**JQUERY**

1)$(document).ready(function(){  
  
  *//jQuerymethodsgohere...*   
  
 *//Also written as $(function(){});*

});

Ans. This function is used to prevent the jquery codes from running before the loading of entire html document with their elements or tags. Reasons are:-

a)trying to hide an html element using jquery, where the element is not loaded yet, then the jquery code becomes invalid

b)if we try to get the size or h1 tag of element which is not yet loaded

2)$("p").click(function(){ //$('selector').action() selector is used to select  //or find html elements ex.p , h1 are selectors.

//$ is used to select the html tags like ‘p’ or ‘h1’ or ‘this’

2) $(*selector*).getJSON(*url,data,success(data,status,xhr))*

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *url* | Required. Specifies the url to send the request to |
| *data* | Optional. Specifies data to be sent to the server |
| *success(data,status,xhr)* | Optional. Specifies the function to run if the request succeeds Additional parameters:  *data* - contains the data returned from the server.  *status* - contains a string containing request status ("success", "notmodified", "error", "timeout", or "parsererror").  *xhr* - contains the XMLHttpRequest objec |

3)Extend function is used to merge objects.

**var** object1 = {

banana: { weight: 52, price: 100 },

cherry: 97

};

**var** object2 = {

banana: { price: 200 },

durian: 100

};

*// Merge object2 into object1*

$.extend( object1, object2 );

1. The prepend() method inserts specified content at the beginning of the selected elements.
2. The find() method returns descendant elements of the selected element.A descendant is a child, grandchild, great-grandchild, and so on.**The DOM tree:** This method traverse downwards along descendants of DOM elements, all the way down to the last descendant. To only traverse a single level down the DOM tree (to return direct children), use the [children()](https://www.w3schools.com/jquery/traversing_children.asp) method.

6)Modal Details:

Bootstrap provides custom events for most plugins' unique actions. Generally, these come in an infinitive and past participle form - where the infinitive (ex. show) is triggered at the start of an event, and its past participle form (ex. shown) is triggered on the completion of an action.

As of 3.0.0, all Bootstrap events are namespaced.

All infinitive events provide preventDefault functionality. This provides the ability to stop the execution of an action before it starts.

//this line was part of alam da’s code…

$('#myModal').on('show.bs.modal', function (e) { if (!data) return e.preventDefault() // stops modal from being shown })

#### .modal('show')

Manually opens a modal. **Returns to the caller before the modal has actually been shown** (i.e. before the shown.bs.modal event occurs).

$('#myModal').modal('show')

Name Type Default Description

|  |  |  |  |
| --- | --- | --- | --- |
| show | boolean | true | Shows the modal when initialized. |

| **Event Type** | **Description** |
| --- | --- |
| show.bs.modal | This event fires immediately when the show instance method is called. If caused by a click, the clicked element is available as the relatedTarget property of the event. |
| shown.bs.modal | This event is fired when the modal has been made visible to the user (will wait for CSS transitions to complete). If caused by a click, the clicked element is available as the relatedTarget property of the event. |
| hide.bs.modal | This event is fired immediately when the hide instance method has been called. |
| hidden.bs.modal | This event is fired when the modal has finished being hidden from the user (will wait for CSS transitions to complete). |
| hidePrevented.bs.modal | This event is fired when the modal is shown, its backdrop is static and a click outside the modal or an escape key press is performed with the keyboard option or data-keyboard set to false. |

**AJAX Terms to know:**

function fetch\_ajax\_joinee\_data($temp\_id){

$sql = $this->db->select('\*')

->from('joinee\_info')

->where('id',$temp\_id)

->get();

return $sql->result();//if this ajax call is set to result, then this is in array form which should be accessed using jquery in the $.each loop by writing data[i].joinee\_name

return $sql->row();//if this ajax call is set to row, then this is in array form which should be accessed using jquery in the $.each loop by writing data.joinee\_name.

$.each(data, function(i,field){

$('#joinee\_name').val(data[i].joinee\_name);

$("#email").val(data[i].email);

});