Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра «Інформаційні системи та мережі»



Звіт

До лабораторної роботи №7

З предмету «Веб-програмування»

Виконав:

ст. гр. ІТ-22

Сарнацький Олекса

Перевірив:

доц. каф. ІСМ Лозицький О. А.

Львів2019

Лабораторна робота №7

«Методи JavaScript DOM»

Теоретичні відомості

Об'єктна модель документа (англ. Document Object Model, DOM) — специфікація прикладного програмного інтерфейсу для роботи зі структурованими документами (як правило, документами XML). Визначається ця специфікація консорціумом W3C.

З точки зору об'єктно-орієнтованого програмування, DOM визначає класи, методи та атрибути цих методів для аналізу структури документів та роботи із представленям документів у вигляді дерева. Все це призначено для того, аби надати можливість комп'ютерній програмі доступу та динамічної модифікації структури, змісту та оформлення документа.

Разом із поширенням та розвитком веб-технологій і веб-переглядачів почали з'являтись різні, часто несумісні інтерфейси роботи із HTML документами в інтерпретаторах JavaScript, вбудованих у веб-переглядачі. Це спонукало World Wide Web Consortium (W3C) узгодити та визначити низку стандартів, які отримали назву W3C Document Object Model (W3C DOM). Специфікації W3C не залежать від платформи або мови програмування.

Через те, що структура документа представляється у вигляді дерева, повний зміст документа аналізується та зберігається в пам'яті комп'ютера. Тому, DOM підходить для застосувань в програмах, які вимагають багаторазовий доступ до елементів документа в довільному порядку. В разі, якщо треба лише послідовний або одноразовий доступ до елементів документа, рекомендується, для пришвидшення переробки та зменшення обсягів необхідної пам'яті комп'ютера, використовувати послідовну модель роботи зі структурованими документами (SAX).

Хід роботи

JavaScript-код:

var checkbox = document.getElementById('themer');

var invertor = document.getElementById('inverter');

checkbox.addEventListener('change', function () {

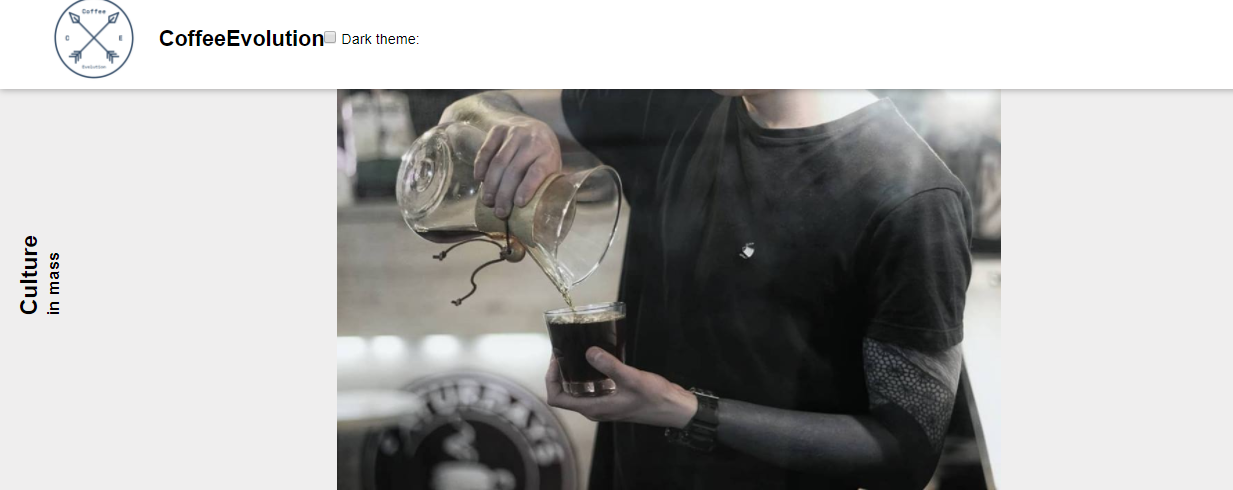
// Triggers repaint in most browsers:

invertor.setAttribute('media', this.checked ? 'screen' : 'none');

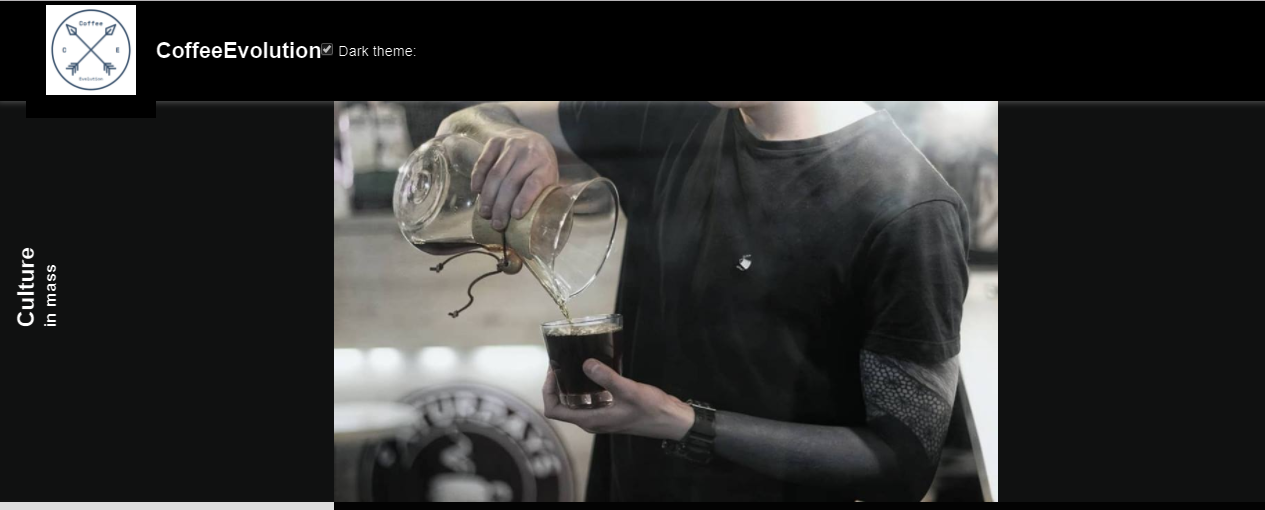
// Forces repaint in Chrome:

invertor.textContent = invertor.textContent.trim();

});



*Рис. 1. Світла тема*



*Рис. 2. Темна тема*

Висновок: на цій роботі я ознайомився з методами змін теми засобами javascript

Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра «Інформаційні системи та мережі»



Звіт

До лабораторної роботи №8

З предмету «Веб-програмування»

Виконав:

ст. гр. ІТ-22

Сарнацький Олекса

Перевірив:

доц. каф. ІСМ Лозицький О. А.

Львів

2019

Лабораторна робота №8

«GIT. Розміщення сайту в Інтернеті»

Теоретичні відомості

Git — розподілена система керування версіями файлів та спільної роботи. Проект створив Лінус Торвальдс для управління розробкою ядра Linux, а сьогодні підтримується Джуніо Хамано (англ. Junio C. Hamano). Git є однією з найефективніших, надійних і високопродуктивних систем керування версіями, що надає гнучкі засоби нелінійної розробки, що базуються на відгалуженні і злитті гілок. Для забезпечення цілісності історії та стійкості до змін заднім числом використовуються криптографічні методи, також можлива прив’язка цифрових підписів розробників до тегів і комітів.

Прикладами проектів, що використовують Git, є ядро Linux, Android, LibreOffice, Cairo, GNU Core Utilities, Mesa 3D, Wine, багато проектів з X.org, XMMS2, Gstreamer, Debian DragonFly BSD, Perl, Eclipse, GNOME, KDE, Qt, Ruby on Rails, PostgreSQL, VideoLAN, PHP, One Laptop Per Child (OLPC), АБІС Koha, GNU LilyPond та Elinks і деякі дистрибутиви GNU/Linux (див. нижче).

Програма є вільною і випущена під ліцензією GNU GPL версії 2.

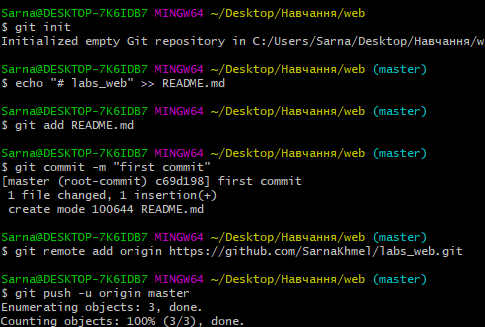
Система спроектована як набір програм, спеціально розроблених з врахуванням їхнього використання у скриптах. Це дозволяє зручно створювати спеціалізовані системи управління версіями на базі Git або користувацькі інтерфейси. Наприклад, Cogito є саме таким прикладом фронтенда до репозиторіїв Git. А StGit використовує Git для управління колекцією латок.

Система має ряд користувацьких інтерфейсів: наприклад, gitk та git-gui розповсюджуються з самим Git.

Віддалений доступ до репозиторіїв Git забезпечується git-демоном, SSH або HTTP сервером. TCP-сервіс git-daemon входить у дистрибутив Git і є разом з SSH найпоширенішим і надійним методом доступу. Метод доступу HTTP, незважаючи на ряд обмежень, дуже популярний в контрольованих мережах, тому що дозволяє використання існуючих конфігурацій мережевих фільтрів.

Хід роботи

1. Створюю порожній репозиторій на GitHub.
2. Клоную його
3. Копіюю файли в репозиторій.
4. Роблю коміт.



1. Пушу зміни на сервер.

Висновок: на цій роботі я запушив свій сайт на GitHub.

Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра «Інформаційні системи та мережі»



Звіт

До лабораторної роботи №5

З предмету «Веб-програмування»

Виконав:

ст. гр. ІТ-22

Сарнацький Олекса

Перевірив:

Лозицький О. А.

Львів

2019

Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра «Інформаційні системи та мережі»



Звіт

До лабораторної роботи №6

З предмету «Веб-програмування»

Виконав:

ст. гр. ІТ-22

Сарнацький Олекса

Перевірив:

Лозицький О. А.

Львів

2019

Лабораторна робота №5

«Інтеграція JavaScript на сторінку. Перевірка полів форми»

Теоретичні відомості

JavaScript (JS) — динамічна, об'єктно-орієнтована прототипна мова програмування. Реалізація стандарту ECMAScript. Найчастіше використовується для створення сценаріїв веб-сторінок, що надає можливість на стороні клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд веб-сторінки.

JavaScript класифікують як прототипну (підмножина об'єктно-орієнтованої), скриптову мову програмування з динамічною типізацією. Окрім прототипної, JavaScript також частково підтримує інші парадигми програмування (імперативну та частково функціональну) і деякі відповідні архітектурні властивості, зокрема: динамічна та слабка типізація, автоматичне керування пам'яттю, прототипне наслідування, функції як об'єкти першого класу.

Мова JavaScript використовується для:

* написання сценаріїв веб-сторінок для надання їм інтерактивності;
* створення односторінкових веб-застосунків (React, AngularJS, Vue.js);
* програмування на стороні сервера (Node.js);
* стаціонарних застосунків (Electron, NW.js);
* мобільних застосунків (React Native, Cordova);
* сценаріїв в прикладному ПЗ (наприклад, в програмах зі складу Adobe Creative Suite чи Apache JMeter);

всередині PDF-документів тощо.

Незважаючи на схожість назв, мови Java та JavaScript є двома різними мовами, що мають відмінну семантику, хоча й мають схожі риси в стандартних бібліотеках та правилах іменування. Синтаксис обох мов отриманий «у спадок» від мови С, але семантика та дизайн JavaScript є результатом впливу мов Self та Scheme.

Хід роботи

Файл script.js

function checkForm() {

var a = document.forms["Regform"]["name"].value;

var b = document.forms["Regform"]["phone"].value;

if (!isNaN(a)) {

alert("Ім`я може містити тільки букви");

document.forms["Regform"]["name"].select();

return false;

}

if (isNaN(b)) {

alert("Прізвище може містити тільки букви");

document.forms["Regform"]["phone"].select();

return false;

}

if (isNaN(b)) {

alert("Прізвище може містити тільки букви");

document.forms["Regform"]["phone"].select();

return false;

}

}

Файл index.html

<form name="Regform" onsubmit="return checkForm()" method="post">

<p>Будь ласка заповніть форму</p>

<hr>

<label for="name"><b>Name</b></label>

<input type="text" placeholder="Введіть ім'я" name="name" id="name" required>

<br>

<label for="phone"><b>Прізвище</b></label>

<input type="text" placeholder="введіть номер" name="phone" required>

<br>

<label for="email"><b>Email</b></label>

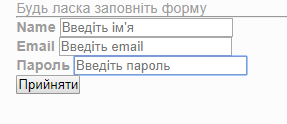
<input type="email" placeholder="Введіть email" name="emailaddress" required>

<br>

<button type="submit" id="accept" class="registerbtn" >Прийняти</button>

</div>

</form>



Висновок: на цій лабораторній роботі я навчився вбудовувати JavaScript в сторінку, а також виконувати перевірку полів форми.

Лабораторна робота №6

«Події у JavaScript»

Теоретичні відомості

JavaScript має низку властивостей об'єктно-орієнтованої мови, але завдяки концепції прототипів підтримка об'єктів в ній відрізняється від традиційних мов ООП. Крім того, JavaScript має ряд властивостей, притаманних функціональним мовам, — функції як об'єкти першого класу, об'єкти як списки, каррінг, анонімні функції, замикання (closures) — що додає мові додаткову гнучкість.

JavaScript має C-подібний синтаксис, але в порівнянні з мовою C має такі корінні відмінності:

* об'єкти, з можливістю інтроспекції і динамічної зміни типу через механізм прототипів
* функції як об'єкти першого класу
* обробка винятків
* автоматичне приведення типів
* автоматичне збирання сміття
* анонімні функції

JavaScript містить декілька вбудованих об'єктів: Global, Object, Error, Function, Array, String, Boolean, Number, Math, Date, RegExp. Крім того, JavaScript містить набір вбудованих операцій, які, грубо кажучи, не обов'язково є функціями або методами, а також набір вбудованих операторів, що управляють логікою виконання програм. Синтаксис JavaScript в основному відповідає синтаксису мови Java (тобто, зрештою, успадкований від C), але спрощений порівняно з ним, щоб зробити мову сценаріїв легкою для вивчення. Так, приміром, декларація змінної не містить її типу, властивості також не мають типів, а декларація функції може стояти в тексті програми після неї.

Семантика мови схожа з семантикою мови Self.

Приклад оголошення і використання класу в JavaScript:

**class** MyClass {

constructor () {

**this**.myValue1 = 1;

**this**.myValue2 = 2;

}

}

**const** mc = **new** MyClass();

mc.myValue1 = mc.myValue2 \* 2;

Хід роботи

<script>

function countClicks() {

for(let i=1; i<=3; i++) {

alert("Клік № " + i);

}

}

</script>

<input type="text" name="Thing" value="" />

<script>

document.getElementsByName("Thing")[0].addEventListener('change', doThing);

function doThing(){

alert('В поле введено "' + this.value + '"!');}

</script>

<script>

input.oncut = input.oncopy = input.onpaste = function(event) {

result.innerHTML = event.type + ' ' + input.value;

return false;

};

</script>

<div>

<button type="submit" id="accept" class="registerbtn" onclick="countClicks()" value="Рахувати">Події</button>

</div>

Висновок: на цій лабораторній роботі я навчився додавати події у JavaScript.