# Строгая / нестрогая типизация

Также называется сильная / слабая типизация. При строгой типизации типы назначаются «раз и навсегда», при нестрогой могут изменяться в процессе выполнения программы. В языках со строгой типизацией запрещены изменения типа данных переменной и разрешены только явные преобразования типов данных. Строгая типизация выделяется тем, что язык не позволяет смешивать в выражениях различные типы и не выполняет автоматические неявные преобразования, например нельзя вычесть из строки число. Языки со слабой типизацией выполняют множество неявных преобразований автоматически, даже если может произойти потеря точности или преобразование неоднозначно.

## Примитивные типы

1. **Number (Число):** Представляет как целые, так и вещественные числа. Пример: `let x = 5;`
2. **String (Строка):** Представляет текст. Строки заключаются в одинарные или двойные кавычки. Пример: `let str = "Hello";`
3. **Boolean (Логический):** Представляет логические значения `true` (истина) или `false` (ложь). Пример: `let isTrue = true;`
4. **Undefined (Неопределенный):** Представляет значение, которое не было присвоено. Если переменная объявлена, но ей не присвоено значение, то её тип будет `undefined`. Пример: `let x;`
5. **Null (Пусто):** Представляет отсутствие значения или ничего. Пример: `let y = null;`

6. **Symbol (Символ):** Введен в ECMAScript 6 (ES6). Каждый символ уникален и используется, например, для создания уникальных свойств объектов. Пример: `let sym = Symbol('description');`

7. **BigInt (Большое Число):** Введен в ECMAScript 2020 (ES11). Представляет целые числа произвольной длины. Заключается в числовом литерале с суффиксом `n`. Пример: `let bigIntValue = 123n;`

Таким образом, в JavaScript есть семь примитивных типов данных: Number, String, Boolean, Undefined, Null, Symbol и BigInt. Каждый из них обладает своими особыми характеристиками и используется для представления различных видов данных.

Примитивные типы являются неизменяемыми, что означает, что значения этих типов не могут быть изменены напрямую. Все операции с примитивами возвращают новое значение, а не изменяют существующее.

## Операторы (Operators):

### Арифметические операторы:

- `+` : Сложение.
- `-` : Вычитание.
- `*` : Умножение.
- `/` : Деление.
- `**` : Возведение в степень.

Пример:

```
let result = 10 + 5; // 15
let product = 3 * 4; // 12
let power = 2 ** 3; // 8
```

### Оператор `typeof`:

Оператор `typeof` используется для определения типа значения.

```
typeof 42;         // "number"
typeof "Hello";    // "string"
typeof true;       // "boolean"
typeof undefined;  // "undefined"
```

# Операторы сравнения (Comparison Operators):

## Стандартные операторы сравнения:

- `==` : Не строгое равенство (производит приведение типов).
- `===` : Строгое равенство (без приведения типов).
- `!=` : Не строгое неравенство.
- `!==` : Строгое неравенство.

## Другие операторы сравнения:

- `<` : Меньше.
- `>` : Больше.
- `<=` : Меньше или равно.
- `>=` : Больше или равно.

Примеры:

```
let x = 5;
let y = "5";


x == y;  // true (произойдет приведение типов)
x === y; // false (строгое сравнение без приведения типов)
x !== y; // true (строгое неравенство)
```

# Приведение типов (Coercion):

Приведение типов происходит, когда JavaScript автоматически изменяет тип значения в определенном контексте.

```
let num = 42;
let str = "The answer is " + num; // Приведение num к строке
```

# Условные операторы (If-Else):

```
let condition = true;


if (condition) {
  // Блок кода выполняется, если условие истинно
} else {
```

```
  // Блок кода выполняется, если условие ложно
}
```

## Оператор Switch Case:

```
let day = "Monday";


switch (day) {
  case "Monday":
    console.log("It's Monday!");
    break;
  case "Tuesday":
    console.log("It's Tuesday!");
    break;
  // ...
  default:
    console.log("It's another day!");
}
```

Оператор `switch` используется для выбора одного из множества блоков кода для выполнения, основываясь на значении переменной.

Эти концепции представляют основные элементы операторов, сравнения и условий в JavaScript.