Полный текст лекции: Введение в JavaScript и DOM

## Введение

Сегодня мы начнем изучение языка JavaScript — одного из важнейших языков веб-разработки. Это язык, который позволяет «оживить» страницу: сделать её интерактивной, реагирующей на действия пользователя. JavaScript выполняется прямо в браузере, без дополнительной установки, и является единственным языком, который изначально поддерживается всеми браузерами.  
  
На стороне сервера могут использоваться другие языки (например, Python, Java, PHP), но JavaScript работает на клиенте, то есть в браузере пользователя. Это делает его незаменимым инструментом как для frontend-разработчиков, так и для тестировщиков, которые анализируют поведение веб-приложений.

## Слайд 1: JavaScript — оживляем страницу

JavaScript — это язык программирования, встроенный в браузеры. Он делает страницы интерактивными: позволяет обрабатывать клики, ввод в поля, менять содержимое страницы без перезагрузки. Без JavaScript сайт был бы статичным. JS — это логика, поведение, реакция на действия пользователя.

## Слайд 2: Первое знакомство с кодом

Пример кода:  
<script>console.log("Привет, мир!");</script>  
  
Чтобы увидеть результат, откройте в браузере инструменты разработчика (нажмите F12 или ПКМ → Инспектировать). Перейдите на вкладку Console. Там вы увидите сообщение 'Привет, мир!'. Это не вывод для пользователя, а отладочная информация для разработчика.

## Слайд 3: Переменные и типы

Переменная — это ячейка для хранения данных. В JS переменные создаются через let и const:  
let name = "Alice"; // строка  
let age = 30; // число  
  
console.log(name, age);  
  
Используйте let для изменяемых значений, const — для неизменяемых. Старый способ var использовать не рекомендуется.

## Слайд 4: Операции

JavaScript поддерживает арифметические операции:  
let x = 5, y = 2;  
let sum = x + y;  
console.log("x+y=", sum); // 7  
  
Также возможна конкатенация — объединение строк:  
let msg = "Hello, " + name;

## Слайд 5: Условия

Для выполнения действий по условию используем конструкцию if/else:  
if (age >= 18) {  
 console.log("Совершеннолетний");  
} else {  
 console.log("Несовершеннолетний");  
}  
  
Сравнения: >, <, ===, !==  
Логические операторы: && (И), || (ИЛИ)

## Слайд 6: Циклы

Цикл позволяет повторить действия:  
for (let i = 1; i <= 5; i++) {  
 console.log(i);  
}  
  
Здесь три части:  
- инициализация (let i = 1)  
- условие (i <= 5)  
- шаг (i++)  
  
Есть также while и do...while, но чаще используют for.

## Слайд 7: Функции

Функции позволяют переиспользовать код:  
function greet(username) {  
 console.log("Привет, " + username);  
}  
greet("Мир");  
  
Функция может принимать параметры и возвращать значения через return.

## Слайд 8: Практика в консоли

Попросите учеников сделать в консоли:  
- 2 + 2  
- let name = "Катя"  
- console.log("Привет, " + name)  
- Написать условие, проверяющее возраст  
  
Цель — почувствовать, что код работает прямо в браузере и реагирует на действия пользователя.

## Слайд 9: Итог по JS

Вы познакомились с основами:  
- Переменные  
- Операции  
- Условия  
- Циклы  
- Функции  
  
Домашнее задание: написать скрипт, который выводит все чётные числа от 1 до 10 в консоль.

## Занятие 5: Работа с DOM

DOM — это представление HTML-страницы в виде объекта. С его помощью можно изменять структуру и содержание страницы через JS.

## Слайд 1: DOM — дерево элементов

HTML превращается в объектное дерево. С каждым элементом можно взаимодействовать как с объектом: получить его, изменить, удалить.

## Слайд 2: Поиск и изменение элемента

HTML:  
<p id="demo">Hello</p>  
  
JS:  
let p = document.getElementById("demo");  
p.innerText = "Привет!";  
  
Также:  
document.querySelector(".highlight") — по классу

## Слайд 3: Демонстрация через консоль

Откройте страницу, нажмите F12, перейдите в Console.  
Выполните: document.getElementById("demo").innerText = "Привет!";  
Изменения произойдут мгновенно.

## Слайд 4: События

HTML: <button id="btn">Нажми меня</button>  
  
JS:  
let btn = document.getElementById("btn");  
btn.addEventListener("click", function() {  
 alert("Кнопка нажата!");  
});

## Слайд 5: Динамическое обновление

HTML: <input id="nameInput"> <span id="greet"></span>  
  
JS:  
document.getElementById("nameInput").addEventListener("input", function() {  
 let name = this.value;  
 document.getElementById("greet").innerText = "Привет, " + name;  
});

## Слайд 6: Инструменты разработчика

DevTools (F12):  
- Вкладка Elements — HTML-структура  
- Console — ввод команд и отладка  
- Мгновенные изменения, не затрагивающие файл.

## Слайд 7: Отладка

Вкладка Sources → поставьте точку останова (breakpoint).  
Когда скрипт дойдет до этой строки, выполнение остановится. Можно шагать по коду, следить за переменными.

## Слайд 8: Пример отладки

Напишите ошибочный код, например: console.log(usrname);  
В консоли будет ошибка: переменная не определена.  
Используйте DevTools, чтобы найти и исправить её.

## Слайд 9: Практика

HTML-файл с кнопкой и полем. Ученики:  
- пишут JS, который при клике показывает текст из поля через alert.  
Наставник помогает, если не работает.

## Слайд 10: Резюме

JS позволяет взаимодействовать с HTML.  
DevTools помогают отлаживать, исправлять и исследовать поведение страницы.

## Дополнительно: Обмен данными с сервером

JS может отправлять и получать данные от backend-сервера:  
- Метод fetch позволяет делать HTTP-запросы.  
  
Пример:  
fetch("https://example.com/api/user")  
 .then(response => response.json())  
 .then(data => console.log(data));  
  
Это позволяет делать запросы к серверу, получать данные и использовать их на странице. Следующий урок будет посвящён этому более подробно.