**Типизация в JS**

*JavaScript* является языком с *динамической слабой неявной* типизацией.

Чтобы объявить переменную в языке программирования*JavaScript*, не требуется указывать тип, он будет определен при присваивании по значению. В любую переменную можно записать любое значение, а при необходимости в выражениях переменные интерпретатором будут приведены к требуемому типу.

Стандарт*ECMAScript* определяет девять типов:

* + Шесть типов данных, являющихся примитивами:
    - * *Undefined* — неопределенный тип; неизвестное, неопределенное значение;
      * *Boolean* — булев (логический) тип данных;
      * *Number* — число;
      * *String* — строка;
      * *BigInt* — большие целые числа;
      * *Symbol;*
  + *null* — ничто;
  + *Object* — объект;
  + *Function* — функция.

**Переменные**

Переменные в *JavaScript* создаются с помощью ключевого слова let:

**let** greeting = 'Привет, мир!';

console.log(greeting);

// -> Привет, мир!

Динамическая типизация позволяет в одни и те же переменные помещать данные разных типов:

**let** anything = 'текст';

console.log(anything);

anything = **5**;

console.log(anything);

// -> текст

// -> 5

Переменная при определении до первого присваивания всегда имеет тип undefined:

**let** anything;

console.log(anything);

console.log(**typeof**(anything));

// -> undefined

// -> undefined

typeof()

typeof() — функция для определения типа значения. Её можно применять как к переменным, так и константам (в том числе именованным).

let требуется прописывать в программном коде для каждой переменной лишь *единожды*.  
В дальнейшем при присваивании, этого делать нельзя. Почему? Слово let говорит интерпретатору, что следующее слово — имя переменной. Это объявление переменной: интерпретатор запоминает, что это переменная, выделяет для нее память. А присваивание — это изменение уже связанного с именем значения, расположенного в памяти.

В этом программном коде происходит сразу две операции:

**let** number = **5**;

* + объявление переменной — let number;
  + присваивание переменной значения — number = 5;

просто они объединены для краткости.

Кроме let, в *JavaScript* может быть использовано ключевое слово var.

В работе let и var есть два существенных различия:

* + для var не существует блочной области видимости:
* Почему предпочтительнее использовать let?
* Причин достаточно большое количество, и все они связаны с ограничением области видимости. Часто (практически всегда) *JavaScript*-код приложение пишется разными разработчиками, которые абсолютно не согласуют взаимные действия. При этом программный код выполняется весь в одном потоке.
* В таком случае  важно писать код таким образом, чтобы он не повлиял на программный код других разработчиков. Если два разработчика используют одинаковое название переменной, которая будет доступна в обоих скриптах, это может привести к непредсказуемым последствиям. Поэтому требуется соблюдение некоторых условий, которые помогают ограничить области видимости. Например, рекомендуется не использовать глобальные переменные (которые видны в глобальной области видимости) и использовать для объявления переменных слово let. Про области видимости вы еще узнаете в дальнейшем курсе, сейчас же просто дадим рекомендацию:
* Ключевое слово var считается устаревшим и *не рекомендуется* к использованию.

**Именованные константы**

Именованные константы в *JS* создаются аналогично переменным, есть два отличия:

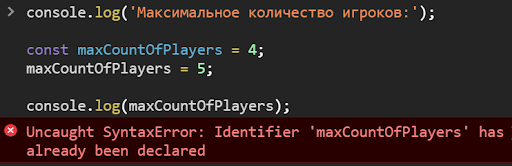
* + используется ключевое слово const;
  + при создании константы требуется сразу указать её значение.

**const** maxCountOfPlayers = **4**;

console.log(maxCountOfPlayers);

// -> 4

Технически, разница между const и let только в возможности изменить значение, связанное с именем:



В *JavaScript* можно объявить константы в любом месте программы:

console.log('Максимальное количество игроков:');

**const** maxCountOfPlayers = **4**;

console.log(maxCountOfPlayers);

// -> Максимальное количество игроков:

// -> 4

Используйте константы, а не переменные, тогда, когда не планируется, что присвоенное значение изменится:

**const** firstOperand = **2**;

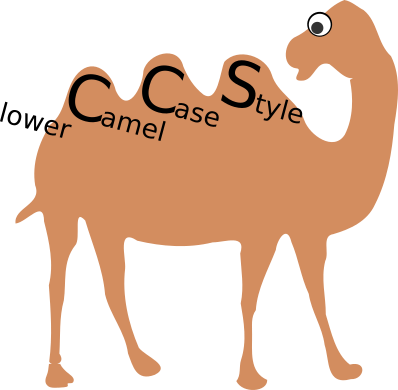
**const** secondOperand = **2**;

**const** sum = firstOperand + secondOperand;

console.log(sum);

// -> 4

**Именование переменных и констант**



*Две основные проблемы в программировании:*

*1. Именование переменных;  
2. Инвалидация кэшированных данных;  
3. Переполнение на единицу.*

*автор неизвестен*

**Ограничения языка для имён переменных:**

* 1. Регистр важен (*page, Page, PAGE* — разные переменные с точки зрения *JS*);
  2. В наименованиях переменных допустимы символы:
     + - латинские буквы (*A-Z, a-z*);
       - цифры (0-9);
       - символы $ и \_.
  3. Имя переменной не может начинаться с цифры;
  4. Имя переменной не может быть зарезервированным в языке словом (например, нельзя создать переменную с наименованием*let*).

Всё остальное (с точки зрения языка) допустимо:

**let** $ = **1**;

**let** \_ = **2**;

**let** abrabrabrabrabrabrabrabrabra = **3**;

**let** PRICE500$ = **4**;

**let** f\_4 = $ \* \_ \* abrabrabrabrabrabrabrabrabra \* PRICE500$;

console.log(f\_4);

// -> 24

Однако интерпретатору требуется лишь выполнить код. Обычно код еще требуется оптимизировать, дополнить или исправить. А это делает человек, для которого, в отличие от интерпретатора, важно значение переменной. **Поэтому переменные должны быть именованы правильно.**

Как правильно именовать переменные? Самое главное правило — наименование переменной должно быть осмысленным. Не используйте транслитерацию, именуйте переменные с помощью слов на английском языке.

Для переменных и именованных констант в*JavaScript* принято использовать составные наименования в стиле lowerCamelCase.

Стили написания именований из нескольких слов

Так как название переменной часто состоит из нескольких слов, а пробел не является знаком, который входит в наименование переменной, появилось несколько стилей написания наименований из нескольких слов:

|  |  |  |
| --- | --- | --- |
| **Название стиля** | **Описание** | **Примеры** |
| snake\_case | Разные слова соединяются с помощью символа подчеркивания вместо пробела, все слова написаны маленькими буквами. | variable\_example foo\_bar default\_app\_settings my\_class main\_page |
| kebab-case, lisp-case | Разные слова соединяются с помощью символа дефиса вместо пробела, все слова написаны маленькими буквами. | variable-example  foo-bar  default-app-settings my-class main-page |
| Train-Case | Разные слова соединяются с помощью символа дефиса вместо пробела, а каждое слово пишется с заглавной буквы. | Variable-Example  Foo-Bar  Default-App-Settings  My-Class Main-Page |
| SCREAMING\_SNAKE\_CASE | Разные слова соединяются с помощью символа подчеркивания вместо пробела, все слова написаны заглавными буквами. | VARIABLE\_EXAMPLE FOO\_BAR DEFAULT\_APP\_SETTINGS MY\_CLASS MAIN\_PAGE |
| CamelCase | Несколько слов пишутся слитно в нижнем регистре без пробелов, при этом каждое слово пишется с заглавной буквы. | VariableExample  FooBar  DefaultAppSettings  MyClass MainPage |
| lowerCamelCase | Отличие от UpperCamelCase (CamelCase) в том, что первое слово пишется в нижнем регистре полностью. | variableExample  fooBar  defaultAppSettings  myClass mainPage |

Старайтесь избегать слишком длинные наименования, например, наименование defaultSettings выигрывает перед settingsForUserWhoDidntSetTheirOwnSettings.

Что касается сокращений — используйте их тогда и только тогда, когда вы полностью уверены, что их смысл будет понятен тому кто читает код. Если сомневаетесь, что так будет, не используйте. Например, positionHor (сокращенно от *horizontal*) менее понятно, чем positionX, хоть и длиннее.

Разумеется, «удачное» наименование переменной или нет — зависит от кода, в котором она участвует. Идеально, если ваш код читается без комментариев:

**const** price = **123**;

**const** count = **4**;

**const** discount = **5**;

**const** totalCost = price \* count \* (**1** - discount / **100**);

console.log('Итоговая стоимость составит', totalCost, 'руб');