

# Advanced JavaScript

## Part 2

[jan.schulz@devugees.org](mailto:jan.schulz@devugees.org)

# 1. Agenda

1. Introduction to JQuery
2. The six core functions of JQuery
3. Benefits of JQuery
4. Introduction to NPM
5. `$()` and `jQuery()`
6. Selectors
7. DOM Traversal
8. Filter
9. Chaining
10. Native DOM Objects

# 1. Introduction to JQuery

- Is a JavaScript library
  - Not a framework
- Automates a bunch of common VanillaJS tasks
  - „1 Line of Code“
- 10 years old
- Cross-Browser Functionality

# 1. Introduction to JQuery



# 1. Introduction to JQuery



## 2. Six Core Functions

### 1. Access DOM elements

`$('div')`

-> access all DIV elements on the page

## 2. Six Core Functions

### 2. Modify the appearance of a page

```
$('div').addClass('highlight');
```

-> add class highlight to all divs

```
$('div').removeClass('highlight');
```

-> remove class highlight from all divs

## 2. Six Core Functions

### 3. Alter the content of a page

```
$('div').append('<div>Hallo World</div>');
```

-> adds a div with Hallo World to all divs



## 2. Six Core Functions

### 4. Animate web page changes

```
$('#div').fadeOut();
```

-> each div on the page fades out and disappears

```
$('#div').slideUp();
```

-> each div on the page slides up and disappears

## 2. Six Core Functions

### 5. Retrieve data from the server

```
$('div').load('content.html');
```

-> loads the content of an HTML files into a div

## 2. Six Core Functions

### 6. Respond to user interaction

```
$('#div').click( function() {  
    alert('Hallo World');  
});
```

-> when clicking on any div, alert Hallo World

# 3. Benefits of JQuery

## 1. Minimal Code

- JQuery operates on **sets of elements**

## 2. Large Library

- Wide variety of inventive and useful modules

## 3. Strong community

- Stack Overflow = Your new best friend

## 4. Cross-Browser Support

- Chrome, Firefox, IE, Safari

## 5. AJAX Support

- Easier than in VanillaJS

# 3. Benefits of JQuery

- Biggest benefit:

**Everybody uses it.**

# 4. Introduction to NPM

- **NPM = Node Package Manager**
  - Manages (Mainly Installs and Removes) packages for **NodeJS** applications
- **NodeJS = Serverside JavaScript**
  - More on that later ...
- Let us install watch-http-server  
**sudo npm install -g watch-http-server**

## 5. `$()` and `jQuery()`

- `$()` is a synonym for `jQuery()`
- `$()` or `jQuery()`
  - selects a DOM element
  - wraps jQuery functionality around it
  - returns it

## 5. \$() and jQuery()

- \$() is a synonym for jQuery()
- \$() or jQuery()
  - selects a DOM element
  - wraps jQuery functionality around it
  - returns it
- **It is not the native JavaScript Object for DOM**



# 6. Selectors

- Most Used Basic Selectors

Selector	Description
*	Selects all elements
.CLASSNAME	Selects all elements with a specific class = CLASSNAME
#IDNAME	Select one element with a specific id = IDNAME
TAG1 TAG2	Selects all TAG2 elements which are children to TAG1 i.e. <b>li a</b> would give all anchor-tags within
TAG1 > TAG2	Selects all TAG2 elements which are <b>direct children</b> to TAG1 i.e. <b>li &gt; a</b> would give all anchor-tags within as
TAG.CLASSNAME	Selects all TAG elements which have the class CLASSNAME
TAG#IDNAME	Selects all TAG elements which have the class IDNAME
TAG1 + TAG2	Selects the first adjacent TAG2 element after the TAG1 element
TAG1, TAG2	Selects all Tags with TAG1 and all tags with TAG2

# 6. Selectors

- Advanced Advanced Selectors

Selector	Description
TAG:gq(0)	Selects all elements of TAG which have index > 0
TAG:eq(0)	Selects all elements of TAG which have index = 0
TAG:lt(2)	Selects all elements of TAG which have index < 0
TR:even	Selects all elements of TR which have even index
TR:odd	Selects all elements of TR which have odd index
TR:nth-child(even)	Selects all elements of TR which have even index
TR:nth-child(odd)	Selects all elements of TR which have odd index
TD:contains('Hallo')	Selects all elements of TD which contain a string 'Hallo'

# 7. DOM Traversal

- Each selected Element can be a starting point for further DOM traversal
    - Traversal = Walking along a path
  - An **element A** can be
    - Parent
    - Child
    - Direct Sibling (by having the same direct parent)
    - Indirect Sibling (by having the same indirect parent)
- ... To an **element B**

# 7. DOM Traversal

`$(E).parent()`

-> gives the direct parent of **E**

`$(E).parents()`

-> gives all parents of **E**

`$(E).children()`

-> gives all children of **E**

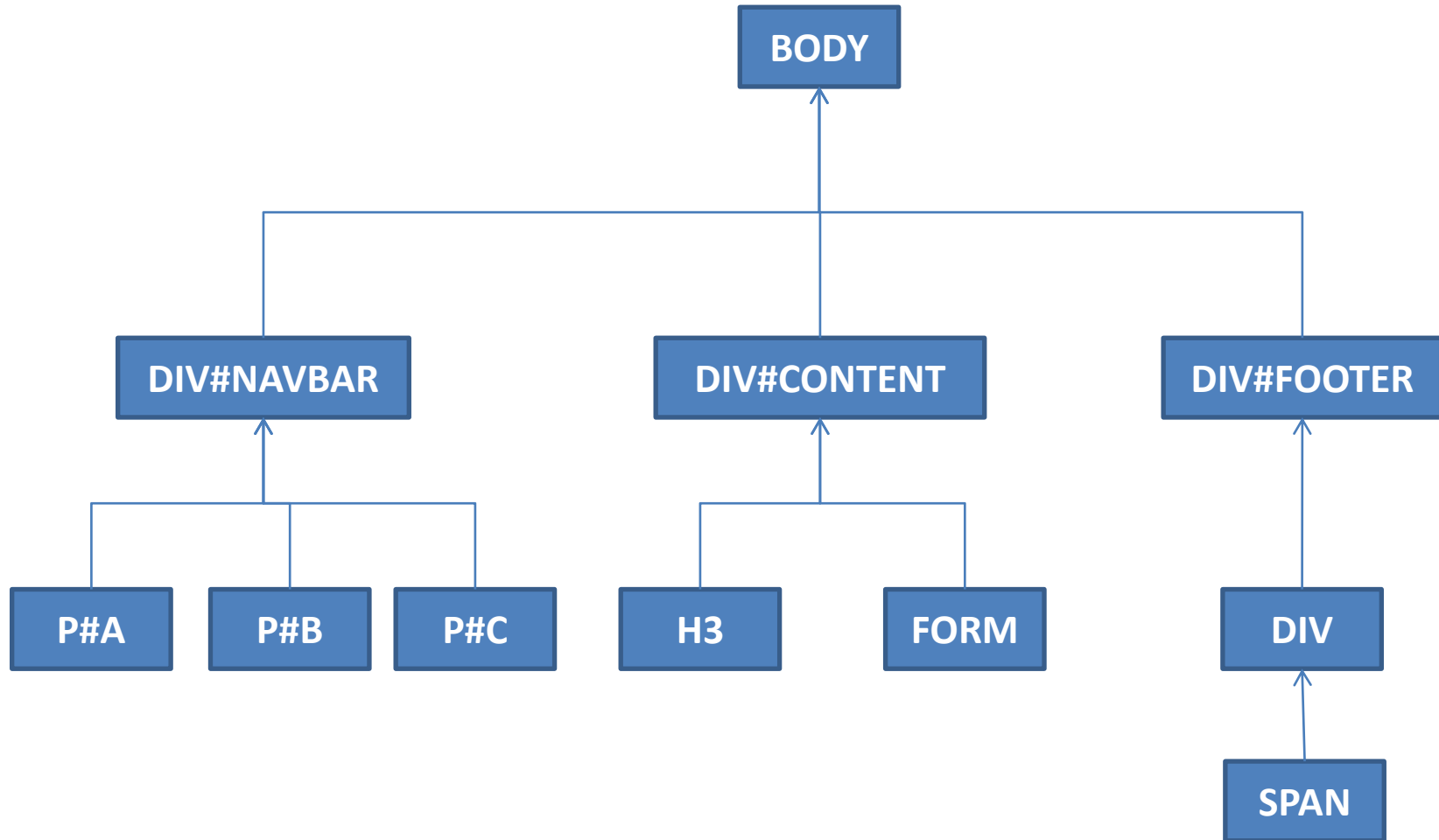
`$(E1).parentsUntil(E2)`

-> gives all parent elements from **E1** to **E2**  
except **E1** and **E2**

`$(E1).closest(E2)`

-> either the closest sibling or the closest  
parent that matches **E2**

# 7. DOM Traversal



# 7. DOM Traversal

`$(E).next()`

-> gives the next direct sibling of **E**

`$(E).prev()`

-> gives the previous direct sibling of **E**

`$(E).nextAll()`

-> gives **all** next direct siblings of **E**

`$(E).prevAll()`

-> gives **all** previous direct siblings of **E**

`$(E).siblings()`

-> gives **all** siblings, including **E**

## 7. DOM Traversal

`$(E).nextAll().first()`

-> gives the **first** direct sibling of **E**

`$(E).nextAll().last()`

-> gives the **last** direct sibling of **E**

# 8. Filter

- Define a function that filters out selected DOM elements
- If the function returns true, the element is considered

```
$(E).filter(function() { return true; })
```

-> considers all **E**, no filter

```
$('a').filter(function() {  
    if (this.hostname.indexOf('google') !== -1)  
        return true;  
});
```

-> Selects all Links that point to Google



## 9. Chaining

- Each JQuery selector returns an Object that methods, too
- Calling a methods returns an Object again (and again, ...)

```
$('div.hallo').parent().closest('p').find('h3.world')
```

# 10. Native DOM Objects

- JQuery does not return native DOM Objects, since it returns a JQuery wrapper around it
- Nevertheless, Native DOM Objects can be extracted using JQuery

```
var el = $('div')[0];
```

```
var el = $('div').get(0);
```