Лабораторна робота № 2

Тема: Форма як головний елемент динаміки та зворотнього зв'язку з сервером

Мета: оволодіти засобами керування формами, внесення інформації для сервера на динамічній web-сторінці, ознайомитись з функціями JavaScript та бібліотеки jQuery

Основні теги форми

Форма HTML містить групу елементів для приймання від користувача та пересилання даних у рядку запиту на web-сервер. Ними є `button`, `checkbox`, поля типу `text`, `input` тощо.

Важливий атрибут `action` вказує на URL сервера та програму на ньому для опрацювання даних запиту. `Атрибут method` вказує на вибір метода передачі даних: `get` чи `post`. Наведемо приклад тегу форми,

```
<form action="http://www.facerec.com/app.php" method="get">
 <fieldset>
  <label>Name: <input type="text" name="name" /></label>
  <label>sex: <input type="text" name="sex" /></label>
  <label>country?
    <input type="checkbox" name="country" />
  </label>
  <input id="sbutton" type="submit" value="Send me your
picture!"/>
 <fieldset>
</form>
```

Для перевірки правильності внесених на форму даних використовуються скрипти JavaScript.

Наприклад, якщо під час натискання кнопки відсилання даних певне поле даних не заповнене, то запит не надсилається. Фрагмент коду для реалізації цього сценарію наведено нижче:

```
. . . . .
let submit = document.getElementById("sbutton");
submit.onclick = submitClick();
. . . . .
   function submitClick(event) {
       if (document.getElementById("name").value == "") {
          event.preventDefault();  // відмінити submit
       }
   }
```

Код передбачає зупинку пересилання даних – `prevent Default.` (IE9 використовує `return false;` замість `preventDefault`).

Після натискання кнопки `submit` браузер формує запит у вигляді послідовності двох груп даних : заголовок `http`-протоколу та рядок з `DNS` адресою сервера, назвою програми для виконання та пар параметрів (назва–значення) :

```
GET /index.html HTTP/1.1
Host: localhost
```

```
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
image/xbm, */*
Accept-Language: en
Connection: Keep-Alive
User-Agent: Mozilla/4.0 (compatible; MSIE 4.5; Mac_PowerPC)
```

http://www.xul.ua/somefile.xml'

```
first=this+is+a+field&second=was+it+clear+%28already%29%3F
```

Форма HTML приймає параметри для передачі на сервер та опрацювання їх відповідним програмним засобом. Для зменшення кількості помилок на сервері, зменшення трафіку пересилання на клієнті використовують скрипти попередньої перевірки правильності та адекватності введених даних.

Розглянемо приклад перевірки відсутності незаповненого поля, правильності формату електронної адреси. На формі (після тегу form) використано два поля для імені та адреси:

```
<form name="myForm" action="demo_form.asp" onsubmit="return
validateForm()" method="post">
First name: <input type="text" name="fname">
Email: <input type="text" name="email">
<input type="submit" value="Submit">
</form>
```

Доступ до елементів форми з допомогою JavaScript

Наступний код JavaScript функції перевіряє правильність заповнення полів параметрами для передачі. Функція викликається під час натискання кнопки submit:

```
function validateForm()
{
let  x=document.forms["myForm"]["fname"].value;
if (x==null || x=="")
  {
  alert("First name must be filled out");
  return false;
  }
var x=document.forms["myForm"]["email"].value;
var atpos=x.indexOf("@");
var dotpos=x.lastIndexOf(".");
if (atpos<1 || dotpos<atpos+2 || dotpos+2>=x.length)
  {
  alert("Not a valid e-mail address");
  return false;
  }
}
```

Для перевірки збігу введених значень пароля використовується така функція:

```
function PwdValidation()
{
  let frm = document.forms["myform"];
  if(frm.pwd1.value != frm.pwd2.value)
  {
```

```
    sfm_show_error_msg('The Password and verified password
does not match!',frm.pwd1);
    return false;
  }
  else
  {
    return true;
  }
}
```

Елементи мови JavaScript

Одним з інструментів підтримки динамічних сценаріїв при перегляді Web-сторінок у межах комп'ютера користувача є мова програмування JavaScript – спрощений варіант мови програмування Java. Нею запезпечується рух об'єктів на сторінці, введення та виведення параметрів, зміна зображень вікон тощо. Програми модифікації, створення гіпертекстових сторінок традиційно називають скриптами (scripts), які інтерпретуються програмою перегляду. Спосіб базується на ідеології об'єктно-орієнтованого програмування. Зупинимось на скриптах, написаних мовою JavaScript.

Мова програмування для реалізації інтерактивності web-сторінок: введення тексту, реакції на події, отримання інформації з сервера, здійснення обчислень, підлягає стандартизації, але не підтримується всіма браузерами. Javascript інтерпретується на браузері та інтегрується за змістом HTML/CSS. Наведемо приклад внесення динамічного тексту:

`document.write("message");` –друкує текст на сторінці

Ввімкнення коду Javascript в HTML можливе трьома способами:

1. У тіло сторінки (виконується під час завантаження):

```
<body>
    ...
    <script type="text/javascript">
    Javascript code
    </script>
    ...
</body>
```

2. У заголовок сторінки (виконується як реакція на подію (викликом)):

```
<head>
    ...
    <script type="text/javascript">
    Javascript code
    </script>
    ...
</head>
```

3. Посиланням на зовнішній .js файл (розміщений у заголовку чи тілі):

```
<script src="filename" type="text/javascript"></script>
<script src="example.js" type="text/javascript"></script>
```

Розглянемо приклади типів даних та команд мови програмування. Тип змінної визначається за присвоєним значенням. Перепризначення значення об'єктів робиться операціями присвоювання. Оголошення змінної здійснюється ключовим словом var (var text = "text").

В JavaScript реалізовані всі типи операторів мов програмування : +,-,*, /, %, >>,<<, +=, -=, ... До того ж оператор "+" під час роботи з рядками означає конкатенацію:

```
s = "string1"+"string2";
```

Операторами структурного програмування є :

- *break* – примусовий вихід з циклу

```
while(i &lt 6)
{      if(i==3) break;      }
```

- *continue* – перехід на кінець циклу

```
while(i &lt 6)
{      if(i==3) continue;   }
```

- *for* – цикл

```
for(i=0;i<9;i++)
{      ...    }
```

- *for* – цикл властивостей об'єкта (змінних класу)

```
for(i in obj)
{      str = obj[i]  }
```

- *if..else* – умовний оператор

```
if(i>0)
{      ...    }
else
{      ...    }
```

- *while* – умовний цикл

```
while(j==k)
{      j++;   k--;   }
```

Зазначені оператори не містять повного переліку операторів JavaScript, але їх достатньо для виконання практичних завдань.

Тип Array дає можливість маніпулювання множинами як об'єктів сторінки, так і нових створених:

```
new_array = new Array();
new_array = new Array(5);
colors = new Array ("red","white","blue");
```

Розмірність масиву може змінюватися динамічно, наприклад,:

```
colors = new Array();
colors[5] = "red";
```

У цьому разі масив colors складається з 6 елементів (перший елемент масиву має індекс 0). Для масивів визначені три методи: join, reverse, sort. Наприклад,

```
colors = new Array("red","white","blue");
string = colors.join("+");
string = "red + white + blue";
```

Метод reverse змінює порядок елементів масиву на зворотний, sort – сортує елементи масиву в порядку зростання властивостей. Масив має дві властивості: length і prototype.

Поряд з масивами програми JavaScript використовують вбудовані масиви, наприклад, зображення (Images) чи гіпертекстові вказівки (Links).

Важливим елементом мови є події, які використовуються для виконання частин програмного коду скрипту, наприклад, onLoad="Scroll();". До найвживаніших можна зарахувати такі:

- onLoad – виконання скрипту чи функції під час завантаження;
- onChange – породжується під час зміни значення елементу форми;
- onClick – породжується під час вибору об'єкта (button, checkbox і т.п.);
- onSelect – породжується під час вибору текстового об'єкта (text, textarea);
- onSubmit – під час натискання на кнопку Submit;
- onUnload – під час переходу до іншої сторінки.

Розглянемо приклади використання вбудованих об'єктів.

```
new_image = new Image();new_image = new Image
(width,height);
```

Об'єкт Image має низку властивостей, зокрема, назва файла:

```
img_array[0].src = "image1.gif"
```

Усі операції в програмі на JavaScript описують дії над об'єктами – елементами робочої області броузера та контейнерами мови HTML. Об'єкти мають властивості та методи. Існують також інші функції, які дають змогу працювати зі стандартними математичними функціями та керувати процесом виконання програми. JavaScript має механізм опрацювання подій перегляду динамічних об'єктів та управління багатовіконним інтерфейсом. Об'єкти JavaScript беруть свій початок від класу Window. Ієрархію основних об'єктів подано на рис. 2.1.

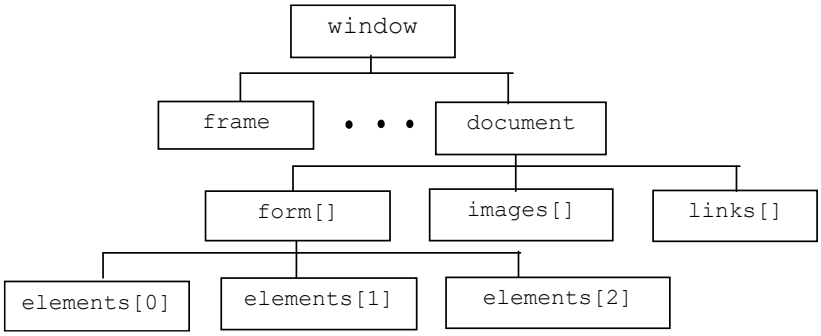

Рис. 2.1. Ієрархія об'єктів у мові JavaScript

Таблиці каскадних стилів (CSS)

Засоби каскадного стилю призначені для опису, подання, планування та вигляду інформації на web-сторінці, на противагу мові HTML, призначеній для опису змісту.

Стиль називається каскадним через керування атрибутами елементів у каскадному порядку:

- стилі браузера за замовчуванням;
- зовнішні файли стилів (<link>) ;
- внутрішні стилі (в межах тега <style> у заголовку);
- рядковий стиль (атрибути стилів елементів HTML).

Вони можуть бути розміщені в тексті коду сторінки або окремим файлом (раціональніший спосіб).

Базовим є синтаксис правила CSS

```
селектор {
    властивість: значення;
    властивість: значення;
    ...
    властивість: значення;
}
```

Наприклад,

```
p {
    font-family: sans-serif;
    color: red;
}
```

Файл CSS складається, як правило, з багатьох правил, кожне з яких починається із селектора. Останній прив'язується до елемента HTML і позначає властивості стилю подання даних.t

Під час створення окремого CSS-файла він приєднується до HTML коду тегом <link> у заголовку коду:

• Назвою без розширення

```
<link rel="stylesheet" type="text/css" href="filename" />
```

• Файлом з розширенням

```
<link rel="stylesheet" type="text/css" href="style.css" />
```

• Посиланням на сервер

```
<link rel="stylesheet" type="text/css"
href="http://www.google.com/uds/css/gsearch.css" />
```

Файлів може бути декілька. У разі конфлікту назв приймається останній.

Підключення JQUERY та його основні елементи

JQuery – це бібліотека JavaScript функцій підтримки зміни змісту на web-сторінці: для опрацювання подій, анімації, модифікації змісту тегів HTML документів та спрощення взаємодії з сервером за технологією Ajax. Технологія JQuery спрощує програмування сценаріїв засобами мови JavaScript.

Синтаксис JQuery подібний до синтаксису стилів CSS (підтримуються всі селектори), а саме: складається з селектора (вказівника) на елемент HTML та функції для виконання певних дій над атрибутами цього елемента: $(selector).action()

Приклади:

$(this).hide() – заховати елемент,

$("p").hide() – заховати всі параграфи,

$("p.test").hide() – заховати параграфи класу (class=«test»),

$("#test").hide() – заховати елементи з id="test".

Функція є бібліотечна або описана відразу після її оголошення. Функція може використовуватись для всього тіла сторінки без вказування конкретного селектора.

Наведемо приклад використання селектора та простої побудованої функції:

```
$(document).ready(function() {
    alert("document is ready");
```

```
});
```

Ця інструкція використовується у разі завантаженого HTML-документа та в готовому DOM навіть при недовантажених графічних об'єктах. Вона позначає функцію для виконання над об'єктами сторінки. В цьому випадку видається повідомлення після завантаження сторінки.

Помилка виникає, якщо:

- елемент не існує,
- зображення не завантажено тощо.

Для підімкнення бібліотеки в HTML-коді необхідно використати теги:

- `<script type='text/javascript' src='jquery.js'></script>`
- `<script type='text/javascript' src='jquery.min.js'></script>`
- `<script type="text/javascript" src="/jsz/jqModal.js"></script>`
- `<script type="text/javascript" src="/jsz/cache/03565abe.js"></script>`
- `<script type="text/javascript" src="/jsz/adriver.core.2.js"></script>`

Усі селектори погруповані в бібліотеках методів: Query Selectors, jQuery Events, jQuery Effects, jQuery Callback, jQuery HTML, jQuery CSS, jQuery AJAX.

Розглянемо приклади представників селекторів з різних бібліотек jQuery, які є обов'язковими елементами інструкцій на виконання.

Синтаксис	Опис
`$("*")`	вибір усіх елементів
`$("p")`	вибір усіх <p> елементів
`$("p.intro")`	вибір усіх <p> елементів з class="intro"
`$("p#intro")`	вибір перших <p> елементів з id="intro"
`$(":animated")`	вибір усіх анімованих елементів
`$(":button")`	вибір усіх елементів <button>, зокрема. для <input>
`$(":even")$(":odd")`	вибір парних (непарних) елементів
`$(this)`	Вибирає поточний HTML-елемент
`$("p#intro:first")`	Вибір першого елемента <p> з id = "intro"
`$(".intro")`	Вибір усіх елементів з класом = "intro"
`$("#intro")`	Вибір першого елемента з id = "intro"
`$("ul li:first")`	Вибір першого елемента Перший

`$("ul li:first-child")`	Вибір першого елемента кожного
`$("[href]")`	Вибирає всі елементи з атрибутом HREF
`$("[href$='.jpg']")`	Вибирає всі елементи з атрибутом HREF, який закінчується «. jpg»
`$("[href='#']")`	Вибирає всі елементи з HREF, значення дорівнює «#»
`$("[href!='#']")`	Вибирає всі елементи з HREF, значення НЕ дорівнює «#»
`$("div#intro .head")`	Вибирає всі елементи з класом = «head» у елемент з id = "intro")

Нижче в таблиці наведемо головні методи з бібліотеки опрацювання подій jQuery Events:

Метод	Опис
`$(document).ready(function)`	Пов'язує функцію з готовим документом
`$(selector).click(function)`	Пов'язує функцію з подією у певному місці
`$(selector).dblclick(function)`	Пов'язує функцію з подією у певному місці (подвійне натискання)
`$(selector).focus(function)`	Пов'язує функцію з фокусом на елементі
`$(selector).mouseover(function)`	Пов'язує функцію з подією у певному місці (курсор на об'єкті)

Повний список методів можна знайти на ресурсі http://www.w3schools.com/jquery/jquery_ref_events.asp

Підключення Bootstrap та його основні елементи

Bootstrap – це фреймворк для розробки клієнтських застосувань (front end). Bootstrap містить шаблони на базі HTML та CSS з текстом, формами, кнопками, таблицями, навігацією, модулями, каруселями зображень та багато інших сучасних елементів веб сторінок, а також вбудовані засоби JavaScript (плагіни). Bootstrap дозволяє створювати адаптивні сторінки у телефонах, планшетах та ноутбуках.. Bootstrap 3 підтримує реалізацію для мобільних телефонів. Bootstrap підтримується у сучасних браузерах (Chrome, Firefox, Internet Explorer, Safari, Opera). Версія 4.0 містить препроцесор CSS (SASS) та підтримку `flex-box`.

Для створення сторінки необхідно завантажити бібліотеки підтримки `CSS Bootstrap, JavaScript та jQuery`:

```
<!-- Latest compiled and minified CSS -->
<link
rel="stylesheet"href="https://maxcdn.bootstrapcdn.com/bootstr
ap/3.3.7/css/bootstrap.min.css">

<!-- jQuery library -->
<scriptsrc="https://ajax.googleapis.com/ajax/libs/jquery/1.12
.4/jquery.min.js"></script>

<!-- Latest compiled JavaScript -->
<scriptsrc="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/j
s/bootstrap.min.js"></script>
```

Розглянемо етапи розробки веб сторінки з допомогою Bootstrap

1. На першому етапі формується заголовок у стилі HTML5 `doctype`, оскільки Bootstrap потребує саме його (включно з атрибутами (`lang i charset`):

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <meta charset="utf-8">
 </head>
</html>
```

2. **Bootstrap** 3 is mobile-first. Засоби Bootstrap 3 дозволяють розробляти сторінки, адаптивні до мобільних пристроїв. Налаштування сторінки під розмір екрану пристрою забезпечується атрибутами в тегу `<meta>` всередині тегу `<head>` :

```
<meta name="viewport" content="width=device-width, initial-
scale=1">
```

Де `width=device-width` для керування динамічним налаштуванням. `initial-scale=1` вказує на початковий рівень зміни розміру сторінки при першому завантаженні.

3. **Containers**. Bootstrap також використовує два елементи (контейнери) для зберігання змісту сторінки : - `.container` клас фіксованої довжини , - `.container-fluid` клас повної ширини з розширенням до повного екрану

Ці елементи є кінцевими вузлами дерева. Приклад тексту сторінки з використанням зазначених елементів наведено нижче:

```
<!DOCTYPE html>
<html lang="en">
<head>
 <title>Bootstrap Example</title>
 <meta charset="utf-8">
```

```
 <meta name="viewport" content="width=device-width, initial-
scale=1">
 <link
rel="stylesheet"href="https://maxcdn.bootstrapcdn.com/bootstr
ap/3.3.7/css/bootstrap.min.css">
<scriptsrc="https://ajax.googleapis.com/ajax/libs/jquery/1.12
.4/jquery.min.js"></script>
<scriptsrc="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/j
s/bootstrap.min.js"></script>
</head>
<body>
<div class="container"> або <div class="container-fluid">
  <h1>My First Bootstrap Page</h1>
 <p>This is some text.</p>
</div>
</body>
</html>
```

Атрибути Bootstrap погруповані в групи подібно W3.CSS . Назва групи вказує на призначення : Grid Basic, Typography, Tables, Images, Jumbotron, Wells, Alerts, Buttons, Button Groups, Glyphicons, Badges/Labels, Progress Bars, Pagination, Pager, List Groups, Panels, Dropdowns, Collapse, Tabs/Pills, Navbar. Forms, Inputs, Input Sizing, Media Objects, Carousel, Modal, Tooltip, Popover, Scrollspy, Affix.

Коротко зупинимось на головних атрибутах різних груп класів. Typography – стилі шрифтів керуються класами Bootstrap і мають незначні відмінності від стилів браузерів за замовчуванням

Таблиці - стилі таблиць підтримуються класами .table, .table-striped - почергові рядкі з фоном, .table-bordered - з окантуванням таблиці, .table-hover – сірий фон на рядки таблиці, .table-condensed - компактна форма таблиці.

Контекстні класи можна використати для спеціального позначення рядків (<tr>) чи клітинок таблиці (<td>)::

.active - колір під курсором в рядку чи клітинці,
.success - вказує на успіх або позитивну дію,
.info - вказує на інформацію щодо змін чи дії,
.warning - попередження для уваги,
.danger - вказує на небезпечні дії.

Клас .table-responsive забезпечує адаптивну таблицю:

```
<div class="table-responsive">
 <table class="table">
  ...
 </table>
</div>
```

Класи зображень. .img-rounded заокруглені кути .img-circle – зображення в колі

```
<img src="cinqueterre.jpg" class="img-circle" alt="Cinque
Terre"width="304" height="236">
```

Клас .img-thumbnail забезпечує мінітюаризацію зображень :

```
<img src="cinqueterre.jpg" class="img-thumbnail" alt="Cinque
Terre"width="304" height="236">
```

Створити адаптивні зображення дозволяє клас .img-responsive , що додається до тегу <img>.

До класу `.img-responsive` можна застосувати властивості `display: block;` `max-width: 100%; height: auto;.`
Створити адаптивні вбудовані об'єкти (відео) дозволяє клас `.embed-responsive-item` :

```
<div class="embed-responsive embed-responsive-16by9">
 <iframe class="embed-responsive-item" src="..."></iframe>
</div>
```

Клас `.well` додає округлену границю, сірий фон та доповнення прогалинами.:

```
<div class="well">Basic Well</div>
```

Розміри керуються відповідними класам від слів `Small,Large` : `.well-sm` `.well-lg` :

```
<div class="well well-sm">Small Well</div>
<div class="well well-lg">Large Well</div>
```

Для кнопок Bootstrap передбачає велике різноманіття стилів : Basic. Default. Primary. Success. Info. Warning Danger. LinkTo, для яких використовуються наступні класи : `.btn .btn-default .btn-primary, .btn-success, .btn-info, .btn-warning, .btn-danger, .btn-link`

```
<button type="button" class="btn">Basic</button>
<button type="button" class="btn btn-
default">Default</button>
```

Кнопки можна використовувати в елементах `<a>`, `<button> та <input>.`

```
<input type="button" class="btn btn-info" value="Input
Button">
<input type="submit" class="btn btn-info" value="Submit
Button">
```

Розміри кнопок керуються класами:`.btn-lg, .btn-md, .btn-sm,.btn-xs`

```
<button type="button" class="btn btn-primary btn-
xs">XSmall</button>
```

Стан кнопки задається класами `.active` та `.disabled`:

```
 <button type="button" class="btn btn-primary
disabled">Disabled Primary</button>
```

Завдання до лабораторної роботи № 2:

Розробити web-сторінку згідно макета (wireframe)
https://cacoo.com/diagrams/ZvVhYS3UpG5PdbBy/EDE3A

Обираємо для розробки шаблон Students Add/Edit
Використовуємо Gitlab (https://gitlab.com)
Сторінка має містити:
- модальне вікно для внесення даних як на макеті з прихованим полем id
- для редагування даних використовується те саме вікно
- кнопки у модальному вікні та реакції на ці кнопки функціями JavaScript/jQuery.
- функція додавання/збереження має сформувати рядок даних, що буде передаватись з даної форми на сервер.
- створити стандартний опис з будь яким кешуванням PWA