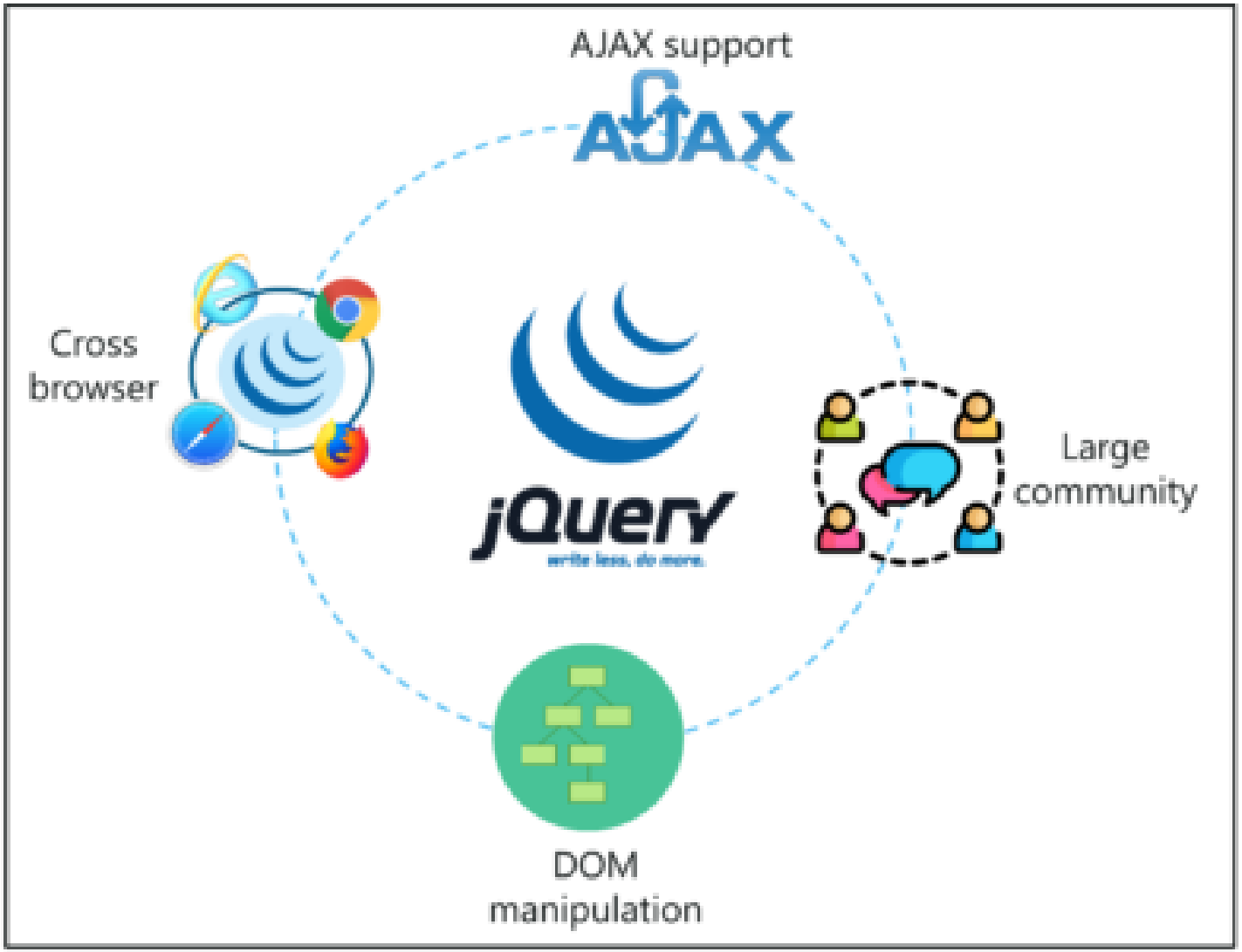
jQuery and AJAX



What is jQuery?

- jQuery simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development.
- jQuery is a JavaScript toolkit designed to simplify various tasks by writing less code.

Core features supported by jQuery −

- DOM manipulation
- Event handling
- AJAX Support
- Animations
- Light weight
- Cross browser support

What is jQuery?

- jQuery simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development.
- jQuery is a JavaScript toolkit designed to simplify various tasks by writing less code.

Here is the list of important core features supported by jQuery −

- **DOM manipulation −** The jQuery made it easy to select DOM elements, negotiate them and modifying their content by using cross-browser open source selector engine called Sizzle.

- **Event handling −** The jQuery offers an elegant way to capture a wide variety of events, such as a user clicking on a link, without the need to clutter the HTML code itself with event handlers.

What is jQuery?

- **AJAX Support** − The jQuery helps you a lot to develop a responsive and featurerich site using AJAX technology.

- **Animations** − The jQuery comes with plenty of built-in animation effects which you can use in your websites.

- **Lightweight** − The jQuery is very lightweight library - about 19KB in size (Minified and gzipped).

- **Cross Browser Support** − The jQuery has cross-browser support, and works well in IE 6.0+, FF 2.0+, Safari 3.0+, Chrome and Opera 9.0+

- **Latest Technology** − The jQuery supports CSS3 selectors and basic XPath syntax.

How to use jQuery?

There are two ways to use jQuery.

- **Local Installation** − You can download jQuery library on your local machine and include it in your HTML code.

```
<head>
<script src="jquery-3.4.1.min.js"></script>
</head>
```

- **CDN Based Version** − You can include jQuery library into your HTML code directly from Content Delivery Network (CDN).

- Google CDN:

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

jQuery Syntax

The jQuery syntax is tailor-made for selecting HTML elements and performing some action on the element(s).

Syntax:

$(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)

Examples:

- $(this).hide( ) - hides the current element.
- $("p").hide( ) - hides all <p> elements.
- $(".test").hide( ) - hides all elements with class="test".
- $("#test").hide( ) - hides the element with id="test".

The Document Ready Event

- All jQuery methods are inside a document ready event:

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

    // jQuery methods go here...

});
```

- This is to prevent any jQuery code from running before the document is finished loading (is ready).
- It is good practice to wait for the document to be fully loaded and ready before working with it.
- This also allows you to have your JavaScript code before the body of your document, in the head section.
- Here are some examples of actions that can fail if methods are run before the document is fully loaded:
  - Trying to hide an element that is not created yet
  - Trying to get the size of an image that is not loaded yet

The Document Ready Event

- All jQuery methods are inside a document ready event:

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

    // jQuery methods go here...

  });
```

OR

```
$(function( )
  {

    // jQuery methods go here...

  });
```

jQuery Selectors

- jQuery selectors allow you to select and manipulate HTML element(s).
- jQuery selectors are used to "find" (or select) HTML elements based on their name, id, classes, types, attributes, values of attributes and much more.
- It's based on the existing CSS Selectors, and in addition, it has some own custom selectors.
- All selectors in jQuery start with the dollar sign and parentheses: $().

jQuery Selectors

The element Selector

- The jQuery element selector selects elements based on the element name.

- You can select all <p> elements on a page like this:
  - $("p")

- Example

When a user clicks on a button, all <p> elements will be hidden:

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

jQuery Selectors

The #id Selector

- The jQuery #id selector uses the id attribute of an HTML tag to find the specific element.
  - $("#test")

- Example
- When a user clicks on a button, the element with id="test" will be hidden:

```
$(document).ready(function( )
 {
     $("button").click(function( )
          {
             $("#test").hide();
           });
   });
```

jQuery Selectors

The .class Selector

- To find elements with a specific class, write a period character, followed by the name of the class:
  - $(".test")

- Example
- When a user clicks on a button, the elements with class="test" will be hidden:

```
$(document).ready(function( )
  {
     $("button").click(function( )
        {
            $(".test").hide();
        });
   });
```

jQuery Exercise

1. Add a list elements within an unordered list element using jQuery.

2. Write a jQuery Code to add a tag at the beginning of the list item, containing the index of the list item.

3. Hide all the input elements within a form.

Test Data :

```
<body>
  <form  name='demo_form'>
  First name: <input type="text" name="fname"> <br>
  Last name: <input type="text" name="lname"> <br>
  <input type="submit" value="Submit">
  </form>
  </body>
```

jQuery Callback Functions

- JavaScript statements are executed line by line.
- However, with effects, the next line of code can be run even though the effect is not finished. This can create errors.
- To prevent this, you can create a callback function.
- A callback function is executed after the current effect is finished.
- Syntax:

  $(selector).hide(speed,callback);

jQuery Callback Functions

- Example:
- a callback parameter that is a function that will be executed after the hide effect is completed:

```
$("button").click(function( )
 {
    $("p").hide("slow", function( )
       {
          alert("The paragraph is now hidden");
        });
  });
```

jQuery Callback Functions

Example:

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/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 with no Callback Functions

- The example below has no callback parameter, and the alert box will be displayed before the hide effect is completed:

- Example without Callback

```
$("button").click(function( )
 {
  $("p").hide(1000);
  alert("The paragraph is now hidden");
});
```

jQuery with no Callback Functions

Example:

```
<!DOCTYPE html>
        <html>
        <head>
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
        <script>
        $(document).ready(function(){
          $("button").click(function(){
            $("p").hide(1000);
            alert("The paragraph is now hidden");
          });
        });
```

jQuery with no Callback Functions

Example:

```
</script>

</head>

<body>

<button>Hide</button>

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

</body>

</html>
```

jQuery Events

- All the different visitors' actions that a web page can respond to are called events.

- An event represents the precise moment when something happens.

Examples:

- moving a mouse over an element
- selecting a radio button
- clicking on an element

The term "fires/fired" is often used with events.

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery hide() and show()

- jQuery hide() : Hides the Syntax or the element of html that you want to hide.

  - $(selector).hide(speed, callback);

- jQuery show() : Shows the syntax or the element of html that you want the user to see

  - $(selector).show(speed, callback);

  - The speed parameter is a optional parameter used for defining the speed of hiding and showing of the content of html.
  - Durations can be specified either using one of the predefined string ‘slow’ or ‘fast’, or in a number of milliseconds, for greater precision; higher values indicate slower animations.

jQuery | Hide/Show, Toggle and Fading methods with Examples

- jQuery hide() and show()

```
<script>
$(document).ready(function( )
{
      $("#hide").click(function( )
        {
        $("p").hide( ;
          });
       $("#show").click(function( )
        {
            $("p").show( );
        });
});
</script>
```

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery toggle( ):

- You can also toggle between hiding and showing an element with the toggle() method.

- Shown elements are hidden and hidden elements are shown

- Syntax:
- $(selector).toggle(speed,callback);

- Example:

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

jQuery Events

- In jQuery, most DOM events have an equivalent jQuery method.

- To assign a click event to all paragraphs on a page, you can do this:

```
$("p").click();
```

- The next step is to define what should happen when the event fires.

- You must pass a function to the event:

```
$("p").click(function( )
    {
      // action goes here!!
    });
```

jQuery Events

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

- click( )

- The click() method attaches an event handler function to an HTML element.

- The function is executed when the user clicks on the HTML element.

- When a click event fires on a <p> element; hide the current <p> element

- Example

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

jQuery Events

- dblclick( )

- The dblclick() method attaches an event handler function to an HTML element.

- The function is executed when the user double-clicks on the HTML element:

- Example

```
$("p").dblclick(function( )
    {
      $(this).hide();
    });
```

jQuery Events

- mouseenter( )

- The mouseenter() method attaches an event handler function to an HTML element.

- The function is executed when the mouse pointer enters the HTML element:

- Example

```
$("#p1").mouseenter(function( )
    {
      alert("You entered p1!");
    });
```

jQuery Events

- mouseleave( )

- The mouseleave() method attaches an event handler function to an HTML element.

- The function is executed when the mouse pointer leaves the HTML element:

- Example

```
$("#p1").mouseleave(function( )
    {
     alert("Bye! You now leave p1!");
    });
```

jQuery Events

- mousedown( )

- The mousedown() method attaches an event handler function to an HTML element.

- The function is executed, when the left, middle or right mouse button is pressed down, while the mouse is over the HTML element:

- Example

```
$("#p1").mousedown(function( )
   {
      alert("Mouse down over p1!");
    });
```

jQuery Events

- mouseup( )

- The mouseup() method attaches an event handler function to an HTML element.

- The function is executed, when the left, middle or right mouse button is released, while the mouse is over the HTML element:

- Example

```
$("#p1").mouseup(function( )
    {
      alert("Mouse up over p1!");
    });
```

jQuery Events

- hover( )
- The hover() method takes two functions and is a combination of mouseenter () and mouseleave() methods.

- The first function is executed when the mouse enters the HTML element, and the second function is executed when the mouse leaves the HTML element:

- Example

```
$("#p1").hover(function( )
   {
      alert("You entered p1!");
    },
    function( )
   {
      alert("Bye! You now leave p1!");
    });
```

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery Fading Methods:

- With jQuery you can fade an element in and out of visibility.

- jQuery has the following fade methods:
  - fadeIn()
  - fadeOut()
  - fadeToggle()
  - fadeTo()

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery fadeIn() Method

- The jQuery fadeIn() method is used to fade in a hidden element.

- Syntax:

```
$(selector).fadeIn(speed,callback);
```

Example:

```
$("button").click(function( )
  {
     $("#div1").fadeIn();
     $("#div2").fadeIn("slow");
     $("#div3").fadeIn(3000);
   });
```

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery fadeOut() Method

- The jQuery fadeOut() method is used to fade out a visible element.

- Syntax:

```
$(selector).fadeOut(speed,callback);
```

Example:

```
$("button").click(function( )
  {
     $("#div1").fadeOut();
     $("#div2").fadeOut("slow");
     $("#div3").fadeOut(3000);
    });
```

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery fadeToggle() Method

- If the elements are faded out, fadeToggle() will fade them in.

- If the elements are faded in, fadeToggle() will fade them out.

- Syntax:

  $(selector).fadeToggle(speed,callback);

- Example:

```
$("button").click(function( )
  {
     $("#div1").fadeToggle();
     $("#div2").fadeToggle("slow");
     $("#div3").fadeToggle(3000);
   });
```

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery fadeTo() Method

- The jQuery fadeTo( ) method allows fading to a given opacity (value between 0 and 1).
- Syntax:

  $(selector).fadeTo(speed,opacity,callback);

- Example:

```
$("button").click(function( )
 {
     $("#div1").fadeTo("slow", 0.15);
     $("#div2").fadeTo("slow", 0.4);
     $("#div3").fadeTo("slow", 0.7);
    });
```

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery Sliding Methods

- With jQuery you can create a sliding effect on elements.

- jQuery has the following slide methods:
  - slideDown()
  - slideUp()
  - slideToggle()

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery slideDown() Method

- The jQuery slideDown() method is used to slide down an element.

- Syntax:

  $(selector).slideDown(speed,callback);

Example

```
$("#flip").click(function( )
{
  $("#panel").slideDown();
});
```

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery slideUp() Method

- The jQuery slideUp() method is used to slide up an element.

- Syntax:

  $(selector).slideUp(speed,callback);

Example

```
$("#flip").click(function( )
{
 $("#panel").slideUp();
});
```

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery slideToggle() Method

- If the elements have been slid down, slideToggle() will slide them up.

- If the elements have been slid up, slideToggle() will slide them down.

- Syntax:

  $(selector).slideToggle(speed,callback);

Example

```
$("#flip").click(function( )
{
  $("#panel").slideToggle();
});
```

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery stop() Method

- The jQuery stop() method is used to stop an animation or effect before it is finished.

- The stop() method works for all jQuery effect functions, including sliding, fading.

- Syntax:

  $(selector).stop(stopAll, goToEnd);

Example

```
$("#stop").click(function( )
{
  $("#panel").stop();
});
```

jQuery Hide/Show, Toggle and Fading methods with Examples

- jQuery stop() Method

- The optional stopAll parameter specifies whether also the animation queue should be cleared or not.

- Default is false, which means that only the active animation will be stopped, allowing any queued animations to be performed afterwards.

- The optional goToEnd parameter specifies whether or not to complete the current animation immediately. Default is false.

- So, by default, the stop() method kills the current animation being performed on the selected element.

jQuery Traversing

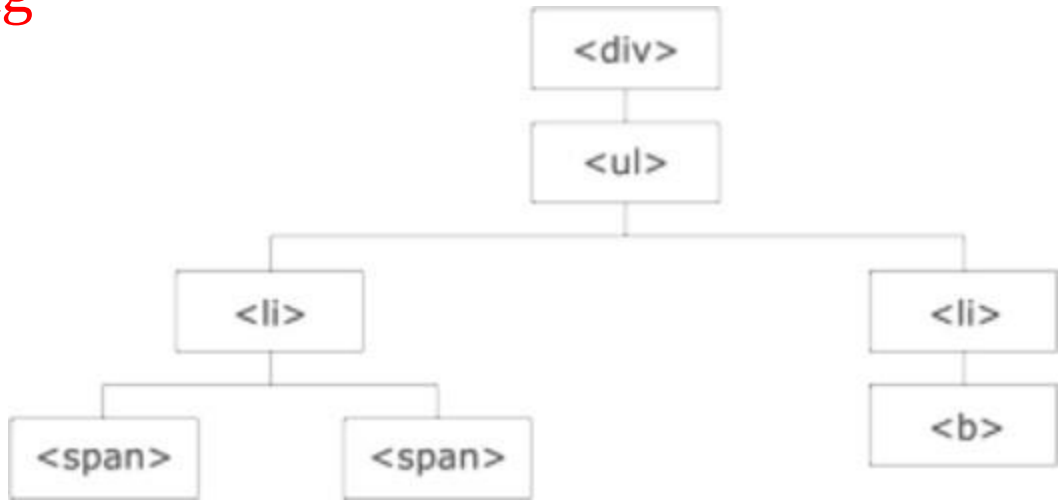
- jQuery traversing, which means "move through", are used to "find" (or select) HTML elements based on their relation to other elements.

- Start with one selection and move through that selection until you reach the elements you desire.

- With jQuery traversing, you can easily move up (ancestors), down (descendants) and sideways (siblings) in the tree, starting from the selected (current) element.

- This movement is called traversing - or moving through - the DOM tree.

jQuery Traversing



- The element is the **parent** of <ul>, and an **ancestor** of everything inside of it
- The <ul> element is the **parent** of both <li> elements, and a **child** of
- The left <li> element is the **parent** of , **child** of <ul> and a **descendant** of
- The element is a **child** of the left <li> and a **descendant** of <ul> and
- The two <li> elements are **siblings** (they share the same parent)
- The right <li> element is the **parent** of <b>, **child** of <ul> and a **descendant** of
- The <b> element is a **child** of the right <li> and a **descendant** of <ul> and

jQuery Traversing the DOM

- jQuery provides a variety of methods that allow us to traverse the DOM.

- The largest category of traversal methods are tree-traversal.

- An ancestor is a parent, grandparent, great-grandparent, and so on.

- Traversing Up the DOM Tree
- Three useful jQuery methods for traversing up the DOM tree are:
  - parent()
  - parents()
  - parentsUntil()

- Traversing Down the DOM Tree
  - children()
  - find()

jQuery Traversing the DOM

- jQuery parent() Method

- The parent() method returns the direct parent element of the selected element.

- This method only traverse a single level up the DOM tree.

- Example

```
$(document).ready(function( )
 {
  $("span").parent();
});
```

- returns the direct parent element of each elements:

jQuery Traversing the DOM

- jQuery parents() Method

- The parents() method returns all ancestor elements of the selected element, all the way up to the document's root element (<html>).

- Example

```
$(document).ready(function( )
    {
      $("span").parents();
    });
```

- returns all ancestors of all elements:

jQuery Traversing the DOM

- jQuery parentsUntil() Method

- The parentsUntil() method returns all ancestor elements between two given arguments.

- Example

```
$(document).ready(function()
 {
   $("span").parentsUntil("div");
 } );
```

- returns all ancestor elements between a and a element

jQuery Traversing the DOM

- jQuery children() Method

- The children() method returns all direct children of the selected element.

- This method only traverses a single level down the DOM tree.

- Example

```
$(document).ready(function( )
{
   $("div").children();
});
```

- returns all elements that are direct children of each elements:

jQuery Traversing the DOM

- jQuery children() Method

- returns all <p> elements with the class name "first", that are direct children of :

- Example

```
$(document).ready(function( )
 {
   $("div").children("p.first");
 });
```

jQuery Traversing the DOM

- jQuery find() Method

- The find() method returns descendant elements of the selected element, all the way down to the last descendant.

- Example

```
$(document).ready(function( )
  {
    $("div").find("span");
  });
```

- returns all elements that are descendants of

jQuery Animations

- jQuery animate() Method

- The jQuery animate() method is used to create custom animations.

- Syntax:

  $(selector).animate({params},speed,callback);

- The required params parameter defines the CSS properties to be animated.

- Example:

```
$("button").click(function()
 {
  $("div").animate({left: '250px'});
 });
```

jQuery CSS Class

- jQuery Manipulating CSS

- jQuery has several methods for CSS manipulation. We will look at the following methods:

  - 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 CSS Class

- jQuery addClass() Method

- Example:

```
$("button").click(function(){
    $("h1, h2, p").addClass("blue");
    $("div").addClass("important");
});
```

OR

- Example:

```
$("button").click(function(){
    $("#div1").addClass("important blue");
});
```

jQuery CSS Class

- jQuery removeClass() Method

- Example:

```
$("button").click(function(){
    $("h1, h2, p").removeClass("blue");
});
```

jQuery CSS Class

- jQuery toggleClass() Method

- Example:

```
$("button").click(function(){
    $("h1, h2, p").toggleClass("blue");
});
```

jQuery data ( ) method

- The data() method is used to attach and get data from the selected elements. It is an inbuilt method in JQuery.
- We can use the removeData() method to remove the data.

- Return data:

  $(selector).data(name)

- Attach data:

  $(selector).data(name, value)

jQuery data ( ) method

- Return data:

```
$(selector).data(name)
```

- The name in the above syntax is an optional parameter.
- It is the data name to retrieve the data. If it is not specified, the data() method returns all stored data for the element as an object.
- This syntax returns the retrieved data for the selected element.

- Attach data:

```
$(selector).data(name, value)
```

- This syntax is used to attach data to selected elements.
- The parameters name and value in the above syntax are the mandatory parameters.
- The name specifies the data name to set, and the value specifies the data value to set.

jQuery data ( ) method

- Example:

```
$("document").ready(function()
{
$("#b1").click(function()
        {
        $("div").data("name", "Welcome to the data method example");
        alert("Data attached");
        }
```

jQuery Utilities

- Jquery provides serveral utilities in the formate of $(name space).
- These methods are helpful to complete the programming tasks, a few of the utility methods are as show below.
  - $.trim( )
  - $.each( )
  - $.inArray( )
  - $.extend( )
  - $.proxy( )
  - $.browser( )
  - $.contains( )
  - $.data( )
  - $.isWindow( )

jQuery Utilities

jQuery trim() method

- The trim() method is used to remove the space, tabs, and all line breaks from the starting and end of the specified string.
- This method does not remove these characters if these whitespace characters are in the middle of the string.

- The commonly used syntax of using this method is given as follows.

jQuery.trim(str)

- The trim() method accepts single parameter str, which is defined as follows.
- str: It is a string that is required to be trimmed.

jQuery Utilities

jQuery each() method

- The each() method in jQuery specifies a function that runs for every matched element.
- The each() accepts a parameter function(index,element) which is a callback function that executes for each selected element.

We can also return false from the callback function to stop the loop early.

Syntax

$(selector).each(function(index, element))

jQuery Utilities

jQuery each() method

- It is one of the widely used traversing methods in JQuery. Using this method, we can iterate over the DOM elements of the jQuery object and can execute a function for every matched element.

Syntax

$(selector).each(function(index, element))

function(index,element): It is a mandatory parameter. It is a callback function that executes for every selected element.

It has two parameter values that are defined as follows.

index: It is an integer value that specifies the index position of the selector.

element: It is the current element. We can use this keyword to refer the currently matched element.

jQuery Utilities

jQuery inArray() method

- The jQuery inArray() method is used to find a specific value in the given array.
- If the value found, the method returns the index value, i.e., the position of the item. Otherwise, if the value is not present or not found, the inArray() method returns -1.

- Syntax

jQuery.inArray(val, arr [, Index])

val: It is the item to search for.

arr: It is an array through which to search.

index: It is the array index at which to start the search.

jQuery Utilities

jQuery isWindow() method

- The isWindow() method is used to test whether the passed argument is a window or not.
- This method returns a Boolean value. If it finds the passed value is a window, it returns true. Otherwise, it returns false.

Syntax

jQuery.isWindow(obj)

obj - It specifies a object to test whether it is a window or not.

jQuery Utilities

jQuery extend() method

- The jQuery extend() method together merges the content of two or more objects into the first object.
- This method returns the merged object.

Syntax

jQuery.extend([deep], target, object1 [, objectN])

jQuery Utilities

jQuery extend() method

This method takes four parameters that are defined as follows.

- deep: It is a Boolean type parameter. If it is specified to true, the merge becomes recursive. Its false value is not supported.
- target: It is an object to extend. It receives the new properties from the passed in additional objects.
- object1: It is an object that contains the additional properties to merge into the target object.
- objectN: It is the additional object that contains the properties to merge into the target object.

If a single parameter is passed to the extend() method, it means that the target parameter is omitted.
In this case, the JQuery object is considered as the target.
If two parameters are passed to the extend() method, properties from all objects added to the target object. The parameters, such as null or undefined, are ignored.

jQuery - AJAX Introduction

- AJAX is the art of exchanging data with a server, and updating parts of a web page - without reloading the whole page.

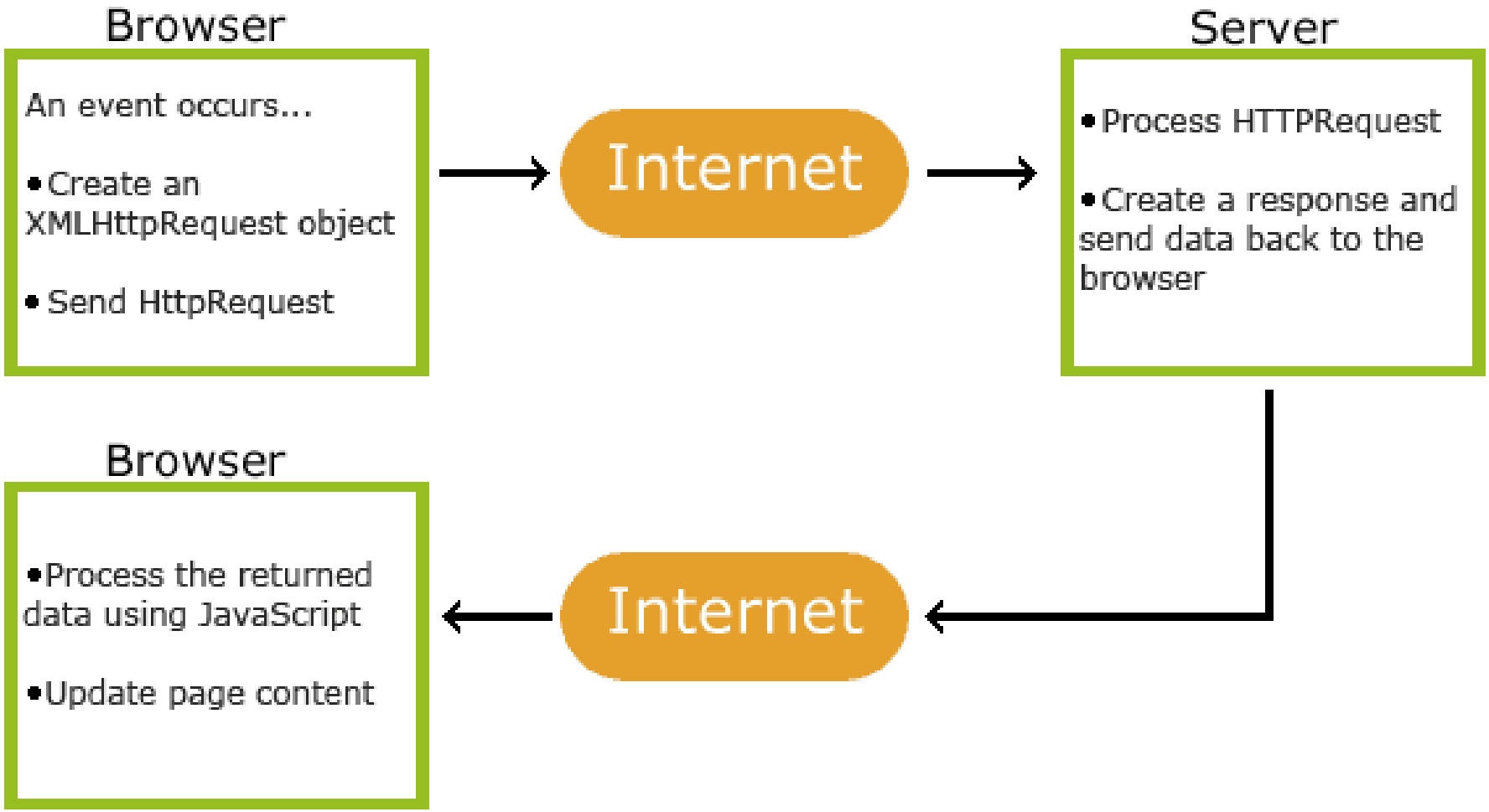
- AJAX = Asynchronous JavaScript And XML.

- AJAX is not a programming language.

- AJAX just uses a combination of:

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

jQuery - AJAX Introduction



jQuery - AJAX Introduction

1. An event occurs in a web page (the page is loaded, a button is clicked)

2. An XMLHttpRequest object is created by JavaScript

3. The XMLHttpRequest object sends a request to a web server

4. The server processes the request

5. The server sends a response back to the web page

6. The response is read by JavaScript

7. Proper action (like page update) is performed by JavaScript

jQuery - AJAX Introduction

AJAX cannot work independently. It is used in combination with other technologies to create interactive webpages.

JavaScript:
Loosely typed scripting language.
JavaScript function is called when an event occurs in a page.
Glue for the whole AJAX operation.

DOM:
API for accessing and manipulating structured documents.
Represents the structure of XML and HTML documents.

CSS:
Allows for a clear separation of the presentation style from the content and may be changed programmatically by JavaScript

XMLHttpRequest:
JavaScript object that performs asynchronous interaction with the server.

jQuery - AJAX Introduction

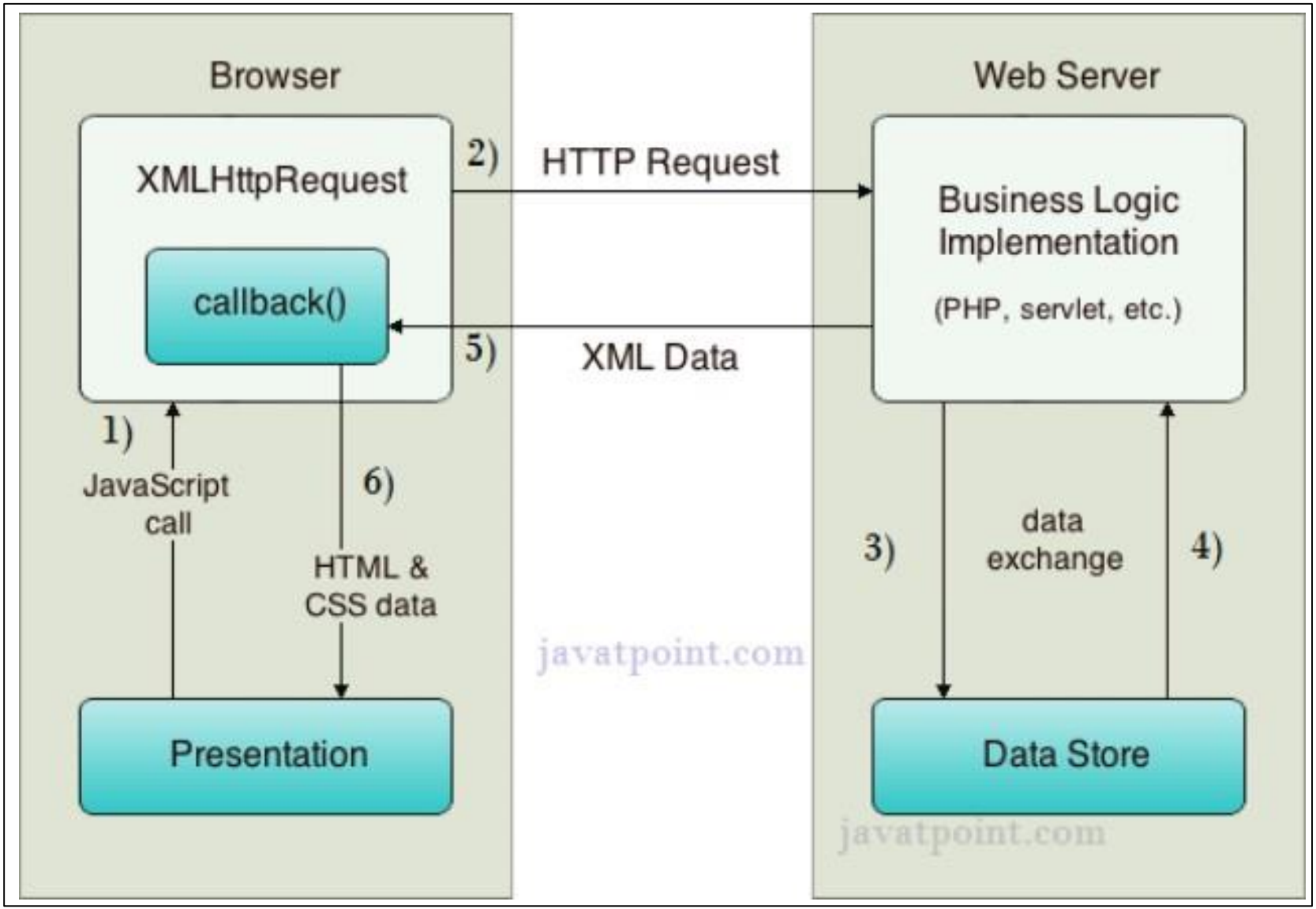
How AJAX works?

- AJAX communicates with the server using XMLHttpRequest object.

1. User sends a request from the UI and a javascript call goes to XMLHttpRequest object.
2. HTTP Request is sent to the server by XMLHttpRequest object.
3. Server interacts with the database using JSP, PHP, Servlet, ASP.net etc.
4. Data is retrieved.
5. Server sends XML data or JSON data to the XMLHttpRequest callback function.
6. HTML and CSS data is displayed on the browser.

jQuery - AJAX Introduction

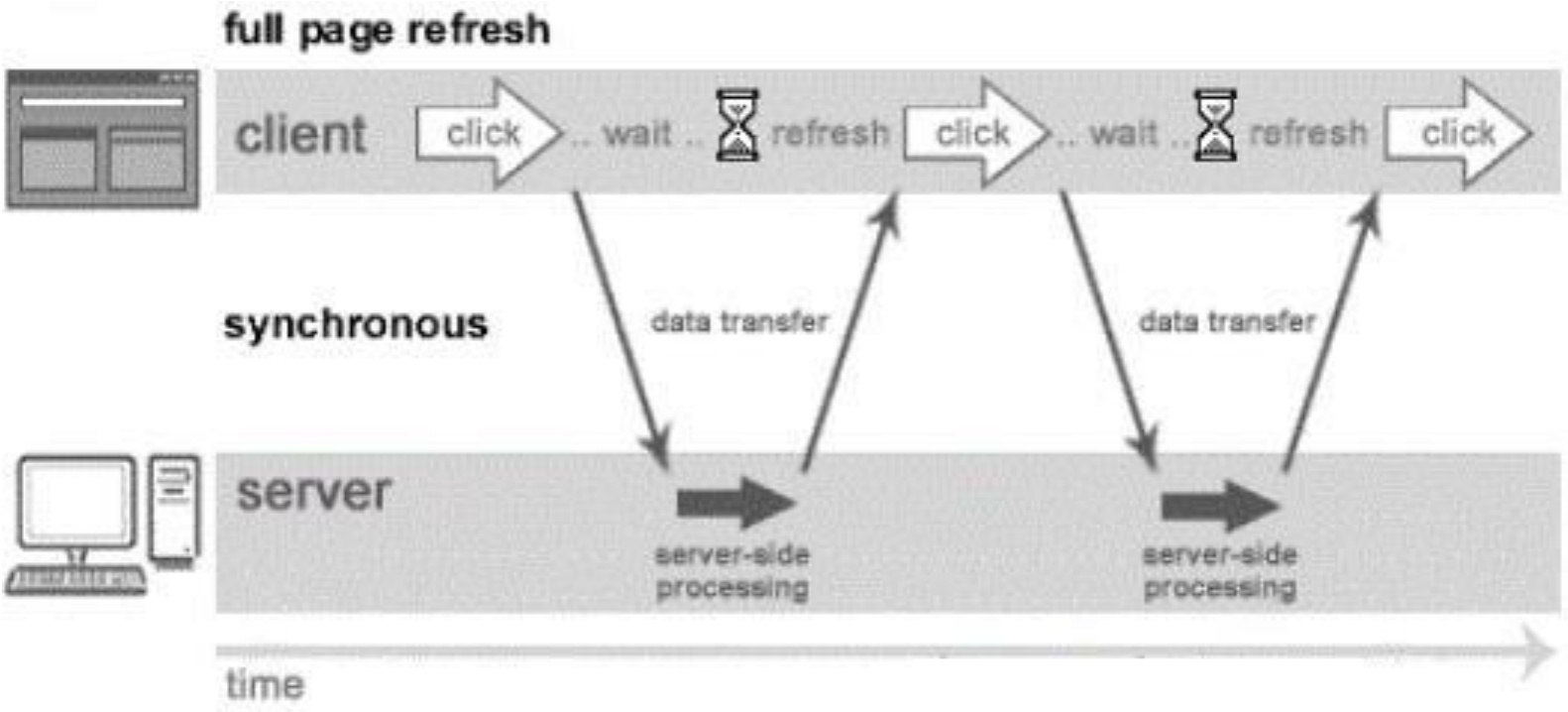
How AJAX works?



jQuery - AJAX Introduction

Understanding Synchronous vs Asynchronous

- Synchronous (Classic Web-Application Model): A synchronous request blocks the client until operation completes i.e. browser is unresponsive.
- In such case, javascript engine of the browser is blocked.

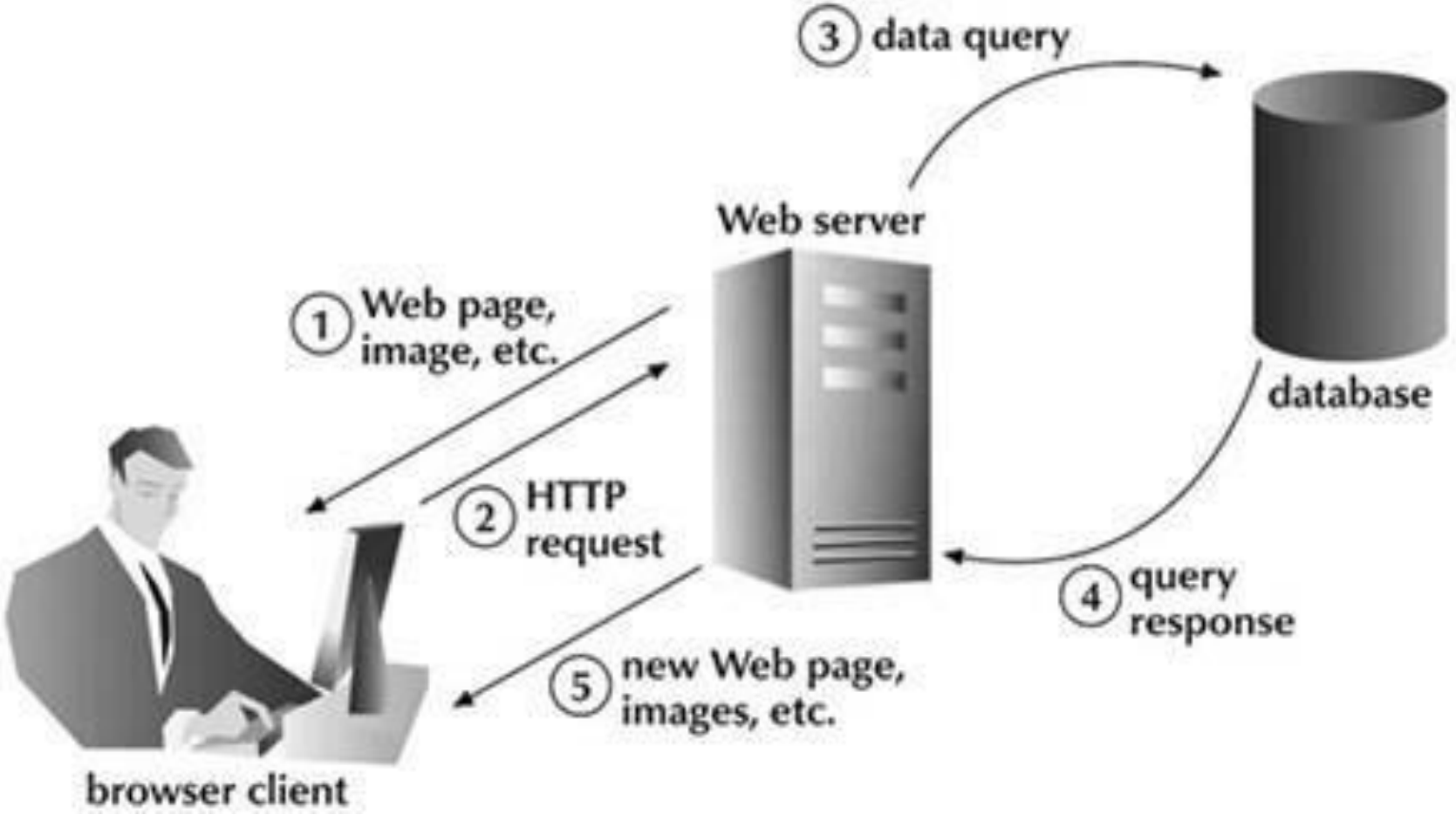


- Full page is refreshed at request time and user is blocked until request completes.

jQuery - AJAX Introduction

Understanding Synchronous vs Asynchronous

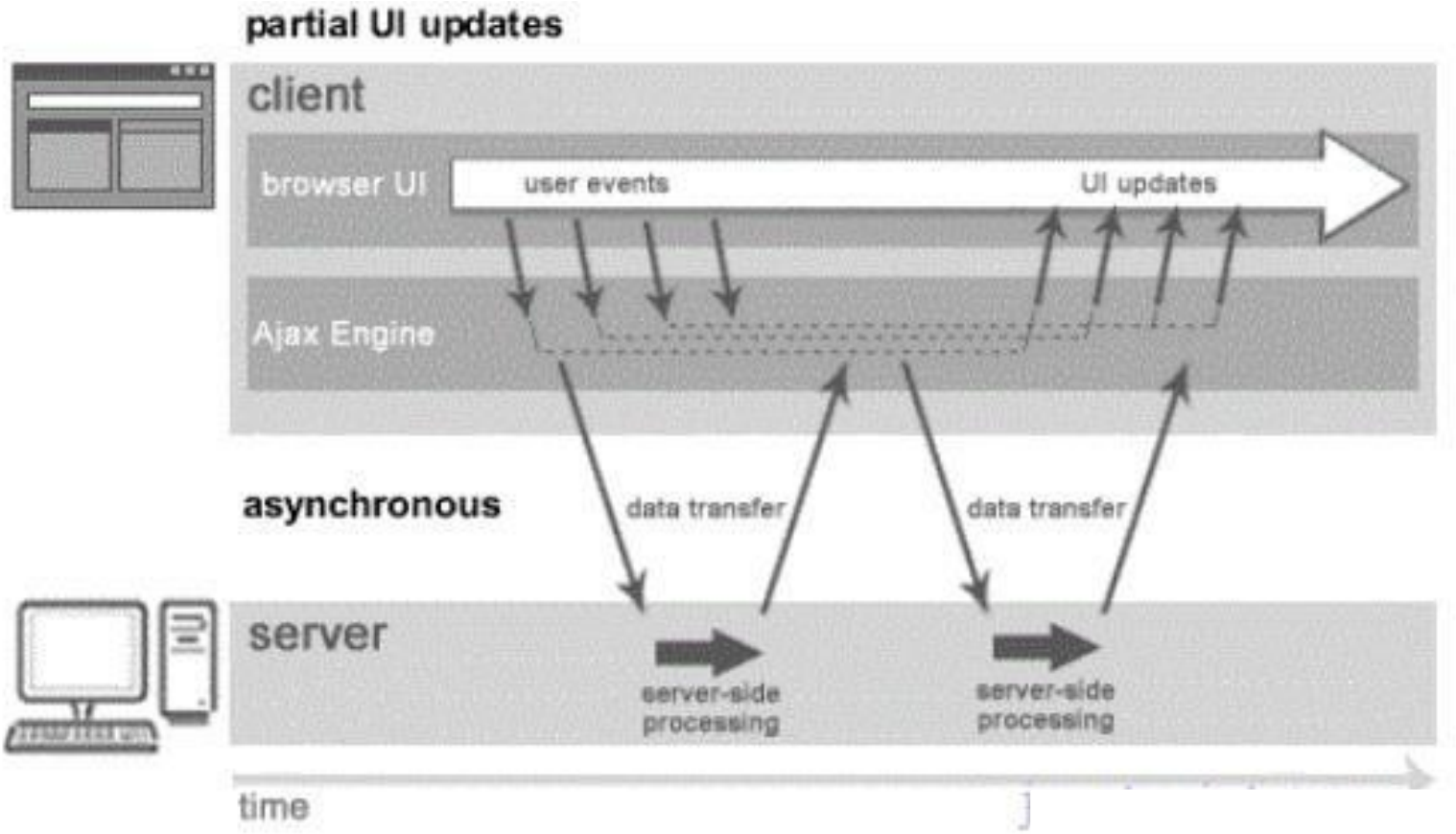
- Synchronous (Classic Web-Application Model):



jQuery - AJAX Introduction

Understanding Synchronous vs Asynchronous

- An asynchronous request doesn't block the client i.e. browser is responsive.
- At that time, user can perform another operations also.
- In such case, javascript engine of the browser is not blocked.

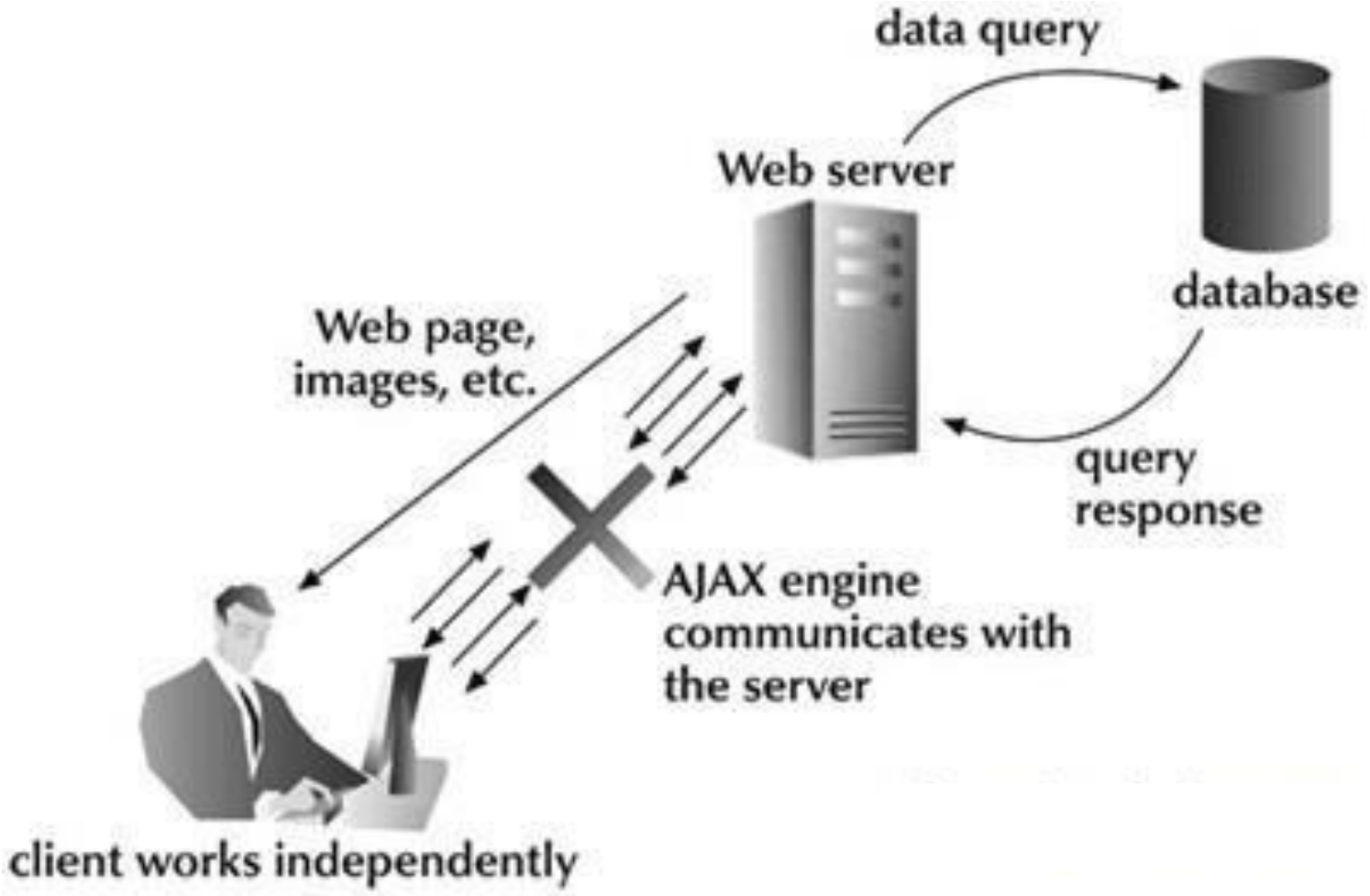


- full page is not refreshed at request time and user gets response from the ajax engine.

jQuery - AJAX Introduction

Understanding Synchronous vs Asynchronous

Asynchronous (Classic Web-Application Model):



jQuery - AJAX Introduction

AJAX Events

• Ajax requests produce a number of different events that you can subscribe to.

Local Events:

•These are callbacks that you can subscribe to within the Ajax request object, like so:

```
$.ajax({
beforeSend: function(){
// Handle the beforeSend event
},
complete: function(){
// Handle the complete event
}
// ......
});
```

jQuery get( ) Method

- The jQuery get() method sends asynchronous http GET request to the server and retrieves the data.

Syntax:

- $.get(url, [data],[callback]);

Parameters Description:

- url: request url from which you want to retrieve the data
- data: data to be sent to the server with the request as a query string
- callback: function to be executed when request succeeds

Example

```
$.get('/data.txt',  // url
      function (data, textStatus, jqXHR) {  // success callback
          alert('status: ' + textStatus + ', data:' + data);
    });
```

jQuery post() Method

- The jQuery post() method sends asynchronous http POST request to the server to submit the data to the server and get the response.

- Syntax:

$.post(url,[data],[callback],[type]);

Parameter Description:

- url: request url from which you want to submit & retrieve the data.
- data: json data to be sent to the server with request as a form data.
- callback: function to be executed when request succeeds.
- type: data type of the response content.

jQuery post() Method

Example: jQuery post() Method

```
$.post('/jquery/submitData',   // url
      {
            myData: 'This is my data.'
      },                                                    // data to be submit
      function(data, status, jqXHR)
    {                                                       // success callback
            $('p').append('status: ' + status + ', data: ' + data);
       })

<p>   </p>
```

jQuery getJSON() Method

- The jQuery getJSON() method sends asynchronous http GET request to the server and retrieves the data in JSON format by setting accepts header to application/json, text/javascript.

- This is same as get() method, the only difference is that getJSON() method specifically retrieves JSON data whereas get() method retrieves any type of data.

- Syntax:

$.getJSON(url,[data],[callback]);

jQuery getJSON() Method

Parameter Description:

- url: request url from which you want to retrieve the data
- data: JSON data to be sent to the server as a query string
- callback: function to be executed when request succeeds
- The following example shows how to retrieve JSON data using getJSON() method.

- Example: jQuery getJSON() Method

```
$.getJSON('/jquery/getjsondata', {name:'Steve'}, function (data, textStatus, jqXHR)
{
    $('p').append(data.firstName);
});
<p></p>
```

jQuery getScript() Method

- The jQuery getScript() method sends http GET request to the server, retrieves the JavaScript file and then executes it.
- Internally, jQuery getScript() method calls get() method and sets dataType to script.

Syntax:

$.getScript(url, [callback]);

Parameter Description:

url: request url from which you want to download JavaScript file

callback: function to be executed when request succeeds

Example: jQuery getScript() Method

```
$.getScript('/Scripts/myJavaScriptFile.js', function(script,status,jqxhr)
{
        alert(status);
    });
```

jQuery load() Method

- The jQuery load() method allows HTML or text content to be loaded from a server and added into a DOM element.

Syntax:
$.load(url,[data],[callback]);

Parameters Description:

url: request url from which you want to retrieve the content
data: JSON data to be sent with request to the server.
callback: function to be executed when request succeeds

Example: Load HTML Content

$('#msgDiv').load('/demo.html');

jQuery - AJAX Introduction

The XMLHttpRequest Object

- All modern browsers support the XMLHttpRequest object.
- The XMLHttpRequest object can be used to exchange data with a server behind the scenes.
- This means that it is possible to update parts of a web page, without reloading the whole page.
- Syntax:

  variable = new XMLHttpRequest();

jQuery - AJAX Introduction

The XMLHttpRequest Object

Example:

```
<!DOCTYPE html>
<html>
<body>
<div id="demo">
<h1>The XMLHttpRequest Object</h1>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>
```

jQuery - AJAX Introduction

```
<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 - AJAX Introduction

Methods of XMLHttpRequest Object

- The important methods of XMLHttpRequest object are as follows:

Method	Description
void open(method, URL)	opens the request specifying get or post method and url.
void open(method, URL, async)	same as above but specifies asynchronous or not.
void open(method, URL, async, username, password)	same as above but specifies username and password.
void send()	sends get request.
void send(string)	send post request.
setRequestHeader(header,value)	it adds request headers.

jQuery - AJAX Introduction

AJAX Events

1. ajaxStart(): The ajaxStart() event is fired whenever an AJAX request begins.
   $(document).ajaxStart(function(){})

2. ajaxSend(): The ajaxSend() event is fired before an AJAX request is sent.
   $(document).ajaxSend(function(function (event, jqxhr, settings){})

3. ajaxSuccess(): The ajaxSuccess() event is fired when an AJAX request completes successfully.
   $(document).ajaxSuccess (function(event, jqxhr, settings){})

4. ajaxComplete(): The ajaxComplete() event is fired when an AJAX request completes.
   $(document).ajaxComplete (function(event, jqxhr, settings){})

jQuery - AJAX Introduction

AJAX Events

5. ajaxStop() : The ajaxStop() event is fired when all AJAX requests have completed.
   $(document).ajaxStop (function(){})

6. ajaxError(): The ajaxError() event is fired when an error is encountered during an AJAX call.
   $(document).ajaxError (function(event, jqxhr, settings, thrownError){})

jQuery - AJAX Introduction

What is JSON in AJAX?

JSON stands for JavaScript Object Notation.

In AJAX, it is used to exchange data between a browser and a server. It is easy to understand, and data exchange is faster than XML.

It supports array, object, string, number, and values.

Example:

```
request.onreadystatechange   = function(){
     if (request.readyState == 4  )
     {
           var jsonObj = JSON.parse(request.responseText);
                        //JSON.parse() returns JSON object
        document.getElementById("date").innerHTML = jsonObj.date;
        document.getElementById("time").innerHTML = jsonObj.time;
     }
  }
```

Advantages of AJAX

- Speed

Reduce the server traffic in both side request. Also reducing the time consuming on both side response.

- Interaction

AJAX is much responsive, whole page(small amount of) data transfer at a time.

- XMLHttpRequest

XMLHttpRequest has an important role in the Ajax web development technique. XMLHttpRequest is special JavaScript object that was designed by Microsoft. XMLHttpRequest object call as a asynchronous HTTP request to the Server for transferring data both side. It's used for making requests to the non-Ajax pages.

- Asynchronous calls

AJAX make asynchronous calls to a web server. This means client browsers are avoid waiting for all data arrive before start the rendering.

Advantages of AJAX

- Form Validation

This is the biggest advantage. Form are common element in web page. Validation should be instant and properly, AJAX gives you all of that, and more.

- Bandwidth Usage

No require to completely reload page again. AJAX is improve the speed and performance. Fetching data from database and storing data into database perform background without reloading page.

Disadvantages of AJAX

1. AJAX application would be a mistake because search engines would not be able to index an AJAX application.

2. Open Source: View source is allowed and anyone can view the code source written for AJAX.

3. ActiveX requests are enabled only in Internet Explorer and newer latest browser.

4 The XMLHttpRequest object itself. For a security reason you can only use to access information from the web host that serves initial pages. If you need to fetching information from another server, it's is not possible with in the AJAX.