

1. Сколько есть видов условных операторов?

В лекции мы рассмотрели 5 видов:

- тернарный оператор (?:)
- условный оператор if
- условный оператор if...else
- условный оператор else if...
- оператор выбора switch

2. Перепишите пример про определение времени суток через тернарный оператор

```
let date = new Date();
let time = date.getHours();

if (time < 10) {
  alert( "Доброе утро!" );
} else {
  alert( "Добрый день!" );
}
```

Запись через тернарный оператор:

```
let date = new Date();
let time = date.getHours();
(time < 10) ? alert( "Доброе утро!" ) : alert( "Добрый день!" );
```

3. Выведется ли alert?

```
if ("0") {
  alert( 'Привет' );
}
```

Да, JavaScript приведёт выражение в условии ("0") к true (поскольку "0" является строкой, а не числом. Если записать так: (0), то условие будет false, alert не выведется).

4. Чему будет равно условие (правда или ложь) в этих случаях, если $x = 6$ and $y = 3$?

- $(x < 10 \ \&\& \ y > 1)$ *логическое И* true – то есть оба выражения истинны
- $(x == 5 \ || \ y == 5)$ *логическое ИЛИ* false – поскольку оба условия false, если бы хотя бы одно было true, вернулось бы true (например, если бы $x = 5$)
- $!(x == y)$ *логическое НЕ* true – все наоборот. Условие ложное, но за счет ! оно превращается в true.

5. Назовите три способа назначения обработчиков событий. Какой из них самый универсальный?

- 1) Назначить обработчик прямо в коде html-документа, как атрибут тега, к которому назначается событие (onclick к <button>).
- 2) Поскольку большинство событий связаны с DOM-элементами, можно назначить обработчик используя DOM-элемент, в коде скрипта: `element.on<событие>`.
- 3) метод `addEventListener` – самый универсальный, поскольку не имеет ограничений по количеству событий, как два предыдущих. Метод вызывается у DOM-элемента. Аргументами нужно передать тип события (справочная информация) и функцию, которую нужно выполнить.

6. Корректна ли такая запись? `button.onclick = hello();`

Нет, поскольку назначить событие нужно без круглых скобок (). Запись со скобками будет вызовом функции, результат выполнения которой будет присвоен к onclick.

7. Какие есть события у клавиатуры?

Когда пользователь нажимает на клавишу клавиатуры, происходит событие `keydown`, как только пользователь отпустил клавишу — произойдёт событие `keyup`.

В функцию-обработчик также передаётся объект события, в котором есть информация о нажатой кнопке:



`key` — символьное представление нажатой клавиши.

`code` — название клавиши (в контексте ее физического расположения на клавиатуре: `KeyS`, `KeyZ`, `KeyU` и т.д.

`altKey` — `true` или `false`, была ли одновременно нажата/отпущена клавиша `Alt`

`ctrlKey` — `true` или `false`, была ли одновременно нажата/отпущена клавиша `Ctrl`

`shiftKey` — `true` или `false`, была ли одновременно нажата/отпущена клавиша `Shift`

`metaKey` — `true` или `false`, была ли одновременно нажата/отпущена так называемая мета-клавиша (на Mac клавиатурах это  `Command`, а в Windows клавиатурах — `Windows key` 

8. Что такое объект события и какие у него могут быть свойства?

Объект события (event) описывает событие, произошедшее на странице — это JavaScript-объект с информацией о событии. Объект создает браузер, записывая детали произошедшего события. То есть: при наступлении события вызывается функция, и браузер при вызове передает в обработчик объект события с помощью аргумента функции.

В объекте события есть как общие свойства (тип события, время события), так и свойства, которые зависят от типа события (например, на какую кнопку нажал пользователь). Примеры свойств событий:

`bubbles` — является ли данное событие всплывающим.

`cancelable` — является ли событие отменяемым.

`currentTarget` — указывает на элемент, на котором установлен обработчик события.

`defaultPrevented` — отменено ли поведение события по умолчанию.

`eventPhase` — указывает на фазу срабатывания события.

`target` — ссылка на объект, которым было инициировано событие. Например, если событие произошло на поле ввода, мы получим ссылку на этот DOM элемент.

`timestamp` — время возникновения события в миллисекундах.

`type` — тип события.

9. Самостоятельно разберитесь, какие бывают операторы сравнения? Напишите сюда как выглядят сравнение "равно", "не равно", "больше чем".

Операторы сравнения выдают логический результат (boolean). Если условие проверки истинно, оператор выдает `true`, а если ложно - `false`.

JavaScript предоставляет три оператора сравнения величин:

- равенство ("двойное равно") `==`,
- строгое равенство (или "тройное равно" или "идентично") `===`,
- больше чем `>`
- и [Object.is](#) (новшество из ECMAScript 6).

Сравнение бывает строгим и нестрогим. При строгом сравнении (===) интерпретатор учитывает типы сравниваемых значений. Когда сравниваем значения нестрого между собой с помощью ==, JavaScript приводит типы самостоятельно: интерпретатор пробует привести типы к одному, чтобы сравнить.

```
5 == "5" // true
```

```
5 === "5" // false
```

Кроме этого, есть другие операторы сравнения: < (меньше чем), <= (меньше чем или равно), >= (больше чем или равно).