**Переменная**

Переменная – это «именованное хранилище» для данных. Мы можем использовать переменные для хранения товаров, посетителей и других данных.

Для создания переменной в JavaScript используйте ключевое слово let.

Приведённая ниже инструкция создаёт (другими словами: *объявляет* или *определяет*) переменную с именем «message»:

let message;

Теперь можно поместить в неё данные, используя оператор присваивания =:

message = 'Hello'; // сохранить строку

alert(message); // показывает содержимое переменной

Для краткости можно совместить объявление переменной и запись данных в одну строку:

let message = 'Hello!'; // определяем переменную и присваиваем ей значение

alert(message); // Hello!

Мы также можем объявить несколько переменных в одной строке:

let user = 'John', age = 25, message = 'Hello';

Такой способ может показаться короче, но мы не рекомендуем его. Для лучшей читаемости объявляйте каждую переменную на новой строке.

Многострочный вариант немного длиннее, но легче для чтения:

let user = 'John';

let age = 25;

let message = 'Hello';

Некоторые люди также определяют несколько переменных в таком вот многострочном стиле:

let user = 'John',

age = 25,

message = 'Hello';

…Или даже с запятой в начале строки:

let user = 'John'

, age = 25

, message = 'Hello';

В принципе, все эти варианты работают одинаково. Так что это вопрос личного вкуса и эстетики.

var вместо let

В старых скриптах вы также можете найти другое ключевое слово: var вместо let:

var message = 'Hello';

Ключевое слово var – *почти* то же самое, что и let. Оно объявляет переменную, но немного по-другому, «устаревшим» способом.

Есть тонкие различия между let и var, но они пока не имеют для нас значения. Мы подробно рассмотрим их в главе Устаревшее ключевое слово "var".

**Имена переменных**

В JavaScript есть два ограничения, касающиеся имён переменных:

1. Имя переменной должно содержать только буквы, цифры или символы $ и \_.
2. Первый символ не должен быть цифрой.

Примеры допустимых имён:

let userName;

let test123;

Если имя содержит несколько слов, обычно используется **верблюжья нотация**, то есть, слова следуют одно за другим, где каждое следующее слово начинается с заглавной буквы: myVeryLongName.

Самое интересное – знак доллара '$' и подчёркивание '\_' также можно использовать в названиях. Это обычные символы, как и буквы, без какого-либо особого значения.

Эти имена являются допустимыми:

let $ = 1; // объявили переменную с именем "$"

let \_ = 2; // а теперь переменную с именем "\_"

alert($ + \_); // 3

Примеры неправильных имён переменных:

let 1a; // не может начинаться с цифры

let my-name // дефис '-' не разрешён в имени

Регистр имеет значение

Переменные с именами apple and AppLE – это две разные переменные.

Нелатинские буквы разрешены, но не рекомендуются

Можно использовать любой язык, включая кириллицу или даже иероглифы, например:

let имя = '...';

let 我 = '...';

Технически здесь нет ошибки, такие имена разрешены, но есть международная традиция использовать английский язык в именах переменных. Даже если мы пишем небольшой скрипт, у него может быть долгая жизнь впереди. Людям из других стран, возможно, придётся прочесть его не один раз.

**Зарезервированные имена**

Существует [список зарезервированных слов](https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Lexical_grammar#%D0%9A%D0%BB%D1%8E%D1%87%D0%B5%D0%B2%D1%8B%D0%B5_%D1%81%D0%BB%D0%BE%D0%B2%D0%B0), которые нельзя использовать в качестве имён переменных, потому что они используются самим языком.

Например: let, class, return и function зарезервированы.

Приведённый ниже код даёт синтаксическую ошибку:

let let = 5; // нельзя назвать переменную "let", ошибка!

let return = 5; // также нельзя назвать переменную "return", ошибка!

Создание переменной без использования use strict

Обычно нам нужно определить переменную перед её использованием. Но в старые времена было технически возможно создать переменную простым присвоением значения без использования let. Это все ещё работает, если мы не включаем use strict в наших файлах, чтобы обеспечить совместимость со старыми скриптами.

// заметка: "use strict" в этом примере не используется

num = 5; // если переменная "num" раньше не существовала, она создаётся

alert(num); // 5

Это плохая практика, которая приводит к ошибке в строгом режиме:

"use strict";

num = 5; // error: num is not defined

**Константы**

Чтобы объявить константную, то есть, неизменяемую переменную, используйте const вместо let:

const myBirthday = '18.04.1982';

Переменные, объявленные с помощью const, называются «константами». Их нельзя изменить. Попытка сделать это приведёт к ошибке:

const myBirthday = '18.04.1982';

myBirthday = '01.01.2001'; // ошибка, константу нельзя перезаписать!

Если программист уверен, что переменная никогда не будет меняться, он может гарантировать это и наглядно донести до каждого, объявив её через const.

**Константы в верхнем регистре**

Широко распространена практика использования констант в качестве псевдонимов для трудно запоминаемых значений, которые известны до начала исполнения скрипта.

Названия таких констант пишутся с использованием заглавных букв и подчёркивания.

Например, сделаем константы для различных цветов в «шестнадцатеричном формате»:

const COLOR\_RED = "#F00";

const COLOR\_GREEN = "#0F0";

const COLOR\_BLUE = "#00F";

const COLOR\_ORANGE = "#FF7F00";

// ...когда нам нужно выбрать цвет

let color = COLOR\_ORANGE;

alert(color); // #FF7F00

Преимущества:

* COLOR\_ORANGE гораздо легче запомнить, чем "#FF7F00".
* Гораздо легче допустить ошибку при вводе "#FF7F00", чем при вводе COLOR\_ORANGE.
* При чтении кода COLOR\_ORANGE намного понятнее, чем #FF7F00.

Когда мы должны использовать для констант заглавные буквы, а когда называть их нормально? Давайте разберёмся и с этим.

Название «константа» просто означает, что значение переменной никогда не меняется. Но есть константы, которые известны до выполнения (например, шестнадцатеричное значение для красного цвета), а есть константы, которые *вычисляются* во время выполнения сценария, но не изменяются после их первоначального назначения.

Например:

const pageLoadTime = /\* время, потраченное на загрузку веб-страницы \*/;

Значение pageLoadTime неизвестно до загрузки страницы, поэтому её имя записано обычными, а не прописными буквами. Но это всё ещё константа, потому что она не изменяется после назначения.

Другими словами, константы с именами, записанными заглавными буквами, используются только как псевдонимы для «жёстко закодированных» значений.

**Придумывайте правильные имена**

В разговоре о переменных необходимо упомянуть, что есть ещё одна чрезвычайно важная вещь.

Название переменной должно иметь ясный и понятный смысл, говорить о том, какие данные в ней хранятся.

Именование переменных – это один из самых важных и сложных навыков в программировании. Быстрый взгляд на имена переменных может показать, какой код был написан новичком, а какой – опытным разработчиком.

В реальном проекте большая часть времени тратится на изменение и расширение существующей кодовой базы, а не на написание чего-то совершенно нового с нуля. Когда мы возвращаемся к коду после какого-то промежутка времени, гораздо легче найти информацию, которая хорошо размечена. Или, другими словами, когда переменные имеют хорошие имена.

Пожалуйста, потратьте время на обдумывание правильного имени переменной перед её объявлением. Делайте так, и будете вознаграждены.

Несколько хороших правил:

* Используйте легко читаемые имена, такие как userName или shoppingCart.
* Избегайте использования аббревиатур или коротких имён, таких как a, b, c, за исключением тех случаев, когда вы точно знаете, что так нужно.
* Делайте имена максимально описательными и лаконичными. Примеры плохих имён: data и value. Такие имена ничего не говорят. Их можно использовать только в том случае, если из контекста кода очевидно, какие данные хранит переменная.
* Договоритесь с вашей командой об используемых терминах. Если посетитель сайта называется «user», тогда мы должны называть связанные с ним переменные currentUser или newUser, а не, к примеру, currentVisitor или newManInTown.

Звучит просто? Действительно, это так, но на практике для создания описательных и кратких имён переменных зачастую требуется подумать. Действуйте.

Повторно использовать или создавать новую переменную?

И последняя заметка. Есть ленивые программисты, которые вместо объявления новых переменных повторно используют существующие.

В результате их переменные похожи на коробки, в которые люди бросают разные предметы, не меняя на них этикетки. Что сейчас находится внутри коробки? Кто знает? Нам необходимо подойти поближе и проверить.

Такие программисты немного экономят на объявлении переменных, но теряют в десять раз больше при отладке.

Дополнительная переменная – это добро, а не зло.

Современные JavaScript-минификаторы и браузеры оптимизируют код достаточно хорошо, поэтому он не создаёт проблем с производительностью. Использование разных переменных для разных значений может даже помочь движку оптимизировать ваш код.

**Итого**

Мы можем объявить переменные для хранения данных с помощью ключевых слов var, let или const.

* let – это современный способ объявления.
* var – это устаревший способ объявления. Обычно мы вообще не используем его, но мы рассмотрим тонкие отличия от let в главе Устаревшее ключевое слово "var" на случай, если это всё-таки вам понадобится.
* const – похоже на let, но значение переменной не может изменяться.

Переменные должны быть названы таким образом, чтобы мы могли легко понять, что у них внутри.