# JQUERY

softserve

## Revision

softserve

## Agenda

jQuery

## WHAT IS JQUERY

jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating and Ajax interactions for rapid web development.

jQuery is a set of typical JavaScript helpers encapsulated in one object.

https://w3techs.com/technologies/overview/javascript_library

softserve

## WHAT IS JQUERY

jQuery library developed by **John Resig**.

v 1.0 of library released on November 26th, 2006.

Now jQuery is maintained as an **Open Source** software.

## FEATURES OF JQUERY

- Cross-browser engine
- DOM element selections
- DOM traversal and modification.
- Events handling.
- CSS manipulation.
- Effects and animations
- AJAX technology

## HOW TO USE JQUERY?

In general base syntax looks like:

```
Find elements.Do something
```

Or more concretely:

```
$(selector).method();
```

Shortcut **$** usually used for calling jQuery object

## SELECTORS

jQuery supports selectors in CSS style:

| | |
|---|---|
| **$(el)** | - selects existing element |
| **$(“*”)** | - selects all elements *(rarely used)* |
| **$(“div”)** | - selects by tag name |
| **$(“#test”)** | - selects by id |
| **$(“.test”)** | - selects by class-name |
| **$(“div#test”)** | - complex selects by tag and id |
| **$(“div.test”)** | - complex selects by tag and class |

## SELECTORS

jQuery supports selectors in CSS style:

| | |
|---|---|
| **$("div *")** | – selects all descendants |
| **$("div span")** | – selects descendants by tag name |
| **$("div>span")** | – selects children by tag name |
| **$("div+img")** | – selects next siblings |
| **$("[checked]")** | – selects by attribute name |
| **$("[type='button']")** | – selects by attribute value |

## SELECTORS

jQuery also supports filters and methods:

| | |
|---|---|
| **$(“selector:filter”)** | – applying of filter |
| **:first, :last, :even, :odd** | – filter by position |
| **:empty, :contains(), :has()** | – filter by content |
| **:eq(**i**)**, **:lt(**i**), :gt(**i**)** | – filter by index |
| **:hidden, :visible** | – filter by displaying |

## SELECTORS

jQuery also supported filters and methods:

| | |
|---|---|
| **$("selector").method("selector")** | – applying of method |
| **.find()** | – selects descendants |
| **.prev(), .next()** | – selects siblings |
| **.children()** | – selects all descendants |
| **.parent()** | – selects direct parent |

## EVENTS

Base syntax of event's assigning in jQuery `$(selector).method(event, handler);`

Methods:

| | |
|---|---|
| **on()** | – add event handler |
| **off()** | – remove event handler |
| **one()** | – add event handler and remove it after first using |

Old Methods [deprecated!]:

| | |
|---|---|
| **bind()** | – add event handler if element exist |
| **live()** | – add event handler if element not exist |

## EVENTS

Base syntax of event’s assigning in jQuery

`$(selector).method(event, handler);`

Events:

**“click”, “”dbclick”, “contextmenu”, “change”, “blur”, “submit”, “mouseup”** and etc.

Handler:

Any function that should be called if event occurred

softserve

## EVENTS

Base syntax of event’s assigning in jQuery $(selector).method(event, handler);

For example:

```
$(“#btn”).on(“click”, function() {
        console.log(“button click detected”);
      });
```

*//message in console will be provided after clicking on button with id “btn”*

softserve

## BUBBLING AND DEFAULT ACTION

For stopping bubbling and prevention of default actions **return false** can be used; instead of native **e.preventDefault()** and **e.stopPropagation()** from JavaScript.

But object **e** still a first parameter of event handler and can be processed if it is needed.

## DOCUMENT READY

jQuery can process synthetic event "ready". This event will execute handler when the DOM is fully loaded:

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

Or more shortly:

```
$(function() { … });  // recommended
```

# BASIC JQUERY METHODS

Some of big collection jQuery methods

For work with content:

| | |
|---|---|
| **.text()** | – Sets or returns the text-content of element |
| **.html()** | – Sets or returns the html-content of elements |
| **.val()** | – Sets or returns the value of form fields |
| **.attr()** | – Sets or gets attribute values |

For work with CSS:

| | |
|---|---|
| **.css()** | – Sets or returns the style attributes [not recommended] |
| **.addClass()** | – Add one of more style classes |
| **.removeClass()** | – Remove one or more selected classes |
| **.toggleClass()** | – Toggles between adding/removing classes |

## BASIC JQUERY METHODS

Some of big collection jQuery methods

For DOM manipulation:

| | |
|---|---|
| **.append()** | – Inserts content at the end of the elements |
| **.prepend()** | – Inserts content at the beginning of the elements |
| **.after()** | – Inserts content after the elements |
| **.before()** | – Inserts content before the elements |
| **.remove()** | – Removes the element (and all its children) |
| **.empty()** | – Removes all children elements |

And many-many other:

**.trigger(), .show(), .hide(), .animate(), .fadeIn(), etc.**

## OPTIMIZATION

If you need to use some DOM-element in different times it will be better to find it once and cash.

```
$("#input").text("some text");
. . .
$("#input").text("any text");
```

# OPTIMIZATION

If you need to use some DOM-element in different times it will be better to find it once and cash.

```
$(“#input”).text(“some text”);
. . .
$(“#input”).text(“any text”);
```

```
let  $el = $(“#input”);
. . .
$el.text(“some text”);
. . .
$el.text(“any text”);
```

# .EACH()

**.each()** - iterates over any collection, executing function-handler for every matched element.

Base syntax:

```
$(selector).each(handler);
$.each(collection, handler);
```

Format of handler:

```
function(index, element) { . . . };
```

## .EACH()

```
$(“li”).each(function(i, el) {
                console.log($(el).text());
           });
```

```
<ul>
  <li>item 1</li>
  <li>item 2</li>
  <li>item 3</li>
</ul>
```

# .EACH()

```
$(“li”).each(function(i, el) {
        console.log($(el).text());
    });
```

```
<ul>
  <li>item 1</li>
  <li>item 2</li>
  <li>item 3</li>
</ul>
```

```
item 1
item 2
item 3
```

## .EACH()

```
$("li").each(function(i, el) {
            console.log($(el).text());
        });
```

```
<ul>
  <li>item 1</li>
  <li>item 2</li>
  <li>item 3</li>
</ul>
```

```
item 1
item 2
item 3
```

// **Note**: second parameter
// of handler is not  jQuery
// Wrapper $(el) is required
// for using jQuery methods

## AJAX IN JQUERY

**.ajax()** – universal method for providing ajax-technology

Base syntax:

**$.ajax(**options**);**

Options:

hash with parameters for setting ajax-dialog:

*data, dataType, type, url, success and error callbacks, etc.*

## AJAX IN JQUERY

```
$.ajax({
    url: '/server/give-me-list-of-cats',
    dataType : ‘json’,
    success: function (list_of_cats) {
        $.each(list_of_cats, function(I, cat) {
            console.log(cat.name);
        });
    }
});
```

softserve

## WRAPPERS

Also jQuery can propose some usable wrappers:

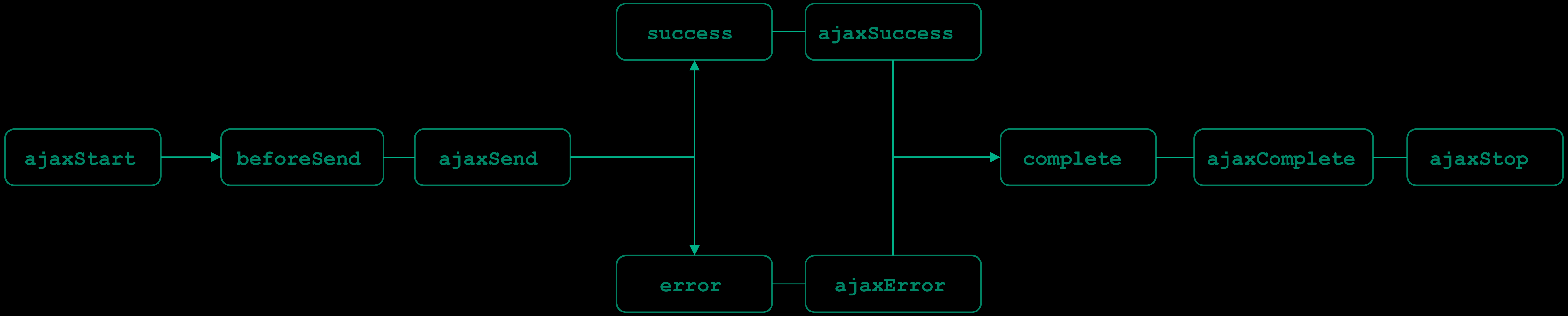
| | |
|---|---|
| **get()** | – sends GET request |
| **post()** | – sends POST request |

Parameters:

| | |
|---|---|
| **url** | – string with URL to which the request is sent. |
| **data** | – optional object or string that is sent to the server |
| **successCallback** | – executed if the request succeeds. |
| **dataType** | – the type of data expected from the server |

## AJAX EVENTS

jQuery supports next event model for ajax-dialog:



Global events: *ajaxStart, ajaxSend, ajaxSuccess, ajaxError, ajaxComplete, ajaxStop*

Internal events: *beforeSend, succes, error, complete*

# AJAX EVENTS

Global event can be used in method **on()**:

```
$(el).on("ajaxSuccess", function() { … });
```

Internal events used by method **ajax**:

```
$.ajax({
    . . .
    error: function() { … }),
    . . .
});
```

softserve

# THANKS FOR YOUR ATTENTION

softserve