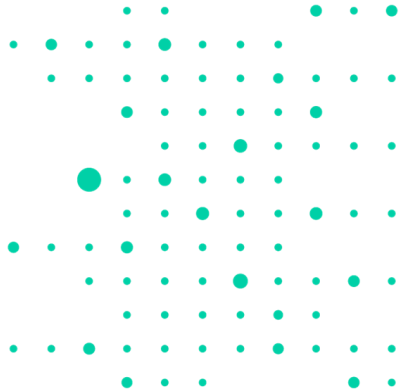




## Front-end overview. Javascript operators & Loops



# Операторы сравнения

---

**==** - не строгое сравнение

**===** - строгое сравнение

**!=** - нестрогое не равно

**!==** - строгое не равно

# Логические операторы

---

**|| (ИЛИ)**

**&& (И)**

**! (НЕ)**

# Операторы ветвления

---

**Инструкция «if»**

**Условный (тернарный) оператор**

**Конструкция "switch"**

# Инструкция «if»

---

## Конструкция

```
if (year == 2015) {  
    alert( 'Правильный ответ!' );  
} else {  
    alert( 'А вот и неправильно!' ); // любое значение  
}
```

# Инструкция «if»

---

## Конструкция

```
if (year == 2015) {  
    alert( 'Это слишком рано...' );  
} else if (year > 2015) {  
    alert( 'Это поздновато' );  
} else {  
    alert( 'Верно!' );  
}
```

# Условный (тернарный) оператор

---

Синтаксис:

```
let result = условие ? значение1 : значение2;
```

# Тернарный оператор

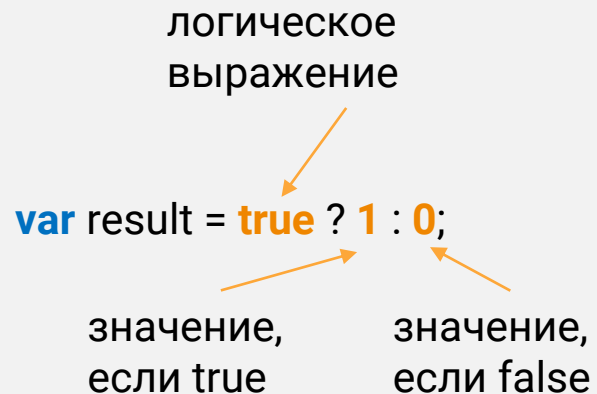
---

логическое  
выражение

`var result = true ? 1 : 0;`

значение,  
если true

значение,  
если false



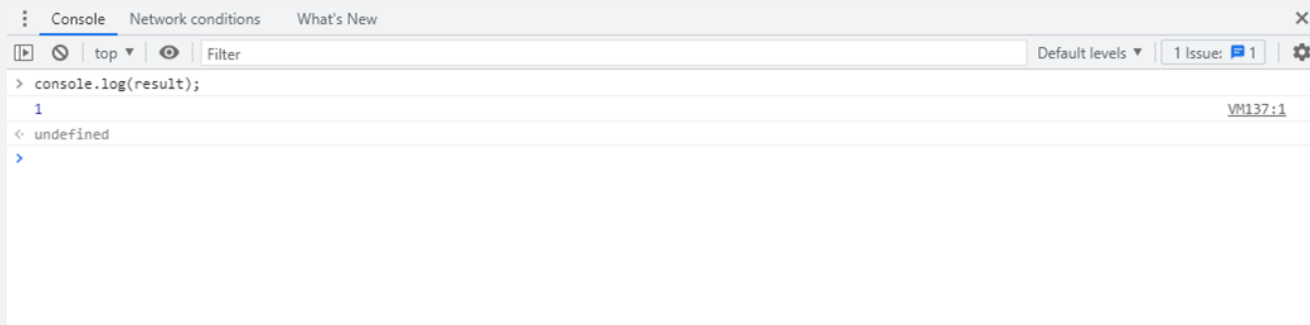
The diagram illustrates the components of a ternary operator in a programming language. It shows the code snippet `var result = true ? 1 : 0;`. Three orange arrows point from descriptive text labels to parts of the code: one from 'логическое выражение' (logical expression) to the word 'true', one from 'значение, если true' (value if true) to the number '1', and one from 'значение, если false' (value if false) to the number '0'.



# Тернарный оператор

---

```
var result = true ? 1 : 0;  
console.log(result);
```

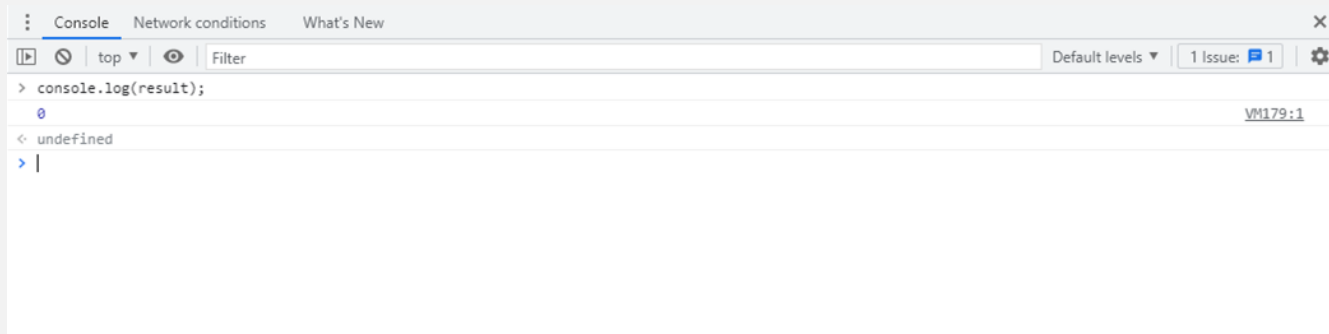


# Тернарный оператор

---

```
var result = false ? 1 : 0;
```

```
console.log(result);
```



# Условный (тернарный) оператор

---

```
let accessAllowed;  
let age = prompt('Сколько вам лет?', '');  
if (age > 18) {  
    accessAllowed = true;  
} else {  
    accessAllowed = false;  
}
```

```
let accessAllowed = (age > 18) ? true : false;
```

# Конструкция "switch"

---

```
let a = 2 + 2;
```

```
switch (a) {  
  case 3:  
    alert( 'Маловато' );  
    break;  
  case 4:  
    alert( 'В точку!' );  
    break;  
  case 5:  
    alert( 'Перебор' );  
    break;  
  default:  
    alert( "Нет таких значений" );  
}
```

# Цикл «while»

---

Конструкция

```
while (condition) {
```

```
    // код
```

```
    // также называемый "телом цикла"
```

```
}
```

# Цикл «while»

---

Конструкция

```
do {  
    // тело цикла  
} while (condition);
```

# Цикл «for»

---

Конструкция

```
for (let i = 0; i < 100; i++) {  
    // тело цикла  
}
```

# Цикл «for»

---

Конструкция

```
for (let key in user) {  
    // тело цикла  
}
```



# Цикл «for»

---

Конструкция

```
for (let elem of numbers) {  
    // тело цикла  
}
```