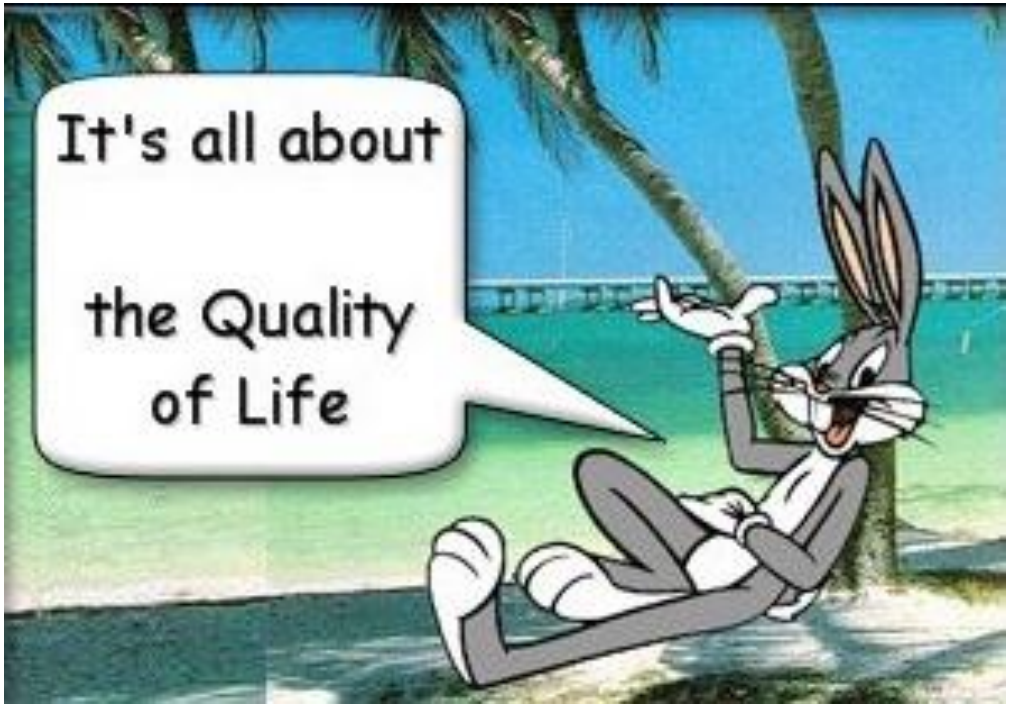



# jQuery

# A Quality of Life by jQuery:

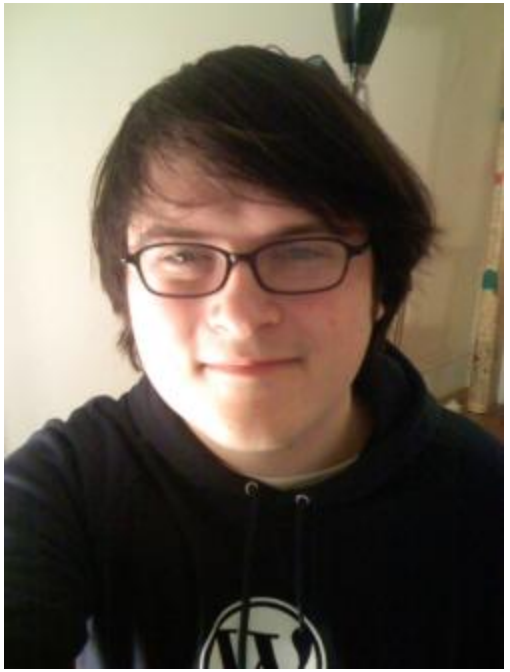
```
$(“#firstName”).text(“Joe Black”);

$(“button”).click(function() {alert “Clicked”;});

$(“.content”).hide();

$(“#main”).load(“content.htm”);

$(“<div/>”).html(“Loading…”).appendTo(“#content”);
```

**Very compact and fluent programming model**


# What is jQuery?

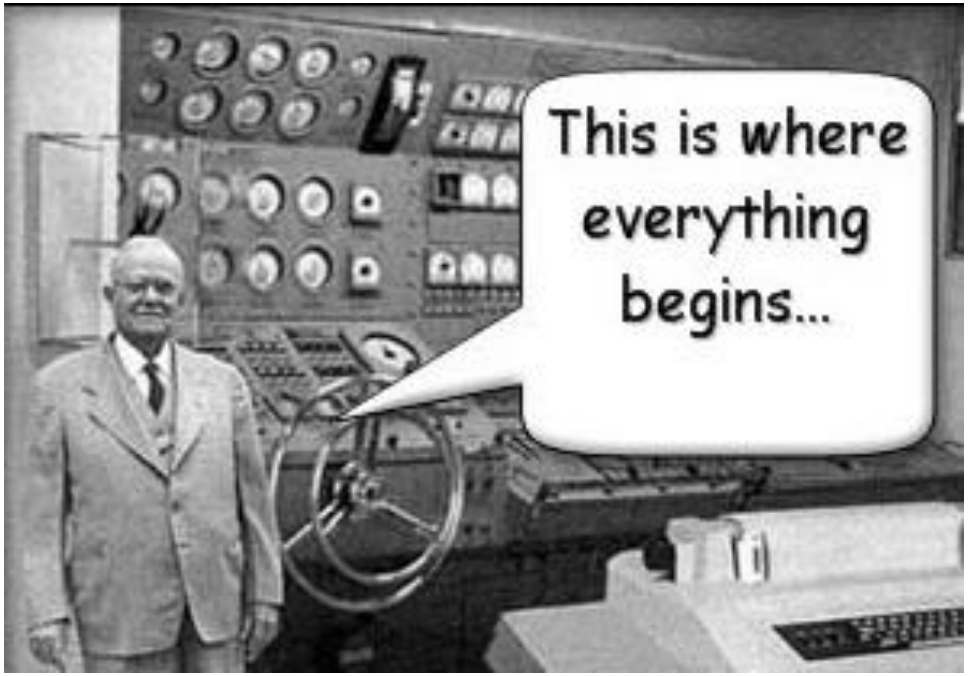
**jQuery** is a lightweight, open-source JavaScript library that **simplifies interaction** between HTML and JavaScript

It was and still being developed by John Resig from Mozilla and was first announced in January 2006



It has a great **community**, great **documentation**, tons of **plugins**, and it was recently adopted by **Microsoft**

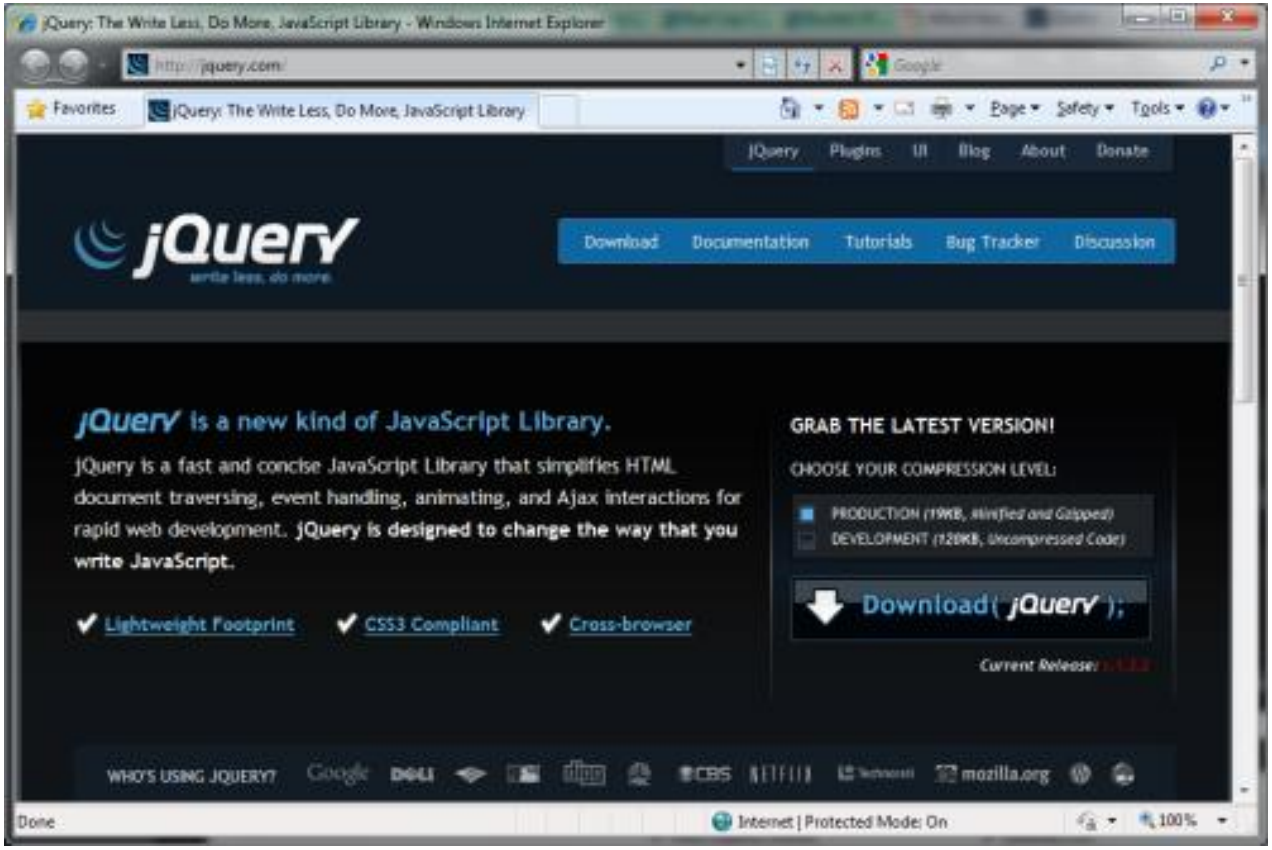
# The current version is 1.8




# Getting Started

# Download the latest version from

http://jquery.com



# To enable itellisense in VS 2008 SP1 install the –vsdoc hotfix:

VS90SP1-KB958502-x86.exe

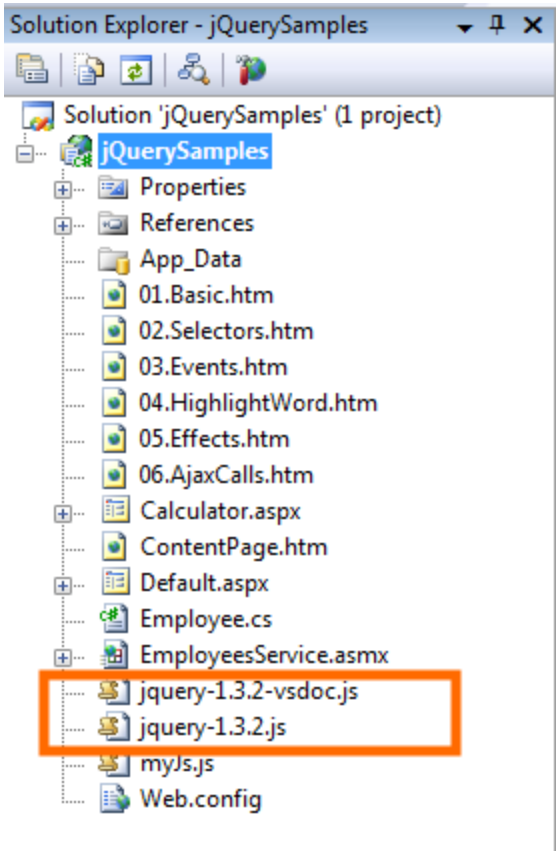
Copy the

**jquery.js**

and the

**jquery-vsdoc.js**

into your application folder



# Reference it in your markup

```
<script src="jquery.js"/>
```

**No need to reference the –vsdoc.js**

# You can also reference it from Google

```
<script src="http://ajax.googleapis.com/
          ajax/libs/jquery/1.2.6/
          jquery.min.js">
</script>
```


# jQuery Core Concepts

# The Magic $() function

```
var el = $(“<div/>”)
```

## Create HTML elements on the fly

# The Magic $() function

```
$(window).width()
```

**Manipulate existing DOM elements**

# The Magic $() function

```
$("div").hide();

$("div", $("p")).hide();
```

**Selects document elements**

**(more in a moment…)**

# The Magic $() function

```
$(function(){…});
```

**Fired when the document is ready for programming.**

**Better use the full syntax:**

```
$(document).ready(function(){…});
```

# The full name of $() function is

```
jQuery("div");
```

**It may be used in case of conflict with other frameworks.**

# The library is designed to be isolated

```
(function(){
 var
     jQuery=window.jQuery=window.$=function(){
          // …
     };
})();
```

**jQuery uses closures for isolation**

# Avoid $() conflict with other frameworks

```
var foo = jQuery.noConflict();
// now foo() is the jQuery main function
foo("div").hide();
```

```
// remove the conflicting $ and jQuery
var foo = jQuery.noConflict(true);
```

# jQuery's programming philosophy is:

## GET >> ACT

```
$("div").hide()

$("<span/>").appendTo("body")

$(":button").click()
```

**Almost every function returns jQuery, which provides a fluent programming interface and chainability:**

```
$(“div”).show()
        .addClass(“main”)
        .html(“Hello jQuery”);
```

# Three Major Concepts of jQuery

The $() function

Get > Act

Chainability


# jQuery Selectors

# All Selector

```
$("*")          // find everything
```

**Selectors return a pseudo-array of jQuery elements**

# Basic Selectors

**By Tag:**

```
$("div")
// <div>Hello jQuery</div>
```

**By ID:**

```
$("#usr")
// <span id="usr">John</span>
```

**By Class:**

```
$(".menu")
// <ul class="menu">Home</ul>
```

**Yes, jQuery implements CSS Selectors!**

# More Precise 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 <li> element of the first <ul> |
| $("ul li:first-child") | Selects the first <li> element of every <ul> |
| $("[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 |

# More Precise Selectors

```
$("div.main")    // tag and class
$("table#data") // tag and id
```

# Combination of Selectors

```
// find by id + by class
$("#content, .menu")
// multiple combination
$("h1, h2, h3, div.content")
```

# Hierarchy Selectors

```
$("table td")          // descendants
$("tr > td")           // children
$("label + input")     // next
$("#content ~ div")    // siblings
```

# Selection Index Filters

```
$("tr:first")    // first element
$("tr:last")     // last element
$("tr:lt(2)")    // index less than
$("tr:gt(2)")    // index gr. than
$("tr:eq(2)")    // index equals
```

# Visibility Filters

```
$("div:visible")// if visible
$("div:hidden") // if not
```

# Attribute Filters

```
$("div[id]")                // has attribute
$("div[dir='rtl']")         // equals to
$("div[id^='main']")        // starts with
$("div[id$='name']")        // ends with
$("a[href*='msdn']")        // contains
```

# Forms Selectors

```
$("input:checkbox")   // checkboxes
$("input:radio")      // radio buttons
$(":button")          // buttons
$(":text")            // text inputs
```

# Forms Filters

```
$("input:checked")    // checked
$("input:selected")   // selected
$("input:enabled")    // enabled
$("input:disabled")   // disabled
```

# Find Dropdown Selected Item

```
<select name="cities">
    <option value="1">Tel-Aviv</option>
    <option value="2" selected="selected">Yavne</option>
    <option value="3">Raanana</option>
</select>
```

```
$("select[name='ddl'] option:selected").val()
```

# SELECTORS DEMO


Great!
So what's next?


# Document Traversal

# A Selector returns a pseudo array of jQuery objects

```
$("div").length
```

**Returns number of selected elements.**

**It is the best way to check selector.**

# Getting a specific DOM element

```
$("div").get(2) or $("div")[2]
```

**Returns a 2nd DOM element of the selection**

# Getting a specific jQuery element

```
$("div").eq(2)
```

**Returns a 2nd jQuery element of the selection**

# each(fn) also passes an indexer

```
$("table tr").each(
   function(i){
        if (i % 2)
             $(this).addClass("odd");
   });
```

***$(this)*** **– convert DOM to jQuery**

***i*** **- index of the current element**

# Traversing HTML

```
.next(expr)      // next sibling
.prev(expr)      // previous sibling
.siblings(expr)  // siblings
.children(expr) // children
.parent(expr)    // parent
```

# Check for expression

```
$("table td").each(function() {
   if ($(this).is(":first-child")) {
        $(this).addClass("firstCol");
   }
 });
```

# Find in selected

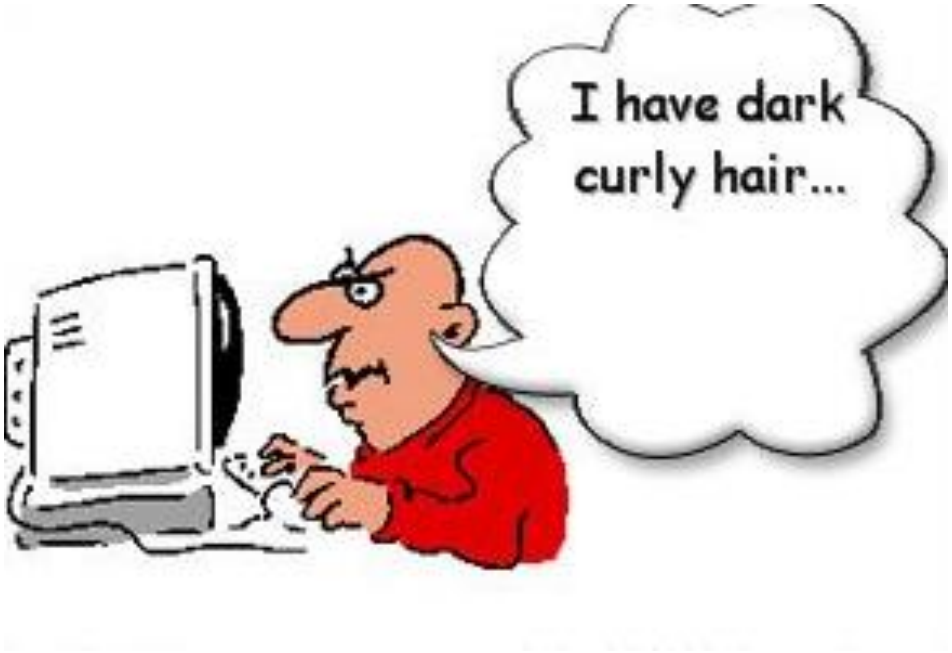
```
// select paragraph and then find
// elements with class ‘header’ inside
$(“p”).find(“.header”).show();

 // equivalent to:
$(“.header”, $(“p”)).show();
```

# Advanced Chaining

```
$("<li><span></span></li>") // li
    .find("span") // span
        .html("About Us") // span
        .andSelf() // span, li
            .addClass("menu") // span,li
        .end() // span
    .end() // li
    .appendTo("ul.main-menu");
```

# Get Part of Selected Result

```
$("div")
        .slice(2, 5)
        .not(".green")
            .addClass("middle");
```




# HTML Manipulation

# Getting and Setting Inner Content

```
$(“p”).html(“<div>Hello $!</div>”);


// escape the content of div.b
$(“div.a”).text($(“div.b”).html());
```

# Getting and Setting Values

```
// get the value of the checked checkbox
 $(“input:checkbox:checked”).val();
```

```
// set the value of the textbox
 $(“:text[name=‘txt’]”).val(“Hello”);
```

```
// select or check lists or checkboxes
 $(“#lst”).val([“NY”,”IL”,”NS”]);
```

# Handling CSS Classes

```
// add and remove class
$("p").removeClass("blue").addClass("red");
```

```
// add if absent, remove otherwise
$("div").toggleClass("main");
```

```
// test for class existance
 if ($("div").hasClass("main")) { //… }
```

# Inserting new Elements

```
// select > append to the end
$("h1").append("<li>Hello $!</li>");

// select > append to the beginning
$("ul").prepend("<li>Hello $!</li>");
```

```
// create > append/prepend to selector
$("<li/>").html("9").appendTo("ul");

$("<li/>").html("1").prependTo("ul");
```

# Replacing Elements

```
// select > replace
$("h1").replaceWith("<div>Hello</div>");
```

```
// create > replace selection
$("<div>Hello</div>").replaceAll("h1");
```

# Replacing Elements while keeping the content

```
  $(“h3”).each(function(){
$(this).replaceWith(“<div>”
                                        +
$(this).html()
                          + ”</div>”);
});
```

# Deleting Elements

```
// remove all children
$("#mainContent").empty();
```

```
// remove selection
$("span.names").remove();
```

```
// change position
$("p").remove(":not(.red)")
                .appendTo("#main");
```

# CSS Manipulations

```
// get style
$("div").css("background-color");
```

```
// set style
$("div").css("float", "left");
```

```
// set multiple style properties
$("div").css({"color":"blue",
              "padding": "1em"
              "margin-right": "0",
              marginLeft: "10px"});
```

# Dimensions

```
// get window height
var winHeight = $(window).height();

// set element height
$("#main").height(winHeight);

//.width() - element

//.innerWidth() - .width() + padding

//.outerWidth() - .innerWidth() + border

//.outerWidth(true) - including margin
```

**The default unit is Pixel (px)**

# Positioning

```
// from the document
$(“div”).offset().top;

// from the parent element
$(“div”).position().left;

// scrolling position
$(window).scrollTop();
```


# Events

# When the DOM is ready…

```
$(document).ready(function(){
   //…
});
```

- Fires when the document is ready for programming.
- Uses advanced listeners for detecting.
- window.onload() is a fallback.

# Attach Event

```
// execute always
$("div").bind("click", fn);

// execute only once
$("div").one("click", fn);
```

## Possible event values:

blur, focus, load, resize, scroll, unload, beforeunload, click, dblclick, mousedown, mouseup, mousemove, mouseover, mouseout, mouseenter, mouseleave, change, select, submit, keydown, keypress, keyup, error

(or any custom event)

# jQuery.Event object

CONTENTS

# Detaching Events

```
$(“div”).unbind(“click”, fn);
```

(Unique ID added to every attached function)

# Events Triggering

```
$("div").trigger("click");
```

Triggers browser's event action as well.

Can trigger custom events.

Triggered events bubble up.

# Events Helpers

```
// attach / trigger

elem.blur(fn) / elem.blur()

elem.focus(fn) / elem.focus()

elem.click(fn) / elem.click()

elem.change(fn) / elem.change()
```

And many others…

# EVENTS DEMO


# Effects

# Showing or Hiding Element

```
// just show
$("div").show();

// reveal slowly, slow=600ms
$("div").show("slow");

// hide fast, fast=200ms
$("div").hide("fast");

// hide or show in 100ms
$("div").toggle(100);
```

# Sliding Elements

```
$(“div”).slideUp();

$(“div”).slideDown(“fast”);

$(“div”).slideToggle(1000);
```

# Fading Elements

```
$(“div”).fadeIn(“fast”);

$(“div”).fadeOut(“normal”);

// fade to a custom opacity
 $(“div”).fadeTo (“fast”, 0.5);
```

Fading === changing opacity

# Detecting animation completion

```
$("div").hide("slow", function() {
        alert("The DIV is hidden");
});


$("div").show("fast", function() {
        $(this).html("Hello jQuery");
}); // this is a current DOM element
```

Every effect function has a (speed, callback) overload

# Custom Animation

```
// .animate(options, duration)
$("div").animate({
                    width: "90%",
                    opacity: 0.5,
                    borderWidth: "5px"

                    }, 1000);
```

# Chaining Animation

```
$(“div”).animate({width: “90%”},100)
        .animate({opacity: 0.5},200)
        .animate({borderWidth: “5px”});
```

By default animations are queued and than performed one by one

# Controlling Animations Sync

```
$(“div”)
    .animate({width: “90%”},
                {queue:false, duration:1000})
    .animate({opacity : 0.5});
```

The first animation will be performed immediately without queuing

# EFFECTS DEMO


# AJAX with jQuery

# Loading content

```
$(“div”).load(“content.htm”);


// passing parameters
 $(“#content”).load(“getcontent.aspx”,
                                {“id”:”33”,

      “type”:”main”});
```

# Sending GET/POST requests

```
$.get(“test.aspx”, {id:1},
        function(data){alert(data);});


$.post(“test.aspx”, {id:1},
        function(data){alert(data);});
```

# Retrieving JSON Data

```
$.getJSON("users.aspx", {id:1},
        function(users)
        {
             alert(users[0].name);
        });
```