PODSTAWY JavaScript i jQuery

Paweł Kaczmarek

Zastosowanie JavaScript

- Modyfikacja wyglądu strony
- Modyfikacja zawartości strony
- Dodanie dynamiki strony
- Asynchroniczne wołania do serwera (AJAX) – real-time:
  - Auto-complete
  - Odświeżanie zawartości strony
  - Walidacja formularzy
  - Komunikacja z API

Zasady JS

- Case sensitive
- Nazwy muszą zaczynać się od litery, _ lub $
- Bloki kodu określamy nawiasami klamrowymi { ... }
- Linie kończymy średnikami ;

Best Practices

- CamelCase
- * Piszemy po angielsku (nie używamy polskich znaków)
- Sensowe nazewnictwo

Słowa kluczowe

abstract	arguments	boolean	break	byte
case	catch	char	class*	const
continue	debugger	default	delete	do
double	else	enum*	eval	export*
extends*	false	final	finally	float
for	function	goto	if	implements
import*	in	instanceof	int	interface
let	long	native	new	null
package	private	protected	public	return
short	static	super*	switch	synchronized
this	throw	throws	transient	true
try	typeof	var	void	while

http://www.w3schools.com/js/js_reserved.asp - Pełna lista

Osadzanie JavaScript - w pliku HTML

Index.html

```
<script>
        console.log('test');
</script>
```

Osadzanie JavaScript - w osobnym pliku .js

Index.html

<script **src = " libFile.js "** ></script>

libFile.js

console.log('test');

tag <script>

Osadzamy wewnątrz:

- body
- head

Osadzanie JavaScript - czego nie robić

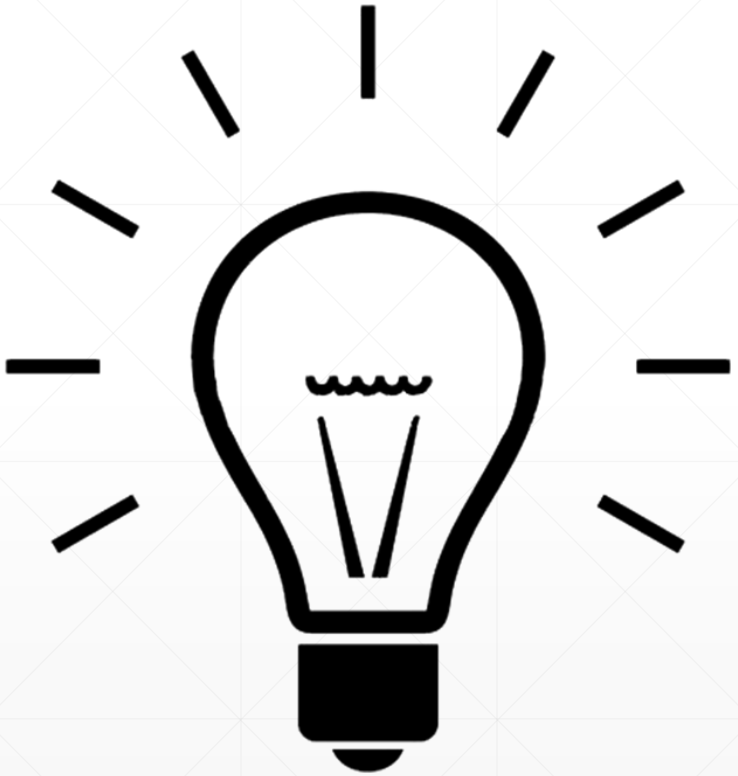
```
<script>
        var text = 'tekst w konsoli'
        console.log(text);
</script>
```

Ten kod się wykona

```
<script  src = " libFile.js " >
        var text = 'tekst w konsoli'
        console.log(text);
</script>
```

Ten kod się nie wykona

0.Sandbox



jQuery

jQuery?

- Bilblioteka JavaScript - JavaScript Query
- Służy do wyszukiwania elementów na stronie i wykonywania na nich akcji
- Czasami upraszcza programowanie w JavaScript
- Działa pod wszystkimi przeglądarkami
- Znajduje się w pojedynczym pliku *.js

Konstrukcja zapytania jQuery

$(*selector*).*action*( )

- $ - zmienna globalna, w której trzymane jest jQuery
- selector - wartość textowa, string, definiuje jakich elementów szukamy
- action - akcja wykonana na wszystkich znalezionych elementach

Przykłady zapytań jQuery

$(*selector*).*action*( )

- $('p').hide()
- $('div').show()
- $('div.news').addClass('przeczytany')

Zapytania jQuery zwracają kolekcje elementów

```
<p class="klikalny">Element 1</p>
<p class="klikalny">Element 2</p>
<p class="klikalny">Element 3</p>
<p class="klikalny">Element 4</p>
```

$(*'p'*)

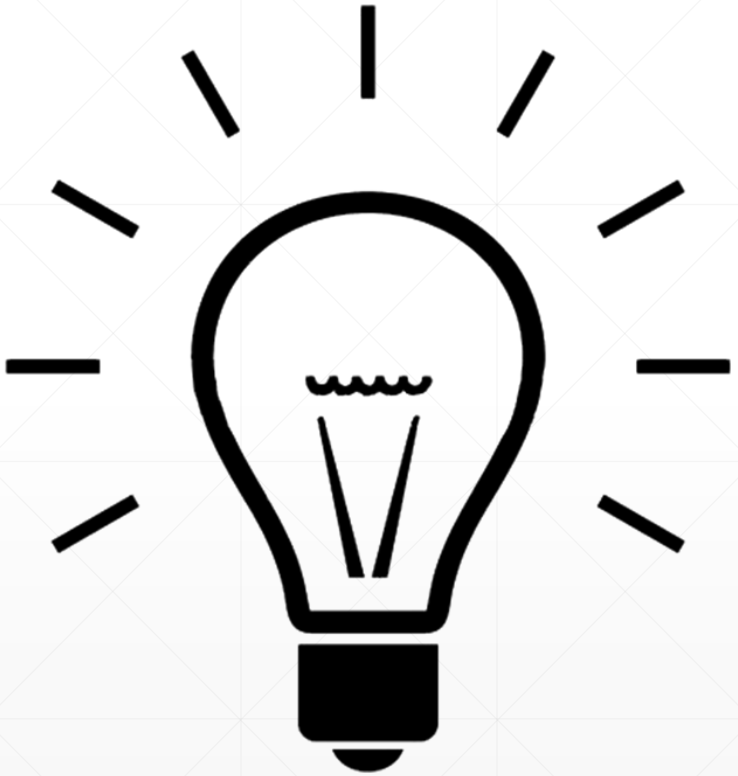
```
$('p')
[  <p class="klikalny">Element 1</p>,
   <p class="klikalny">Element 2</p>,
   <p class="klikalny">Element 3</p>,
   <p class="klikalny">Element 4</p>]
```

jQuery - Proste selectory

- $('div')
- $('.artykul')
- $('#naglowek')
- $('div.artykul')
- $('div, p')
- $('div p')
- $('div p.artykul')

http://api.jquery.com/category/selectors/ - Pełna lista

1.jq.warmup



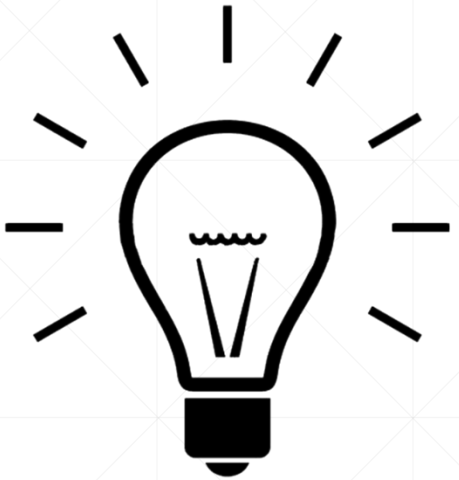
Events (zdarzenia)

Eventy HTML

- Zdarzenia przytrafiające się elementom HTML
- onchange, onclick, onmousedown….
- **asynchroniczne !!**

http://www.w3schools.com/jsref/dom_obj_event.asp - Pełna lista eventów

Obsługa eventów – przez atrybut HTML



- <div **onclick**=" **console.log ('kliknięcie') "** >Click on this text!
- <div **onclick**=" **mojaFunkcjaGlobalna() "** >Click on this text!

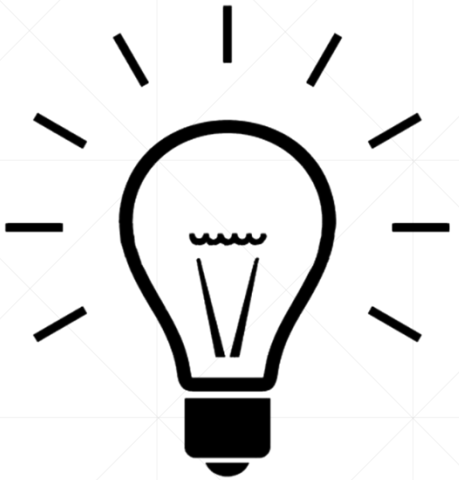
jQuery - Obsługa eventów

- click
- dblclick
- change
- mousedown
- hover
- keydown

```
$(      selector      ).action( )
```

```
$( '.nazwaKlasy' ).click ( function() {    } );
$( '.nazwaKlasy' ).dblclick( function() {    } );
$( '.nazwaKlasy' ).mousedown ( function() {    } );
$( '.nazwaKlasy' ).change( function() {    } );
```

jQuery - Obsługa eventów



```
$( '.nazwaKlasy' ).click ( function() {
        $(  this  ).hide();
} );
```

anonymous functions and callbacks

Funkcja nazwana

function **nazwaFunkcji** () { }

nazwaFunkcji ();

Funkcja anonimowa

function () { }

...

Funkcja nazwana

function **nazwaFunkcji** () { }

nazwaFunkcji ();

Funkcja anonimowa

function () { }

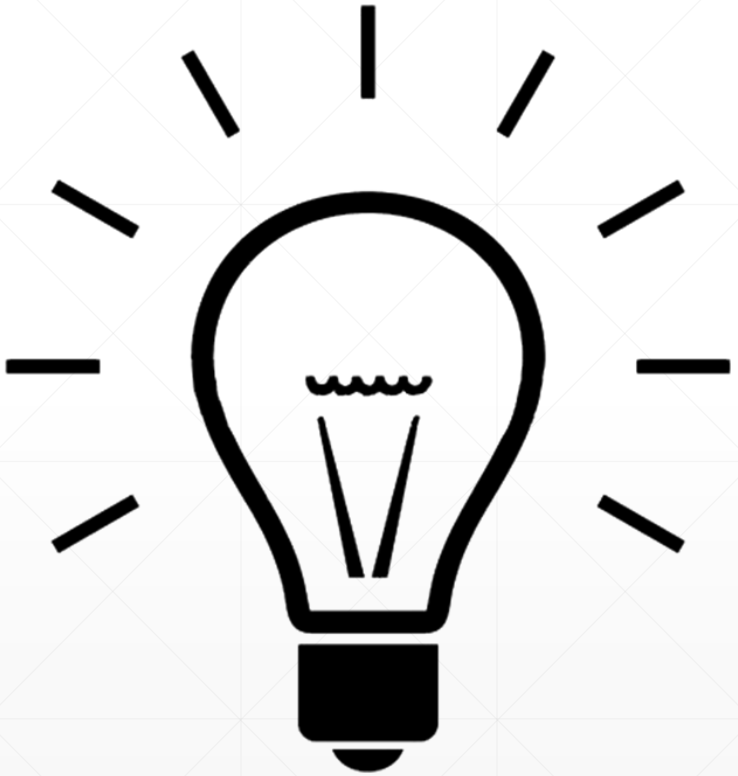
var **nazwaFunkcji** = *function* () { }

nazwaFunkcji ();

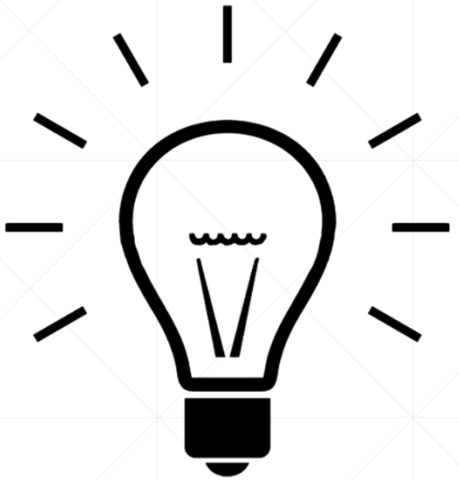
callback

- funkcja przekazana do innej funkcji jako parametr
- bardzo często jest funkcją anonimową

2.jq.events



jQuery - funkcja .find()



```
$( '.nazwaKlasy' ).click ( function() {
        $(  this  ).find('.podelement').hide();
} );
```

jQuery - pokazywanie, ukrywanie elementów

var el = $('naszSelector');

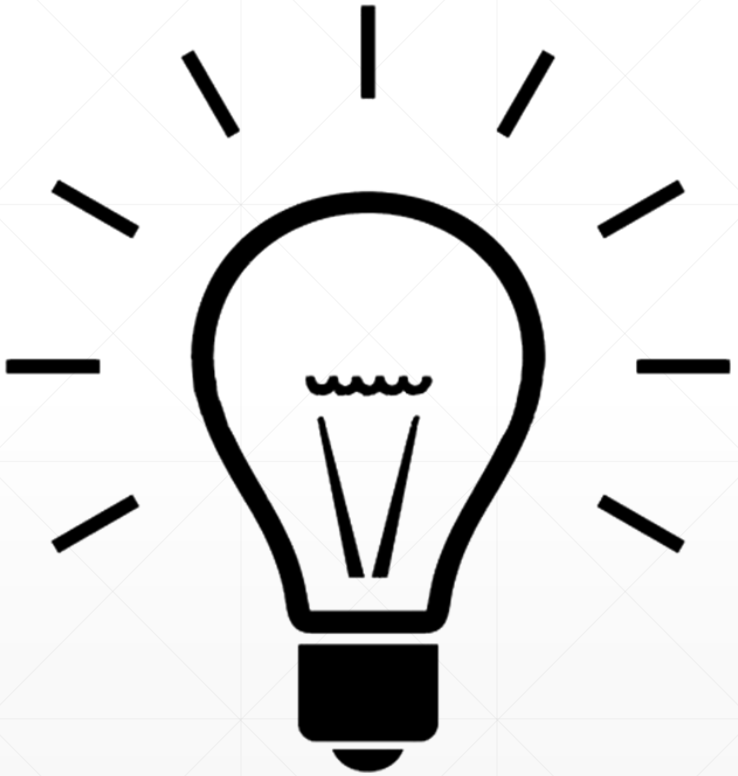
- el.show()
- el.hide()
- el.fadeIn(*400*)
- el.fadeOut(*400*)
- el.fadeToggle(*400*)
- el.fadeTo(*400, 0.5*)
- el.slideUp(*500*)
- el.slideDown(*500*)
- el.slideToggle(*500*)
- el.stop()

jQuery - Atrybut class

var els = $('naszSelector');

- els.hasClass(*'klikalny'*) - ***true***, jeżeli **przynajmniej jeden** element posiada klasę *'klikalny'*
- els.addClass(*'klikalny'*) - dodaje klasę *'klikalny'* do wszystkich elementów
- els.removeClass(*'klikalny'*) - usuwa klasę *'klikalny'* do wszystkich elementów
- els.toggleClass(*'klikalny'*) - dodaję/usuwa klasę *'klikalny'* do/z wszystkich elementów

3.jq.classes

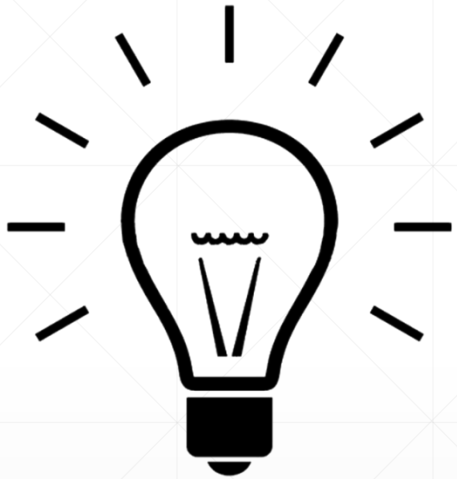


Uruchamiamy zapytania jQuery dopiero po załadowaniu całego dokumentu

```
$( document ).ready( function() {

    $(  'p'  ).click(....)

})
```

jQuery - tworzenie nowych elementów HTML

```
var nowyDiv = $('<div>'),
    nowyParagraf = $('<p>');
```



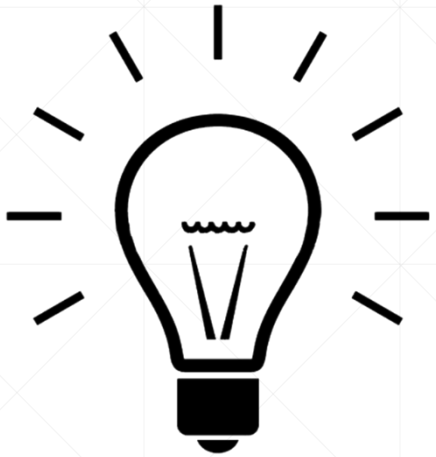
jQuery - Kolejkowanie funkcji - method chaining

```
var newElement = $('<div>').addClass('hero').text('Batman');
```

jQuery - funkcja append()

```
var newElement = $('<div>').addClass('hero').text('Batman');
$('#heroes').append(newElement);
```

- dodawanie dynamicznie stworzonych elementów do strony



jQuery- dokumentacja

http://api.jquery.com

Array (tablica)

Array - tablice

```
var myTable  =  [  ] ;

var liczby =  [ 10 ,  11,  12 ] ;

var cars  =  [ 'Saab', 'Volvo',  'BMW' ];
```

array.forEach()

```
var heroes = ['Batman', 'Robin', 'Gordon'];

heroes.forEach(function(hero, index) {
    console.log(hero, index);
})
```

- wywołuje funkcję dla każdego elementu tablicy
- za każdym wywołaniem przekazuje do funkcji dwa parametry:
  - pierwszy - aktualny element tablicy
  - index - index aktualnego elementu tablicy

array.map()

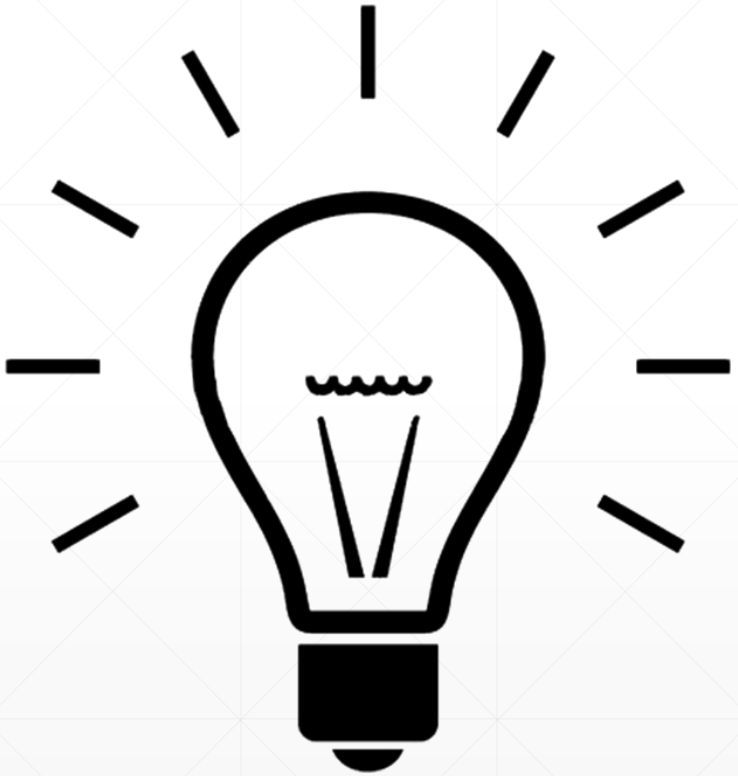
```
var heroes = ['Batman', 'Robin', 'Gordon'];

var modifiedHeroes = heroes.map(function(hero) {
    return 'Hero - ' + hero;
});
```

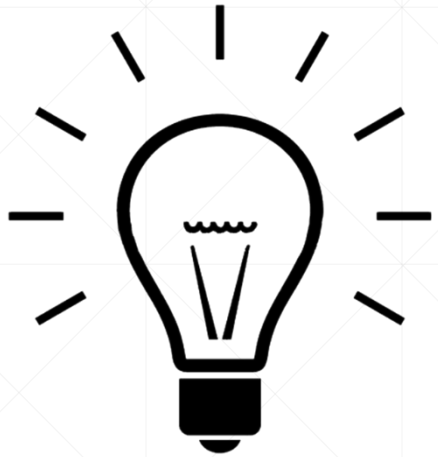
- tworzy nową tablicę
- nowa tablica zawiera nowe elementy, stworzone w oparciu o elementy ze starej tablicy
- stara tablica pozostaje niezmieniona

Array - Kolejkowanie funkcji - method chaining

```
['Batman', 'Robin', 'Gordon']
    .map(function (hero) {
        return 'Hero - ' + hero;
    })
    .forEach(function(hero) {
        console.log(hero);
    })
```

4.Array.forEach





Array - dokumentacja

https://developer.mozilla.org

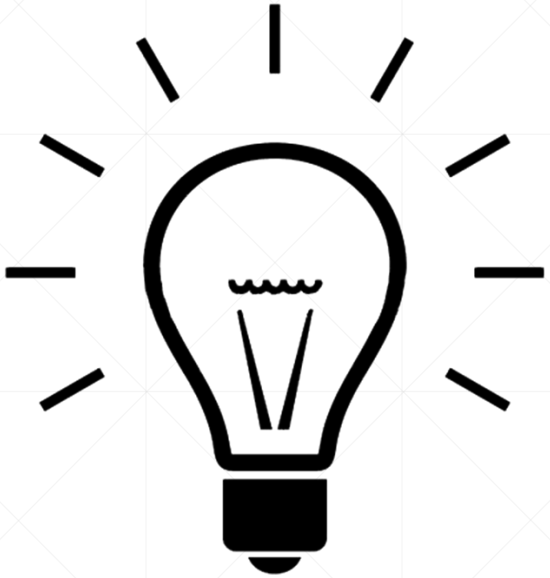
Array

Do elementów tablicy dostajemy się po indeksie

Array – odczyt

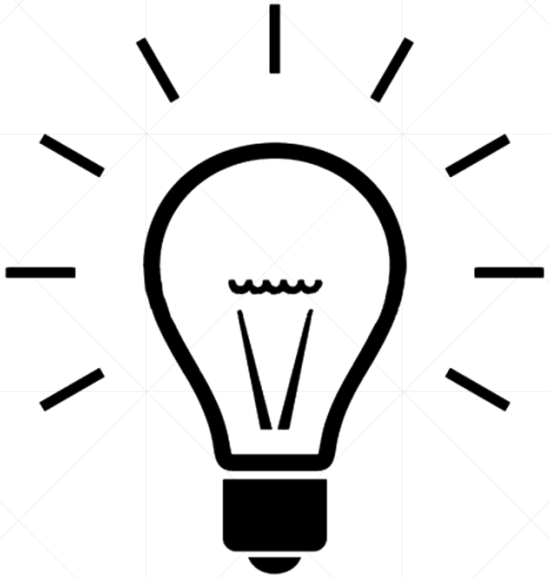
```
var cars  =  [  'Saab',  'Volvo',  'BMW'  ];
```

```
cars [ 0 ]   ===   'Saab'
cars [ 1 ]   ===   'Volvo'
cars [ 2 ]   ===   'BMW'
```



Array – zapis

```
var cars  =  [ 'Saab',  'Volvo',  'BMW' ];


cars [ 2 ]  =  'Niemiecki wóz'
cars [ 3 ]  =  'Trabant'
cars [ 4 ]  =  'Czarna Wołga'
```



Array.length - liczba elementów w tablicy

```
var a = [ ];
var b = [ "Freeze" ];
var c = [ "Batman", "Robin", "Freeze", "Riddler" ]
```

a**.length** == 0

b**.length** == 1

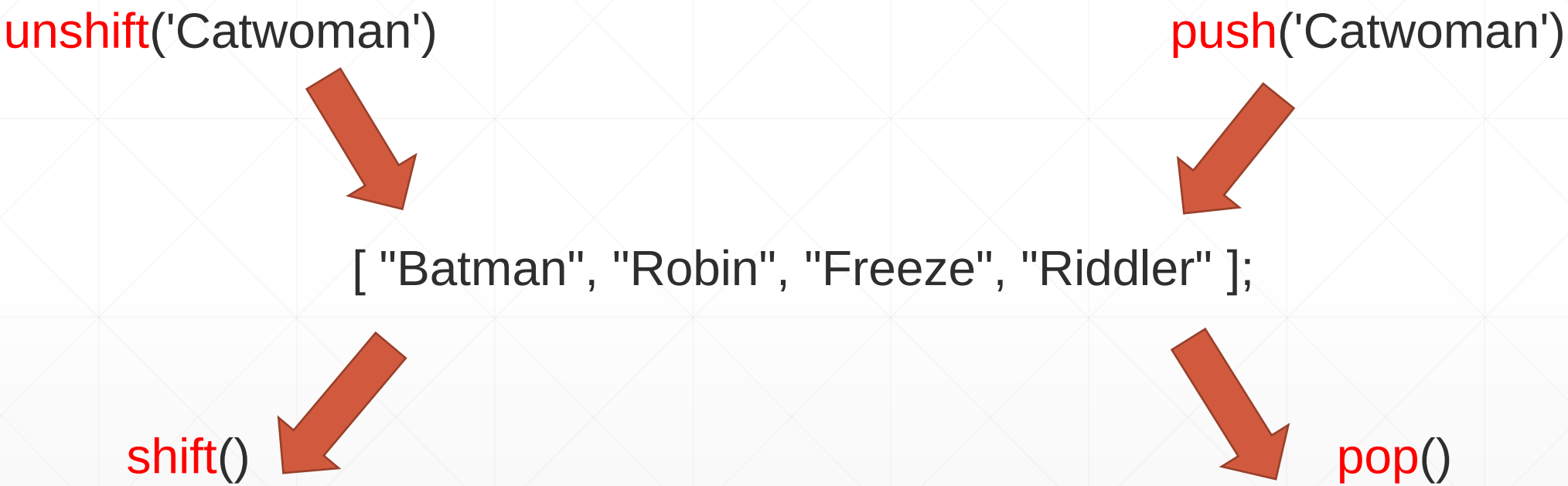
c**.length** == 4

Array.concat - sklejanie tablic

```
var a = [ "Batman", "Robin" ];

var b = [ "Freeze", "Riddler" ];

var c = a.concat(b)
```

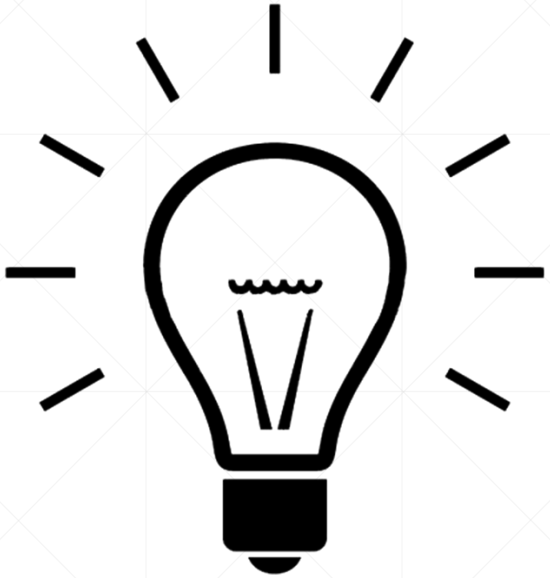
```
a == [ "Batman", "Robin" ];

b == [ "Freeze", "Riddler" ];

c == [ "Batman", "Robin", "Freeze", "Riddler" ];
```

Array - pop, push, shift, unshift



Array - metody

```
var  heroes  =  [ "Batman", "Robin", "Freeze", "Riddler" ];
heroes.length;
heroes.sort();
heroes.push(  "CatWoman"  );
heroes.pop();
heroes.shift();
heroes.unshift(  "Batman" );
heroes.concat(  [ "Gordon",  "Oracle” ]  );
```



jQuery - animacje

```
$(selector).animate(
        {
                parametr_1_css:    wartość_docelowa_1  ,
                parametr_2_css:    wartość_docelowa_2  ,
                parametr_3_css:    wartość_docelowa_3
        },
        czasTrwaniaWMilisekundach
);
```

Animacje - przykład

```
$('#mojIdentifikator') . animate( {
        opacity: 0.6,
        width: 400
}, 3000 );
```

4.animacje