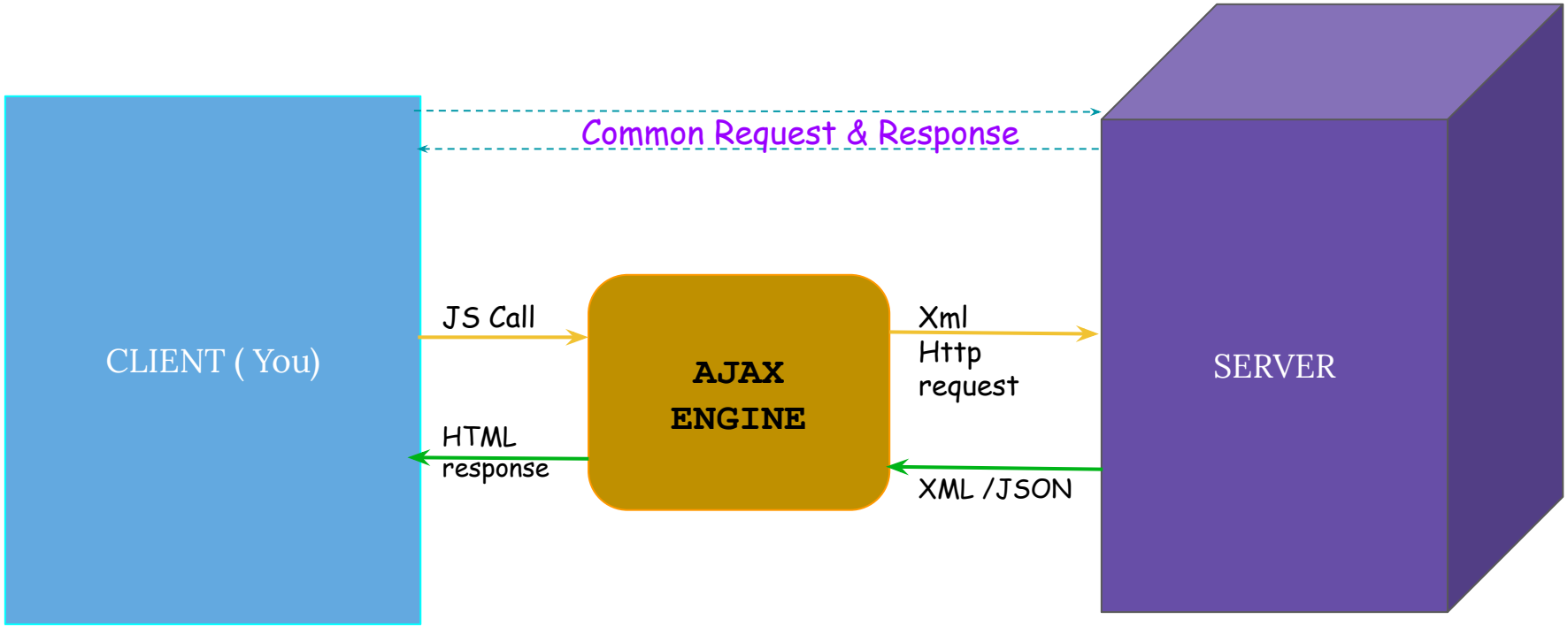
The jQuery library includes various methods to send Ajax requests. These methods internally use XMLHttpRequest object of JavaScript.



Ajax - Asynchronous JavaScript and XML.



How Ajax call works ?



Step 1 – How to make an HTTP request

```
// Old compatibility code, no longer needed.
if (window.XMLHttpRequest) { // Mozilla, Safari, IE7+ ...
    httpRequest = new XMLHttpRequest();
} else if (window.ActiveXObject) { // IE 6 and older
    httpRequest = new ActiveXObject("Microsoft.XMLHTTP");
}
```

1

```
httpRequest.onreadystatechange = nameOfTheFunction;
```

OR

```
httpRequest.onreadystatechange = function(){
    // Process the server response here.
  };
```

2

```
httpRequest.open('GET', 'http://www.example.org/some.file', true);
httpRequest.send();
```

3

Step 1 – How to make an HTTP request

The parameter to the send() method can be any data you want to send to the server if POST-ing the request. Form data should be sent in a format that the server can parse, like a query string:

```
"name=value&anothername="+encodeURIComponent(myVar)+"&so=on"
```

```
httpRequest.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
```

3

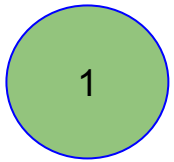
```
httpRequest = new XMLHttpRequest();
if (!httpRequest) {
    alert('Giving up :( Cannot create an XMLHTTP instance');
    return false;
}
httpRequest.onreadystatechange = alertContents;
httpRequest.open('POST', url);
httpRequest.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
httpRequest.send('userName=' + encodeURIComponent(userName));
```

Step 2 – Handling the server response

When you sent the request, you provided the name of a JavaScript function to handle the response:

```
httpRequest.onreadystatechange = nameOfTheFunction;
```

```
if (httpRequest.readyState === XMLHttpRequest.DONE) {
    // Everything is good, the response was received.
} else {
    // Not ready yet.
}
```



`readyState` values

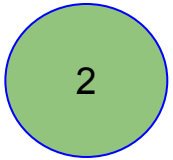
- 0 (uninitialized) or (**request not initialized**)
- 1 (loading) or (**server connection established**)
- 2 (loaded) or (**request received**)
- 3 (interactive) or (**processing request**)
- 4 (complete) or (**request finished and response is ready**)

1. Informational responses (100–199)
2. Successful responses (200–299)
3. Redirects (300–399)
4. Client errors (400–499)
5. Server errors (500–599)

Status Code : https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

Step 2 – Handling the server response

```
if (httpRequest.status === 200) {
    // Perfect!
} else {
    // There was a problem with the request.
    // For example, the response may have a 404 (Not Found)
    // or 500 (Internal Server Error) response code.
}
```



After checking the state of the request and the HTTP status code of the response, you can do whatever you want with the data the server sent. You have two options to access that data:

- **`httpRequest.responseTex`**`t` – returns the server response as a string of text
- **`httpRequest.responseXML`** – returns the response as an **`XMLDocument`** object you can traverse with JavaScript DOM functions.

```
if (httpRequest.readyState === XMLHttpRequest.DONE) {
    if (httpRequest.status === 200) {
        var response = JSON.parse(httpRequest.responseText);
        alert(response.computedString);
    } else {
        alert('There was a problem with the request.');
}
```


Practice

Ajax Method of jQuery

jQuery Ajax Methods	Description
ajax()	Sends asynchronous http request to the server.
get()	Sends http GET request to load the data from the server.
Post()	Sends http POST request to submit or load the data to the server.
getJSON()	Sends http GET request to load JSON encoded data from the server.
getScript()	Sends http GET request to load the JavaScript file from the server and then executes it.
load()	Sends http request to load the html or text content from the server and add them to DOM element(s).

Ajax Events of jQuery

jQuery Ajax Events	Description
ajaxComplete()	Register a handler function to be called when Ajax requests complete.
ajaxError()	Register a handler function to be called when Ajax requests complete with an error.
ajaxSend()	Register a handler function to be called before Ajax request is sent.
ajaxStart()	Register a handler function to be called when the first Ajax request begins.
ajaxStop()	Register a handler function to be called when all the Ajax requests have completed.
ajaxSuccess()	Register a handler function to be called when Ajax request completes successfully.

Ajax Method of jQuery

```
$.ajax({
  type: "POST",
  url: url,
  data: data,
  success: success,
  dataType: dataType

  });
```

```
$.ajax({
    url:'/urlString',
    method: 'GET',

  accepts: application/json,

  async  true,

  beforeSend: function(jqXHR, settings){
        return true;
   },
    cache: true,
    contentType: 'application/x-www-form-urlencoded; charset=UTF-8',

  crossDomain: false,
    data: { firstname: "Board", lastname: "Infinity" },
    dataFilter: function(responseData, dataType){
        return responseData;
  },
```

Ajax Method of jQuery

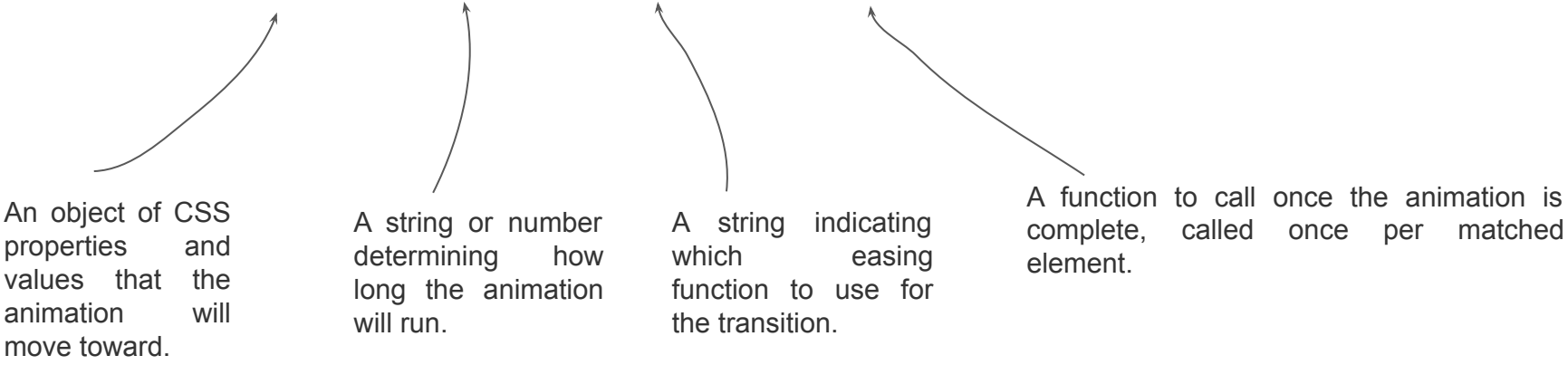
```
dataType: json,
    global: true,
   headers: {},
   statusCode: {
    404: function() {
      alert( "page not found" );
    },
  timeout: 200000
})
.done(function(responseData, textStatus, jqXHR){
// success code
})
.fail(function(jqXHR, textStatus, errorThrown){
    // failure code
})
.always(function(jqXHR, statusText){
        return true;
});
```

jQuery Effects

animate()	Perform custom animation using element's style properties.
queue()	Show or manipulate the queue of functions to be executed on the specified element.
stop()	Stop currently running animations on the specified element(s).
fadeIn()	Display specified element(s) by fading them to opaque.
fadeOut()	Hides specified element(s) by fading them to transparent.
fadeTo()	Adjust the opacity of the specified element(s)
fadeToggle()	Display or hide the specified element(s) by animating their opacity.
hide()	Hide specified element(s).
show()	Display specified element(s).
toggle()	Display hidden element(s) or hide visible element(s).
slideUp()	Hide specified element(s) with sliding up motion.
slideDown()	Display specified element(s) with sliding down motion.
slideToggle()	Display or hide specified element(s) with sliding motion.

jQuery Effects

.animate(properties [, duration] [, easing] [, complete])



An object of CSS properties and values that the animation will move toward.

A string or number determining how long the animation will run.

A string indicating which easing function to use for the transition.

A function to call once the animation is complete, called once per matched element.

```
$( "#clickme" ).click(function()
{
 $( "#book" ).animate({
   opacity: 0.25,
   left: "+=50",
   height: "toggle"
 }, 5000, function() {
   // Animation complete.
 });
});
```

References / Links / Articles / Books /ebooks

→ https://datatables.net/examples/data_sources/ajax
→ https://rapidapi.com/collection/list-of-free-apis
→ https://any-api.com/
→ https://public-apis.io/
→ https://developers.google.com/apis-explorer
→ https://searchapparchitecture.techtarget.com/definition/open-API-public-API
→ https://developer.mozilla.org/en-US/docs/Web/Guide/AJAX/Getting_Started
→ https://courses.cs.washington.edu/courses/cse490h/07sp/readings/ajax_adaptive_path.pdf
→ https://www.digitalocean.com/community/tutorials/submitting-ajax-forms-with-jquery
→ https://restfulapi.net/json-with-ajax/
→ https://morioh.com/p/fa4b8280ddcc

References / Links / Articles / Books /ebooks

https://addyosmani.com/resources/essentialjsdesignpatterns/book/
https://jsbooks.revolunet.com/
https://www.freecodecamp.org/news/javascript-map-reduce-and-filter-explained-with-examples/
https://oscarrobertrodriguez.github.io/exercisesModernDeveloper/javascript-events-in-depth/
Event handlers - https://www.cs.uct.ac.za/mit_notes/web_programming/html/ch15.html
https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Building_blocks/Events
https://eloquentjavascript.net/15_event.html
https://www.kirupa.com/html5/javascript_events.htm
https://www.freecodecamp.org/news/javascript-event-handlers/
https://www.samanthaming.com/tidbits/86-window-location-cheatsheet/
https://digifloat.io/blog/free-3d-illustration-websites/
https://www.codexworld.com/post-get-json-data-from-php-script-jquery-ajax/
https://www.airpair.com/js/jquery-ajax-post-tutorial
https://makitweb.com/how-to-send-get-and-post-ajax-request-with-javascript/