Wprowadzenie do jQuery

W celu wykorzystania możliwości biblioteki jQuery należy:

a) pobrać bibliotekę ze strony projektu *http://jquery.com/* lub po prostu umieścić URL do pliku z biblioteką (ze strony *https://code.jquery.com/*) w dokumencie HTML:

```
<script   src="https://code.jquery.com/jquery-2.2.3.min.js"
  integrity="sha256-a23g1Nt4dtEYOj7bR+vTu7+T8VP13humZFBJNIYoEJo="
  crossorigin="anonymous">
</script>
```

b) w pliku *html* w znaczniku <*script*> (lub pliku z JS) umieścić kod:

```
<script>
$(document).ready(function () {
      //akcje do wykonywania po załadowaniu strony
});
</script>
```

c) przetestować działanie biblioteki wykonując prostą akcję na znaczniku <*body*> polegającą na dodaniu tekstu „Test jQuery”:

```
<script type="text/javascript">

$(document).ready(function () {
      $('body').text('Test jQuery!');
});
</script>
```

Wywołanie akcji/funkcji w jQuery wygląda następująco:

```
$('element').akcja('parametr');
```

gdzie:

`$()` – alias do funkcji o nazwie jQuery(), tworzącej obiekt jQuery;

`element` – wybieranie elementów w oparciu o selektory CSS;

`akcja` – nazwa akcji, która będzie wykonana na wybranych elementach;

`parametr` – parametry dla danej akcji.

W tabelach 1-4 przedstawiono podstawowe elementy niezbędne do pracy z biblioteką.

Tab. 1. Wybrane selektory (http://api.jquery.com/category/selectors/)

Funkcja	Opis	Przykład
('*')	Wyszukiwanie wszystkich elementów niezależnie od ich typu.	
('tag')	Wyszukiwanie wszystkich elementów będących wskazanym znacznikiem HTML.	`$("body")` `$("div,p,span")`
('.klasa')	Wyszukiwanie wszystkich elementów ze wskazaną klasą CSS.	`$(".innnerClass")`
('#id')	Wyszukiwanie elementu o identyfikatorze id.	`$("#myDiv")`
('przodek potomek')	Wyszukiwanie elementów potomek, które mają wskazanego przodka.	`$("div span")`
('rodzic > dziecko')	Wyszukiwanie elementów dziecko, które mają wskazanego rodzica.	`$("ul.topnav > li")`
('elem[atrybut]')	Wyszukiwanie elementów zawierających wskazany atrybut.	
('elem[atrybut=wartosc]')	Wyszukiwanie elementów zawierających atrybuty ze wskazaną wartością.	`$('input[value="Java Script"]')`
(':first'), (':last'),	Pseudoklasa dopasowująca się do	`$("tr:first")`

(':even'), (':odd')	pierwszego/ostatniego elementu /parzystych/nieparzystych elementów.	
(':first-child'), (':last-child')	Pseudoklasa dopasowująca się do pierwszego/ostatniego dziecka każdego pasującego rodzica.	`$("div span:first-child")`
(':enabled'), (':disabled'), (':checked'), (':hidden') , (':selected')	Pseudoklasy pomocne w obsłudze formularzy.	

Tab. 2. Wybrane funkcje animacji (http://api.jquery.com/category/effects/)

Funkcja	Opis	Przykład
show()	Efekt pojawienia się elementu.	`$('p').show();` `$('p').show("slow");`
hide()	Efekt ukrycia elementu.	`$('p').hide();` `$('p').hide(600);`
toggle()	Element jest zwijany i rozwijany.	`$('p').toggle();` `$('.divA').toggle('fast');`
fadeIn()	Efekt pojawiania się kolejnych elementów.	`$("span").fadeIn(100);` `$("div:hidden:first").fadeIn("slow");`
fadeOut()	Efekt znikania kolejnych elementów.	`$("p").fadeOut("slow");`

animate()	Efekt animacji elementu.	`$(".block1").animate({"left": "+=50px"}, "slow");`
		
		`$( ".block2" ).animate({ width: "90%" }, 1000 )`
		`.animate({ fontSize: "24px" }, 1000 )`
		`.animate({ borderLeftWidth: "15px" }, 1000 );`
		
		`$(".block3").animate({`
		`width: "70%",`
		`opacity: 0.4,`
		`marginLeft: "0.6in",`
		`fontSize: "3em",`
		`borderWidth: "10px"`
		`}, 1500 );`

delay()	Opóźnienie wykonania kolejnego efektu.	`$('div').slideUp(300).delay(800).fadeIn(400);`
slideUp()	Efekt wsuwania elementu.	`$("div").slideUp();`

Tab. 3. Wybrane funkcje manipulacji (http://api.jquery.com/category/manipulation/)

Funkcja	Opis	Przykład
addClass()	Dodanie klasy do elementu.	`$("p").addClass("selected");` `$("p").addClass("selected highlight");`
append()	Wstawienie zawartości do elementu.	`$('body').append('<p>Test</p>');` `$("div").append( $("h1") );`
css()	Pobranie/ustawienie właściwości css elementu.	`$('body').css("background-color");` `$('.div').css( "width","+=200" );`
attr()	Pobranie/ustawienie atrybutu elementu.	`$("a").attr("title");` `$('#photo').attr('alt', 'Great photo');`
empty()	Usunięcie tekstu elementu, a także elementów potomnych.	`$('body').empty();`
html()	Pobiera/ustawia kod html danego elementu.	`$("p").html();` `$("div").html("<p>New paragraph</p>");`
text()	Pobiera/ustawia tekst danego elementu.	`$("p:first").text();` `$("p").text("<b>Some</b> new text.");`

val()	Pobiera/ustawia wartość danego elementu.	`$("#l1").val();`
		`$("#l1").val(l2);`
		`$("p").text( $("input").val() );`

Tab. 4. Wybrane funkcje zdarzeń (http://api.jquery.com/category/events/)

Funkcja	Opis	Przykład
click()	Dodanie obsługi zdarzenia kliknięcia elementu.	`$("p").click(function () { $(this).slideUp(); });`
dblclick()	Dodanie obsługi zdarzenia dwu-kliknięcia elementu.	`$("p").dblclick( function () { alert("Hello World!"); });`
mouseover()	Dodanie obsługi zdarzenia umieszczenia kursora myszy nad elementem.	
mouseout()	Dodanie obsługi zdarzenia usunięcia kursora myszy znad elementu.	
select()	Dodanie obsługi zdarzenia wyboru elementu.	`$(":input").select( function () { $("div").text("Text was selected"); });`
submit()	Dodanie obsługi zdarzenia wysłania elementu.	`$("form").submit(function() { $("span").text("Not valid!").show(); return false; });`

Przykład zwięzłego wykorzystania funkcji jQuery:

```
var div = $('#mojDiv');

 div.html('<p>jQuery!</p>');
 div.addClass('klasaZcss');
 div.css('display', 'none');
 div.fadeIn('slow');
```

```
$('#mojDiv').html('<p>jQuery!</p>')
        .addClass('klasaZcss')
        .css('display', 'none')
        .fadeIn('slow');
```

Ćw.1. Podstawy pracy z jQuery

W dokumencie *obliczenia.html* do wyznaczania rat spłaty kredytu, zamiast własnych funkcji JavaScript skorzystaj z biblioteki jQuery.

Na początek przetestuj działanie podstawowych metod:

a) ustal kolor tła elementu **#tresc,**
b) zmień tło pól tekstowych i dodaj do nich klasę *bold* (definicję klasy *.bold* z pogrubioną czcionką dodaj do pliku *style.css*)
c) zmień kolor tła pól tekstowych z wynikami obliczeń

Na koniec obsłuż akcję kliknięcia przycisku „Oblicz” – wykorzystaj funkcje: *parseInt*, *parseFloat* oraz *isNaN*. W przypadku kiedy wynikiem jest *NaN* – wyświetl czytelne dla użytkownika komunikaty w polach wynikowych.

Ćw.2. Galeria z wykorzystaniem jQuery lightbox

Pobierz dodatkową wtyczkę do jQuery *lightbox2-master.zip* ze strony:
http://lokeshdhakar.com/projects/lightbox2/
Po rozpakowaniu znajdziesz tam m.in. foldery: *css*, *js* i *image*s. Folder *images* dodaj jako kolejny folder do naszego projektu WWW a pliki z folderów *css* i *js* skopiuj do odpowiednich folderów istniejących już w *WWW*. Zastosuj *ligthbox* do strony *galeria.html*.

W tym celu:

a) w nagłówku strony dołącz bibliotekę *jQuery* i arkusz *lightbox.css*:

```
<script   src="https://code.jquery.com/jquery-2.2.3.min.js"
integrity="sha256-a23g1Nt4dtEYOj7bR+vTu7+T8VP13humZFBJNIYoEJo="
crossorigin="anonymous"></script>
<link   rel="stylesheet" type="text/css" href="css/lightbox.css" />
```

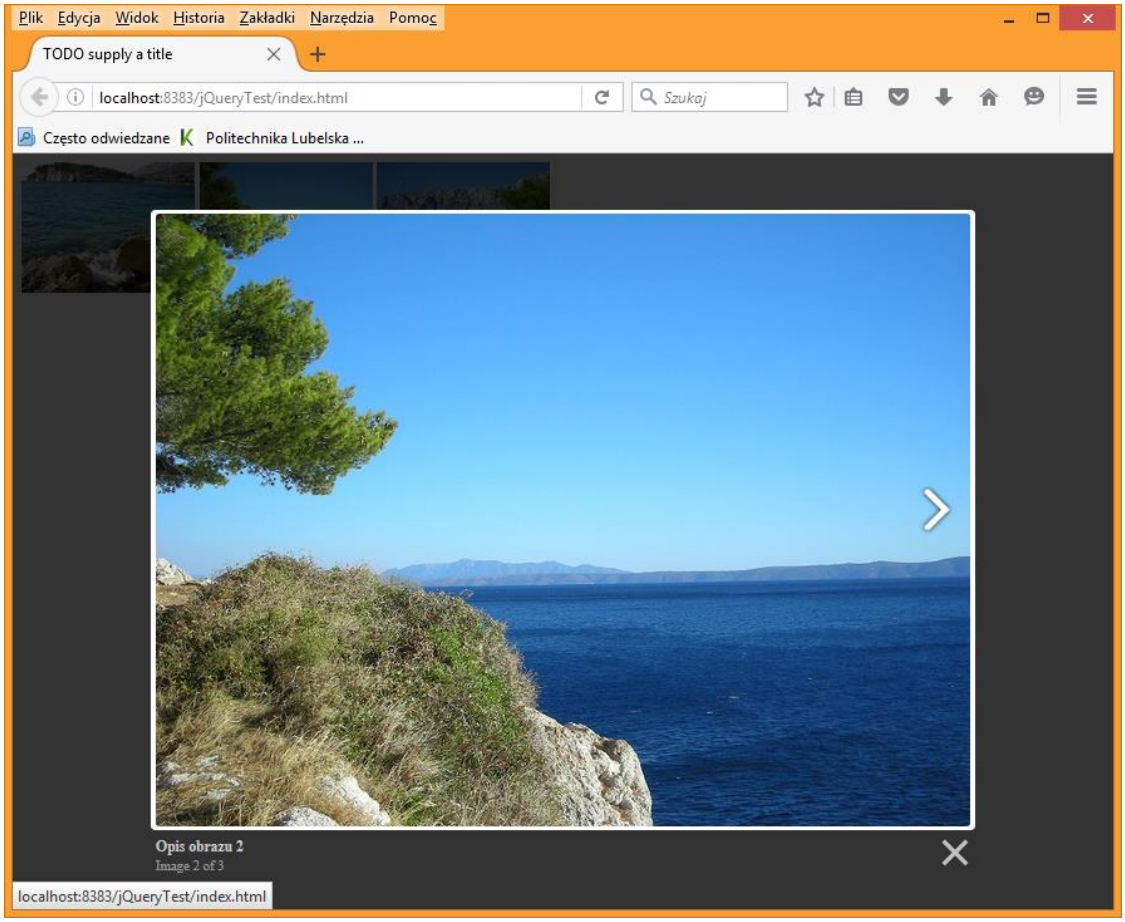
zamień hiperłącza tak, by korzystały z *lightbox*:

```
<a href="zdjecia/obraz1.jpg" data-lightbox="galeria" data-title="Opis
obrazu 1">  <img src="miniaturki/obraz1.jpg" alt="obraz1" />
</a>
<a href="zdjecia/obraz2.jpg" data-lightbox="galeria" data-title="Opis
obrazu 2" > <img src="miniaturki/obraz2.jpg" alt="obraz2"  />
</a>
```

b) UWAGA! Wtyczkę *lightbox* dołącz tuż przed zamknięciem elementu </body>:

```
<script src="js/lightbox.js"></script>
```

Punkty a i b wystarczą już do tego, aby nasza galeria przyjęła postać jak na rys.1.



Rys.1. Galeria z wykorzystaniem jquery lightbox

Ćw.3.

Do przestrzeni funkcji biblioteki **jQuery** można również dodać własne funkcje pomocne np. przy walidacji formularza. Na *Listingu 1* pokazano w jaki sposób można zdefiniować funkcję o nazwie ***SprawdzPole*** w przestrzeni funkcji jQuery.

Listing 1. Definicja funkcji pomocniczej w przestrzeni jQuery

```
(function ($) {
      //parametrem funkcji jest obiekt wyrażenia regularnego (reg):
      $.fn.sprawdzPole = function (reg) {
      if(!reg.test(this.val()))
            return (false);
      else
            return (true);
      };
})(jQuery);
```

Z tak zdefiniowanej funkcji ***SprawdzPole*** można już korzystać w ten sam sposób, jak ze standardowo dostępnych funkcji biblioteki (Listing 2). Do kodu z Listingu 2 dodaj kolejne reguły walidacji dla formularza z pliku formularze.html. Zwróć uwagę na to, aby definicja funkcji znalazła się przed skryptem, który z niej korzysta.

Listing 2. Wykorzystanie funkcji SprawdzPole do walidacji formularza

```
$(document).ready(function () {
$('#tresc').css('background','#80ffff');
$('input').addClass('bold').css('background','#00dbdb');
});
//walidacja formularza po kliknięciu na przycisk submit o
id='wyslij'
$(function() {
      $('#wyslij').click(function(){
      obiektNazw = /^[a-zA-ZąćęłńóśżźĄĆĘŁŃÓŚŻŹ]{2,20}$/;
      ok=true;
      if (!$("#nazw").sprawdzPole(obiektNazw)) {
        $("#nazw_error").text("Wprowadź poprawnie nazwisko!");
        ok=false;
      }
      else {
        $("#nazw_error").text("");
      }
      //sprawdź pozostałe warunki
      // ...
      if (ok) { //wyślij formularz:
        $("#form1").submit();
        return true;
      }
      else return false;
});
```