

J is for Jedi?

Wheel you use,
invent it you don't.



jQuery intro

Up until now, we have been enjoying vanilla (JS)



not knowing there are
many ways to make it
tastier...

- Meet jQuery



jQuery has changed the way
millions write JavaScript.

jQuery intro

jQuery is a fast, small, and feature-rich JavaScript **library**.

Library – A collection of reusable code

Making our life easier!

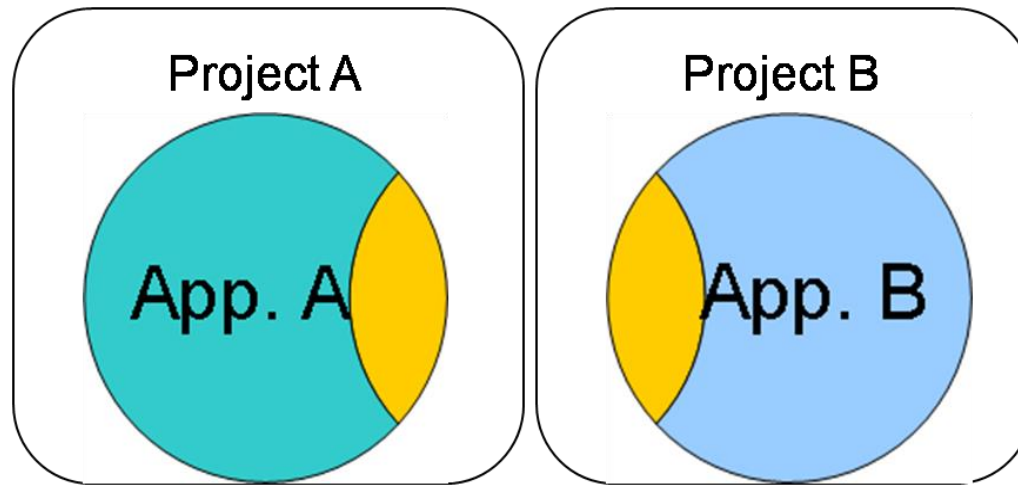
- HTML document traversal and manipulation
- event handling
- Animation
- Ajax

What comes with a library?

Library is a new dependency

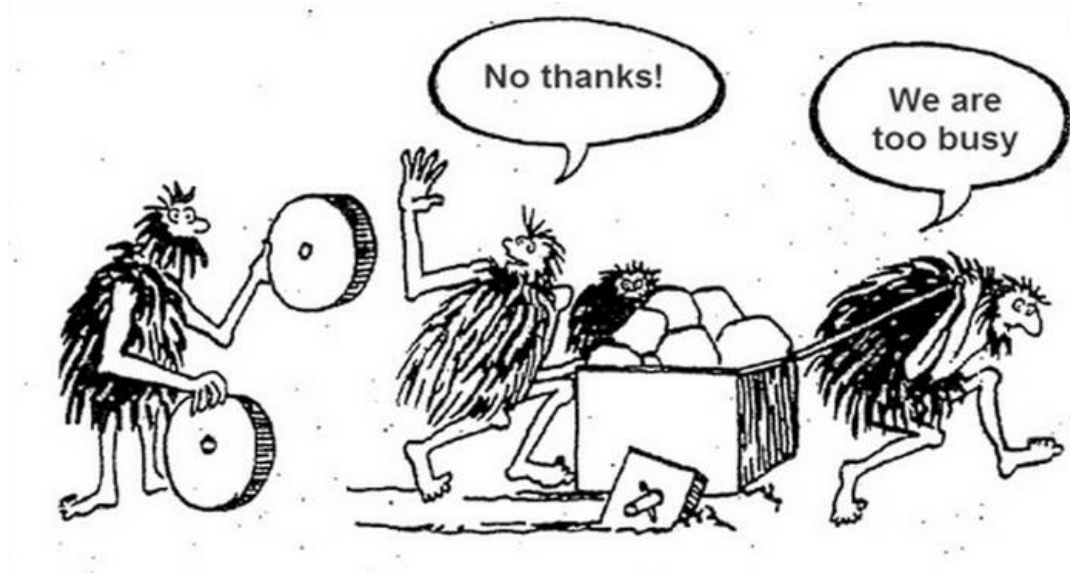
- An API we need to learn
- Possible conflicts
- Adds code we never use
- The cost/benefit ratio for jQuery is very high

Why use a library?



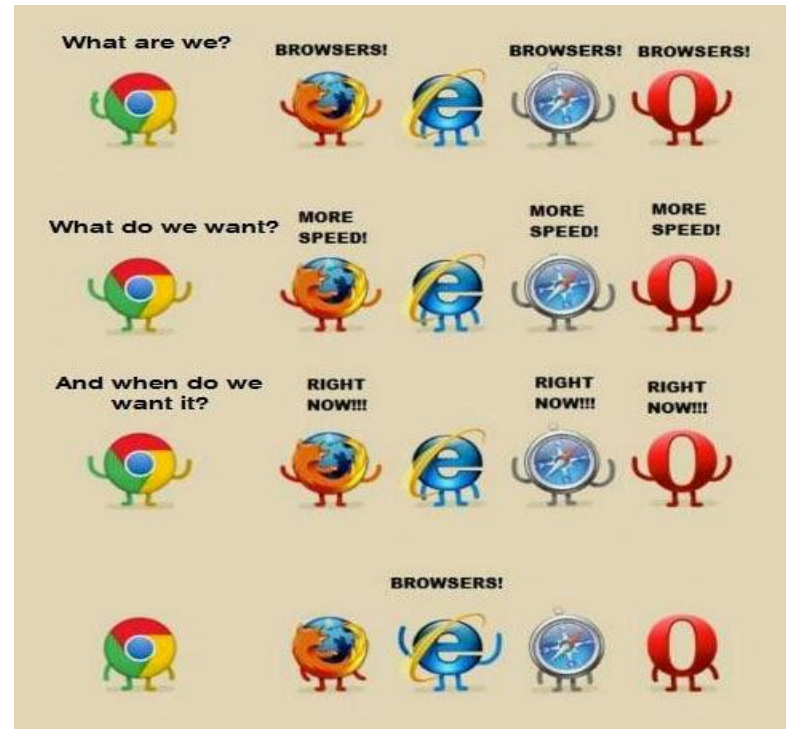
Code Reuse

Why use a library?



Use the wheel, don't re-invent it.

Why use a library?



Cross browser support



From ninja to Jedi in a few ~~easy~~
steps

Step 1

Include the jQuery script in our HTML

There are two options to do that:

- Download the library from the jQuery website and put it in our JS folder
- Place a direct link to a jQuery CDN

Option 1



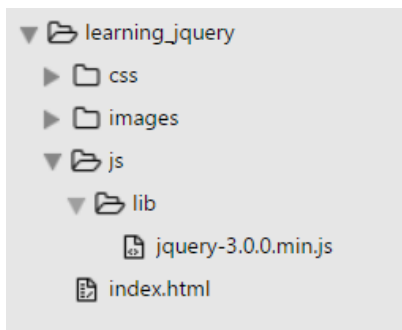
Download the library



Download jQuery
v3.0.0



Create a sub folder under our JS folder called lib (in order to distinguish external libraries and our own code)



Add a script tag to our HTML code with the path:

```
<script src="./js/lib/jquery-3.0.0.min.js"></script>
```

Option 2



Just add the CDN URL in the script tag instead of your folder path

```
<script src="https://code.jquery.com/jquery-3.0.0.min.js"></script>
```



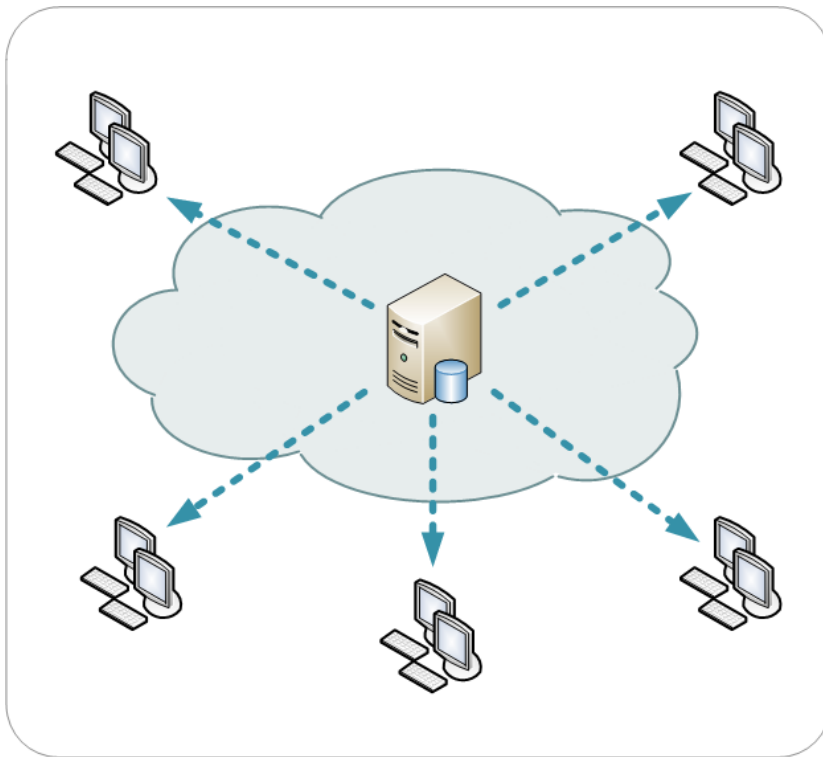
CDN - Content Delivery Network

Distributed network of servers that delivers Web content based on the geographic locations

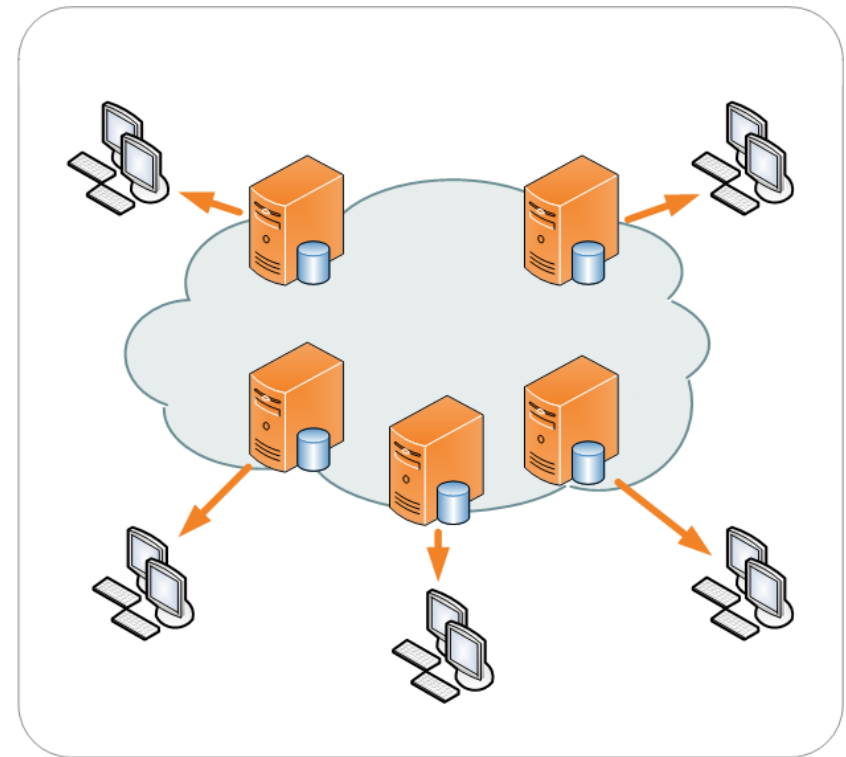
Option 2

- **CDN - Content Delivery Network**

Single server distribution



CDN distribution



Option 2

Using a CDN is always preferable

- Google servers are more reliable
- You don't pay for traffic

Unless you want to be able to code offline (like on a plane)



Step 2

Every time we want to use the jQuery library, we need to use its' namespace .

The obvious namespace is just jQuery,
But the shorter version is just the **\$** sign.

`$("#test").hide()` `jQuery("#test").hide()`



Questions



Questions?

Step 3

So...

How can we use it to make our vanilla tastier?

Selecting elements from the DOM



Step 3

Selectors and the jQuery Object

Step 3

Selecting an element with the id of "control-panel" in JS

```
<body>  
  <div id="control-panel"></div>  
</body>
```

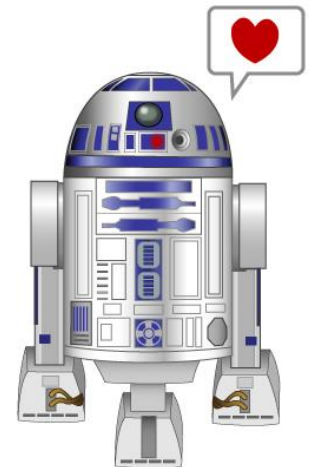


```
var controlPanel = document.getElementById("control-panel");
```

In jQuery



```
var controlPanel = $("#control-panel");
```



Step 3

What is the difference?

```
var controlPanel = document.getElementById("#control-panel");  
var controlPanel = $("#control-panel");
```

- Shorter and more elegant
- jQuery selector function **\$** returns a jQuery object.
Different than the object returned by **getElementById**

Step 3

There are more functions

But let's look at some examples of selectors.



Step 3

Translate to English:

```
var controlPanel = $(".main.menu #control-panel");
```

Element with id "**control-panel**" that is a descendant of an element with both "**main**" and "**menu**" classes

```
var listItems = $("ul li.list-item");
```

All elements with tag **li** and class "**list-item**" under element with tag "**ul**"

Wait... but that can return more than one element...
More on that later, it is time for the next step!



Step 3

jQuery selectors can receive any CSS selector you know (and more)

```
var myFirstButton= $(".container button:first-child");
```

```
var flippedMemoryCards= $(".flipped.card:not(.disabled)");
```

Questions



Questions?

[illegible]

Step 4

So we have a jQuery object, what now?

```
var controlPanel = $("#control-panel");
```

Changing **one** CSS property

```
controlPanel.css("background-color", "red");
```

Changing **many** CSS properties

```
controlPanel.css({  
  "background-color": "red",  
  "width": "100px",  
  "display": "inline-block"  
});
```

Step 4

Getting an attribute:

```
var panelId = controlPanel.attr("id");
```

Setting an attribute:

```
myImage.attr("src", "./images/my_cat.jpg");
```



Actions on classes

- We can add, remove or toggle a class

```
// get all the elements with the class 'nice'.
```

```
var elms = $('.nice');
```

```
// add to all of them the class 'some-class'
```

```
elms.addClass('some-class');
```

```
// remove from all the class 'some-other-class'
```

```
elms.removeClass('some-other-class');
```

```
// remove the class open if it exists
```

```
// else, add "open"
```

```
elms.toggleClass('open');
```

Don't worry.
Syntax is quiet weird, but you'll get used to it soon.



Step 4

Clear HTML content:

```
controlPanel.empty();
```

Changing text content:

```
myTitle.text("Hello world");
```

Hiding an element (will set display to none):

```
controlPanel.hide();
```

Showing an element:

```
controlPanel.show();
```



Step 4

Fading in an element!

```
controlPanel.fadeIn();
```

Fading out an element!

```
controlPanel.fadeOut();
```

The full API (list of available functions)

Can be found in the jQuery website



Questions



Questions?

Step 4

What happens when jQuery returns several results?
Consider the following HTML:

```
<ul id="my-menu">  
  <li class="nav-option">Home</li>  
  <li class="nav-option">About</li>  
  <li class="nav-option">Gallery</li>  
  <li class="nav-option">Contact Us</li>  
</ul>
```

What will the following code do?

```
var listItems = $("#my-menu .nav-option");  
listItems.hide();
```

Hide all of
the li elements

Step 4

Every result of the **\$** query function is actually a collection (similar to array) of jQuery objects

```
var listItems = $("#my-menu .nav-option");
```

And as one it has the length property:

```
console.log(listItems.length);
```

Will print **4**

[Selectors Examples](#)

Step 5

Creating/removing elements The easy way

Step 5

Creating an element using jQuery is super easy:

```
var navBar = $("<div/>");  
navBar.addClass("nav-bar");
```

Just like JS we need to append it to the document

```
$("body").append(navBar);
```

Step 5

Let's create the following structure dynamically:

```
<ul id="my-menu">
  <li class="nav-option">Home</li>
  <li class="nav-option">About</li>
  <li class="nav-option">Gallery</li>
  <li class="nav-option">Contact Us</li>
</ul>
```

```
var menuOptions = ["Home", "About", "Gallery", "Contact Us"];
var navBar = $("<ul/>");
navBar.attr("id", "my-menu");
for (var i=0; i < menuOptions.length; i++){
  var myItem = $("<li/>");
  myItem.addClass("nav-option");
  myItem.text(menuOptions[i]);
  navBar.append(myItem);
}
$("body").append(navBar);
```

Step 5

Removing an element is also simple

Remove an element with id "to-delete"

```
$("#to-delete").remove();
```



Creating elements

- `append()` - Inserts content **inside** the selected elements, at the **end**
- `prepend()` - Inserts content **inside** the selected elements, at the **beginning**
- `after()` - Inserts content **after** the selected elements
- `before()` - Inserts content **before** the selected elements

Creating Elements

3. Before

2. Prepend

This is the target div to which new elements are associated using jQuery

1. Append

4. After

Questions



Questions?

Step 6

Selecting elements with context

Step 6

Sometimes we want to query elements within a specific parent

Let's say we want all of the elements with class "to-delete" under a div with class "board2"

We have two options:

```
var elementsToRemove = $("div.board2 .to-delete");
```

OR

```
var board2 = $("div.board2");  
var elementsToRemove = board2.find(".to-delete");
```

Performance

Whenever we are using a selector,
jQuery is querying the whole DOM
And it takes time

This is why this

```
var board2 = $("div.board2");  
var changeColorTo = board2.find(".colored");  
var elementsToRemove = board2.find(".to-delete");  
var replaceText = board2.find(".replace-me");
```

Is better than this:

```
$("div.board2 .colored");  
$("div.board2 .to-delete");  
$("div.board2 .replace-me");
```

Questions



Questions?

Step 7

jQuery is forgiving

Step 7

An important fact about jQuery's forgiving nature

Consider the following HTML

```
<ul id="menu">
  <li class="nav-option">Home</li>
  <li class="nav-option">About</li>
  <li class="nav-option">Gallery</li>
  <li class="nav-option">Contact Us</li>
</ul>
```

The following code Will not cause an error

```
var hideMenuButtons = function () {
  var menu = $("#this-is-not-the-menu-id");
  menu.find("li.nav-option").hide();
};
```

Step 7

To know if the query returned results, we can always use the following method:

```
var hideMenuButtons = function(){  
    var menu = $("#this-is-not-the-menu-id");  
    if (menu.length > 0){  
        menu.find("li.nav-option").hide();  
    }  
};
```


Where should we put the js files?

If we put them in the head, this code from the previous steps (fixed) will not work:

```
var hideMenuButtons = function(){  
    if (menu.length > 0){  
        menu.find("li.nav-option").hide();  
    }  
};
```

```
hideMenuButtons();
```

Why?

We need to wait for the browser to load
The DOM

Step 8

Waiting for the DOM to load

Loading a webpage requires resources

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Israel Tech Challenge</title>
5   <meta charset="utf-8">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
7   <meta name="description" content="">
8
9   <meta name="viewport" content="width=950, maximum-scale=1">
10  <link href="/favicon.ico" type="image/x-icon" rel="icon" /><link href="/favicon.ico" type="image/x-icon" rel="shortcut icon" />
11
12  <link href='https://fonts.googleapis.com/css?family=Roboto:400,300,500,700' rel='stylesheet' type='text/css'>
13
14  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.5.0/css/font-awesome.min.css">
15
16
17  <link rel="stylesheet" type="text/css" href="/css/normalize.css?1490429561" />
18  <link rel="stylesheet" type="text/css" href="/fancybox/jquery.fancybox-1.3.4.css?1490429561" />
19  <link rel="stylesheet" type="text/css" href="/formalize/css/formalize.css?1490429561" />
20  <link rel="stylesheet" type="text/css" href="/css/main.css?1490429561" />
21  <link rel="stylesheet" type="text/css" href="/css/style.css?1490429561" />
22  <script>
23    window.App = {baseUrl: 'https://israeltechchallenge.com/'};
24  </script>
25  <script>(function() {
26    var _fbq = window._fbq || (window._fbq = []);
27    if (!_fbq.loaded) {
28      var fbds = document.createElement('script');
29      fbds.async = true;
30      fbds.src = '//connect.facebook.net/en_US/fbds.js';
31      var s = document.getElementsByTagName('script')[0];
32      s.parentNode.insertBefore(fbds, s);
33      _fbq.loaded = true;
34    }
35    _fbq.push(['addPixelId', '1465970323696088']);
36    })();
37    window._fbq = window._fbq || [];
38    window._fbq.push(['track', 'PixelInitialized', {}]);
39  </script>
40  <noscript></noscript>
41 </head>
42 <body>
43   <a id="top"></a>
44   <!-- start header -->
45   <div class="header">
46     <div class="header-top">
47       <!-- start box -->
48       <div class="box">
```

Favicon

Fonts

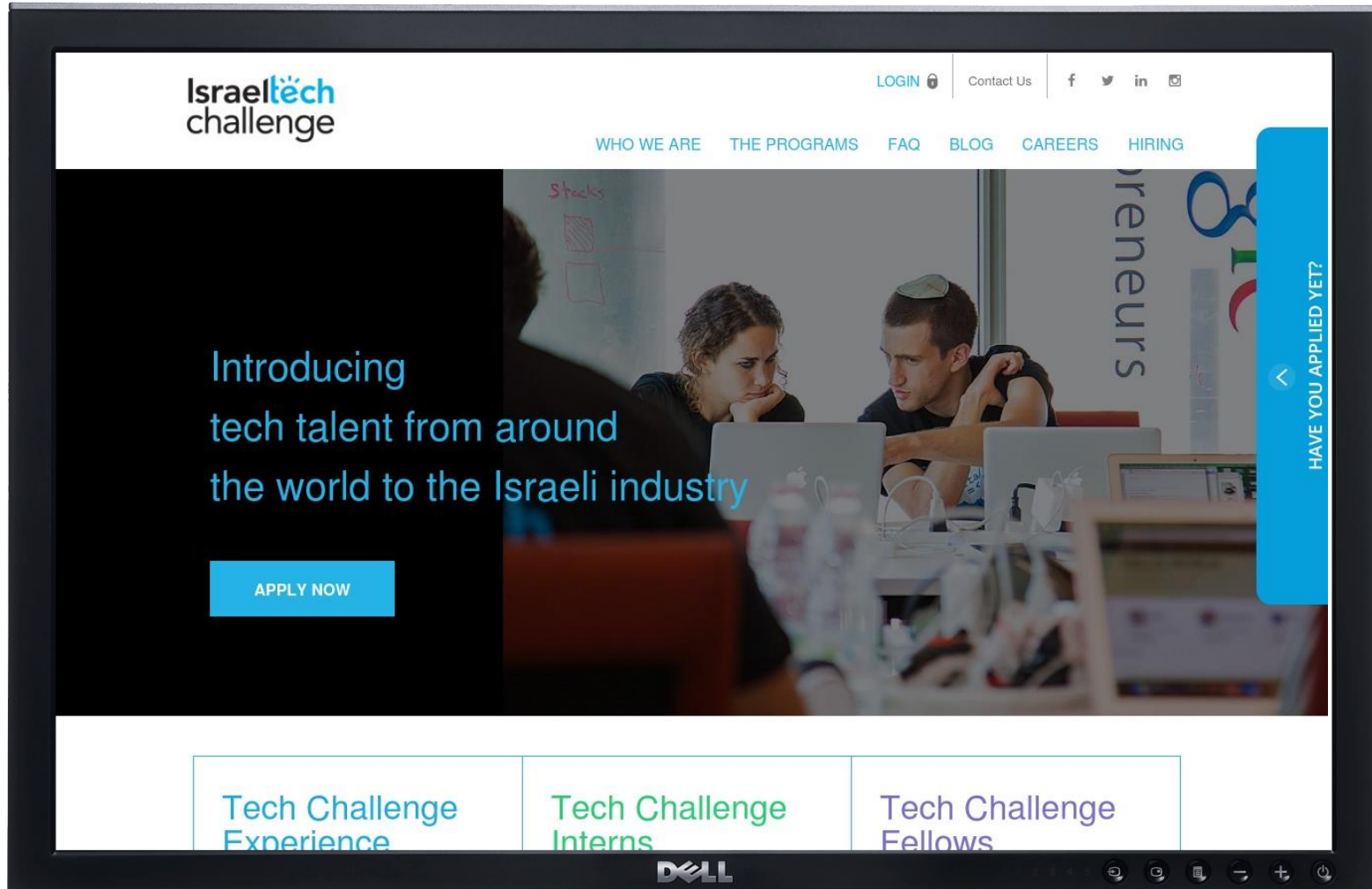
CSS

JS

JPEG

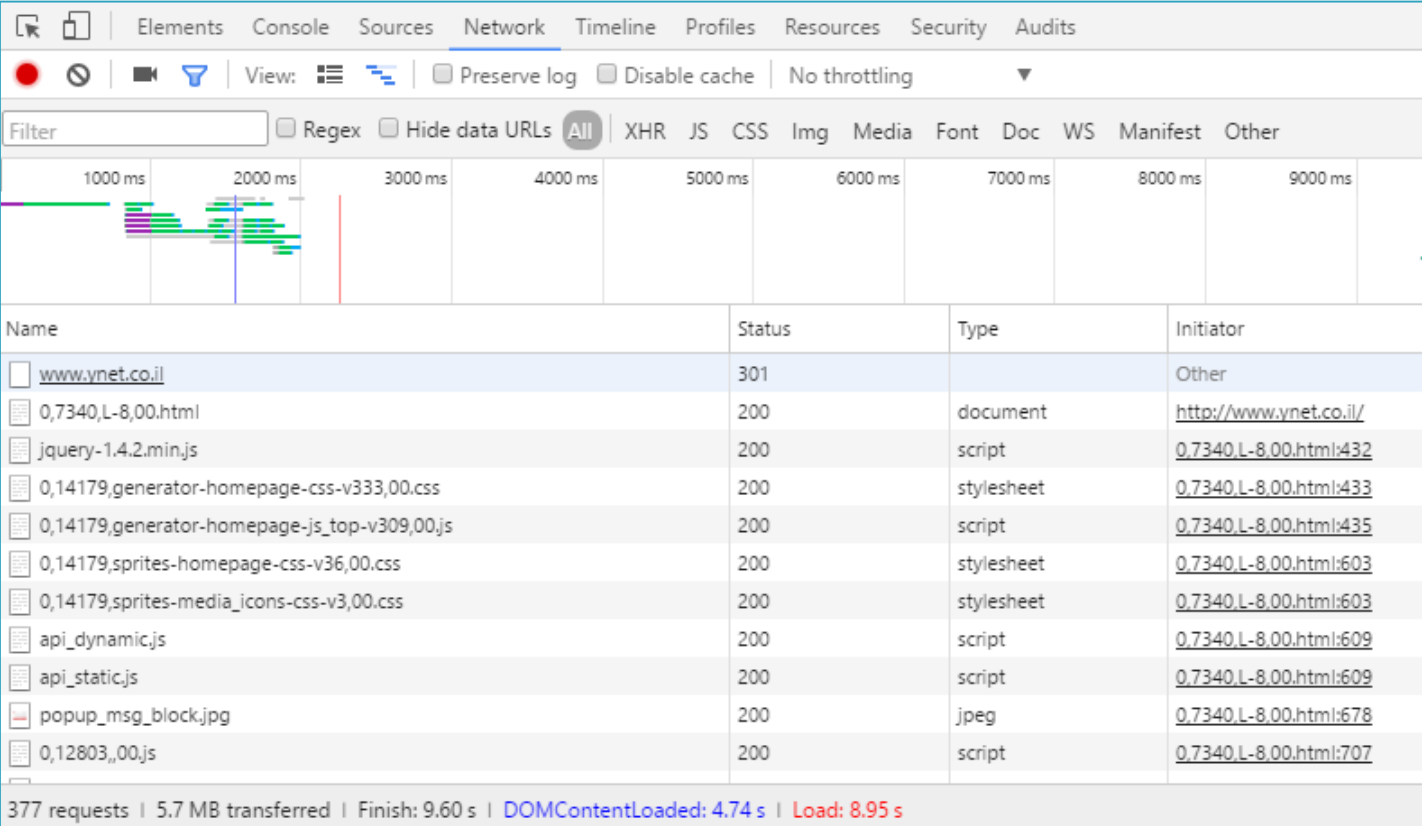
PNG

Loading a webpage requires resources



Loading resources

- Viewing the loaded resources is simple
- Use the **network** tab in the Chrome dev tools



The screenshot displays the Chrome DevTools Network tab. The top toolbar includes icons for refreshing, pausing, and recording, along with a filter input and checkboxes for 'Preserve log', 'Disable cache', and 'No throttling'. Below the toolbar, a filter bar shows 'All' selected, with other filters like XHR, JS, CSS, etc. A timeline view at the top shows a sequence of requests over time. The main table lists the following resources:

| Name | Status | Type | Initiator |
|--|--------|------------|---|
| www.vnet.co.il/ | 301 | | Other |
| 0,7340,L-8,00.html | 200 | document | http://www.vnet.co.il/ |
| jquery-1.4.2.min.js | 200 | script | 0,7340,L-8,00.html:432 |
| 0,14179,generator-homepage-css-v333,00.css | 200 | stylesheet | 0,7340,L-8,00.html:433 |
| 0,14179,generator-homepage-js_top-v309,00.js | 200 | script | 0,7340,L-8,00.html:435 |
| 0,14179,sprites-homepage-css-v36,00.css | 200 | stylesheet | 0,7340,L-8,00.html:603 |
| 0,14179,sprites-media_icons-css-v3,00.css | 200 | stylesheet | 0,7340,L-8,00.html:603 |
| api_dynamic.js | 200 | script | 0,7340,L-8,00.html:609 |
| api_static.js | 200 | script | 0,7340,L-8,00.html:609 |
| popup_msg_block.jpg | 200 | jpeg | 0,7340,L-8,00.html:678 |
| 0,12803,,00.js | 200 | script | 0,7340,L-8,00.html:707 |

At the bottom, a summary bar shows: 377 requests | 5.7 MB transferred | Finish: 9.60 s | DOMContentLoaded: 4.74 s | Load: 8.95 s

Notice the colors

378 requests | 5.7 MB transferred | Finish: 39.25 s | DOMContentLoaded: 4.74 s | Load: 8.95 s

- **DOMContentLoaded** – The DOM is loaded and parsed (the structure of the page), not including CSS, images, scripts etc.
- **Load** – the time when the images, videos and so on finished loading.

Document Ready

Let's start with JS:

DOMContentLoaded

```
document.addEventListener("DOMContentLoaded", function(event){  
    console.log("DOM fully loaded and parsed");  
});
```

Load

```
document.addEventListener("load", function(event){  
    console.log("All resources finished loading");  
});
```

Why use `$(document).ready()`?

- Loading a page takes an unknown time
- We want to make sure jQuery finds the right elements

```
$(document).ready(function() {  
    // Document is loaded and DOM is ready  
    alert("Document is ready");  
});
```


As we saw before, jQuery is a forgiving library.

When our code will execute:

- “menu” element is not ready yet and
- The selector will return nothing

In order to verify that the DOM has finished loading, we can use the “**ready**” function, which is equivalent to **DOMContentLoaded** .

DOMContentLoaded ~ ready (jQuery)

Load = load

Our code will now look like this:

```
var hideMenuButtons = function(){  
    var menu = $("#menu");  
    menu.find("li.nav-option").hide();  
};
```

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

Using the actual document object and not a string

The ready function receives a function as a parameter (we chose to use an anonymous one)

Our code will execute only after the DOM has finished loading

Questions



Questions?



Events



Some inspiration

- Lets see what jQuery will allow us to do:
- [jQuery UI Demo](#)
- [Magnifier Effect](#)
- [Bubble Navigation](#)
- [Circular Cool Things](#)

Chaining

jQuery provides many functions for every element

- Add a class to it
- Add text to it
- Append it to the body

```
var someDiv = $("<div/>");  
someDiv.addClass("big-div");  
someDiv.text("someText");  
someDiv.appendTo($ (document.body));
```

Chaining

- Alternatively, we can do the same using function chaining
- Every function operates on the result of the previous functions in the chain.

```
$ ("<div/>")  
  .addClass ("big-div")  
  .text ("someText")  
  .appendTo ($ (document.body) ) ;
```

Event listeners in jQuery

jQuery is far more elegant!

- jQuery

```
$(".new-game-btn").on('click', function () {  
    MemoryGame.start(imgArr);  
});
```

- Javascript – long an tedious.....

```
var newGameBtn = document.getElementsByClassName("new-game-btn");  
for(var i=0; i < newGameBtn.length; i++) {  
    //define event listeners for the click on the new game buttons  
    newGameBtn[i].addEventListener('click', function () {  
        MemoryGame.start(imgArr);  
    });  
}
```


Adding event listeners

- We can add any event listeners to any jQuery object:

```
var btn = $('.btn');  
btn.on('click', function (eventObj) {  
    var btnClicked = $(this);  
    btnClicked.toggleClass("red-text");  
});
```

- Another option is just use .click()

```
btn.click(function (eventObj) {  
    var btnClicked = $(this);  
    btnClicked.toggleClass("red-text");  
});
```

Removing event listeners

- we can also remove listeners from any jQuery object

```
var btn = $('.btn-1');  
btn.on('click', function (eventObj) {  
    var btnClicked = $(this);  
    btnClicked.off('click');  
    btnClicked.toggleClass("red-text");  
});
```

- Once the specific button is clicked once, it will not trigger the event again.

Questions



Questions?

What triggers an event?

```
<div class="container-fluid">
  <div class="row first">
    Some text
  </div>
  <div class="row second">
    Some text
  </div>
</div>
```


- Let's add an event listener to the parent class

```
$( '.container-fluid' ).on( "mouseover mouseout", (function(event) {
    $(this).toggleClass( 'red-bg' );
}));
```

- This adds 2 event listeners to every jQuery element of class **container-fluid**

What triggers an event?


- What will happen if we move our mouse over a child element of `$('.container-fluid')` ?



Some text
Some text

- What happens if we change `$(this)` to `$(event.target)` ?

```
$( '.container-fluid' ).on( "mouseover mouseout", (function(event) {  
    $( event.target ).toggleClass( 'red-bg' );  
}));
```



Some text
Some text

Triggering an event via code

```
$ (" #some-btn" ) .click ( ) ;
```

Select a specific element



Click on it



- What happens if there is no event listener defined on that element? **Nothing!**

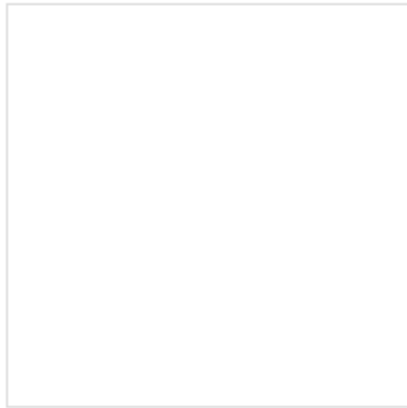
Events

- The Full event list [here](#) (Online, like everything else 😊)
 - click
 - mousedown
 - mouseout
 - dblclick
 - blur
 - focus
 - keyup
 - keypress
 - Hover
 - ...

Emotional coloring

We have button that will paint our box with different color according to the emotion.
Here is a mockup:

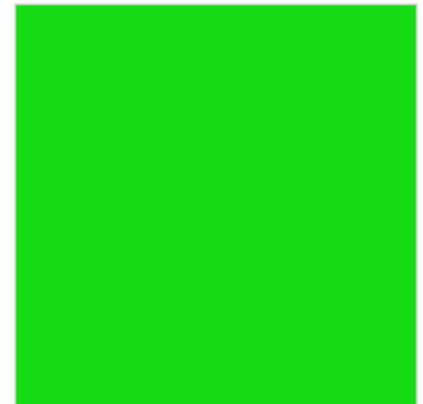
loving fresh happy



When we click on fresh we want the box to be painted green.



loving fresh happy



How can we do that?

Here is the code

Data elements

- We can add arbitrary data to the jQuery objects using `.data()`
- Setting data on jQuery elements:

```
$( '.btn-1' ).data( "dataname", "value" );
```

- Getting data from a jQuery element:

```
$( '.btn-1' ).data( "dataname" );
```

This data will only exist in the jQuery object!



Emotional coloring

What we want to do is connect between a DOM element and a color.

Can you do it?

We can do that with the data feature:

```
$("#button:nth-child(1)").data("color", "#e12e2e");  
$("#button:nth-child(2)").data("color", "#14db14");  
$("#button:nth-child(3)").data("color", "#fee11b");  
  
$("#button").click(function(){  
    $(".result").css("background", $(this).data("color"));  
});
```

Iterating over children

When we get multiple elements using a jQuery selector we can iterate them with the each function.

```
$( 'input' ).each( function () {  
    console.log( $( this ).val() );  
} );
```

This can be useful when you want to read value or data from multiple elements.

Questions



Questions?

Further reading:

- <https://api.jquery.com/category/traversing/>
- <https://learn.jquery.com/using-jquery-core/>
- <http://tutorialzine.com/2011/06/15-powerful-jquery-tips-and-tricks-for-developers/>
- <https://learn.jquery.com/performance/optimize-selectors/>

Further reading:

- <http://api.jquery.com/> documentation
- <https://jqueryui.com/> good widgets.
- Compare between using jQuery and Javascript
<http://youmightnotneedjquery.com/> and
<http://vanilla-js.com/>.
- <http://lab.abhinayrathore.com/jquery-standards/>
style and performance rules recommendations.

Live examples from the lecture

- Selectors examples

<http://jsfiddle.net/2bhere4u/h054Lkdn/17/>

- DOM Manipulation examples

<http://jsfiddle.net/2bhere4u/nhoyyjx6/9/>

- Document ready example

<http://jsfiddle.net/2bhere4u/om1aomf5/>

Jquery Cheat Sheet

Add Jquery in ascript tag

```
<script src="https://code.jquery.com/jquery-3.0.0.min.js"></script>
```

Selectors

```
$("#control-panel");
```

Change CSS

```
jqueryObject.css("background-color","red");
```

Add class

```
controlPanel.addClass("minified");
```

Remove class

```
controlPanel.removeClass("mobile-mode");
```

Get Attribute

```
var panelId = controlPanel.attr("id");
```

Remove class

```
myImage.attr("src","./images/my_cat.jpg");
```

Clear HTML content

```
controlPanel.empty();
```

Change text content

```
myTitle.text("Hello world");
```

Hide Element

```
controlPanel.hide();
```

Show Element

```
controlPanel.show();
```

Fadein animation

```
controlPanel.fadeIn();
```

Append

```
$("body").append(navBar);
```

Remove

```
$("#to-delete").remove();
```

Select under a specific node:

```
var board2 = $("div.board2");
```

```
var element = board2.find(".to-delete");
```

Remove

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


jQuery events Cheat Sheet

Toggle class

```
elems.toggleClass('open');
```

On (add event) + Off

```
btn.on('click', function (eventObj) {  
    $(this).off('click');  
});
```

Add Data

```
$('.btn-1').data("dataname", "value");
```

Get Data

```
$('.btn-1').data("dataname");
```

Loop over elements

```
$('input').each(function () {  
    console.log($(this).val());  
});
```