Главная задача. Спросить у пользователя логин и пароль. При верном написать правильная авторизация.



Чтобы понять действие условных операторов, мы предлагали представить их в виде разветвлений на дороге, по которой двигается интерпретатор JavaScript. Циклы можно представить как разворот на дороге, возвращающий обратно, который заставляет интерпретатор многократно проходить через один и тот же участок программного кода.

В языке JavaScript имеется четыре цикла: while, do/while, for и for/in. Каждому из них посвящен один из следующих подразделов. Одно из обычных применений циклов - обход элементов массива.

**Цикл while**

Оператор if является базовым условным оператором в языке JavaScript, а базовым циклом для JavaScript можно считать цикл while. Он имеет следующий синтаксис:

*while (выражение)*

*{*

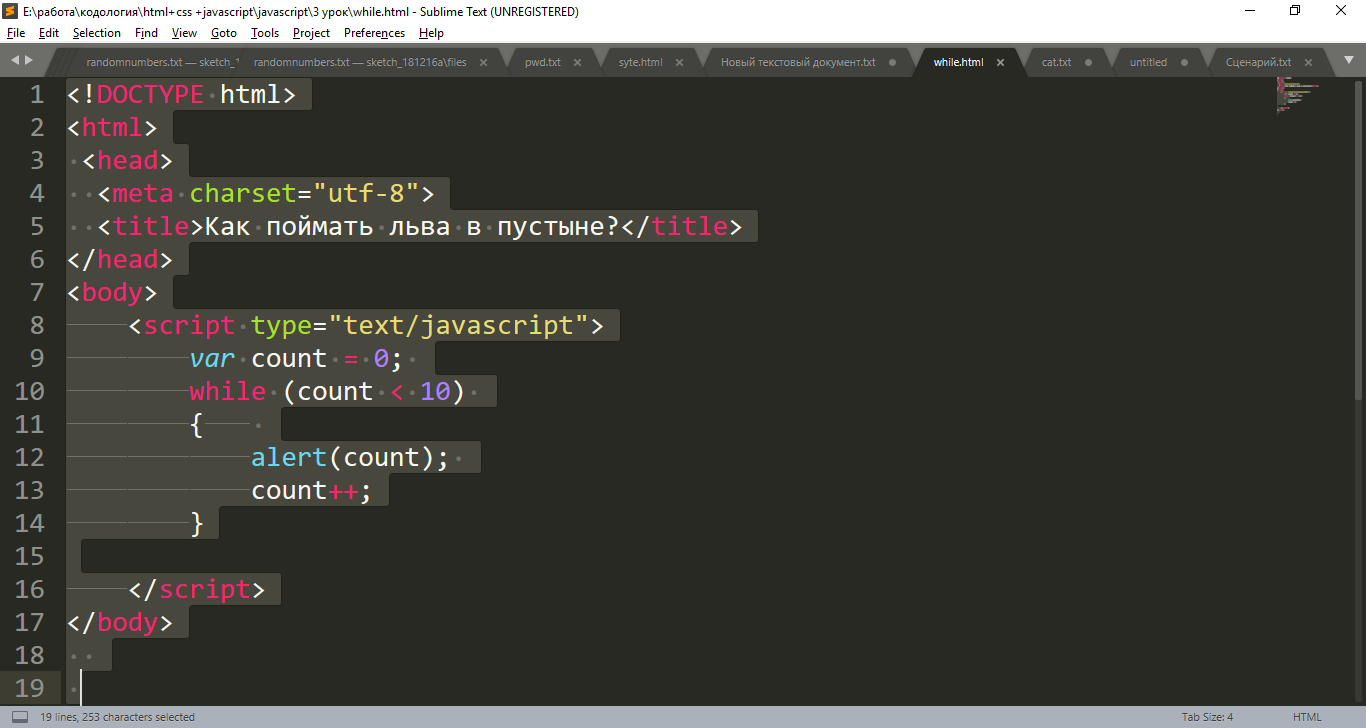
*инструкция*

*}*

Цикл while начинает работу с вычисления выражения. Если это выражение имеет ложное значение, интерпретатор пропускает инструкцию, составляющую тело цикла, и переходит к следующей инструкции в программе. Если выражение имеет истинное значение, то выполняется инструкция, образующая тело цикла, затем управление передается в начало цикла и выражение вычисляется снова. Иными словами, интерпретатор снова и снова выполняет инструкцию тела цикла, пока значение выражения остается истинным. Обратите внимание, что имеется возможность организовать бесконечный цикл с помощью синтаксиса while(true).

Обычно не требуется, чтобы интерпретатор JavaScript снова и снова выполнял одну и ту же операцию. Почти в каждом цикле с каждой итерацией цикла одна или несколько переменных изменяют свои значения. Поскольку переменная меняется, действия, которые выполняет инструкция, при каждом проходе тела цикла могут отличаться.

Кроме того, если изменяемая переменная (или переменные) присутствует в выражении, значение выражения может меняться при каждом проходе цикла. Это важно, т.к. в противном случае выражение, значение которого было истинным, никогда не изменится и цикл никогда не завершится! Ниже приводится пример цикла while, который выводит числа от 0 до 9:



Как видите, в начале переменной count присваивается значение 0, а затем ее значение увеличивается каждый раз, когда выполняется тело цикла. После того как цикл будет выполнен 10 раз, выражение вернет false (т.е. переменная count уже не меньше 10), инструкция while завершится и интерпретатор перейдет к следующей инструкции в программе. Большинство циклов имеют переменные-счетчики, аналогичные count. Чаще всего в качестве счетчиков цикла выступают переменные с именами i, j и k, хотя для того чтобы сделать программный код более понятным, следует давать счетчикам более наглядные имена.

## Цикл for

Цикл for представляет собой конструкцию цикла, которая часто оказывается более удобной, чем цикл while. Цикл for упрощает конструирование циклов, следующих шаблону, общему для большинства циклов. Большинство циклов имеют некоторую переменную-счетчик. Эта переменная инициализируется перед началом цикла и проверяется перед каждой итерацией. Наконец, переменная-счетчик инкрементируется или изменяется каким-либо другим образом в конце тела цикла, непосредственно перед повторной проверкой переменной. Инициализация, проверка и обновление - это три ключевых операции, выполняемых с переменной цикла. Инструкция for делает эти три шага явной частью синтаксиса цикла:

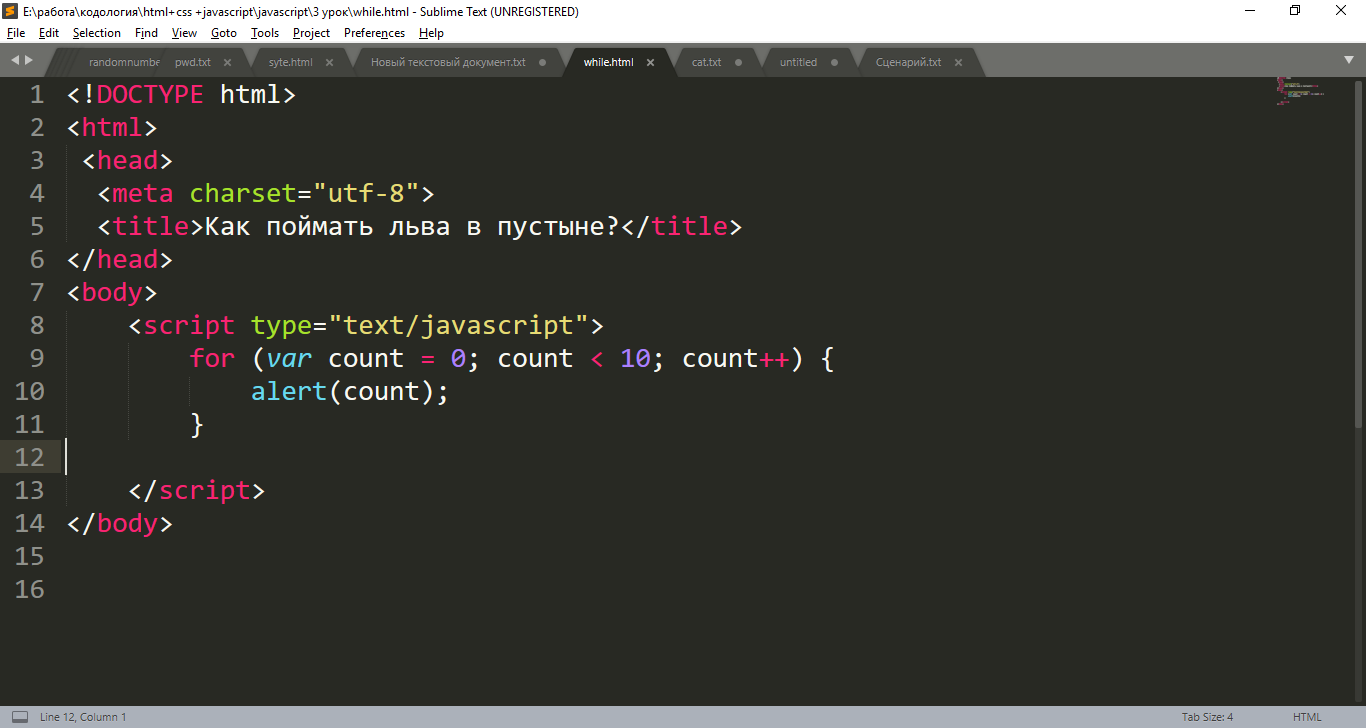
*for(инициализация; проверка; инкремент)*

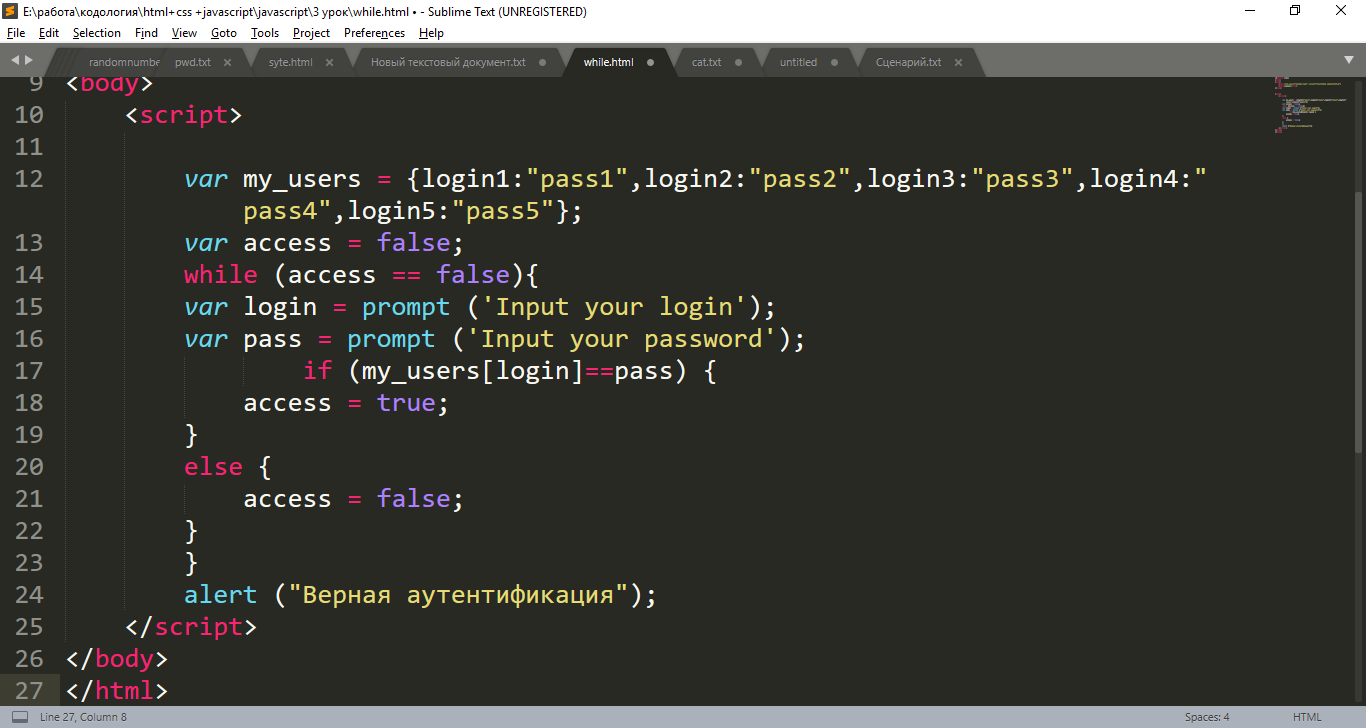
*{*

*инструкция*

*}*

Инициализация, проверка и инкремент - это три выражения (разделенных точкой с запятой), которые ответственны за инициализацию, проверку и увеличение переменной цикла. Расположение их в первой строке цикла упрощает понимание того, что делает цикл for, и не позволяет забыть инициализировать или увеличить переменную цикла.





document.location.href = "http://www.ya.ru";

Функции и формы:

# Функции

Зачастую нам надо повторять одно и то же действие во многих частях программы.

Например, красиво вывести сообщение необходимо при приветствии посетителя, при выходе посетителя с сайта, ещё где-нибудь.

Чтобы не повторять один и тот же код во многих местах, придуманы функции. Функции являются основными «строительными блоками» программы.

Примеры встроенных функций вы уже видели – это alert(message), prompt(message, default) и confirm(question). Но можно создавать и свои.

