

## • Лабораторная работа №7

- *Изучение основ JavaScript, создание простейших функций и использование базовых операторов*

- JavaScript – язык программирования с динамической типизацией, очень широко распространен в веб-разработке. JavaScript изначально создавался для того, чтобы сделать web-странички «живыми». Программы на этом языке называются скриптами. В браузере они подключаются напрямую к HTML и, как только загружается страница – тут же выполняются. Сценарии на языке JavaScript загружаются браузером с сервера и выполняются на компьютере пользователя, соответственно для многих задач при работе на JS не нужно отсылать лишних запросов на сервер и ждать ответа для обработки. В JS есть широкие возможности обмена информацией с сервером без перезагрузки веб-страницы, что активно используется во многих современных приложениях.

- Для просмотра JavaScript-кода веб-страницы, откройте в браузере панель разработчика, перейдите во вкладку Source и затем выберите файл, код которого вам нужен.

- Приступим к написанию первого скрипта на JavaScript. Скопируйте ваш проект из прошлой лабораторной работы в директорию **lab7**. Затем, в папке **articles/static/** создайте папку **js**. В директории **js** создайте файл **helloworld.js**.

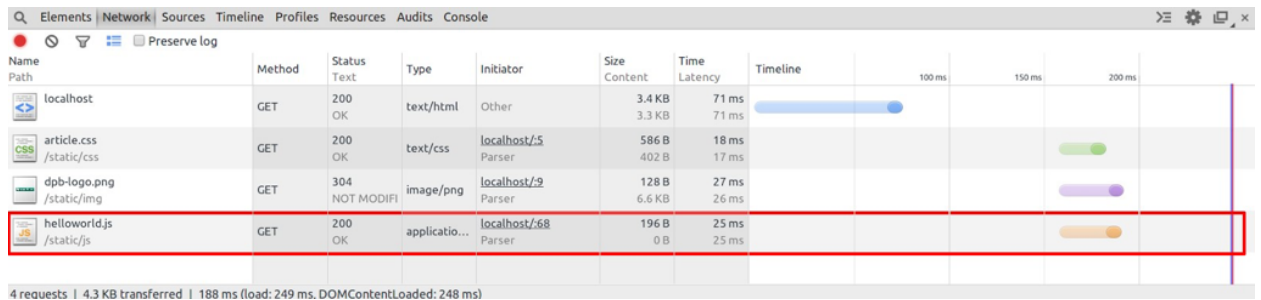
- Для того, чтобы подключить JavaScript-файл к веб-странице, необходимо добавить тег в html-шаблон:

- `<script src="{{ STATIC_URL }}js/helloworld.js"></script>`

- Данный тег может находиться в любом месте файла (но не за пределами `<head>` и `<body>`), однако, желательно скрипты подключать ближе к закрывающемуся тегу `<body>`, так как таким образом загрузка сценариев не будет мешать загрузке остального контента (текст, картинки, стили, шрифты).

- Затем подключите файл `helloworld.js` к `html`-шаблону `archive.html`. Запустите проект и пройдите по адресу `http://127.0.0.1:8000/`. Запустите панель отладчика, перейдите на вкладку `Network` и убедитесь, что `js`-файл есть в списке загруженных файлов, как это показано на рисунке 7.

- 



Name Path	Method	Status Text	Type	Initiator	Size Content	Time Latency	Timeline
localhost	GET	200 OK	text/html	Other	3.4 KB 3.3 KB	71 ms 71 ms	
article.css /static/css	GET	200 OK	text/css	localhost/5 Parser	586 B 402 B	18 ms 17 ms	
dpb-logo.png /static/img	GET	304 NOT MODIFI	image/png	localhost/9 Parser	128 B 6.6 KB	27 ms 26 ms	
helloworld.js /static/js	GET	200 OK	applicatio...	localhost/68 Parser	196 B 0 B	25 ms 25 ms	

4 requests | 4.3 KB transferred | 188 ms (load: 249 ms. DOMContentLoaded: 248 ms)

- Рисунок 7. JS-файл в списке загруженных

- После того, как файл был загружен, напишите его исходный код. Выполните задание, схожее с заданием из лабораторной работы №1. Создайте список студентов с полями: имя, фамилия, группа, оценки.

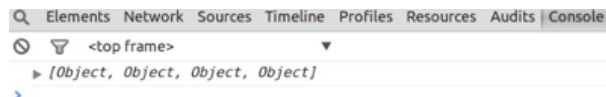
- *Пример:*

```
var groupmates = [
  {
    "name": "Александр",
    "surname": "Иванов",
    "group": "БВТ1702",
    "marks": [4, 3, 5]
  },
  {
    "name": "Иван",
    "surname": "Петров",
    "group": "БСТ1702",
    "marks": [4, 4, 4]
  },
  {
    "name": "Кирилл",
    "surname": "Смирнов",
    "group": "БАП1801",
```

- `"marks": [5, 5, 5]`
- `}`
- `];`

- Для вывода переменной на экран воспользуйтесь методом `console.log()`, в качестве аргументов функция принимает то, что нужно вывести на экран (в консоль). Например: `console.log(groupmates);`

- Перезагрузите страницу в браузере и в панели разработчика перейдите во вкладку Console. Там можно увидеть созданный вами массив, но в свернутом виде. Для того, чтобы посмотреть содержимое массива, нажмите на стрелку перед массивом.



#### • Рисунок 8. Созданный массив в свернутом виде

- Такая форма вывода не очень удобна. Создайте функцию, которая будет выводить содержимое вашего массива в виде таблицы, наподобие таблицы в лабораторной работе №1.

- ```
var rpad = function(str, length) {
    • // js не поддерживает добавление нужного количества
      символов
    • // справа от строки, т.е. аналога ljust из Python здесь нет
    • str = str.toString(); // преобразование в строку
    • while (str.length < length)
      • str = str + ' '; // добавление пробела в конец строки
    return str; // когда все пробелы добавлены, вернуть строку
  };
  •
  •
  • var printStudents = function(students){
    • console.log(
      • rpad("Имя", 15),
      • rpad("Фамилия", 15),
      • rpad("Группа", 8),
      • rpad("Оценки", 20)
    • );
    • // был выведен заголовок таблицы
    • for (var i = 0; i<=students.length-1; i++){
    • // в цикле выводится каждый экземпляр студента
      • console.log(
        • rpad(students[i]['name'], 15),
        • rpad(students[i]['surname'], 15),
        • rpad(students[i]['group'], 8),
```

```

    • rpad(students[i]['marks'], 20)
    • );
    • }
    • console.log('\n'); // добавляется пустая строка в конце
      вывода
    • };
    • printStudents(groupmates);
    • В данном случае, функция rpad была реализована для
      обеспечения форматирования строки с помощью добавления пробелов, так
      как в JS нет встроенных средств форматирования. Функция printStudents –
      выводит список всех студентов.

```

- **Задание:**

- Напишите функцию, которая фильтрует студентов по группе. Наименование группы, по которой будет проводиться фильтрация, вводится пользователем с клавиатуры.;
- Вам необходимо написать функцию фильтрации студентов по средней оценке, так, чтобы на экран выводился список студентов, средний балл которых выше заданного. Средний балл, по которому будет проводиться фильтрация, вводится пользователем с клавиатуры.

- 

- *Работа с элементами DOM с помощью JavaScript*

- Основным инструментом работы и динамических изменений на странице является DOM (Document Object Model) – объектная модель, используемая для XML/HTML-документов.

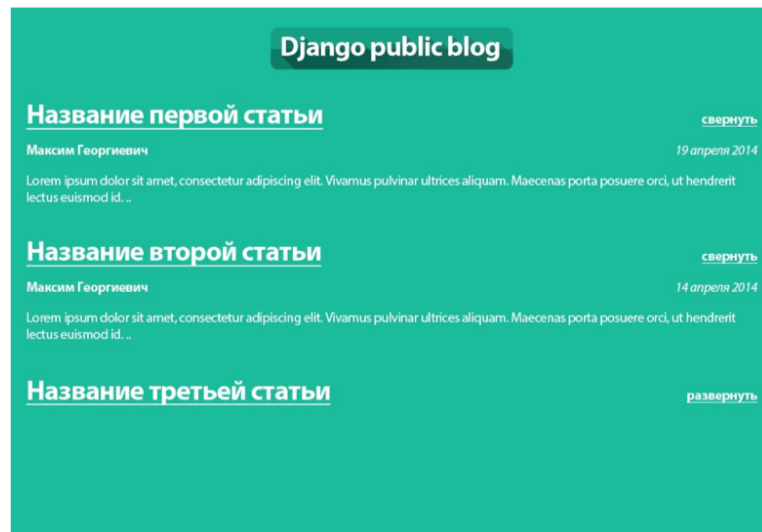
- Согласно DOM-модели, документ является иерархией, деревом. Каждый HTML-тег образует узел дерева с типом «элемент». Вложенные в него теги становятся дочерними узлами. Для представления текста создаются узлы с типом «текст».

- DOM – это представление документа в виде дерева объектов, доступное для изменения через JavaScript.

- С помощью JavaScript можно добавить динамичности веб-странице. Добавьте вашей странице со списком статей возможность скрыть и

развернуть информацию по каждому посту. Пример отображения представлен на рисунке 9. При нажатии на кнопку «свернуть», должны скрываться имя автора, время создания и сам текст, кнопка «свернуть» меняется на «развернуть».

- Для начала добавьте к каждой статье в html-шаблоне кнопку «свернуть», при этом добавив ей класс `fold-button`. Затем создайте файл **fold-post.js** в папке **articles/static/js/**.



- Рисунок 9. Страница отображения всех статей с возможностью сворачивания и разворачивания информации
- Затем создайте функцию, которая будет вызываться каждый раз, когда пользователь будет кликать по кнопке, функция должна выводить сообщение в консоль. JS-скрипт:

- ```
var foldBtns = document.getElementsByClassName("fold-button");
```
- ```
for (var i = 0; i < foldBtns.length; i++)
```
- ```
{ foldBtns[i].addEventListener("click", function(event) {
```
- ```
    console.log("you clicked ", event.target);
```
- ```
});
```
- ```
}
```
- `document` – глобальная переменная, для нее вызывается метод `getElementsByClassName()`, который ищет все элементы с указанным классом в структуре DOM и возвращает их массив. Затем цикл `for` проходит по всем элементам массива и к каждому элементу, с помощью метода `addEventListener`, добавляет обработчик событий `click`, который принимает

два параметра: название события и функцию, которая будет вызываться при каждом событии. Обратите внимание, что функция-обработчик может принимать аргумент (имя которого не имеет значения), и именно в этот аргумент будет вноситься объект, содержащий основную информацию о событии (ссылку на сам элемент, по которому кликнули, координаты курсора в момент события, какие были зажаты в этот момент клавиши и т. п.).

- Зайдите в панель разработчика в вашем браузере и поставьте брейкпоинт (точку останова, breakpoint) на пятой строке, нажав на номер строки (строка, в которой указан код: `console.log("you clicked ", event.target);`). Затем нажмите на кнопку «свернуть» для того, чтобы запустить код на исполнение. Теперь вам доступна отладка скрипта пошагово.

- Далее приступим к разработке и реализации появления и скрывания информации о статье. Добавьте данный код во внутрь функции-обработчика:

- ```
foldBtns[i].addEventListener("click", function(e) {  
    • event.target  
        • .parentElement  
        • .getElementsByClassName('article-author')[0]  
        • .style.display = "none";  
    • event.target  
        • .parentElement  
        • .getElementsByClassName('article-created-date')[0]  
        • .style.display = "none";  
    • event.target  
        • .parentElement  
        • .getElementsByClassName('article-text')[0]  
        • .style.display = "none";  
    • e.target.innerHTML = "развернуть";  
});
```

- Сначала в операторе с помощью атрибута `parentElement` определяется родитель элемента по которому произошел клик, затем внутри этого родителя производится поиск всех элементов с классом “article-author” с помощью метода `getElementsByClassName()`. Этот метод возвращает массив элементов, потому что по умолчанию один и тот же класс может иметь несколько элементов (даже если в узле DOM-а, где производится поиск, есть

всего один элемент с указанным классом, метод все равно вернет список, хоть и с одним элементом). В соответствии с созданной вами html-разметкой внутри родителя one-post только у одного блока есть такой класс, значит метод возвращает список с одним элементом, доступ к которому можно получить по нулевому номеру. Теперь остается этому элементу с помощью атрибутов style и display указать значение none. Именно css-свойство display: none позволяет скрывать элемент на странице.

- После необходимо у элемента, по которому кликнули, изменить текст, эта операция выполняется с помощью атрибута innerHTML.

- После реализации скрытия необходимо вернуть исходное состояние при повторном клике на эту кнопку. Так как при любой ситуации функция-обработчик, созданная вами ранее, будет вызвана, нужно именно в ней решить эту проблему. Сделайте так, чтобы при нажатии на кнопку «свернуть», самой кнопке добавлялся класс folded: `e.target.className = "fold-button folded"`;

- Обратите внимание, что у кнопки уже есть класс fold-button, который не следует удалять, поэтому через пробел был добавлен описывающий текущее состояние класс. Однако, раз у кнопки, которая уже была нажата, есть дополнительный класс, значит вы можете в начале функции-обработчика проверять наличие этого класса, и если он есть, то не скрывать информацию о записи, а наоборот показывать её. Для этого добавьте проверку и в случае, если у кнопки есть класс folded, сделайте вновь видимой скрытую информацию:

- ```
foldBtns[i].addEventListener("click", function(e) {  
    • if (e.target.className == "fold-button folded"){  
        • e.target.innerHTML = "свернуть";  
        • e.target.className = "fold-button";  
        • var displayState = "block";  
    • }  
    • else{  
        • e.target.innerHTML = "развернуть";  
        • e.target.className = "fold-button folded";  
        • var displayState = "none";  
    • }  
})
```

- `event.target`
  - `.parentElement`
  - `.getElementsByClassName('article-author')[0]`
  - `.style.display = displayState;`
- `event.target`
  - `.parentElement`
  - `.getElementsByClassName('article-created-date')[0]`
  - `.style.display = displayState;`
- `event.target`
  - `.parentElement`
  - `.getElementsByClassName('article-text')[0]`
  - `.style.display = displayState;`
- `});`
- **Задание:**
  - Сделайте другую реализацию функции-обработчика, которая была бы более компактна. Предложенный вариант довольно громоздок, однако он хорошо иллюстрирует некоторые возможности манипулирования DOM с помощью JavaScript. Весь описанный выше функционал можно реализовать с помощью CSS-классов и их изменения через функцию-обработчик. Сделайте так, чтобы класс `folded` добавлялся не кнопке, по которой кликнули, а родителю всего поста (это тот самый элемент, у которого в html-разметке установлен класс `one-post`). В таком случае свойство `display` можно изменять через CSS-стили:
    - `.one-post.folded .article-author{`
    - `/* данный стиль применится только для элементов класса`
    - `.article-author, у которых родитель с классом one-post`
    - `имеет также класс folded */`
    - `display: none;`
    - `}`
  - По аналогии с разработанной вами функцией, установите через CSS-стили «исчезновение» остальных элементов поста.