Министерство образования и науки Российской Федерации

ФГАОУ ВО «Севастопольский государственный университет»

Институт информационных технологий и управления в технических системах

Кафедра «Информационные системы»

ОТЧЁТ

по Лабораторной работе №2

по дисциплине

«Веб-технологии»

Выполнила студентка гр. ИС/б-19-1-з

Петрова Арина Алексеевна

Принял:

г. Севастополь

2022 г.

**Тема:** «Исследование возможностей программирования на стороне клиента. Основы языка JavaScript»  
**Цель:** исследовать особенности написания программ для приложений на стороне клиента. Изучить основы языка JavaScript и объектной модели браузера. Приобрести практические навыки проверки HTML-форм с использованием JavaScript.  
Тексты разработанных программ:  
1. Альбом  
let fotos = [

{url: 'https://sun9-60.userapi.com/impg/iTmU9ly\_ad-8ACEtm8Zl8Qc8ePjkYR0RvmYc0A/e1parfYWeCc.jpg?size=1633x2160&quality=95&sign=af98b59eaab1cdc9f5f941f750579f1d&type=album'},

{url: 'https://sun9-84.userapi.com/impg/f7ri31BsNPgkpjBn7iZRjOvWHVMNhxf\_x08oiA/o4\_x8K4GoE8.jpg?size=1682x2160&quality=95&sign=6923509ee35b4a67c2884e273515688c&type=album'},

{url: 'https://sun9-12.userapi.com/impg/plzXNeEvOX2\_x\_4cLsNVooN9GMUY6BuLdMVhYg/Ld3nOBGs-Xk.jpg?size=2560x1920&quality=95&sign=4d4db8e55c73d6166277fb61b44d24fb&type=album'},

{url: 'https://sun9-20.userapi.com/impg/-LTJx8Pl08E9ixHGP97zhXxGoKT4JgxY6cnaaw/CYErSDz4o48.jpg?size=2159x2160&quality=95&sign=e973914ad952d2b1923abd40700df82a&type=album'},

{url: 'https://sun9-38.userapi.com/impg/vIo7R3QJEKwdsQppr4QFEINvAfq8BbJ28cnfHQ/1i18EhFNTHw.jpg?size=1887x2160&quality=95&sign=69e92db45869f0fcd1b2b5b9d8e770b9&type=album'},

{url: 'https://sun9-31.userapi.com/impg/JY0j0MNQIWM-0hkfu-e7fUziC2EGGmOnekQreg/ioJaB0mpxVM.jpg?size=1703x1640&quality=95&sign=05e20eb78b15f4c43a81885de378e42f&type=album'},

{url: 'https://sun9-15.userapi.com/impg/5WO\_LGe-Tcs1QwDcbPCg3kj-4M47rJ\_CaBBx7g/HxsvUn-SJ0M.jpg?size=1668x1668&quality=96&sign=6059cf809b52e0f1fdbcee9e897e8822&type=album'},

{url: 'https://sun9-83.userapi.com/impg/dmLmBgNO-V-shMckVIiNwakcGD3tQjjTO9SpqQ/oj5dYo8hH-Y.jpg?size=2160x2160&quality=96&sign=97062c800d9cfe83550a9667edfeb9f3&type=album'},

{url: 'https://sun9-42.userapi.com/impg/59mcI7TYCmCqfB9J6i3bS-RzjjoMBRKinClZOQ/Ge1yMJ9D9HI.jpg?size=2160x2160&quality=96&sign=87f15f24e3a59ae6f18118ff4120579c&type=album'},

{url: 'https://sun9-39.userapi.com/impg/-q5mSf4wJW7HMs03-7MxeNYme1eYXdALrxcH5g/21VT8aj7pCk.jpg?size=2500x2063&quality=96&sign=84d8cbc857580d5bdbd65cee881de95d&type=album'},

{url: 'https://sun9-54.userapi.com/impg/ZT\_\_8E4nSGtQluXSrD5gmeBICiPDs8OoT3eUTQ/Uf2O83k6OUs.jpg?size=2160x2160&quality=96&sign=7ea51996c06f51478d4ef1888bd8641d&type=album'},

{url: 'https://sun9-57.userapi.com/impg/gkgFEAp14syHWpFrM01iGanJMTIzToYMwQMGAA/7Lh\_GdOQmKw.jpg?size=2500x1590&quality=96&sign=9b6f07a4708abe01b7597ab4f858536c&type=album'},

{url: 'https://sun9-29.userapi.com/impg/tp5xjoTXyNXYlZHstN2OP2fIvKfgbMfnCTQqzg/TI9pR4ZiQDo.jpg?size=1500x1100&quality=96&sign=78d8bae54eb3e8ec81a8f348adbe75b0&type=album'},

{url: 'https://sun9-32.userapi.com/impg/HesU7f3e\_wxTM12m-tDC565cKWzS2yA2WTxViA/5\_rrPIEQJ2k.jpg?size=840x579&quality=96&sign=2086ac52066922d4451cb8dc6bdf566e&type=album'},

{url: 'https://sun9-63.userapi.com/impg/amVLq0yvONhxfF8knUCrOxbX7DNrf4g2dwY6og/mLag3LwFu9I.jpg?size=1747x2160&quality=95&sign=8125b9af4d7fd88b5a562029b2a28f71&type=album'}

]

let titles = [

{title: 'Ху Тао фан арт'},

{title: 'Тигнари фан арт'},

{title: 'Оригинальный арт'},

{title: 'Бутылочка'},

{title: 'Оригинальный арт'},

{title: 'Оригинальный арт'},

{title: 'Оригинальный арт'},

{title: 'Дракониха'},

{title: 'Кроссовер'},

{title: 'Оригинальный арт'},

{title: 'Янь Фэй фанарт'},

{title: 'Оригинальный арт'},

{title: 'Стадик'},

{title: 'Оригинальный арт'},

{title: 'Аяка фанарт'}

]

const imagesPerRow = 3;

const tableContainer = document.querySelector('#table-container');

const table = document.createElement('table');

for(let rowIndex = 0; rowIndex < fotos.length / imagesPerRow; rowIndex++)

{

const row = document.createElement('tr');

for(let cellIndex = 0; cellIndex < imagesPerRow; cellIndex++)

{

const photoIndex = rowIndex \* imagesPerRow + cellIndex;

const photo = fotos[photoIndex];

const ptitle = titles[photoIndex];

const cell = document.createElement('td');

const img = document.createElement('img');

const title = document.createElement('span');

img.src = photo.url;

title.innerHTML = ptitle.title;

cell.appendChild(img);

cell.appendChild(title);

row.appendChild(cell);

}

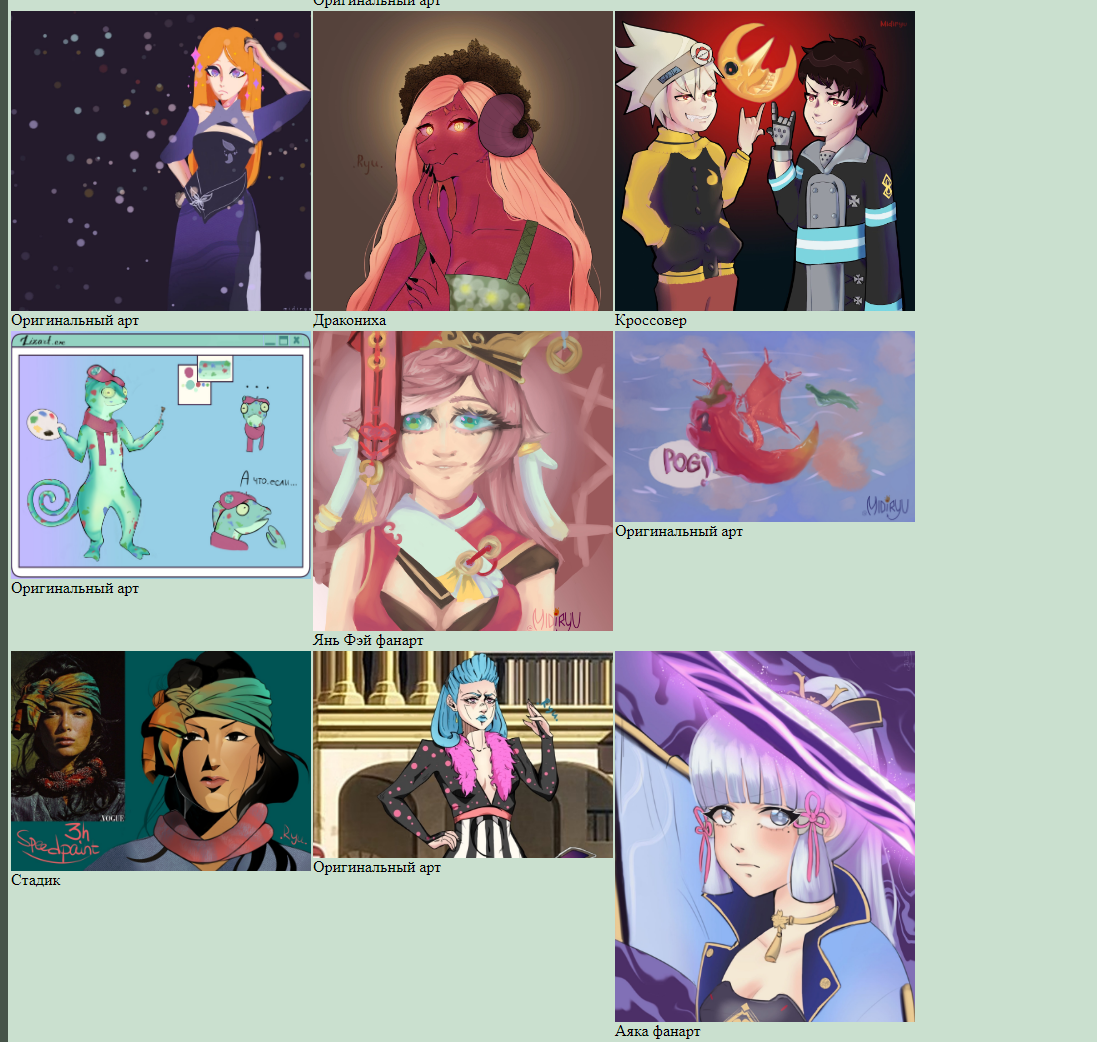
table.appendChild(row);

}

tableContainer.appendChild(table);

Результат:





2. Страница Обо мне:  
const containerSelector = '#lists-container';

const interests = [

{

headerId: 'hobby',

title: 'Моё хобби',

items: [

'Рисование',

'Пение',

'Настольные игры'

]

},

{

headerId: 'books',

title: 'Мои любимые книги',

items: [

'Роберт Шекли "Абсолютное оружие"',

'Сага о Ведьмаке',

'Гарри Гаррисон "Специалист по этике"',

'Антуан де-сент Экзюпери "Маленький Принц"'

]

},

{

headerId: 'music',

title: 'Моя любимая музыка',

items: [

'Rock',

'DSBM',

'slowcore',

'synthwave',

'EDM',

'noise',

'jazz',

'gabba',

'uptempo',

'maidcore',

'grindcore',

'pop',

'classic',

'death metal'

]

},

{

headerId: 'films',

title: 'Мои любимые фильмы',

items: [

'Ослеплённый желаниями',

'Невероятная жизнь Уолтера Митти',

'500 дней лета',

'Твин Пикс',

'Паприка',

'Хеллсинг OVA',

'Судьба: Начало'

]

},

{

headerId: 'games',

title: 'Мои любимые игры',

items: [

'Dark Souls',

'Bloodborne',

'Salts and Sanctuary',

'Skyrim',

'Gothic',

'Oblivion',

'Fallout: New Vegas',

'Metro',

'S.T.A.L.K.E.R.',

'Slay the Spire',

'Deponia',

'The Whispered World',

'The Witcher',

'Elden Ring'

]

}

];

const listsContainer = document.querySelector(containerSelector);

drawLists(interests, listsContainer);

function drawList(list, container) {

const listContainer = document.createElement('div');

const header = document.createElement('header');

header.id = `#${list.headerId}`;

const title = document.createElement('h2');

title.innerHTML = list.title;

header.appendChild(title);

listContainer.appendChild(header);

const listElement = document.createElement('ul');

list.items.forEach(item => {

const listItem = document.createElement('li');

listItem.innerHTML = item;

listElement.appendChild(listItem);

});

listContainer.appendChild(listElement);

container.appendChild(listContainer);

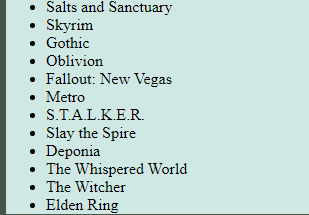
}

function drawLists(lists, container){

lists.forEach(list => drawList(list, container));

}

Результат:



3. Страница Контакты

const regEmail = /\S/;

const regFio = /\D/;

const regPhone = /^[+]\d/;

const email = document.querySelector('#email');

const pNumber = document.querySelector('#phone');

const fio = document.querySelector('#fio');

const btn = document.getElementById('submit');

btn.addEventListener('click', inputHandler);

function inputHandler(event) {

if(! validateFio(fio)) {

event.preventDefault();

}

if(! validatePhone(pNumber)) {

event.preventDefault();

}

if(! validateEmail(email)) {

event.preventDefault();

}

}

function validateEmail(email)

{

if (email.value.trim() === "") {

alert("Заполните поле");

email.focus();

return false;

}

if (email.value !== "") {

console.log(regEmail);

if (! regEmail.test(email.value)) {

alert("В поле присутствуют лишние символы");

email.focus();

return false;

}

}

return true;

}

function validateFio(fio) {

if (fio.value.trim() === "") {

alert("Заполните поле");

fio.focus();

return false;

}

if(! regFio.test(fio.value)) {

alert("В поле присутствуют лишние символы");

fio.focus();

return false;

}

if (((fio.value).split(" ").length - 1) != 2) {

alert("Форма ФИО введена некорректно");

fio.focus();

return false;

}

return true;

}

function validatePhone(pNumber) {

if (pNumber.value.trim() === "") {

alert("Заполните поле");

pNumber.focus();

return false;

}

if(! regPhone.test(pNumber.value)) {

alert("В поле присутствуют лишние символы");

pNumber.focus();

return false;

}

if(! (/^[+]3/.test(pNumber.value) || /^[+]7/.test(pNumber.value))) {

alert("Номер телефона должен начинаться с +7 или +3");

pNumber.focus();

return false;

}

if(/\s/.test(pNumber.value)) {

alert("Номер телефона не должен содержать пробелы");

pNumber.focus();

return false;

}

if((pNumber.value).length < 10 || (pNumber.value).length > 12) {

alert("Длинна номера телефона должна быть от 9 до 11 цифр");

pNumber.focus();

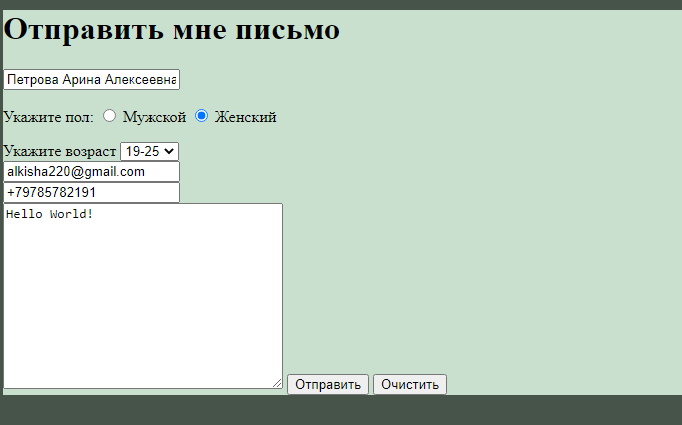
return false;

}

return true;

}

Результат:



4. Страница Тест

const thirdQ = querySelector('#thirdq');

function validateField(thirdQ)

{

if (thirdQ.value.trim() === "") {

alert("Заполните поле");

thirdQ.focus();

return false;

}

if (! /\D/.test(thirdQ.value)) {

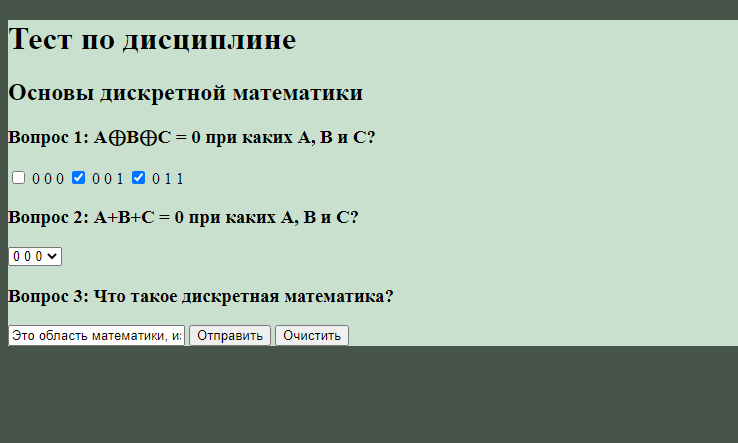
alert("В поле присутствуют цифры");

thirdQ.focus();

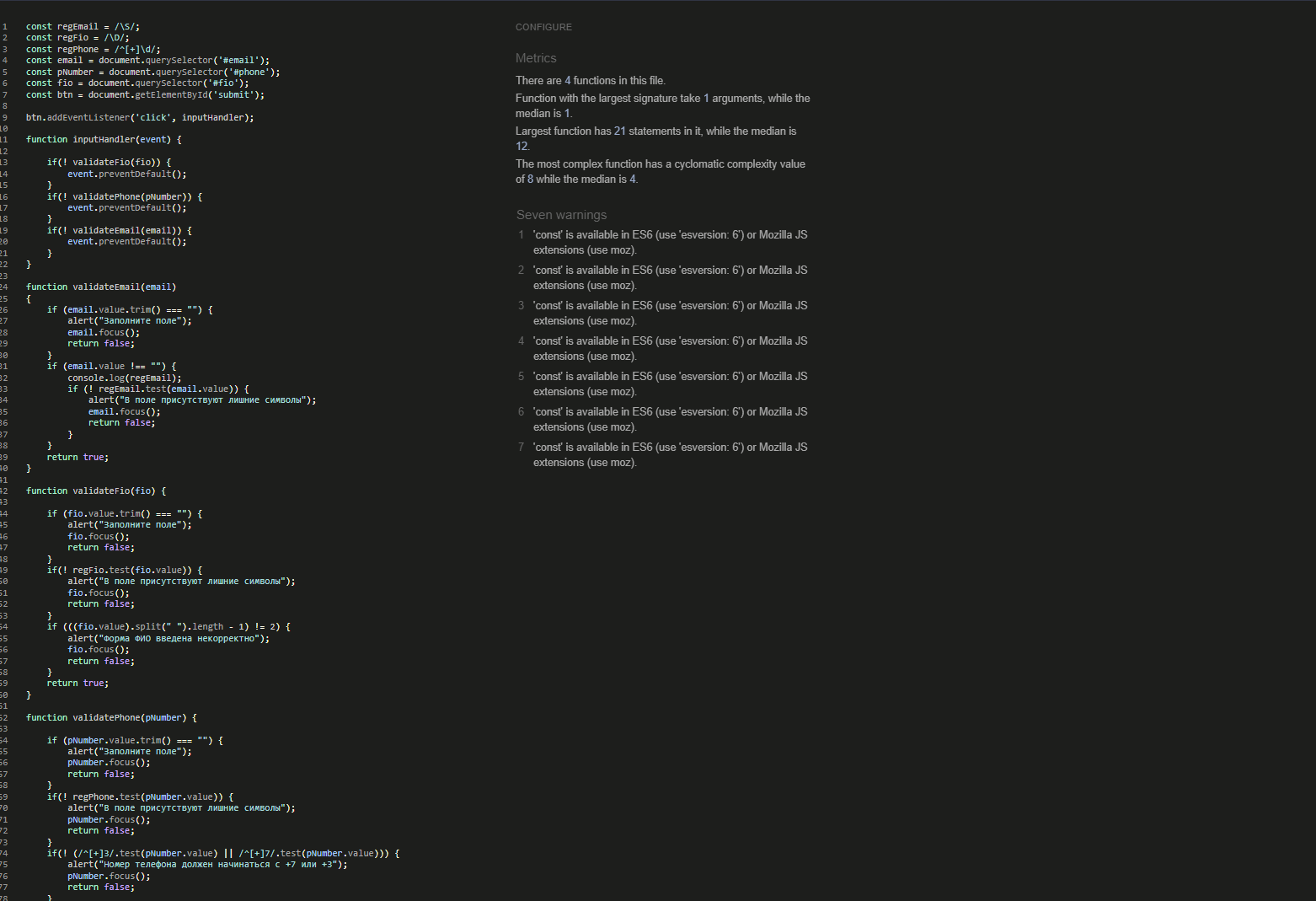
return false;

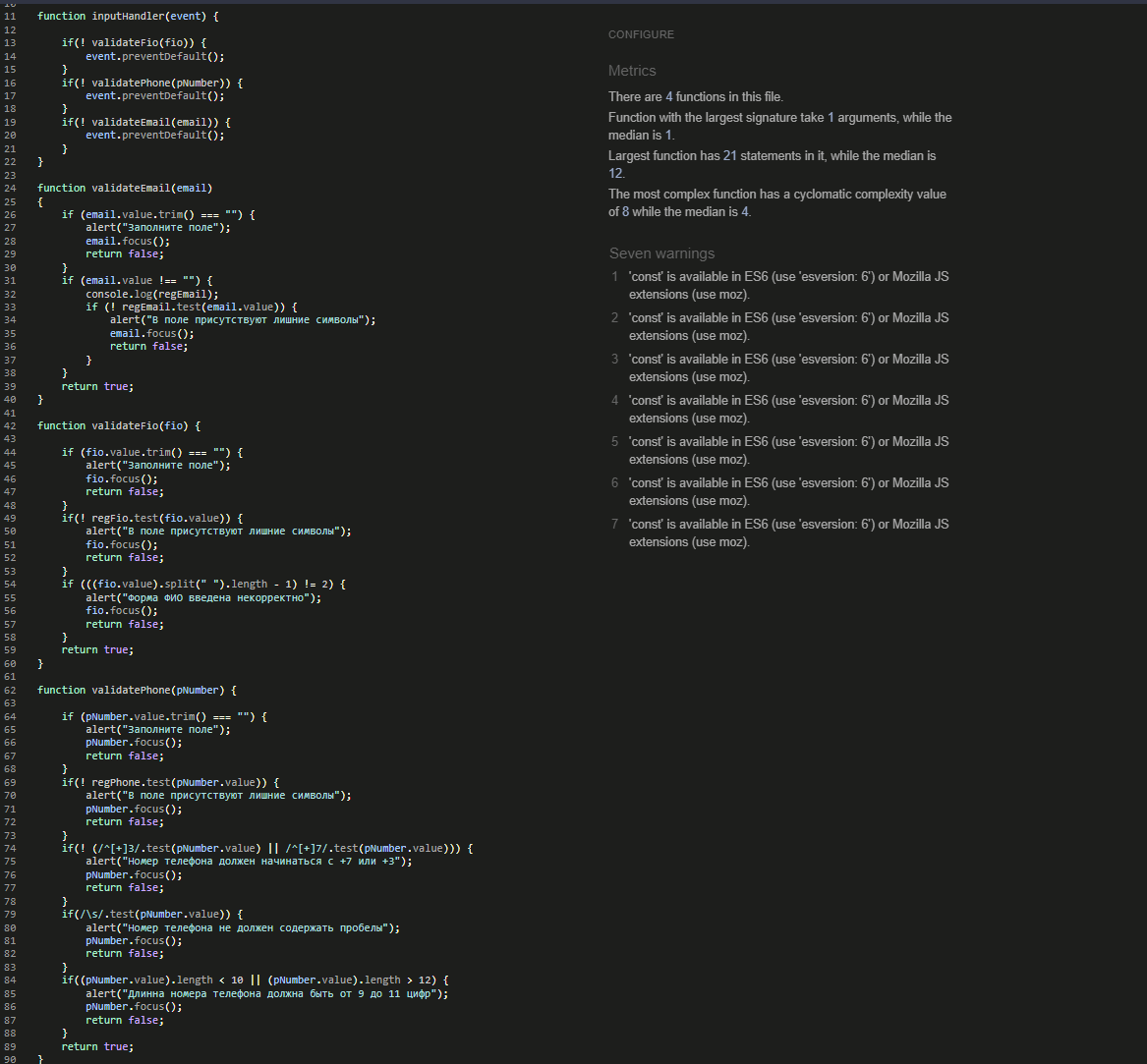
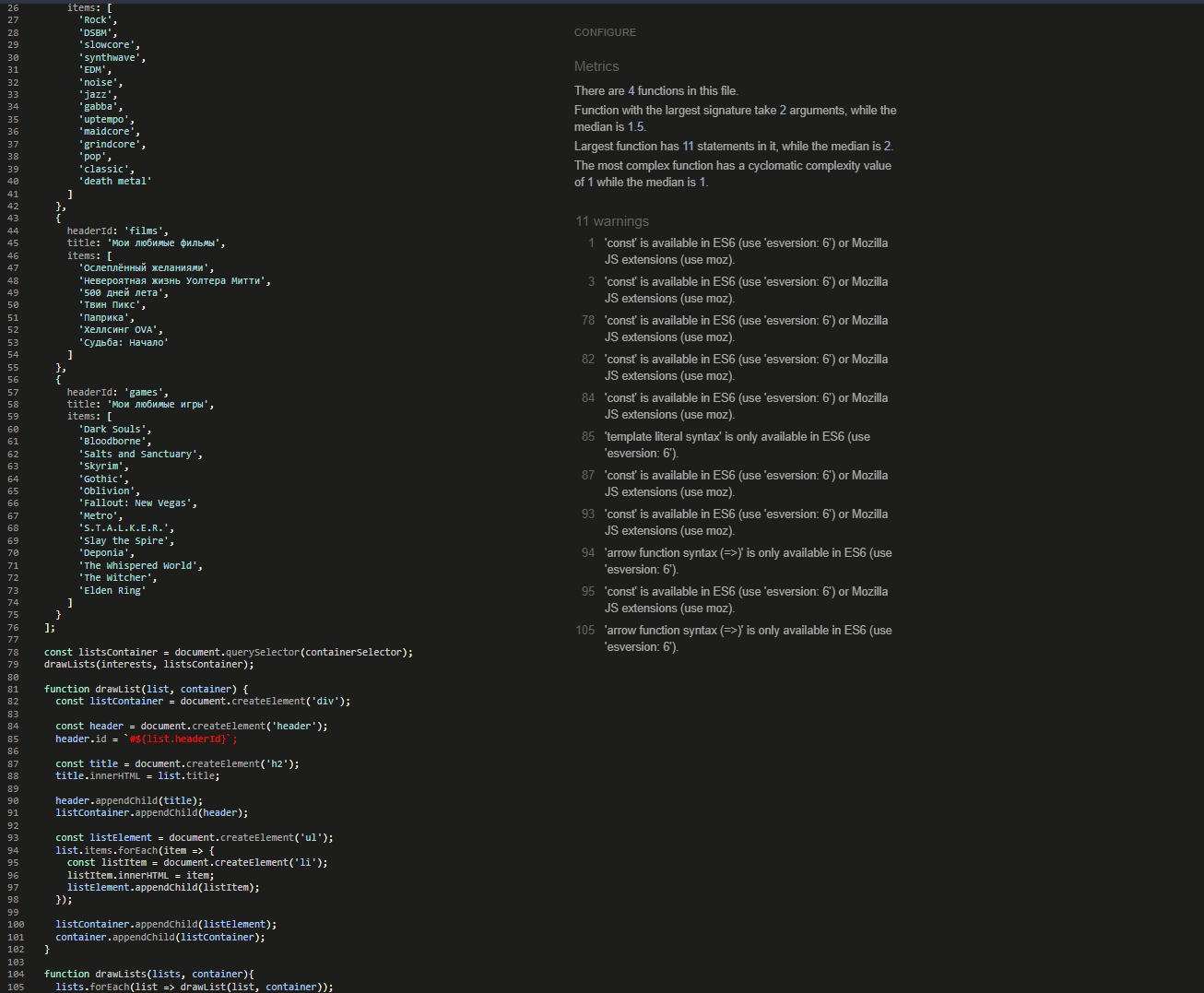
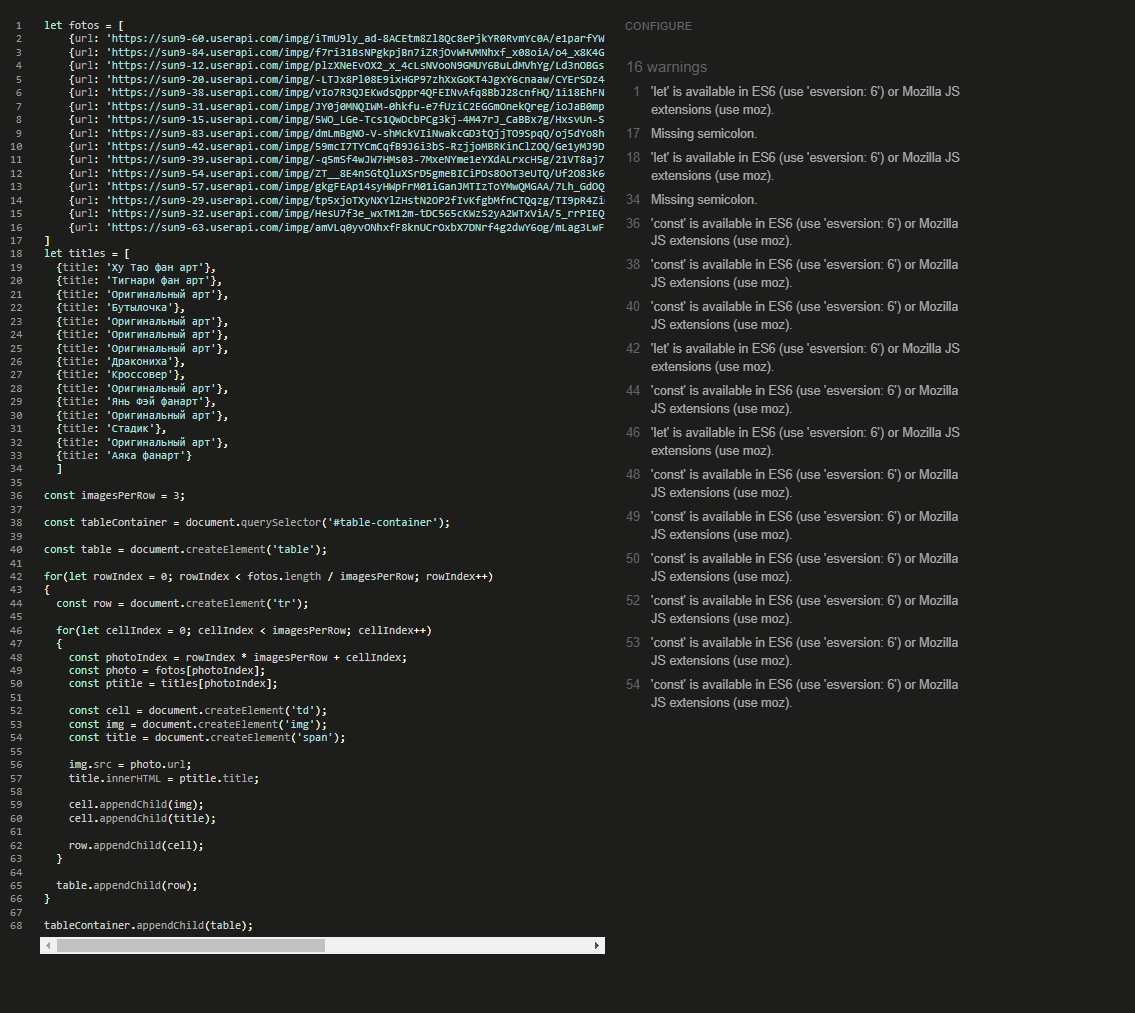
}

return true;

}  
Результат:  


Проверка кодов с помощью jshint:





Выводы: при помощи Javascript можно расширить возможности вёрстки страницы. Html и css позволяли размечать абзацы и оформлять страницу, а обрабатывать события.

Контрольные вопросы

1. Приведите теги HTML, внутри которых располагаются команды JavaScript?   
<script></script>

2. Где могут располагаться скрипты JavaScript?   
Внутри тега <script> в любом месте html документа

3. Расскажите об объектной модели браузера в JavaScript?

На самом верху этой модели находится глобальный объект window. Он представляет собой одно из окон или вкладку браузера с его панелями инструментов, меню, строкой состояния, HTML страницей и другими объектами. Доступ к этим различным объектам окна браузера осуществляется с помощью следующих основных объектов: navigator, history, location, screen, document и т.д. Так как данные объекты являются дочерними по отношению к объекту window, то обращение к ним происходит как к свойствам объекта window.

window – самый главный объект в браузере, который отвечает за одно из окон (вкладок) браузера. Он является корнем иерархии всех объектов доступных веб-разработчику в сценариях JavaScript. Объект window кроме глобальных объектов (document, screen, location, navigator и др.) имеет собственные свойства и методы, которые предназначены для:

* открытия нового окна (вкладки);
* закрытия окна (вкладки) с помощью метода close();
* распечатывания содержимого окна (вкладки);
* передачи фокуса окну или для его перемещения на задний план (за всеми окнами);
* управления положением и размерами окна, а также для осуществления прокручивания его содержимого;
* изменения содержимого статусной строки браузера;
* взаимодействия с пользователем посредством следующих окон: alert (для вывода сообщений), confirm (для вывода окна, в котором пользователю необходимо подтвердить или отменить действия), prompt (для получения данных от пользователя);
* выполнения определённых действий через определённые промежутки времени и др.

Если в браузере открыть несколько вкладок (окон), то браузером будет создано столько объектов window, сколько открыто этих вкладок (окон). Т.е. каждый раз открывая вкладку (окно), браузер создаёт новый объект window связанный с этой вкладкой (окном).

navigator – информационный объект с помощью которого Вы можете получить различные данные, содержащиеся в браузере:

* информацию о самом браузере в виде строки (User Agent);
* внутреннее "кодовое" и официальное имя браузера;
* версию и язык браузера;
* информацию о сетевом соединении и местоположении устройства пользователя;
* информацию об операционной системе и многое другое.

history – объект, который позволяет получить историю переходов пользователя по ссылкам в пределах одного окна (вкладки) браузера. Данный объект отвечает за кнопки forward (вперёд) и back (назад). С помощью методов объекта history можно имитировать нажатие на эти кнопки, а также переходить на определённое количество ссылок в истории вперёд или назад. Кроме этого, с появлением HTML5 History API веб-разработчику стали доступны методы для добавления и изменения записей в истории, а также событие, с помощью которого Вы можете обрабатывать нажатие кнопок forward (вперёд) и back (назад).

location – объект, который отвечает за адресную строку браузера. Данный объект содержит свойства и методы, которые позволяют: получить текущий адрес страницы браузера, перейти по указанному URL, перезагрузить страницу и т.п.

screen – объект, который предоставляет информацию об экране пользователя: разрешение экрана, максимальную ширину и высоту, которую может иметь окно браузера, глубина цвета и т.д.

document – HTML документ, загруженный в окно (вкладку) браузера. Он является корневым узлом HTML документа и "владельцем" всех других узлов: элементов, текстовых узлов, атрибутов и комментариев. Объект document содержит свойства и методы для доступа ко всем узловым объектам. document как и другие объекты, является частью объекта window и, следовательно, он может быть доступен как window.document.

4. Как описываются переменные в JavaScript?

Const a описывает константы, let a – переменные. Есть устаревшая версия var. Разница в том, что let существует только внутри той функции, в которой объявлена.

Описать переменную можно как при объявлении, так и отдельно.

Let a = 5; или let a; a = 5;

5. Расскажите о свойствах и методах строк в JavaScript?

Свойства: caller (возвращает функцию, откуда был вызов), constructor (создаёт новый объект), length (длинна), name (название)  
Методы: string.fromCharCode() – Возвращает строку, созданную из указанной последовательности значений Юникода.  
String.fromCodePoint() – Возвращает строку, созданную из указанной последовательности кодовых точек Юникода.

String.raw – Возвращает строку, созданную из сырой шаблонной строки.

Ещё методы: String.quote, String.substring, String.toLowerCase, String.toUpperCase, String.charAt, String.charCodeAt, String.indexOf, String.lastIndexOf, String.startsWith, String.endsWith, String.trim, String.trimLeft, String.trimRight, String.toLocaleLowerCase, String.toLocaleUpperCase, String.localeCompare, String.match, String.search, String.replace, String.split, String.substr, String.concat, String.slice

6. Какие методы реализованы для объекта Array в языке JavaScript?

'quote', 'substring', 'toLowerCase', 'toUpperCase', 'charAt', 'charCodeAt', 'indexOf', 'lastIndexOf', 'startsWith', 'endsWith', 'trim', 'trimLeft', 'trimRight', 'toLocaleLowerCase',

'toLocaleUpperCase', 'localeCompare', 'match', 'search', 'replace', 'split', 'substr', 'concat', 'slice'

7. Как преобразуются типы данных в Java Script?   
Существует 3 наиболее широко используемых преобразования: строковое, численное и логическое.

**Строковое** – Происходит, когда нам нужно что-то вывести. Может быть вызвано с помощью String(value). Для примитивных значений работает очевидным образом.

**Численное** – Происходит в математических операциях. Может быть вызвано с помощью Number(value).

**Логическое** – Происходит в логических операциях. Может быть вызвано с помощью Boolean(value).

8. Расскажите о свойствах и методах массива elements?

Свойства: tagName – имя тега, innerText – получить текстовое содержимое тега, innerHTML – получение или установка кода html, offsetWidth и offsetHeight определяют соответственно ширину и высоту элемента в пикселях, clientWidth и clientHeight также определяют ширину и высоту элемента в пикселях, но уже без учета границы

Методы: getAttribute(attr): возвращает значение атрибута attr

setAttribute(attr, value): устанавливает для атрибута attr значение value. Если атрибута нет, то он добавляется

removeAttribute(attr): удаляет атрибут attr и его значение

getBoundingClientRect(): возвращает объект со свойствами top, bottom, left, right, которые указывают на смещение элемента относительно верхнего левого угла браузера

9. Для чего используются в JavaScript объект form и массив forms?

Для работы с формами и обработки событий на ней (выборы checkbox, radio, кнопки, текстовые поля, списки) и обращению к свойствам форм.

10. Расскажите об операторах JavaScript?

Существуют операторы присваивания (=), сравнения (==, ===, >=, <=, >,< ,!=, !==), арифметические (%, ++, --, +, -, \*, \*\*), битовые (&, |, ^, ~ ,<<, >>, >>>) , логические (&&, ||, !), строковые (+, +=), условные (?), унарные (delete, typeof, void), отношения (in, instanceof), this, new, super