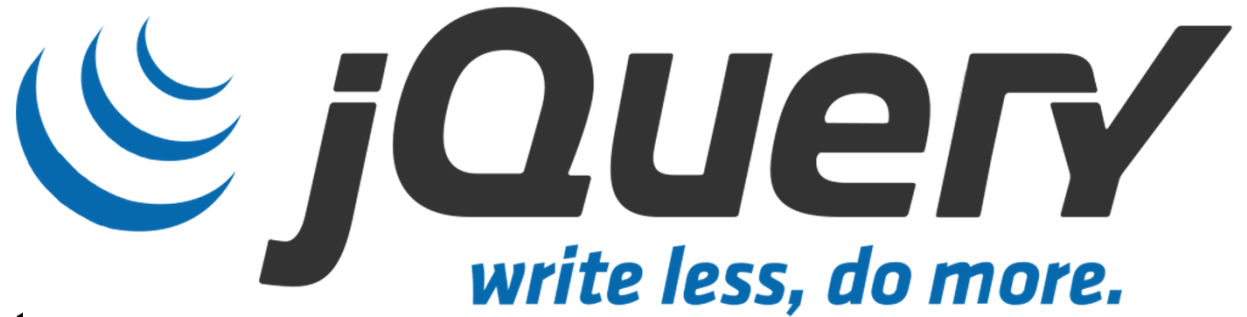


COMP4021  
Internet Computing

jQuery

David Rossiter & Gibson Lam

# jQuery



- jQuery is a JavaScript library that makes writing JavaScript code easier and more powerful
- jQuery code is typically more concise and cleaner than most JavaScript code
- The jQuery hides various some issues with different browsers e.g. older browsers from the programmer

jQuery tries to handle older browsers

jQuery is a library for JS, which itself is written in JS.

# Using jQuery

All jQuery codes can be re-written in pure JS

- It is **not** necessary to use jQuery to do things in a web page
- However, jQuery helps a lot if you do use it
- 74% of all websites use the jQuery library
- To use jQuery, the first thing to do is to link to it from your web page

*Data from [https://w3techs.com/technologies/overview/javascript\\_library/all](https://w3techs.com/technologies/overview/javascript_library/all)*

# Where Is the jQuery Library You Use?

- There's 2 main approaches:

- **Approach 1**

- You handle the jQuery file in your web site**

- Download jQuery e.g. from <http://jquery.com/> to your folder
    - Then add a link to the library from your HTML page

```
<head>                                The jQuery file you downloaded
...
<script src="jquery-3.4.1.min.js"></script>
...
</head>
```

# Using the jQuery Library

- **Approach 2**

- You use the jQuery file from somewhere else**

- There's lots of copies of the jQuery library on the web
- Some organisations make a CDN (Content Delivery Network) which means the jQuery library is distributed around the world, and you will automatically receive the file from the closest server
- Google, Microsoft and the jQuery people do that

# jQuery CDNs

- For example, to link to the google jQuery CDN, you would do something like this:

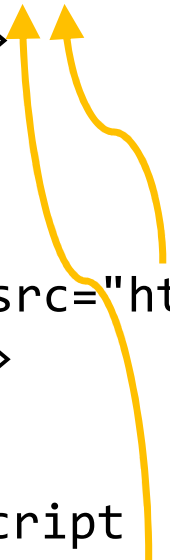
```
<head>  
...  
<script  
src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js">  
</script>
```

- To link to the Microsoft CDN, use this instead:

```
<script src="https://ajax.aspnetcdn.com/ajax/jquery/jquery-3.4.1.min.js">  
</script>
```

- To link to the jQuery CDN, use this instead:

```
<script  
  src="https://code.jquery.com/jquery-3.4.1.min.js">  
</script>
```



# Compressed or Uncompressed?

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

- min means the file is the 'minimum' version, sometimes called the 'compressed' version

- This version uses clever tricks to make the file much smaller e.g. no spaces unless necessary, variables names that use just 1 or 2 letters, etc
- The min version is about 88kB
- The regular version is about 290kB
- jQuery gets used billions of times a day, so this is good!

# Basic jQuery Use

- The basic way of coding in jQuery is similar to using DOM functions, i.e.:
  - Accessing some elements from the DOM
  - Then doing something with those elements
- Often jQuery code is triggered by an event
- We will briefly look at these ideas in this presentation



# An Example HTML File 1/2

```
<!DOCTYPE html>
<html>
<head>
  <title>Mac and Cheese</title>
</head>
<body>
  <h1 id="name">Mac and Cheese</h1>
  <h2>Ingredients</h2>
  <ul id="ingredients">
    <li>1 box of <a href="...">macaroni</a></li>
    <li>1/4 cup of <a href="...">butter</a></li>
    <li>1/4 cup of <a href="...">flour</a></li>
    <li>1/2 tsp of <a href="...">salt</a></li>
    <li>2 cups of <a href="...">milk</a></li>
    <li>2 cups of <a href="...">cheddar cheese</a></li>
  </ul>
```

- Some of the following jQuery code is created for this HTML file

*Links are not shown here, to save space*



- Continued on the following slide



# An Example HTML File 2/2

```
<h2>Directions</h2>
<ol id="directions">
  <li class="step">Cook the macaroni</li>
  <li class="step">Mix the butter, flour and salt in a saucepan</li>
  <li class="step">Add and stir the milk until thick</li>
  <li class="step">Slowly add the cheese until fully melted</li>
  <li class="step">Mix with the macaroni</li>
</ol>
</body>
</html>
```

*Using class is not just for doing pretty CSS things, it is also used for controlling behaviour*

# What it Looks Like

- This is what the example HTML file looks like in a browser

## Mac and Cheese

### Ingredients


- 1 box of [macaroni](#)
- 1/4 cup of [butter](#)
- 1/4 cup of [flour](#)
- 1/2 tsp of [salt](#)
- 2 cups of [milk](#)
- 2 cups of [cheddar cheese](#)

### Directions

1. Cook the macaroni
2. Mix the butter, flour and salt in a saucepan
3. Add and stir the milk until thick
4. Slowly add the cheese until fully melted
5. Mix with the macaroni

# Everything Starts From \$

- In jQuery, everything that you write starts from the '\$' symbol

 • This is sometimes called the *jQuery function*

\$ ...

# Everything Starts From \$

- The \$ symbol can be used to give you access to one or more things in the web page

```
var thebody = $("body");
```

get a node in the DOM structure

- The \$ symbol can be used to give you access to several useful functions by writing \$.

```
if ($.isNumeric("3.3")) ...
```

isNumeric is a jQuery function, not a function you defined before.

# jQuery Element Selector

- As you know, to access the DOM using DOM functions (using JavaScript, no jQuery), you can use `document.getElementById()` or `document.getElementsByTagName()`
- In jQuery, you use `$` to select elements using CSS selectors
- You learnt CSS selectors previously
- See the next few slides for examples

# Selecting Using Id

```
<h1 id="name" >  
    Mac and Cheese  
</h1>
```

- Here is a simple example of selecting an element from the DOM using the `id` attribute in jQuery:

- This is part of the example HTML file shown a few slides ago

看起来theh1和JS object没啥区别但其实有很多不同

```
var theh1 = $("#name");
```

- The above jQuery code gives access to the `<h1>` element and stores it in the `theh1` variable
- `theh1` is a jQuery object

# The jQuery Object

- One important thing you need to know is the DOM element returned by `$` is inside a jQuery object
- That means some things you saw before e.g. `innerHTML` have to be done differently in jQuery, see next slide



# Using the jQuery Object

- Let's extend the example code shown before
- It doesn't work this way:

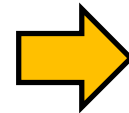
```
var h1 = $("#name");  
alert(h1.innerHTML);
```



This page says  
undefined

- In jQuery you do it like this:

```
var h1 = $("#name");  
alert(h1.html());
```



This page says  
Mac and Cheese

```
<h1 id="name" >  
    Mac and Cheese  
</h1>
```

# More jQuery Examples

- Selecting all `<h2>` in the HTML: `$("h2")`

`<h2>Ingredients</h2>`

`<h2>Directions</h2>`

- Selecting all `<li>` which use the class `step`:

`$("li.step")`

in CSS selector, "." means class

`<li class="step">Cook the macaroni</li>`

`<li class="step">Mix the butter, flour and salt in a saucepan</li>`

`<li class="step">Add and stir the milk until thick</li>`

`<li class="step">Slowly add the cheese until fully melted</li>`

`<li class="step">Mix with the macaroni</li>`

# More jQuery Examples

- Selecting all `<li>` under the unordered list called ingredients <sup>with id</sup> :

```
$("#ul#ingredients li")
```

```
<li>1 box of <a href="...">macaroni</a></li>
<li>1/4 cup of <a href="...">butter</a></li>
<li>1/4 cup of <a href="...">flour</a></li>
<li>1/2 tsp of <a href="...">salt</a></li>
<li>2 cups of <a href="...">milk</a></li>
<li>2 cups of <a href="...">cheddar cheese</a></li>
```

# Having Multiple Elements

- Most of the operations under the jQuery object can work with multiple elements
- If you want to know the number of elements in a jQuery object, you can read its `length` property, for example:

```
$("#title").length // returns 1
```

```
<title>Mac and Cheese</title>
```

```
$("#h2").length // returns 2
```

```
<h2>Ingredients</h2>
```

```
<h2>Directions</h2>
```

```
<li>1 box of <a href="...">macaroni</a></li>
```

- Once a jQuery object is obtained from a selection you can read or change its attribute using `.attr()`
  - For example, this will **read** the attribute of the **first** `<a>` element:

Change HTML attribute

```
$("a").attr("href")
```



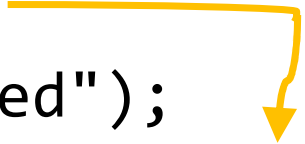
- This changes the attribute of **all** `<a>` elements:

```
$("a").attr("href", "https://en.wikipedia.org")
```

```
<li>1 box of <a href="...">macaroni</a></li>
<li>1/4 cup of <a href="...">butter</a></li>
<li>1/4 cup of <a href="...">flour</a></li>
<li>1/2 tsp of <a href="...">salt</a></li>
<li>2 cups of <a href="...">milk</a></li>
<li>2 cups of <a href="...">cheddar cheese</a></li>
```

## Attributes

# Working with CSS Styles

- You can read or change a jQuery object's CSS style using `.css()`
- For example, this will change the h1 to red:  
`$("#name").css("color", "red");`  
 `<h1 id="name">  
Mac and Cheese</h1>`
- This will change all h2 to red:  
`$("h2").css("color", "red");`  
 `<h2>Ingredients</h2>  
<h2>Directions</h2>`
- This will change all things which have the class step to red:  
`$(".step").css("color", "red");`  
  
`<li class="step">Cook the macaroni</li>  
<li class="step">Mix the butter, flour and salt in a saucepan</li>  
<li class="step">Add and stir the milk until thick</li>  
<li class="step">Slowly add the cheese until fully melted</li>  
<li class="step">Mix with the macaroni</li>`

# Working with CSS Styles

- For example, you can change **all** odd `<li>` elements which have class `step` to red:

```
$("#li.step:nth-child(odd)").css("color", "red");
```

➡ `<li class="step">Cook the macaroni</li>` ←

~~`<li class="step">Mix the butter, flour and salt in a saucepan</li>`~~

➡ `<li class="step">Add and stir the milk until thick</li>` ←

~~`<li class="step">Slowly add the cheese until fully melted</li>`~~

➡ `<li class="step">Mix with the macaroni</li>` ←

# Reading the Element Content

- You have previously seen the use of `.textContent` and `.innerHTML` (using JavaScript, no jQuery) to do something with the content of an HTML tag e.g. a `<p>`
- In jQuery, you can do that using `.text()` and `.html()`
- The main difference between `.text()` and `.html()`:
  - `.html()` can read, also can create, HTML tags
  - `.text()` can read, also can create, simple text



# Using .text() and .html()

```
<ul id="ingredients">  
  <li>1 box of <a href="...">macaroni</a></li>  
</ul>
```

- For example, reading the .text() content of the first ingredient:

```
$("#ul#ingredients li:first-child").text()
```

shows the text of  
the <li> element

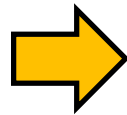


This page says

First ingredient is: 1 box of macaroni

OK

- If you change the  
text() to html()  
it will show the  
HTML code instead



This page says

First ingredient is: 1 box of <a href="https://en.wikipedia.org/wiki/Macaroni">macaroni</a>

OK

# Using Events

- jQuery code is typically run 'inside' events
- Perhaps the most common event is the `ready` event, which can be set up using jQuery like this:

```
<script>  
$(document).ready(function() {  
    alert("The page is ready for you!");  
});  
</script>
```

# The Ready Event

- The document ready event is very similar to the onload event you saw before
- The ready event will trigger when the DOM has finished loading, but it won't wait for things like images to be loaded. The onload event will only trigger after the DOM and all its components, including any images, have finished loading

This page says

The page is ready for you!

OK

# Using Other Events

- It is easy to attach event listener functions using jQuery
- All you need to do is to provide the name of the event and a callback to the `on()` function, like this:

```
$("#a").on("click", function() {  
    if (!confirm("Go to wiki?")) {  
        return false;  
    }  
});
```

- This is called the callback function

Apply the event to  
**all** `<a>` elements

The event is  
the click event

The callback  
function

```
$("#a").on("click", function() {  
    if (!confirm("Go to wiki?")) {  
        return false;  
    }  
});
```

! means not

If the user says no, i.e. clicking 'Cancel', the callback returns false and cancels the click event

A confirm box asks if the user wants to go to the wiki link

## An Example of on()

This page says

Go to wiki?

OK

Cancel

# Useful Effect Functions

- In simple HTML you can use the CSS `display` property to show or hide an HTML element e.g.

```
<p style="display:none">You can't see this!</p>
```

```
<p style="display:block">You can see this!</p>
```

- In jQuery, you can hide and show things nicely using these:

`.show()`, `.hide()` and `.toggle()` 切换

- These functions work with the `display` property
- They can also use some nice animations

- this means the thing that was clicked on

## .show() and .hide()

- Here is some example code to show or hide part of the HTML page by clicking on the <h2> elements

```
$( "h2" ).on( "click", function() {  
    var id = $(this).text().toLowerCase();  
    var mylist = $( "# " + id );  
    if ( mylist.css( "display" ) == "none" )  
        mylist.show();  
    else  
        mylist.hide();  
});
```

*Change any text to lower case letters e.g. CAT becomes cat*

*If the list under the h2 is invisible then show it;  
Otherwise it is visible, so hide it*

# Explanation

- If you click on this, the text 'Ingredients' is extracted, converted to lower case, and the element with that name 'ingredients' is shown/hidden

```
<h2>Ingredients</h2>  
<ul id="ingredients">  
  <li>1 box of <a href="...">macaroni</a></li>  
  . . .  
</ul>
```

- The other list has the same behavior:

```
<h2>Directions</h2>  
<ol id="directions">  
  <li class="step">Cook the macaroni</li>  
  . . .  
</ol>
```


- Click to show/ hide the list



# Using .toggle()

- jQuery also provides a `toggle()` function, which toggles between showing and hiding some elements
- The `if` statement shown works fine but can be simplified:

```
if (mylist.css("display") == "none")  
    mylist.show();  
else  
    mylist.hide();
```



```
mylist.toggle();
```

# Creating Animations

- jQuery can automatically add some nice animations when you show or hide elements
- You can do this by giving a duration (in milliseconds) to `.show()` , `.hide()` or `.toggle()`
- For example:

```
mylist.toggle(500);
```

- This code shows or hides the list using an animation that lasts for 500 milliseconds (=half a second)

# Changing the Animation

- The default animation contains fading and sliding movement
- If you only want the fading part, you can use:

`.fadeIn()`, `.fadeOut()` and `.fadeToggle()`

- If you only want the sliding part, you can use:

`.slideDown()`, `.slideUp()` and `.slideToggle()`