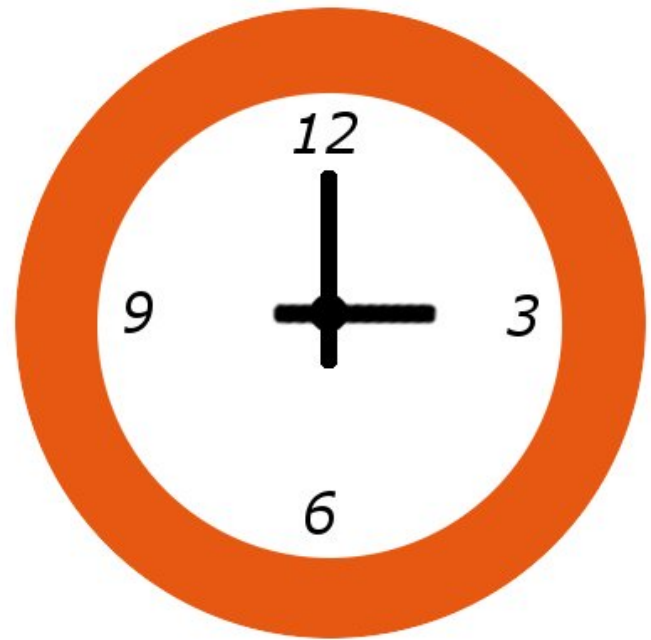
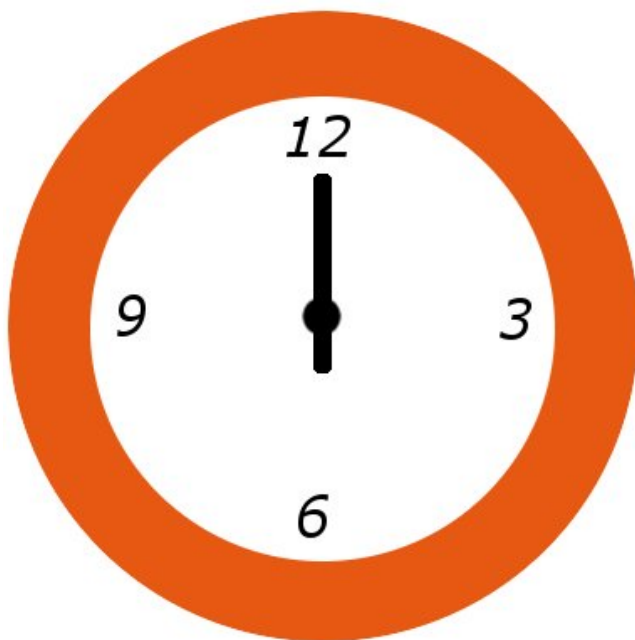


Лаврик Дмитрий

**Как перестать
бояться изучать
JavaScript за 3 часа!?**



Всем привет! Итак, у нас с Вами есть чуть менее трёх часов, чтобы перестать бояться изучать JavaScript. Но думаю, что мы справимся намного быстрее! Итак, поехали!

0. Где писать код

Самый первый и самый важный вопрос – где JavaScript-код-то писать? Сейчас мы будем делать это самым простым способом – использовать парный тег `<script></script>`.

Внимание! Задание

Наберите следующий код:

```
<html>
  <head>
    <title>Hello, JavaScript!</title>
    <meta content="text/html; charset=Windows-1251" http-
      equiv="content-type"/>
    <script language="javascript">
      alert('Hello, JavaScript!');
    </script>
  </head>
  <body>Веб-страничка</body>
</html>
```

Как успехи? Появилось всплывающее окошко? Если всё переписали правильно, то просто не могло не появиться! Поздравляю, сейчас Вы написали свой первый скрипт на данном языке программирования! В дальнейшем весь JavaScript-код мы будем писать вот здесь

Это было совсем просто, переходим к более существенным страхам.

1. Какой магией JavaScript меняет страницу

Для начала Вам нужно понять, что вся интерактивность появляется от того, что JavaScript производит какие-то изменения с элементами нашей страницы. Но! Делает он это отнюдь не каким-то магическим образом и не используя невероятные супер технологии. Ну-ка, вспомните, какую разметку у нас читает браузер? Правильно – html! А что мы используем для оформления страниц? Правильно – css!

Так вот, с помощью JavaScript-а мы просто меняем либо html-код, либо css-стили. Но ведь это, должно быть, жутко неудобно, - наверняка подумали Вы, - ведь из той html-строчки, которую вернул сервер, очень сложно получать нужные элементы, и как переписывать их значения?!

Сейчас мы подходим к первому важному моменту, после которого Ваши опасения должны серьёзно уменьшиться! Дело в том, что браузер уже считал весь ответ от сервера и построил объектную модель документа.

DOM (Document Object Model — «объектная модель документа») - это не зависящий от платформы и языка программный интерфейс, позволяющий программам и скриптам получить доступ к содержимому HTML, XHTML и XML-документов, а также изменять содержимое, структуру и оформление таких документов.

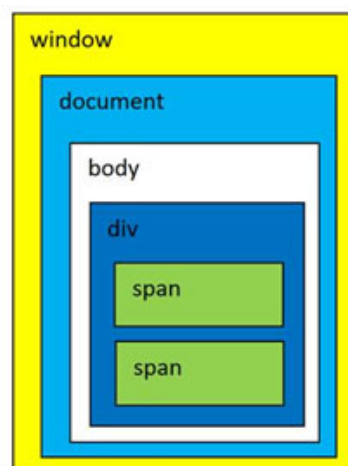
Т.е. JavaScript-ом Вы будете работать со специальной удобной конструкцией, с помощью которой удобно дотягиваться до элементов и менять их содержимое и структуру. Когда же у нас эта загадочная DOM формируется? Рассмотрим схему клиент-серверного взаимодействия:

1. КЛИЕНТ: набираете в браузере адрес
2. КЛИЕНТ-СЕРВЕР: HTTP запрос
3. СЕРВЕР: обработка запроса, генерация html
4. СЕРВЕР-КЛИЕНТ: html документа
5. КЛИЕНТ: браузер парсит html, строит DOM и отображает страницу на экране

Внимание! Задание

Напишите html-код, который Вы видите на картинке слева.

```
<html>
<head>
  <title>Привет, JavaScript!</title>
</head>
<body>
  <div>
    <span>Раз</span>
    <span>Два</span>
  </div>
</body>
</html>
```



При разборе такого html, браузер сформирует DOM, которую Вы видите на той же картинке справа. Пока что мы не будем останавливаться на объектах window и document, а займёмся понятными нам сущностями, которые мы только что прописали в html-коде. Мы видим, что в объект body вложен объект div, в который, в свою очередь, вложены 2 объекта span. И поверьте, что дотянуться до нужного нам объекта и изменить его содержимое будет совсем несложно!

Однако, перед тем, как мы это сделаем, необходимо разобраться с ещё одним существенным моментом.

2. Какими событиями вызывается JavaScript

Возможно, Вы где-то слышали или читали о том, по каким событиям происходит вызов JavaScript-а, про загрузку страницы, отложенную загрузку и т.п. Выкидываем на ближайший час это из головы! Пока что весь наш код мы будем писать внутри следующей конструкции:

```
<script language="javascript">
  window.onload = function()
  {
    // здесь будет наш код
  }
</script>
```

Благодаря этому, код начнёт выполняться только после загрузки всего документа, т.е., когда DOM полностью сформирована. Это обеспечит корректную и понятную работу кода, который мы в дальнейшем будем писать. А про остальные возможности ещё вспомним, но... только после того, как Вы перестанете бояться!

3. Как же трудно дотянуться до нужного объекта

Ну, вот и пришло время поработать с элементами страницы! Мы воспользуемся самым простым способом – присвоим нужному элементу id, а затем получим его с помощью специального метода.

Внимание! Задание

Сначала создаём идентификатор. В тот код, который Вы переписывали с картинки, добавляем div-у атрибут id = "main". А в тег <head></head>

добавьте конструкцию из предыдущего пункта. На картинке ниже Вы можете увидеть код, который должны были получить.

```
1 <html>
2 <head>
3     <title>Привет, JavaScript!</title>
4     <script language="javascript">
5         window.onload = function()
6         {
7         }
8     </script>
9 </head>
10 <body>
11     <div id="main">
12         <span>Раз</span>
13         <span>Два</span>
14     </div>
15 </body>
16 </html>
17
```

А теперь дотянемся до желанного объекта. Для этого внутри window.onload пишем

```
document.getElementById('main');
```

Однако, во-первых, мы никак не увидели результат работы данной записи, а во-вторых, пока что получили только сам объект. Для того, чтобы получить содержимое любого объекта, нужно обратиться к его свойству innerHTML. Давайте добавим это к предыдущей строке кода и поместим её в alert, чтобы увидеть результат работы на экране. Итак, пишем:

```
alert(document.getElementById('main').innerHTML);
```

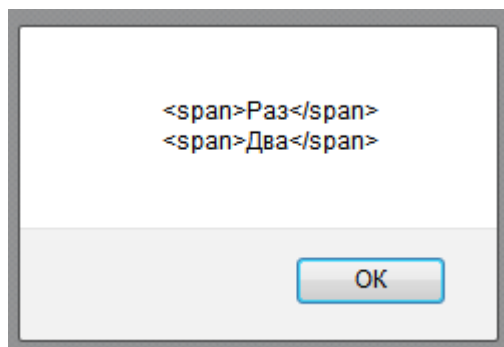
Ниже можете посмотреть скриншот того, что у Вас должно быть набрано.

```
4 <script language="javascript">
5     window.onload = function()
6     {
7         alert(document.getElementById('main').innerHTML);
8     }
9 </script>
```

Что ж, сохраняйте изменения и открывайте страничку в браузере!

Внимание! Пишите символы в таком же регистре!

Если вы всё правильно сделали, то должны были увидеть всплывающее окошко со следующим содержимым:



(именно такое окно выдаёт Mozilla. В других браузерах оно выглядит иначе)

Давайте проверим, всё ли правильно? Действительно, именно такой html-код находился внутри нашего div-а!

Так же легко мы можем менять содержимое интересующих нас элементов. Для этого, получив содержимое элемента, воспользуемся оператором присваивания:

```
document.getElementById('main').innerHTML = '<h1>Новое содержимое</h1>';
```

```
4 <script language="javascript">
5   window.onload = function()
6   {
7       document.getElementById('main').innerHTML = '<h1>Новое содержимое</h1>';
8   }
9 </script>
```

Сохраните изменения и обновите страничку. Не правда ли, контент изменился!

Вот таким нехитрым образом можно дотягиваться до элементов страницы и менять их содержимое. До сих пор страшно?

4. С содержимым понятно, а стили-то как менять?

Также несложно. Дело в том, что у каждого объекта есть свойство style. Дотянемся до него:

```
document.getElementById('main'). style;
```

Теперь дело остаётся за малым. Суть в том, что style хранит в себе все свойства, оформляющие данный элемент. Названия почти такие же, как в css,

а дотянуться до них можно, опять же, с помощью записи через точку.
Давайте, зададим нашему div-у несколько свойств оформления:

```
document.getElementById('main').style.color = 'red';  
document.getElementById('main').style.fontSize = '136px';  
document.getElementById('main').style.marginLeft = '100px';
```

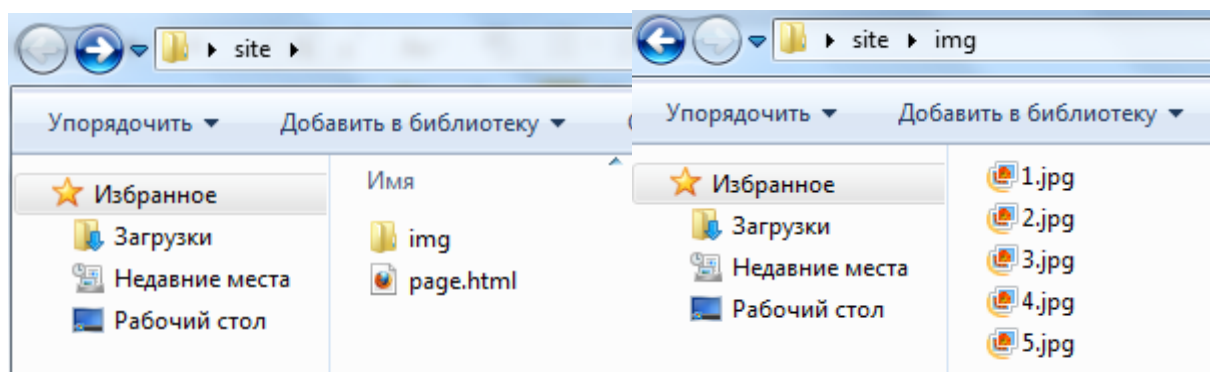
```
4 <script language="javascript">  
5     window.onload = function()  
6     {  
7         // Красный цвет  
8         document.getElementById('main').style.color = 'red';  
9  
10        // Размер шрифта - 136 пикселей  
11        document.getElementById('main').style.fontSize = '136px';  
12  
13        // Отступ слева - 100 пикселей  
14        document.getElementById('main').style.marginLeft = '100px';  
15    }  
16 </script>
```

Ну, вот, и оформление научились менять!

5. Это только пока что всё просто, а написать что-то более серьёзное очень сложно

Неправда! Сейчас мы с Вами напишем самый настоящий слайдер фотографий, и это будет совсем нестрашно.

Для начала определимся со структурой сайта. В корне сайта у нас будет страничка page.html и папка с фотографиями – img. Пусть у нас будет 5 фото, и назовём мы их 1.jpg, 2.jpg ... 5.jpg.



(корневая директория)

(папка с изображениями)

Внимание! Задание

Создайте данную структуру сайта.

Теперь в `<body></body>` напишите следующий код:

```
<body>
  <div id="photo">
    
  </div>
  <span id="prev">Предыдущее фото</span>
  <span id="next">Следующее фото</span>
</body>
```

Мы создаём один `div`, как контейнер для фотографий, и два `span`-а, которые станут кнопками смены фото. Запустите данную страницу.

Внимание! Если Вы делаете данный пример не в новом html-документе, а продолжаете предыдущие примеры, не забудьте сейчас стереть старый JavaScript-код!

Если Вы всё сделали правильно, то на экране должны увидеть примерно следующее (разумеется, с другим фото):



Предыдущее фото Следующее фото

Теперь нам нужно создать 2 обработчика событий: по нажатию на `span` с `id` «prev», и на `span` с `id` «next». Для этого получим данные элементы и воспользуемся свойством `onclick`. Смысл записи, которую Вы увидите ниже, заключается в следующем: свойству `onclick` конкретного элемента мы присваиваем функцию, которая отработает после клика по данному элементу.

```
document.getElementById('next').onclick = function()
{
    // здесь мы будем писать код, который отработает по клику на элемент
    // с id «next»
}
```

```
document.getElementById('prev').onclick = function()
{
    // здесь мы будем писать код, который отработает по клику на элемент
    // с id «prev»
}
```

Внимание! Это мы пишем также внутри `window.onload`

Что ж, обработчики событий готовы, но сами по себе они нам ничего не дают. Необходимо придумать логику показа фотографий. Очевидно, для того, чтобы показать пользователю другую фотографию, нам придётся поменять содержание нашего контейнера, например, с

```

```

на

```

```

Для удобства мы назвали фотографии 1.jpg, 2.jpg ... 5.jpg. Это означает, что мы легко можем определить, какую фотографию необходимо показать пользователю. Например, если в данный момент загружено фото 3.jpg, и была нажата кнопка «Следующее фото», то мы подгружаем картинку с номером на один больше, т.е. 4.jpg. Соответственно, для загрузки предыдущей фотографии – с номером на 1 меньше.

В связи с этим нам удобно было бы хранить номер фотографии в какой-то переменной и менять его в зависимости от действий пользователя. Давайте эту переменную создадим:

```
var i = 1; // 1, потому что у нас изначально загружена 1-ое фото
```

Внимание! Объявите переменную перед обработчиками событий.

На данный момент у Вас должен быть написан следующий код:

```
1 <html>
2 <head>
3     <title>Привет, JavaScript!</title>
4     <script language="javascript">
5         window.onload = function()
6         {
7             var i = 1;
8             document.getElementById('next').onclick = function()
9             {
10            }
11            document.getElementById('prev').onclick = function()
12            {
13            }
14        }
15    </script>
16 </head>
17 <body>
18     <div id="photo">
19         
20     </div>
21     <span id="prev">Предыдущее фото</span>
22     <span id="next">Следующее фото</span>
23 </body>
24 </html>
```

Теперь ещё немного обдумаем логику. Если пользователь, просматривая первую фотографию, нажмёт «предыдущее фото», то мы не должны на это реагировать (в самом деле, не формировать же ссылку на нулевую фотографию). Аналогичная ситуация, когда при просмотре пятой фотографии будет нажата кнопка «следующее фото».

Т.е., в обработчике события для кнопки «предыдущее фото» мы должны поставить проверку, не равно ли сейчас значение переменной *i* единице, а в обработчике события для кнопки «следующее фото» - не равно ли оно пяти. Если данные проверки вернут истину, то значение счётчика мы трогать не будем. В противном же случае, либо увеличим его на единицу (если нажата кнопка «следующее фото»), либо уменьшим («предыдущее фото»). Затем, согласно текущему значению счётчика, сформируем новую ссылку на фотографию.

Итак, что же у нас получается? Формировать ссылки мы будем следующим образом:

```
'';
```

Помещаем необходимые проверки в соответствующие обработчики.

```

4 <script language="javascript">
5   window.onload = function()
6   {
7     var i = 1;
8
9     document.getElementById('next').onclick = function()
10    {
11      if(i != 5)
12        i++;
13      document.getElementById('photo').innerHTML = '';
16    }
17
18    document.getElementById('prev').onclick = function()
19    {
20      if(i != 1)
21        i--;
22      document.getElementById('photo').innerHTML = '';
25    }
26  }
27 </script>

```

Ура! Поздравляю! Только что мы с Вами создали настоящий слайдер фотографий! И весь код занял всего лишь 21 строчку!

Ну что ж, надеюсь, что после проделанной работы Вы стали бояться JavaScript-а намного меньше, и теперь, наоборот, захотите его изучать!