JS Libraries & jQuery overview

by Aleksandr Motsjonov

soswow@gmail.com

TALLINK SILJA LINE

Agenda

- Demo – Teaser
- Chapter I – JS Libraries
- Chapter II – What is jQuery
- Chapter III – Deep into the Code
- Demo – Full Version

Demo – Teaser

Chapter I – JS Libraries

JS Libraries

- Reasons. Why JS Libraries in the first place?
- Lot of them. Question of choose.
- Some interesting stuff.

Why JS Libraries?

- Less code
- More free time
- More money left
- Easier to write

Why JS Libraries?

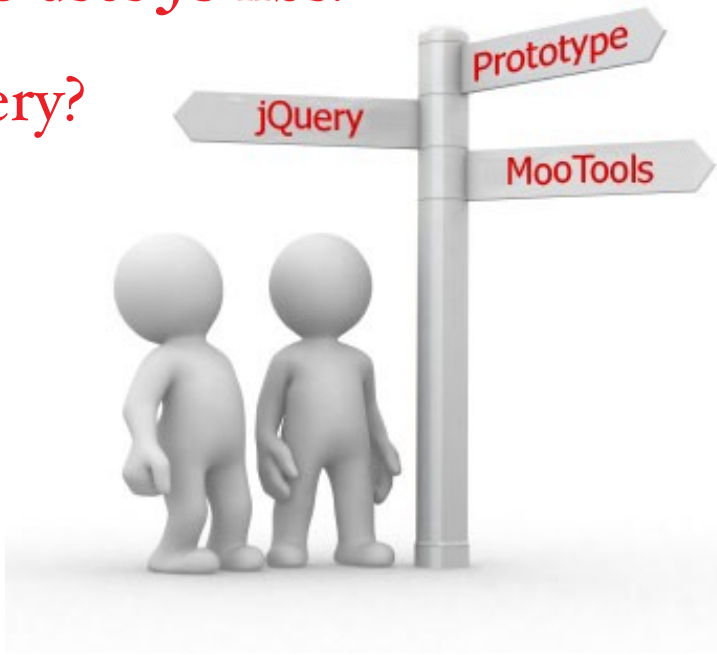
- JS Lib as cross browser abstraction layer
  - Less headache
  - Less bugs
  - Less time spend
  - More money left

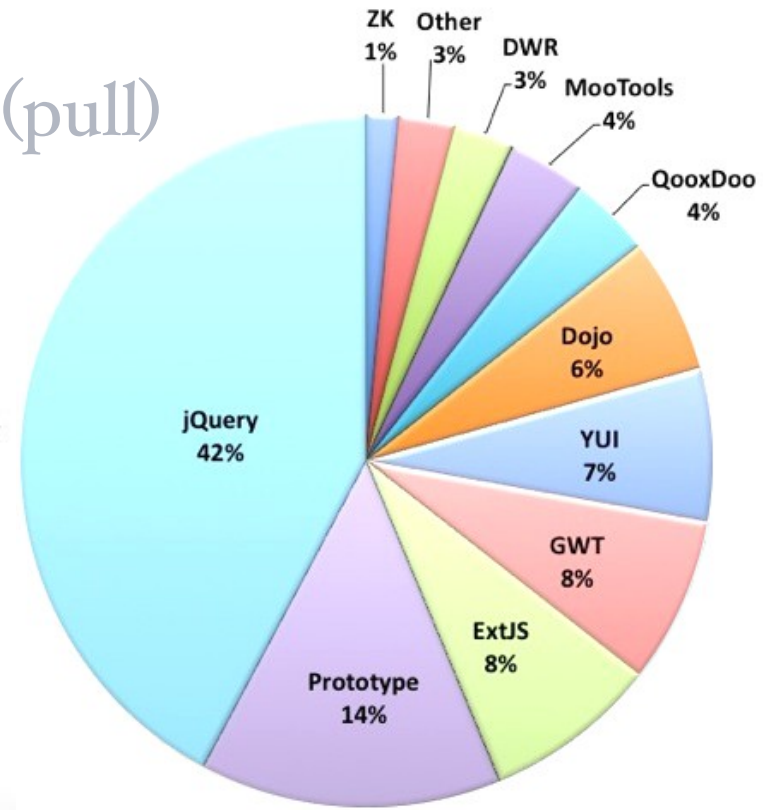
Why JS Libraries?

- No need to reinvent the wheel.
  - Methods like selectByClassName
  - All other utility methods already written (Plug-ins)

Which one to choose?

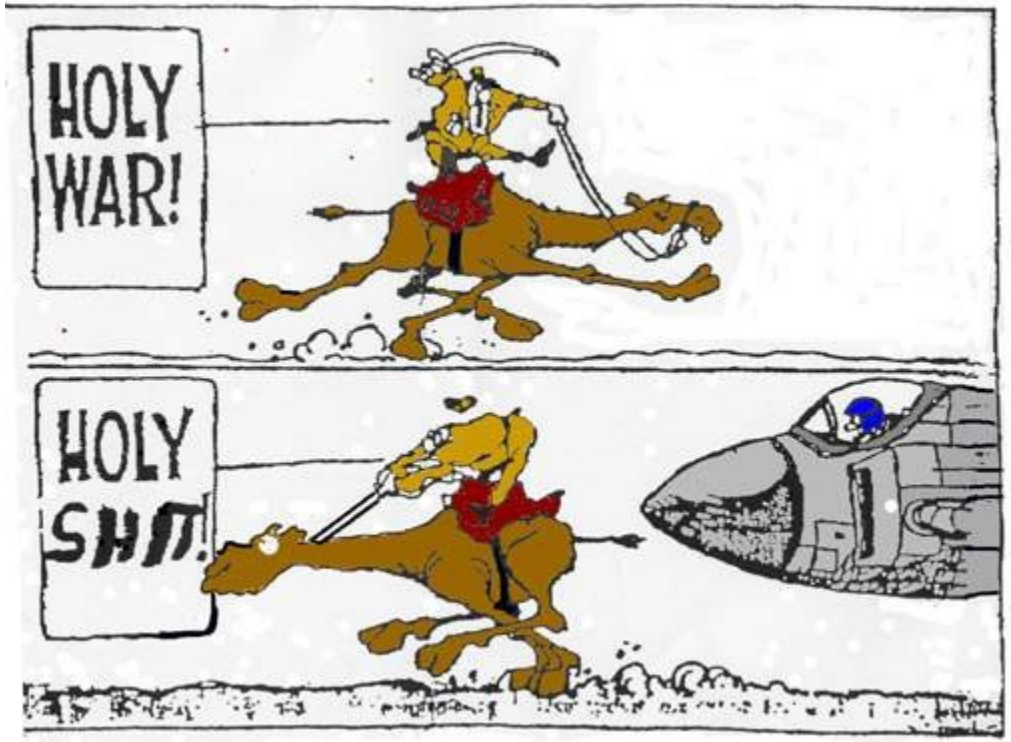
- There is a lot of JS Libraries. (pull)
  - Who uses JS libs?
  - jQuery?





Choose of JS Library

- Everyone have right to choose whatever she likes.
- I respect other libs and criticism about jQuery.



Choose of JS Library

- Different needs – different JS libs.
  - Working with DOM
  - OOP Abstraction
  - UI Toolkit
  - Ajax framework
  - General Purpose

Choose of JS Library

- Why I choose jQuery?
  - Honestly I’ve been starting with Prototype =)
  - http://www.rsdn.ru/article/inet/jQuery.xml

```
//Plain JavaScript
var tables = document.getElementsByTagName("table");
for ( var t = 0; t < tables.length; t++ ) {
  var rows = tables[t].getElementsByTagName("tr");
  for ( var i = 1; i < rows.length; i += 2 )
    if ( !/(^|s)odd(s|$)/.test( rows[i].className ) )
      rows[i].className += " odd";
}
```

Last things about JS libs

- ECMAScript 5.0
- Google AJAX Libraries API
  - http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js

Now You will know who and why need to choose jQuery.

Chapter II – What is 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.
- Write less, do more.
- http://jquery.com/
- Author: John Resig
- Last Release: v. 1.3.2
- Size (production): 19kb

jQuery

- Inside the Code
- Outside the Code
- Additional Bonuses

What jQuery has Inside Code?

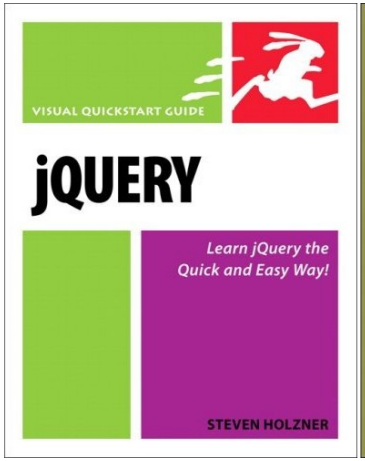
- DOM Manipulation
  - Node/Attributes Modification
  - Traversing
- CSS Queries
- Ajax
- Simple Visual Effects
- Utility methods
- Custom Plug-ins development

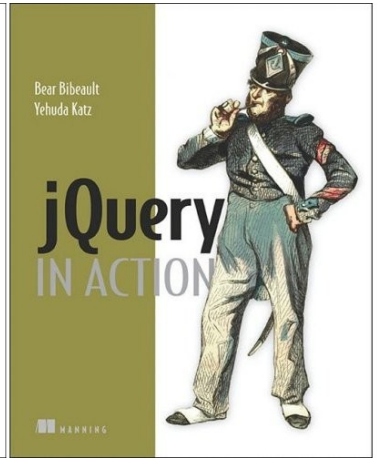
Outside code - Popularity

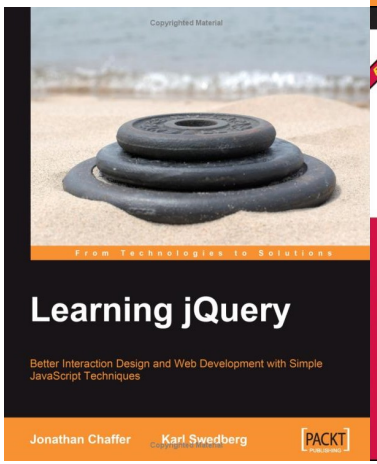
- Who Uses jQuery?
  - Most popular sites: Google, Amazon, IBM, Twitter, Dell, NBC, BBC, Digg, Intel, Oracle, Slashdot, Technorati, SourceForge, GitHub, Science, New York Post, FeedBurner, ...
  - OpenSource projects: Mozilla, WordPress, Drupal, Trac, Joomla, Symfony, Zend Framework, PEAR, ...

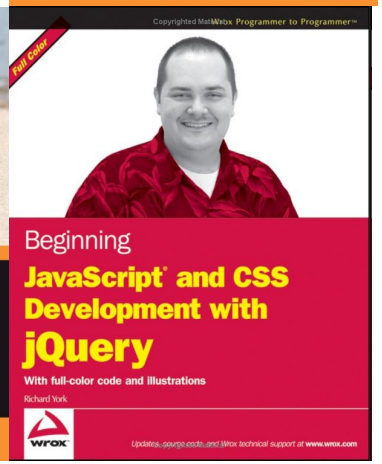
Outside code - Popularity

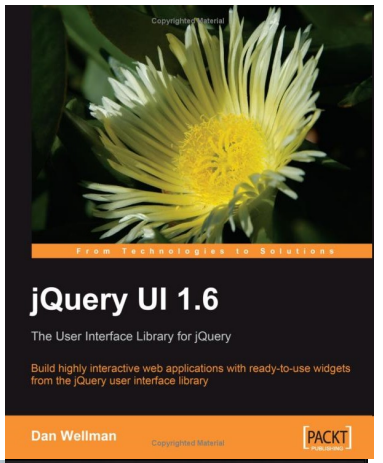
- Big Community
  - 3200 messages in Google Groups
  - 2600 Plug-ins in central repository
  - Countless number of tutorials and articles
- Books





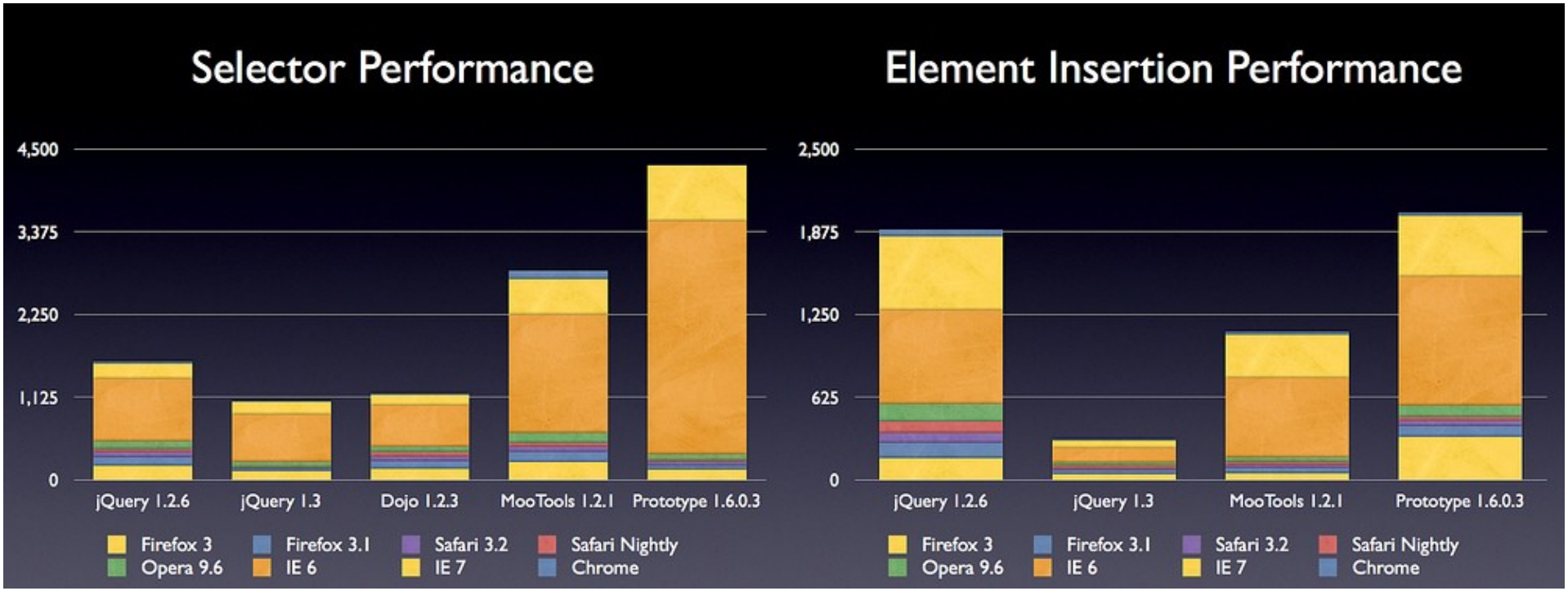


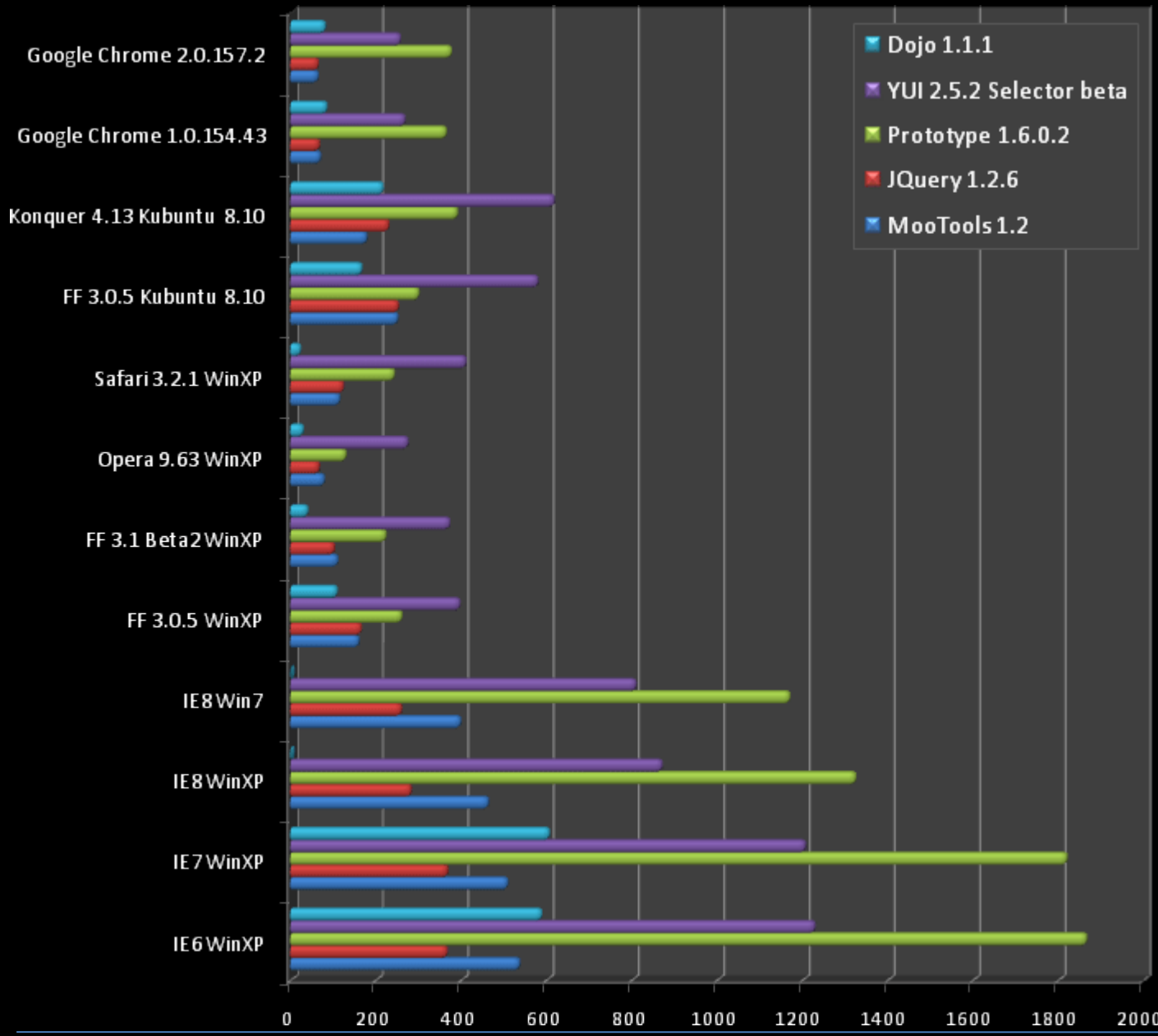




Additional Bonuses

Additional Bonuses - Performance





Additional Bonuses

- Browser Compatibility
  - Firefox 2.0+
  - Internet Explorer 6.0+
  - Safari 3+
  - Opera 9+
  - Chrome 1+
  - Known Issues in (FF 1.0, IE 1-5, Safari 1-2, Opera 1-8, lynx =)
- jQuery Loves other Libraries =)
  - noConflict()
  - Don't touch standard Objects (Object, String, itc.)

Chapter III – Meet the jQuery Code

Deep into the Code

- CSS Query
- DOM
- Events
- Ajax
- Other

Basics

- jQuery Object concept
  - Almost everything in jQuery is jQuery Objects. Every jQuery method you call from such objects.
  - Almost every method return jQuery object. You can make chains with it.

$ is the Mother of jQuery

- $ - is shorthand for jQuery object/function – main object, which need to be called to use jQuery. $ can be called with 4 types of arguments:
  - HTML Text.
    For creating new DOM element (returns jQuery object)
  - CSS Selector.
    For searching and retrieving existing DOM elements.
  - DOM element.
    For wrapping it with jQuery to get jQuery object.
  - Function.
    Shorthand for document onDOMLoad event handler.

$ - Examples

```
//jQuery(expression, context) Returns: jQuery
jQuery("TABLE.bigTable A"); //Returns jQuery object
$("TABLE.bigTable A");  //Returns jQuery object

//jQuery(html, ownerDocument)    Returns: jQuery
$("<div>");  //return Created DOM Element (wraped in jQuery)
$("<div id='someId' class='someClass'>");
//New DOM element with some Attributes

//jQuery(elements)   Returns: jQuery
var element = document.getElementById("someId"); //DOM element
$(element); //jQuery Object

//jQuery(callback)
$(function(){
   //This will be called on DOM onload event (Not Document onload)
});
```

CSS as Query language for DOM tree

- jQuery (like many other JS Libraries) uses CSS as Query language to get DOM elements.
- Main method to make CSS Queries is $(), but there is also methods, which arguments also CSS statement.
- $() could return 0, 1 or many elements, which match CSS statement. No exceptions.

Main methodology of jQuery

```
$("BODY DIV:visible").hide("slow");
```

Get (Find) some jQuery elem. Do something with it.

CSS Support

- CSS 1
- CSS 2
- CSS 3 (subset)
- Custom CSS selectors

CSS Selectors in Action

- Basic
- Hierarchy
- Basic Filters
- Content Filters
- Visibility Filters
- Attribute Filters
- Child Filters
- Forms
- Form Filters

Basic Selectors

```
$("#someElementId"); //1 element with such ID

$("TABLE"); //All Table elements

$(".someClassName"); //All with such Class Name

$(".firstClass.secondClass"); //All which have both classes

$("*"); //All DOM elements on the page.

//(Grouping) All tables AND all elements with such class
$("TABLE, .someClass");
```

Hierarchy Selectors

```
$("DIV SPAN"); //All SPAN in all DIV (at any level)

$("DIV > SPAN"); //All SPAN in all DIV (at NEXT level)

$("H1 + P"); //P which next immediate sibling to H1

$("H1 ~ P"); //All P which are sibling to H1
```

Basic Filters

```
$("DIV P:first"); //Only First P inside DIV (also :last)

$("P:not(.someClass)"); //All P but not with such class

$("TR:odd TD"); //All TD in odd ROWS (also :even)

$("TR:eq(0) TD:gt(1)");
//All TD which index Greater then 1 in First TR
//(with index 0)
//Also :lt(index) for less then
```

Content Filters

```
$("P:contains(someText)"); //P with such text

$("P:empty"); //Empty P

$("P:has(SPAN.someClass)");
//P which have inside itself SPAN with such class

$("P:parent"); //P if its parent to something else
```

Visibility Filters

```
$("DIV:visible"); //Visible DIVs

$("DIV:hidden"); //Hidden DIVs
```

Attribute Filters

```
$("INPUT[name=someName]"); //INPUT such name value

$("INPUT[name*=form]"); //Contains

$("INPUT[name$=Name]"); //Ends with

$("INPUT[name^=some]"); //Begins with

$("INPUT[name!=someName]");
//Input without name or it has other value

$("INPUT[name=someName][value=1]"); //With such name and
value attributes
```

Child Filters

```
$("TR:nth-child(1) TD, TR:nth-child(even) TD");
//All TD in 1st, 2nd, 4th, 6th, etc rows

$("TR:nth-child(3n+2) TD");
//All TD in 2nd, 5th, 8th, 11th, etc rows

$("TD:first-child"); //First child TD of every its parent
//Also :last-child, :only-child
```

Forms

```
$(":input");       //<input />
$(":text");        //<input type="text" />
$(":password");    //<input type="password" />
$(":radio");       //<input type="radio" />
$(":checkbox");    //<input type="checkbox" />
$(":submit");      //<input type="submit" />
$(":image");       //<input type="image" />
$(":reset");       //<input type="reset" />
$(":button");      //<input type="button" />
$(":file");        //<input type="file" />
$(":hidden");      //<input type="hidden" />
```

Form Filters

```
$("INPUT:enabled"); //<input />

$("INPUT:disabled");//<input disabled="disabled" />

$("INPUT:checked"); //<input type="radio" checked="checked" />

$("INPUT:selected");//<option selected="selected" />
```

DOM Traversing

- Filtering
- Finding
- Chaining

DOM Traversing - Filtering

```
$("TD").eq(0); //Leave only one element.
$("DIV").slice(1, 3); //LEave elements from 1 to 3

$("DIV, SPAN").filter(":visible");
//Leave only Visible DIVs and SPANs in result
$("INPUT").filter(function(){
   if($(this).val() > 5){
      return true;
   }else{
      return false;
   }
});
$("DIV").not("#someId, .someClass");
//Filters out #someId and .someClass divs from init set
$("INPUT").map(function(){
   return $(this).val();
}); //Returns Array of values of all inputs.

$("#hiddenTextId").is(":visible"); //Return Boolean!
```

DOM Traversing - Finding

```
$("TD").filter(".someClass").add("#someId");
// == $("TD.someClass, #someId");

$("TR").children(".someClass");
// == $("TR > .someClass");

$("span").closest(".parentPuk");
//Closest Parent witch match expression

$("P:eq(0)").contents(); //REturns All child element (Including
text nodes)

$("TABLE, DIV").find("SPAN"); //Find inside.

$("span").parents(".parentPuk"); //Get All Parents
```

Chaining

```
$("#someId")                  //DIV#someId
   .find("H1")                //H1
   .addClass("selected")      //H1
.end()                        //DIV#someId
   .find(">p:visible")        //P, P
      .andSelf()              //DIV#someId, P, P
      .addClass("selected1")  //DIV#someId, P, P
   .end()                     //P, P
.end();                       //DIV#someId
```

Attributes

```
$("H1").attr("title"); //get title attribute
$("H1").attr("title","Hello To You"); //Set title attribute
$("TABLE").attr({ height: 200, width: 400});
//Set Attributes with object map

$("H1").addClass("selected");
$("H1").hasClass("selected"); //Return Boolean
$("H1").removeClass("selected");
$("H1").toggleClass("selected");

$("DIV").html(); //innerHTML equivalent
$("DIV").html("some text"); //set innerHTML

$("#someId").text();  //get all Text nodes.

$("INPUT").val(); //Get value property
$("INPUT").val(newvalue); //Set new value
```

DOM Manipulation

append

appendTo

preppend

preppendTo

after

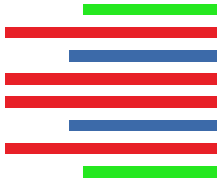
insertAfter

before

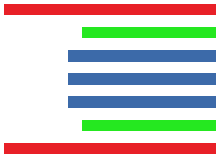
insertBefore

wrap

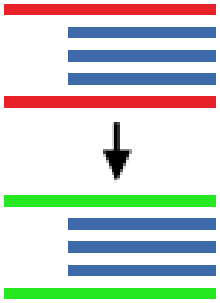
wrapAll



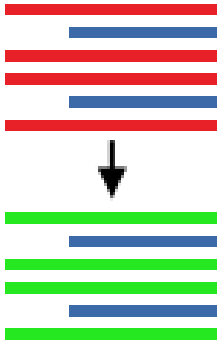
wrapInner



replaceWith



replaceAll



```
$("#ID").append("<div></div>");
$("div").appendTo("#ID");
```

DOM Manipulation

```
$("DIV").empty();

$("DIV").remove();

$("DIV").clone();
```

CSS

```
$("DIV").css("background-color");

$("DIV").css("background-color", "#fff");

$("DIV").css({
   backgroundColor: "#FFF",
   color: "#F00"});
```

Positioning

```
$("DIV").offset();
//Returns: Object{top,left} Relative to the
document.

$("DIV").position();
//Returns: Object{top,left} Relative to its
offset parent.

$("DIV").scrollTop();    //Returns: Integer
$("DIV").scrollTop(val);   //Returns: jQuery

$("DIV").scrollLeft();  //Returns: Integer
$("DIV").scrollLeft(val);  //Returns: jQuery
```

Height and Width

```
$("DIV").height();    //Returns: Integer
$("DIV").height(val);    //Returns: jQuery

$("DIV").width(); //Returns: Integer
$("DIV").width(val); //Returns: jQuery

//With Paddings
$("DIV").innerHeight(); //Returns: Integer
$("DIV").innerWidth();  //Returns: Integer

//With Margins
$("DIV").outerHeight(margin); //Returns: Integer
$("DIV").outerWidth(margin);  //Returns: Integer
```

Events

```
$("input:button").bind("click", function(){
   console.log("Do it always");
}).one("click", function(){
   console.log("Do it once");
}).live("click", function(){
   console.log("Now and forever");
}).hover(
   function(){
      console.log("Mouse Over");
   }, function(){
      console.log("Mouser Out");
});

$("input:button").trigger("click");
$("input:button").unbind("click");
```

Events

```
//Trigger events
blur(), change(), click(), dblclick(), error(),
focus(), keydown(), keypress(), keyup(), select(),
submit()

//Event Handlers
select(fn), keyup(fn), load(fn), mousedown(fn),
mouseenter(fn), mouseleave(fn), mousemove(fn),
mouseout(fn), mouseover(fn), mouseup(fn),
resize(fn), scroll(fn), keypress(fn), keydown(fn),
focus(fn), error(fn), dblclick(fn), click(fn),
change(fn), blur(fn), submit(fn), unload(fn)
```

Ajax

```
$.ajax( {
    type : "POST",
    url : "some.php",
    data : "name=John&location=Boston",
    success : function(msg) {
        alert("Data Saved: " + msg);
    },
    error : function(msg) {
        alert("Error: " + msg);
    }
});
```

- Function: success, beforeSend, complete, dataFilter, error, xhr
- Boolean: async, cache, ifModified, processData, global
- String, Object: contentType, data, dataType, jsonp, password, scriptCharset, type, url, username

Ajax

```
//Load a remote page using an HTTP request.
jQuery.ajax(options)
//Load HTML from a remote file and inject it into the DOM.
load(url, data, callback)
//Load a remote page using an HTTP GET request.
jQuery.get(url, data, callback, type)
//Load JSON data using an HTTP GET request.
jQuery.getJSON(url, data, callback)
//Loads, and executes, a local JavaScript file using an HTTP
GET request.
jQuery.getScript(url, callback)
//Load a remote page using an HTTP POST request.
jQuery.post(url, data, callback, type)
```

Other

- Effects
  - show(), hide(), slideDown(), fadeIn(), animate()
- Utilities
  - jQuery.support, .each(), .map(), .inArray(), .unique(), .isFunction(), .isArray()
  - .get(i), .eq(i), .index()

Demo – Full Version

The End.

TALLINK SILJA LINE