Session 5. jQuery

Programació Multiplataforma i Distribuïda

Grau d'informàtica. EPSEVG

Setembre 2023

Jordi Esteve jesteve@cs.upc.edu

jQuery

jQuery is a JavaScript library focused in the interaction between the HMTL DOM and JavaScript language that simplifies the code writing. Jquery is:

- Extremely concise
- Easy to learn

jQuery philosophy:

1. FIND something
2. DO something to it

```
$("p#id1").text("New text");
```

FIND DO

```
<p id="id1"></p>

      is converted to

<p id="id1">New text</p>
```

jQuery. How to use it

The jQuery library is loaded with the **script** tag placed at the end of the body of the Web page (and previous to other JavaScript libraries). Usually a compressed or minified version is enough (without comments, returns and unnecessary spaces).

1. It can be download from https://jquery.com/download/ and saved next to other JavaScript files of our Web page.

```
<script src="jquery-3.7.1.min.js"></script>
```

2. A jQuery CDN (Content Delivery Network) can be used to get the library very quickly. The content of a CDN is near to the end user so our server is not overloaded. And the same content can be used in different Web pages, so the content could be obtained from the cache of the browser.

```
<script src="https://code.jquery.com/jquery-3.7.1.min.js"
integrity="sha256-/JqT3SQfawRcv/BIHPThkBvs0OEvtFFmqPF/lYI/Cxo="
crossorigin="anonymous"></script>
```

   New browsers can check the integrity of the content they have received.

jQuery. $ function

jQuery library uses one function to do (almost) every operation:

jQuery("p#id1")

jQuery("div.c2 div:last")

It is called so often that **$()** function is an alias to **jQuery()** function:

$("p#id1")

$("div.c2 div:last")

jQuery. Selectors

CSS selectors	Magic selectors
$("p#id1")	$(":hidden")
$("div.c2")	$(":visible")
$("a[target]")	$(":header")
$("a[target=_blank]")	$(":input"), $(":text"), ...")
$("a[href^=http://]")	$("p:contains(Text)")
$("ul#id1 > li")	$("tr:even")
$("li#id1 ~ li")	$("tr:odd")
$("li:first-child")	$("li:first")
$("li:last-child")	$("li:last")
$("li:nth-child(3)")	$(":animated")

jQuery. Collections

$() function returns a **jQuery collection**. It is like an array and we can call methods on it:

```
$("p.c1").length;              // Number of matched elements
$("p.c1").size();              // Number of matched elements
$("p.c1").[0];                 // The first matched element
$("p.c1").[0].text();          // The text of the first matched element
$("p.c1").[0].text("Bye");     // Changes the text of the first matched element
$("p.c1").addClass("c2");      // Adds class c2 to the matched elements
$("p.c1").each((i, e) => console.log(`Paragraph ${i} has ${$(e).text()}`));
    // Prints the index and the text of the matched elements
$("p.c1").click(event => {$(event.target).hide(); return false;});
    // When the matched elements will be clicked, they will be hidden
```

Chaining methods

Most jQuery methods return another jQuery object, usually the same collection: The output of a method can be the input for next method.

```
$("div.c2").addClass("c3").css('color', 'green').show();
```

Some methods return a different collection. With **.end()** we revert to the previous collection.

```
$("div.c2").addClass("c3").find("a").css('color', 'green').end().find("p").hide();
```

One method, many uses

```
$("p.c1").html(); // Returns the HTML content of 1st matched element
$("p.c1").html("<em>Hi</em>"); // Sets HTML content of matched elements
$("p.c1").html((i, oldhtml) => oldhtml.toUpperCase());
// Sets HTML content of matched elements with the result of the function
```

jQuery. Methods (I)

Traversing: find(), first(), last(), end(), is(), filter(), map(), next(), nextAll(), prev(), prevAll(), siblings(), parent(), parents(), …

HTML: html(), text(), val()

```
$("p.c1").text("Hi");
$("p.c1").html("<em>Hi</em>");
```

Attributes: attr(), removeAttr(), css(), addClass(), removeClass(), toggleClass()

```
$("a#id1").attr("href", "https://www.upc.edu");
$("a#id1").attr({
      "href": "https://www.upc.edu",
      "target": "_blank"
});
$("a#id1").css({'color': 'green', 'font-style': 'italic'});
```

jQuery. Methods (II)

Effects: show(), hide(), toggle(), fadeIn(), fadeOut(), slideDown(), animate(), ...

```
$("div.c1").hide('fast');
$("div.c1").fadeOut('slow').slideUp(1000);
$("div.c1").animate({
    opacity: 0.25,
    left: "+=50",
    height: "toggle"
}, 5000, () => {// Actions to do when animation is completed.});
```

Moving elements: append(), appendTo(), before(), after(), …

```
$("div.c1").append("<p>Text</p>"); // Appends Text paragraph inside div.c1
$("<p>Text</p>").appendTo("div.c1"); // Appends Text paragraph inside div.c1
$("div.c1").before("<p>Text</p>"); // Puts Text paragraph before div.c1
$("div.c1").after("<p>Text</p>"); // Puts Text paragraph after div.c1
```

jQuery. Methods (III)

Events: on(), off(), trigger(), click(), …

```
// Adds an event handler: The function is called when "click" event happens on #id1
$("#id1").on("click", () => {
  alert( $(this).text() );
});

$("#id1").click(() => { // click() is an alias of on("click")
  alert( $(this).text() );
});

// A "click" event is triggered on #id1 like if it was triggered naturally by the user.
$("#id1").trigger("click");

$("#id1").off("click"); // Removes an event handler
```

jQuery. Methods returning first matched element

Some methods return the result from the first matched element:

```
let width = $('div.intro').width();
let height = $('div.intro').height();
let src = $('img.photo').attr('src');
let p_html = $('p').html();
let has_c1 = $('p').hasClass('c1');
let email = $('input:email').val();
let top = $('div.intro').css('top');
```

jQuery. Execution after the Web page is loaded.

We can use the **.ready()** method on document tag to pass a function that is executed after the Web page is loaded:

```
$(document).ready(function() {
    alert('The DOM is ready!');
});
```

It can be simplified using the **$()** function:

```
$(function() {
    alert('The DOM is ready!');
});
```

You can review your knowledge about jQuery doing these online resources:

- **jQuery quiz**
- **jQuery exercises**

jQuery. Memory game exercise

Simplify the code of the memory game created at the end of previous session:

1. Add the jQuery library to the memory_game.html file.
2. Simplify the creation of the initial random board by using jQuery functions instead of the document/element JavaScript ones.
3. Simplify the code that shows the card when it is clicked and hides it one second later by using jQuery functions.

Now you know all the tools necessary to finish the memory game that allows to play, turning couples of cards until all the pairs with the same figure are discovered.

You can improve the game adding a counter of the number of moves (turning a couple of cards is one move) and a clock showing the time spent. You can change the difficulty level allowing different board sizes, for example 8, 12, 16 cards.