Zajęcia z programowania aplikacji WWW laboratorium nr 3 – Podstawy JavaScript

Po nabyciu umiejętności projektowania responsywanych stron internetowych przyszedł czas na wprowadzenie do strony elementów reaktywności i dynamiki. Strona powinna zacząć reagować na zdarzenia generowane przez użytkownika. Jest to możliwe wykorzystując Java Script ewentualnie najbardziej znaną bibliotekę do obsługi GUI – Jquery. W tym laboratorium zaznajomimy się ze sposobami korzystania z natywnego JavaScript. JQuery poświęcone będzie kolejne laboratorium.

Celem ćwiczenia jest zapoznanie studenta z możliwościami przetwarzania danych użytkownika wprowadzonych w formularzu po stronie klienta w oparciu o skrypt JavaScript i obiektowy model dokumentu DOM (ang. document object model). Realizując ćwiczenia student zapozna się z elementami języka JavaScript oraz możliwościami wykorzystania go do dynamicznej generacji zawartości strony internetowej oraz dostępu do elementów formularza.

Szczegółowe informacje na temat JavaScript można znaleźć w materiałach wykładowych
Home.agh.edu.pl/~rogus/PAW/w2.zip
lub w niezawodnym
http://www.w3schools.com/js/default.asp

Zadania dla początkujących w JavaScript

Zadanie nr 1.
Wyświetlić na stronie aktualną datę w formie tabelki

2005	12	25

Informacja pomocnicza! - ***Obsługa daty i czasu***
Wśród standardowych klas języka JavaScript istnieje klasa *Date* pozwalająca pozyskiwać informacje od dacie i czasie. Obiekty tej klasy udostępniają zestaw metod pozwalających pozyskiwać informacje o dacie i czasie w różnorodnej formie.

Metoda	Opis
Date()	Konstruktor tworzący obiekt zawierający aktualną datę i czas
Date(parametry)	Konstruktor tworzący obiekt zawierający datę i czas zdefiniowaną przez parametry
getYear()	Dwucyfrowa forma roku (<2000) lub czterocyfrowa(>=2000)
getFullYear()	Czterocyfrowa forma roku
getMonth()	Miesiąc roku (0-11)
getDate()	Dzień miesiąca (1-31)
getDay()	Numer dnia tygodnia od niedzieli do soboty (0-6)
getHours()	Godzina (1-24)
getMinutes()	Minuta (0-59)
getSeconds()	Sekund (0-59)
getMilliseconds()	Milisekunda (0-999)
getTime()	Liczba milisekund od 01.01.1970 00:00:00
getTimezoneOffset()	Różnica czasu lokalnego i GMT
toTimeString()	Tekst czasu w formie międzynarodowej
toDateString()	Tekst daty w formie międzynarodowej
toString()	Tekst daty i czasu w formie międzynarodowej
toLocateTimeString()	Tekst czasu w formie lokalnej (ustawienia przeglądatrki)
toLocateTimeString()	Tekst daty w formie lokalnej (ustawienia przeglądatrki)
toLocaleString()	Tekst daty i czasu w formie lokalnej (ustawienia przeglądatrki)

Przykład użycia

```
<script type="text/javascript">
        var d=new Date(); // utworzenie obiektu zawierającego aktualny czas
        document.write('<i>Dzisiejsza data:</i>'+d.toLocaleDateString());
        if (d.getHours()>16)
                        alert('Pora kończyć pracę !');
</script>
```

Zadanie nr 2.

Napisać program, który dla wczytanej liczby całkowitej wypisuje, czy jest ona parzysta czy nieparzysta. (spróbuj wypisać te informacje na wszystkie znane Ci sposoby - info na ten temat możne znaleźć w materiałach wykładowych).

Zadanie nr 3.

Napisać skrypt dzięki któremu użytkownik będzie mógł wprowadzić swoje imię i po kliknięciu na przycisk „Przywitajmy się” zostanie wyświetlone okienko „Witaj <imię>”

– utworzyć formularz w dokumencie html

– stworzyć skrypt odczytujący wprowadzone dane i wyświetlający okienko dialogowe – otworzyć stronę w przeglądarce, wprowadzić dane do pola i kliknąć na przycisk OK.

Zadanie nr 4.

Zdefiniować funkcję własną *DzienTygodnia*, która dla podanego numeru dnia tygodnia (niedziela=0) zwraca jego polską nazwę. We wnętrzu funkcji wykorzystać instrukcję *switch*. Wstępnie przetestować z wnętrza funkcji alert.

Zadanie nr 5.

W kolejnej komórce tabel z punktu 1 wypisać nazwę aktualnego dnia wykorzystując funkcję *DzienTygodnia*.

Zadanie nr 6

Napisać kod, który w wypadku, gdy aktualnym dniem tygodnia jest dzień laboratoriów z *Programowanie WWW* wyświetli okienko z zapytaniem, czy użytkownik jest zadowolony z tego przedmiotu. Przewidzieć reakcje (wyświetlając odpowiednie okienko *alert*) na dwie możliwe odpowiedzi.

Obsługa elementów strony internetowej z poziomu skryptu

Obiekt document modelu DOM daje możliwość uzyskania dostępu do wszystkich składowych stronie HTML takich jak elementy formularzy oraz znaczniki. W tym celu należy wykorzystać jedną z metod dostępu do elementów strony.

Metoda	Opis
`getElementById(id_elementu)`	Zwraca referencję do jednego elementu strony o niepowtarzalnym id
`getElementsByName(name_elementu)`	Zwraca tablicę referencji do elementów o podanej nazwie
`getElementsByTagName(znacznik)`	Zwraca tablicę referencji do podanego rodzaju znaczników

Po uzyskaniu referencji możemy wykorzystać charakterystyczne dla danego rodzaju elementu właściwości i metody.

Właściwość, metoda	Opis
`value`	Wartość elementu formularza
`type`	Typ elementu formularza
`selectedIndex`	Indeks aktualnie wybranej opcji listy formularza (od 0)
`checked`	Stan pola *radio* lub *checkbox*
`style`	Styl elementu
`innerText`	Zawartość znacznika. Znaczniki zagnieżdżone są wyświetlane.
`innerHTML`	Zawartość znacznika. Znaczniki zagnieżdżone formatują treść.
`submit()`	Wysłanie formularza
`reset()`	Wyczyszczenie formularza

Najpopularniejsze właściwości to:

- `innerHTML` – umożliwia określenie kodu HTML który zostanie umieszczony w elemencie
- `innerText` – umożliwia określenie ciągu znaków który zostanie umieszczony w elemencie

 `className` – umożliwia określenie klasy CSS użytej do prezentacji elementu.

Przykład użycia:

```
<html >
	<head>
		<title>Test</title>
		 <style type="text/css">
			.wyroznienieCzerwone { color: Red; font-weight: 600; }
		 </style>
	 </head>
	<body>
		<div id="nazwa"></div>
		<script type="text/javascript">
			document.getElementById("nazwa").innerHTML = "Witaj świecie";
			document.getElementById("nazwa").className = "wyroznienieCzerwone";
		</script>
	 </body>
</html>
```

Zawartość elementu, w tym przypadku div o id="nazwa" może zostać zmieniona w momencie ładowania strony.

Inny Przykład

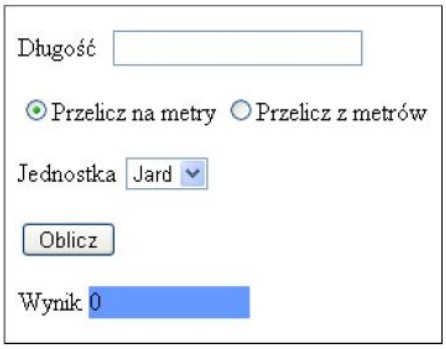
Zastosowanie skryptu do stworzenia strony obliczającej dla podanego przez użytkownika promienia objętość wybranej bryły.

```
<html>
<head>
<title>Objętości brył</title>
<script type="text/javascript">
	function Oblicz()
	{
		var r=Number(document.getElementById('promien').value);
		var V;
		switch(document.getElementById('bryla').value)
		{
		case 'K': V=4.0/3.0*Math.PI*Math.pow(r,3); break;
		case 'S': V=1.0/3.0*Math.PI*Math.pow(r,2)*5; break;
		case 'W': V=Math.PI*Math.pow(r,2)*5; break;
		}
		document.getElementById('wynik').innerHTML=
		'Objętość=<b>'+V.toFixed(2)+'</b>';
		document.getElementById('wynik').style.backgroundColor='yellow';
		document.getElementById('formularz').reset();
	}
</script>
</head>
<body>
<form id="formularz">
Promień bryły: <input id="promien" type="text" value="0" /> <br />
Rodzaj bryły:
<select id="bryla">
```

```
<option value="K">Kula</option>
<option value="S">StoŜek o wysokoŝci 5</option>
<option value="W">Walec o wysokoŝci 5</option>
</select> <br />
<input type="button" value="Oblicz" onclick="Oblicz();">
<p id="wynik" style="width:300px">0</p>
</form>
</body></html>
```

Zadanie 7. Przetwarzanie danych formularza

1.Przygotować formularz jak na rysunku. Uwzględnić następujące jednostki długości: jard, węzeł, stopa.



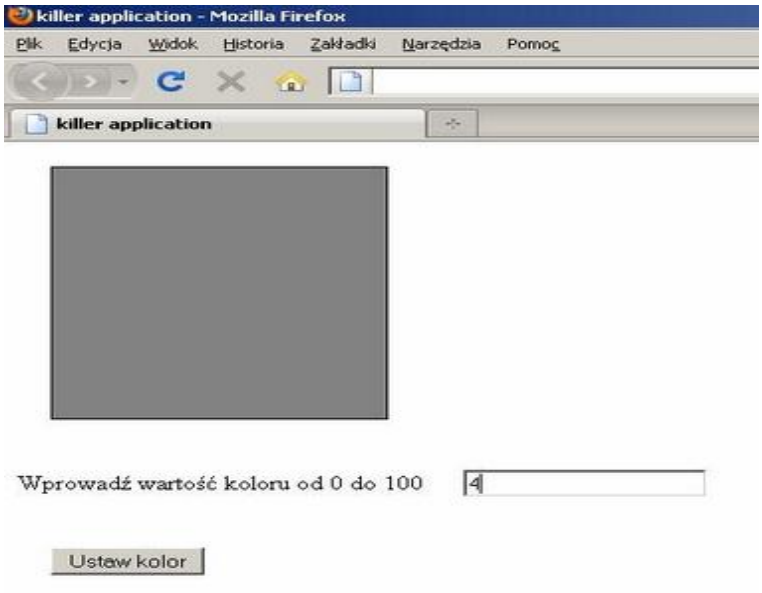
2. Zdefiniuj funkcję *Przelicz*, która przeliczy w zależności od wyboru użytkownika wartość z metrów na wybraną jednostkę lub odwrotnie. Wywołaj funkcję w chwili kliknięcia w przycisk *Oblicz*.

Jednostka	**Przelicznik**
Jard	0.9144 metra
Węzeł	14,63 metra
Stopa	0,30 metra

3. Sformatuj wyświetlenie wyniku z dokładnością do 3 miejsc po przecinku.
4. Dodaj funkcjonalność polegającą na zmianie koloru tła strony (przez styl) w zależności od wybranej jednostki (jard – czerwony, węzeł – niebieski, stopa – zielony).
5. Wykrywaj i ostrzegaj (okienko *alert*) o niepoprawnej liczbie wpisanej w pole długości. Blokuj w takim przypadku obliczenia.

Zadanie 8.

Proszę stworzyć aplikację, która czyta liczbę z przedziału od 0 do 100, wpisaną przez użytkownika w polu tekstowym, a następnie ustawia kolor w skali szarości odpowiadający wprowadzonej liczbie dla zadanej sekcji (kwadratowy obszar na **Rys. 1**) po kliknięciu na przycisk **Ustaw kolor**. Dla liczby 4 (jak na rysunku) kolor tła ma uzyskać wartość rgb(4%,4%,4%).



Pobieranie informacji od użytkownika

Najprostszą metodą komunikacji jest wciśnięcie przycisku lub innego elementu na stronie. Każdy element na stronie umożliwia obsługę zdarzenia kliknięcia za pomocą właściwości `onclick`. Podajemy w niej jako parametr nazwę funkcji która zostanie wywołana. Zmieniając zawartość body z powyższego przykładu otrzymujemy:

```
<div id="nazwa" onclick="zmien()"></div>

<script type="text/javascript">
	document.getElementById("nazwa").innerHTML = "Witaj świecie";
	function zmien() {
		document.getElementById("nazwa").innerHTML = "Witaj ważny świecie!";
		document.getElementById("nazwa").className = "wyroznienieCzerwone";
	}
</script>
```

Po kliknięciu na napis **Witaj świecie** pojawia się na czerwono napis **Witaj ważny świecie**. Kod umieszczony w znacznikach jest wykonywany od razu, natomiast kod umieszczony w funkcji dopiero po jej wywołaniu po kliknięciu na aktywną zawartość.

Kolejną możliwością pobierania informacji od użytkownika są pola tekstowe. Fragment strony umożliwiający wyświetlenie w miejscu div id="wynik" informacji wpisanej w kontrolce `input` o id="dane".

```
<input id="dane" type="text" />
<input type="button" value="Wyświetl" onclick="Wyswietl()" />
<div id="wynik" ></div>

<script type="text/javascript">
	function Wyswietl() {
		document.getElementById("wynik").innerHTML = "Wpisałeś " +
	document.getElementById("dane").value;
	}
</script>
```

Jak wynika z powyższego przykładu konstrukcja `document.getElementById` służy nie tylko do określania właściwości ale również do ich pobierania. Inną możliwością jest przekazanie do funkcji formularza i odwoływanie się do elementów przy pomocy właściwości value. W przypadku formularza przypisanie powyższe będzie miało postad:

```
document.getElementById("wynik").innerHTML= "Wpisałeś " + form.imie.value ;
```

Zdarzenia w JavaScript

Zdarzenie to inaczej zajście jakiejś sytuacji np. kliknięcie myszką, przesuwanie kursora myszki nad obiektem lub wpisywanie treści do kontrolki. Metoda obsługi zdarzenia to w języku JavaScript funkcja, która jest wywoływana w momencie wystąpienia określonego zdarzenia

Zdarzenie	Opis
onChange	Zawartość elementu uległa zmianie.
onClick	Użytkownik kliknął na tym elemencie.
onDblClick	Użytkownik kliknął dwukrotnie na tym elemencie.
onFocus	Użytkownik wybrał dany element.
onKeyDown	Użytkownik nacisnął klawisz.
onKeyPress	Użytkownik nacisnął lub zwolnił klawisz.
onKeyUp	Użytkownik zwolnił klawisz.
onMouseDown	Użytkownik wcisnął przycisk myszki.
onMouseOut	Użytkownik przesunął kursor myszki poza element.
onMouseOver	Użytkownik przesunął kursor myszki poza element.
onSubmit	Użytkownik wcisnął przycisk wysyłający dane z formularza.

Xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

tutorial

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Walidacja formularza

W zadaniu można wykorzystać dowolną stronę którą wykonano na zajęciach lab2 np. wynik zadania 2 lub 3. W sekcji main (głównej) należy umieścić kod formularza, który zamierzamy stworzyć i oprogramować.

Ćwiczenie 1. Formularz kontaktowy

Korzystając z http://getbootstrap.com/css/#forms oraz http://www.w3schools.com/html/html_forms.asp utwórz na stronie formularz kontaktowy, zawierający pola:

1. Imię i nazwisko (wymagane)

2. Adres email (wymagane)

3. Informacja (wymagane, ograniczone do 250 znaków)

4. Przycisk Wyślij

Proszę stosować elementy zgodne ze specyfikacją HTML5.

Dokładnie prześledź przykład poniżej:

Przykład. Tworzenie funkcji do weryfikacji pól formularza

Walidacja myForm

Język JavaScript jest często używany do wstępnego sprawdzenia poprawności danych wprowadzanych przez użytkowników. Przyjrzyjmy się formularzowi poniżej.

```
<form action="index.html#kontakt" method="post" onsubmit="return checkForm();">
      <fieldset>
            <legend>Formularz kontaktowy</legend>
            <label for="contactName">Imię</label>
            <input type="text" id="contactName"/>
            <label for="contactEmail">Email</label>
            <input type="text" id="contactEmail" />
            <input type="submit" value="Wyślij " />
      </fieldset>
</form>
```

Za sprawdzanie poprawności danych odpowiada funkcja *checkForm()*. Funkcja ta powinna zająć się sprawdzeniem kompletności danych. Odwołując się do właściwości *value* każdego z pól, można stwierdzić, czy zawiera ono jakieś dane, czy też pusty ciąg znaków. Funkcja *checkForm()* musi zwrócić wartość *true*, jeśli *myForm* ma zostać wysłany (czyli kiedy dane są kompletne), oraz *false* w przeciwnym razie wypadku.

```
<script>
function checkForm()
{
   var error=false; //to znaczy, że danych nie brakuje
```

```
   var errorText=""; //komunikat z błędem
   var contactName = document.getElementById("contactName") ;
   var contactEmail = document.getElementById("contactEmail") ;
```

// jeśli nic nie wpisano w *contactName* to jest błąd - sprawdzamy czy *contactName* jest puste, jeśli tak to dodajemy do *errorText* pole imię i oznaczamy, że brakuje danych

```
   if (contactName.value == ""){
      errorText += "imię\n"
      error=true;
   } // jeśli nic nie wpisano w contactEmail to jest błąd - sprawdzamy czy
```

contactEmail jest puste, jeśli tak to dodajemy do *errorText* pole email i oznaczamy, że brakuje danych

```
   if (contactName.value == ""){
      errorText += "email\n"
      error=true;
    } // jeśli nie brakuje danych wysyłamy formularz, jeśli brakuje
pojawia się komunikat i formularz nie zostanie wysłany
   if (! error) return true;
   else{
       alert ("Nie wypełniłeś następujących pól: \n" + errorText);
       return false;
   }
}
</script>
```

Ostatecznie, jeśli zmienna *error* ma wartość *true*, za pomocą metody *alert* jest wyświetlane okno z błędem, kiedy natomiast ma wartość *false*, jest wykonywana metoda *submit* wysyłająca dane z formularza.

Sprawdzanie poprawności adresu email

W kolejnym kroku możemy sprawdzić, czy podany adres email jest poprawny. W tym celu modyfikujemy tylko tę część:

```
if(contactEmail.value == ""){
    errorText += "email\n";
    error=true;
}
else
{
   var email = contactEmail.value;
   var regex = /^[a-zA-Z0-9._-]+@([a-zA-Z0-9. -]+\.)+[a-zA-Z0-9. -
    ]{2,4}$/;
   if(regex.test(email)==false)
   {
      errorText += "błędny email\n";
      error=true;
```

```
    }
}
```

Ćwiczenie 2. Walidacja formularza

1. Korzystając z powyższego przykładu napisz funkcję sprawdzającą pola Twojego formularza i umieść ją w osobnym pliku *forms.js* w katalogu *js*.

2. W formularzu w każdej warstwie `<div class=”form-group”>` nad `<label>` wstaw akapit z komunikatem o błędzie i indywidualnym *id*:
`<p id=”errorName”>Podanie imienia jest wymagane!</p>`

3. Ukryj akapity dopisując im klasę *hide.* Bootstrap sam zadba o ich schowanie.

4. Zmodyfikuj skrypt funkcji sprawdzającej formularz w taki sposób, żeby zamiast komunikatu o błędach wyświetlał ukryte akapity.

W tym celu usuń komunikat o błędach z końca skryptu, natomiast w poszczególnych warunkach sprawdzających czy elementy są właściwie wypełnione zamiast *errorText* wstaw dodawanie odpowiedniej klasy z błędem:
`document.getElementById('errorName') .className=' alert alertdanger'; 5. Wprowadź walidację poprawności adresu email i zmień komunikat o błędzie, jeśli adres jest podany, ale w złym formacie (`*innerHTML*).

Ćwiczenie 3. Podświetlanie kontrolek

1. Korzystając z podpowiedzi na http://getbootstrap.com/css/#forms-controlvalidation zmodyfikuj skrypt w taki sposób, żeby w momencie, gdy formularz jest wysyłany z błędem, kontrolki formularza otrzymywały odpowiednie podświetlenie i ikonkę – zielone dla pól poprawnie podświetlonych i czerwone dla niepoprawnych.

PODPOWIEDŹ: Dodaj identyfikator do warstwy *form-group* i wykorzystaj przypisywanie klas z zadania powyżej.

Zadanie samodzielne

1. Wykorzystaj zdarzenie *onblur* oraz *onchange* oraz wiedzę z przykładów powyżej i napisz funkcje pomocnicze, które będą sprawdzały wartości w polach formularza od razu po ich wypełnieniu (a nie podczas wysyłania formularza) i stosowały właściwe podświetlenie oraz komunikat o błędzie.

Xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Zadania dla mających więcej doświadczenia w JavaScript

Zadanie 1. Rozszerzony formularz walidacji

Twoim zadaniem jest przygotowanie i walidacja ankiety dla agencji modelek i modeli „Modelinki". Strona ankiety musi umożliwić pobranie od każdego chętnego do pracy w charakterze modela następujących informacji:

- imię
- nazwisko
- adres zamieszkania
- telefon
- e-mail
- wiek
- wzrost
- waga
- kolor włosów
- kolor oczu
- rozmiar ubrania
- numer butów
- doświadczenie (wybierane od 1 do 5)
- płeć

Po wybraniu płci użytkownikowi zostaje wyświetlona dalsza część formularza w wersji dla kobiet albo dla mężczyzn, który umożliwia pobranie następujących informacji:

w przypadku kobiet:

- obwód biustu
- wielkość miseczki
- talia
- biodra
- długość nogi

w przypadku mężczyzn:

- klatka
- pas
- długość nogi

Ponieważ wszystkie pola są wymagane opracuje schemat sprawdzania pola w odpowiedniej funkcji. Sprawdzenie poprawności wpisanych wartości ma odbywać się dla wszystkich pól

Dla poniższych pol zastosuj podane reguły , dla reszty samodzielnie zdefiniuj reguły walidacji.

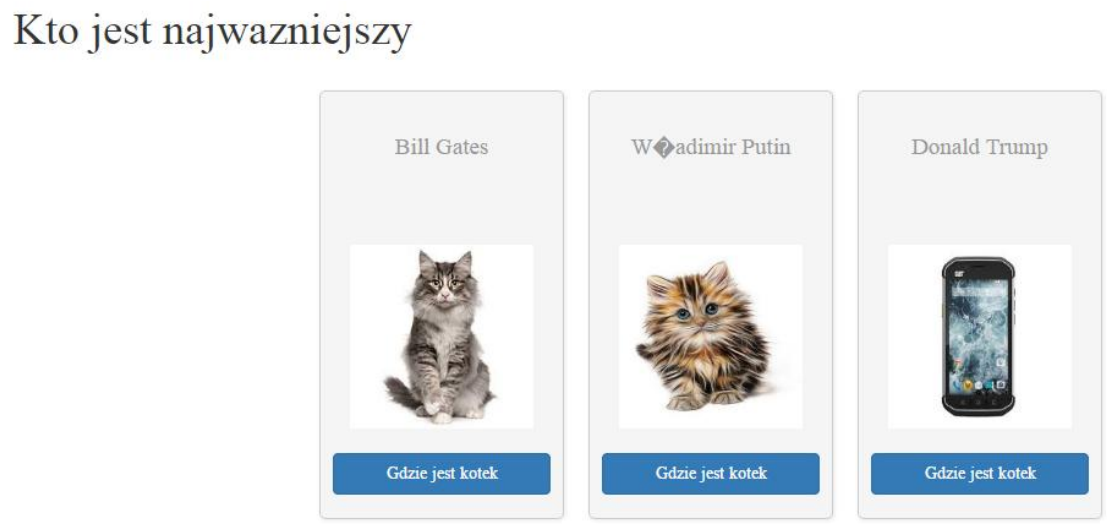
- E-mail (e-mail w formacie nazwa@serwer.domena)

- Kod pocztowy (kod odpowiedni dla naszego kraju)
- Wzrost – liczba w zakresie od 40 do 250 cm,
- Waga – liczba w zakresie od 20 do 80 kg

Wszystkie operacje sprawdzenia poprawności i wyświetlania elementów dla kobiet i mężczyzn maja być realizowane po stronie klienta w przeglądarce! Proszę zastosować odpowiednie wyrażenia regularne pozwalające na poprawną weryfikację danych.

Zadanie 2. Dynamiczne tworzenie zawartości strony.

W katalogu zad2 mamy strone index.html wraz z plikiem do stylizacji oraz 3 zdjecia. Proszę stworzyć plik main.js za pomocą którego wygenerujemy następującą zawartość strony. Dane należy zdefiniować bezpośrednio w pliku js w postaci tablicy obiektów.



Format tablicy jest następujący:

```
var vip = [
  {
    title: "Bill Gates",
    where: "http://www.onet.pl",
    url: "cat1.jpg"
  },
  {
    title: "Władimir Putin",
    where: "http://www.interia.pl",
    url: "cat2.jpg"
  },
  {
    title: "Donald Trump",
    where: "http://www.wp.pl",
```

```
    url: "cat3.jpg"
}];
```

Zadanie 3. Slajder zdjęć.

Napisz slader zdjeć. Zdjecia przesuwają się horyzontalnie po naciśnięciu na zdjęciu. (Zachownie podobne do obiektu typu caruzela w Bootstrap.). Materiały znajdziesz w katalogu zad3.