SOBESEDOVANIE#1

Самые нужные.

created by: webDev(канал на YouTube).

written by: s2trv(Aziz Sattorov).

.№1.

#1

Question:

Answer:

DOCTYPE - используется для указания типа документа. Добавляется он первой строкой любого HTML или XHTML документа. Служит для того, чтобы браузер мог понять как ему интерпритировать страницу.

#2

Question:

Базовая структура HTML-страницы?

Answer:

```
<!-- doctype -->
<!DOCTYPE html>
<!-- html -->
<html lang="en">
  <!-- head -->
<head>
  <meta charset="UTF-8">
  <title>Document</title>
</head>
  <!-- body -->
<body>
  <h1>Hello world!</h1>
</body>
</html>
```

На самом верху обязательно идёт - doctype.

За ним следует - Основной Тег HTML.

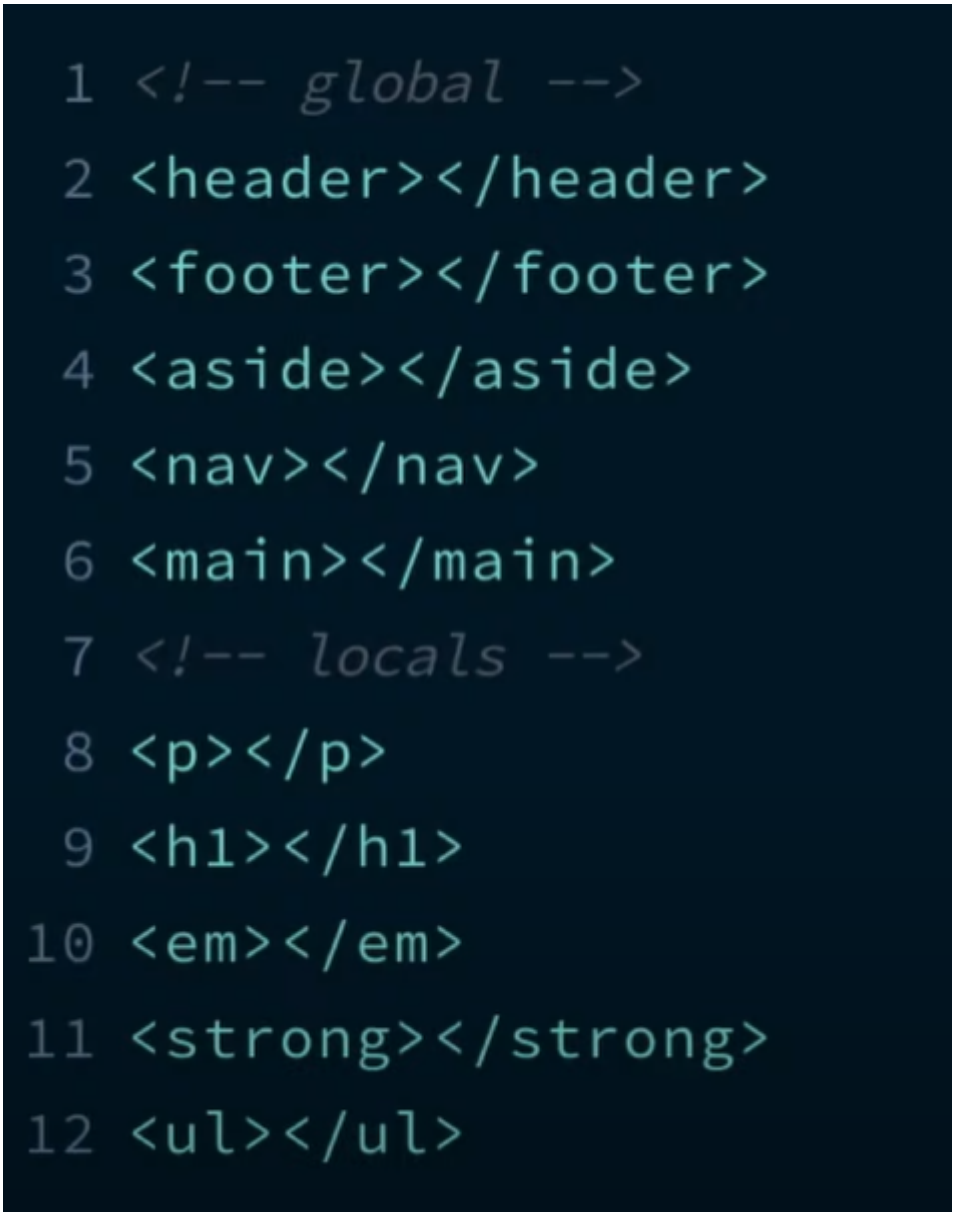
Внутри HTML - идут 2 основных блока HEAD и BODY.

#3

Question:

Что такое семантика? Какие семантичные тэги вы знаете?

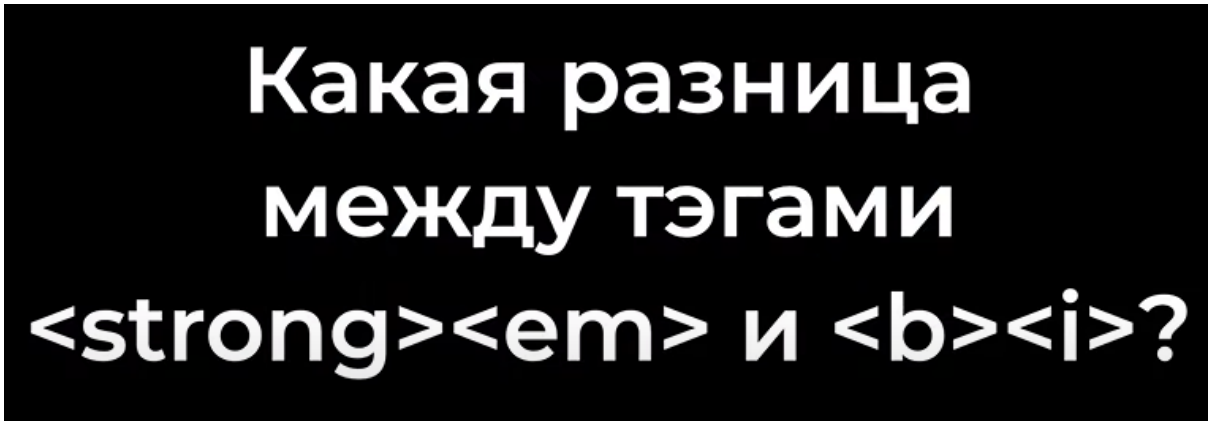
Answer:



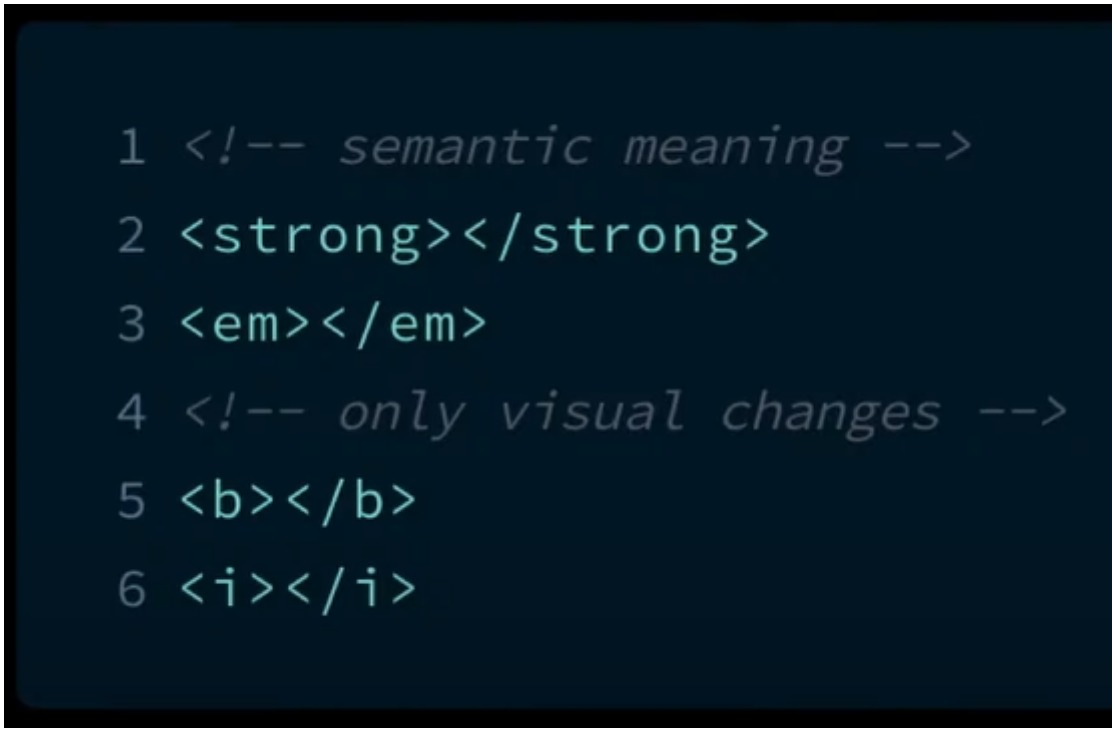
Семантика - это использование правильных тегов, описывающих содержимых контекста

#4

Question:



Answer:



Если смотреть результат в браузере, то тэг - strong и b делает строку жирным.

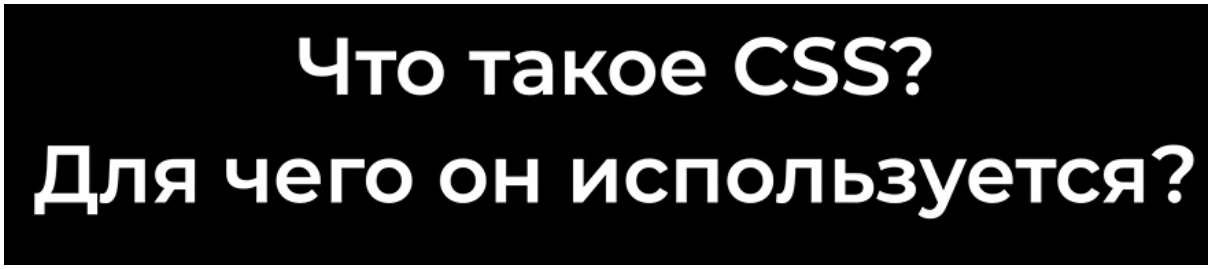
А тэги - em и i делает его курсивным.

Но, однако - strong и em предназначаны для обернутых элементов + при Screen reader’е они выделяются.

A - b и i просто изменяют визуальный вид элемента и не выделяются при screen reader’e.

#5

Question:



Answer:

Рассшифровается как - Cascading Style Sheets.

Предназначаны они для добавления стилей на HTML страницу.

#6

Question:

Answer:

```
/* inline styles */
<div style="background-color:red;"></div>

/* globa styles */
<head>
  <style>
    div { background-color: red; }
  </style>
</head>

/* external file */
<head>
  <link rel="stylesheet" href="css/styles.css" />
</head>

/* import inside CSS */
@import "style/media.css";
@import "style/footer.css";
```

Существует 4 основных способа:

(показано выше)

#7

Question:

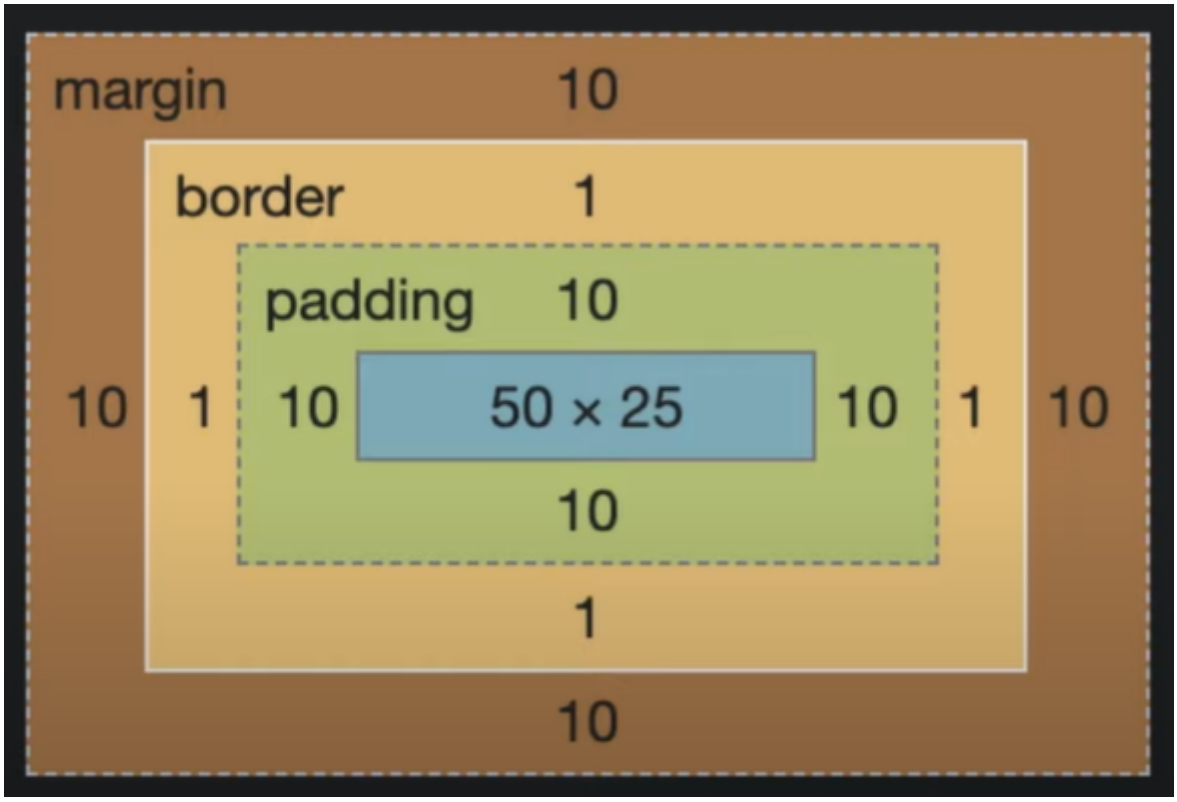
Answer:

Static	Дефолтный тип позиционирования
Relative	Элемент позиционируется относительно своего текущего положения
Absolute	Позиционирование относительно другого элемента у которого позиционирование не static. Если такого нет, то относительно окна браузера
Fixed	Позиционирование только относительно окна браузера
Sticky	В видимой области экрана элемент ведёт себя, как fixed. При дальнейшей прокрутке, скроллится вместе с родителем

#8

Question:

Answer:



Блочная модель позволяет рассчитать какое итоговое пространство будет занимать элемент на странице.

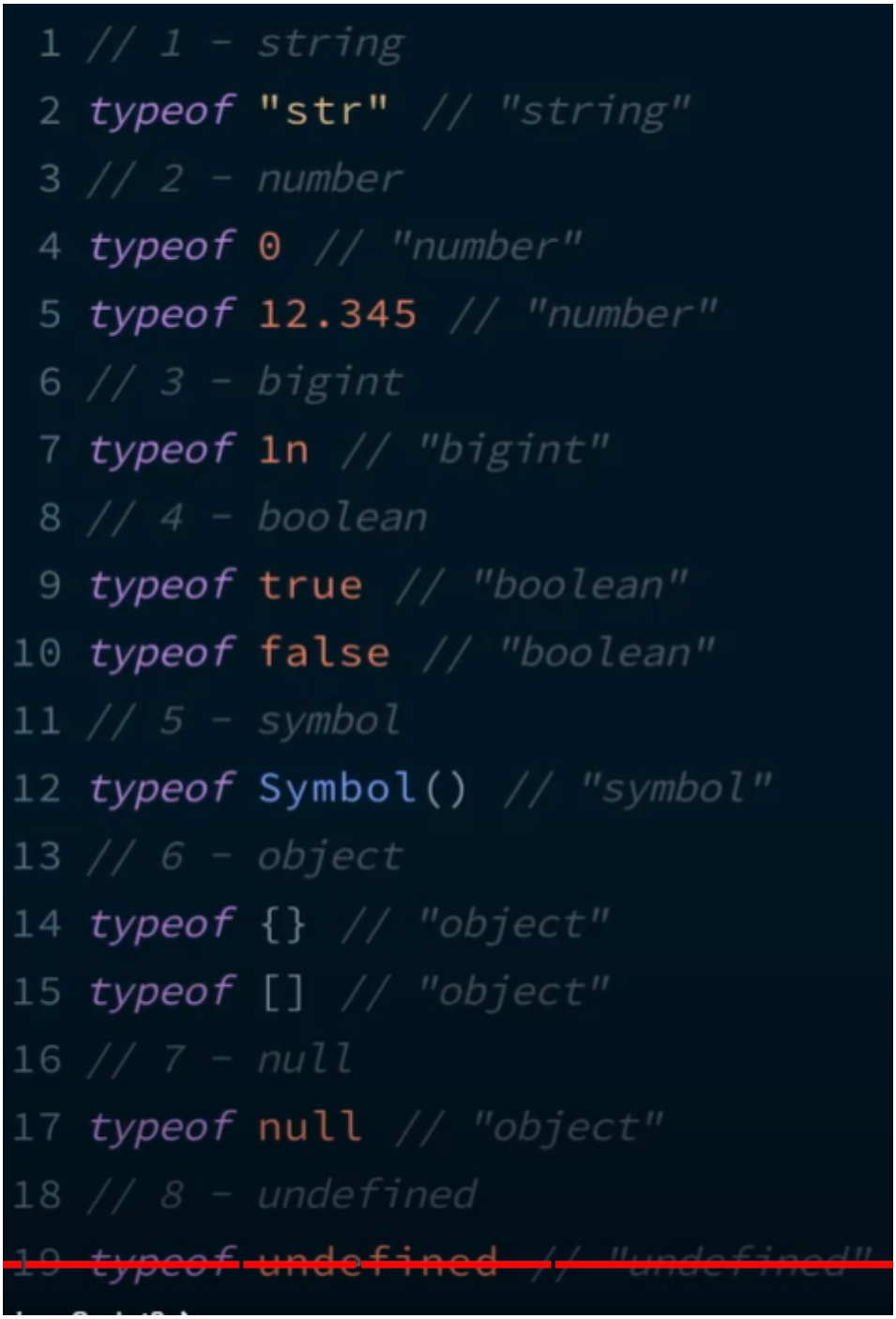
В неё входит:

- сам контент;
- padding;
- border;
- margin.

#9

Question:

Answer:



На данный момент существует 8 основных типов данных.

Это:

- string - строка;
- number - цифра;
- bigint - (индикатор bigint,это n значение в конце числа);
- symbol - символы;

- object - объекты;
- null - нал;
- undefined - андэфайнд.

Question:

Answer:

```
1 1 == '1' // true
2 1 === '1' // false
```

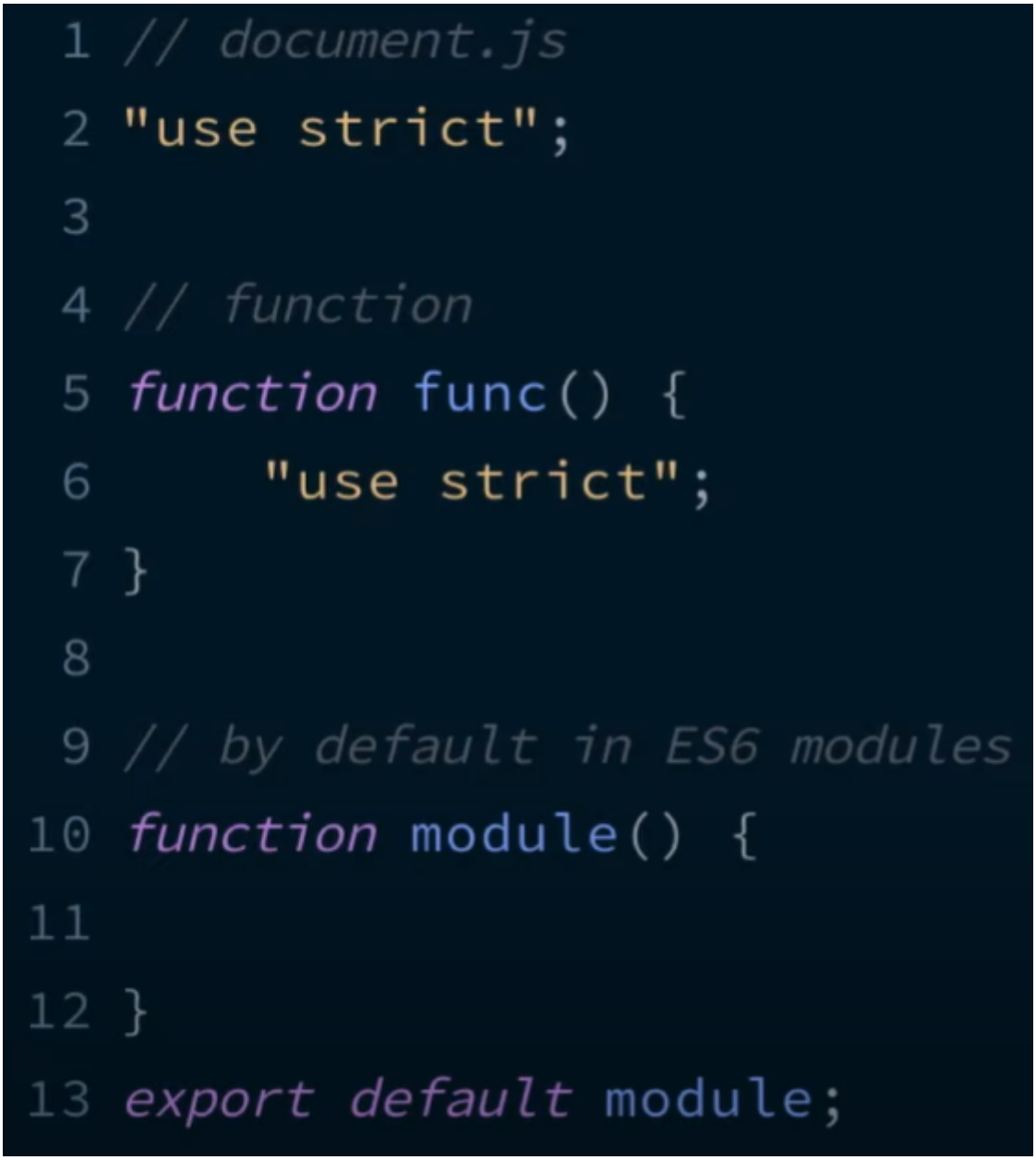
Не строгое равенство (==) - сравнивает просто их значение.

Строгое равенство (===) - дополнительно сравнивает их типы

#11

Question:

Answer:



```
// document.js
"use strict";

// function
function func() {
    "use strict";
}

// by default in ES6 modules
function module() {

}
export default module;
```

Strict mode - позволяет использовать более строгий вариант JS.

Он заменяет исключениями некоторые ошибки которые JS интерпритатор пропускает по умолчанию.

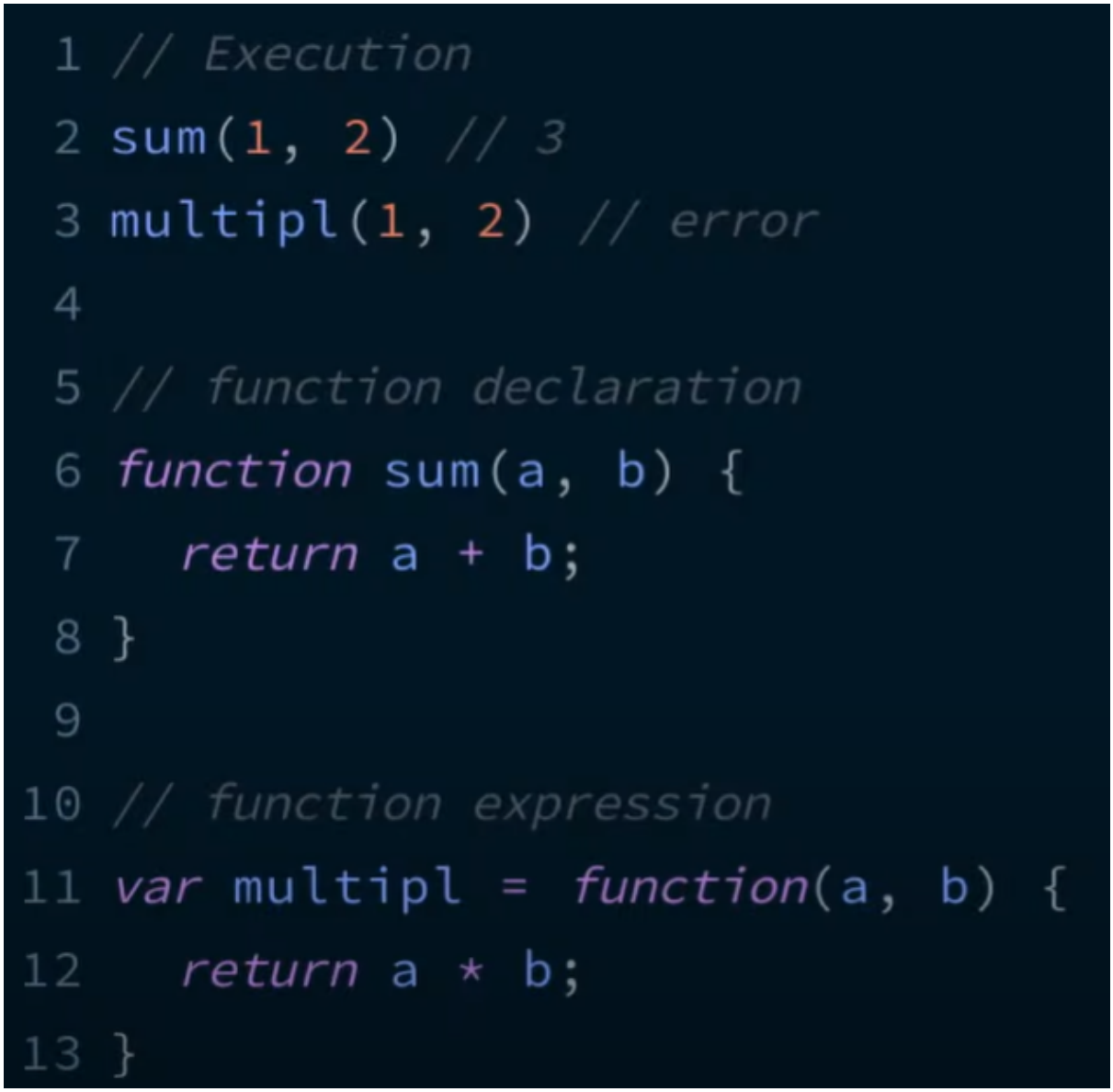
По этому в большинстве случаев, такой режим всегда старается включать.

#12

Question:

Function declaration vs. Function expression?

Answer:



```
// Execution
sum(1, 2) // 3
multipl(1, 2) // error

// function declaration
function sum(a, b) {
  return a + b;
}

// function expression
var multipl = function(a, b) {
  return a * b;
}
```

Function declaration - это функция,созданная в основном в потоке документа.

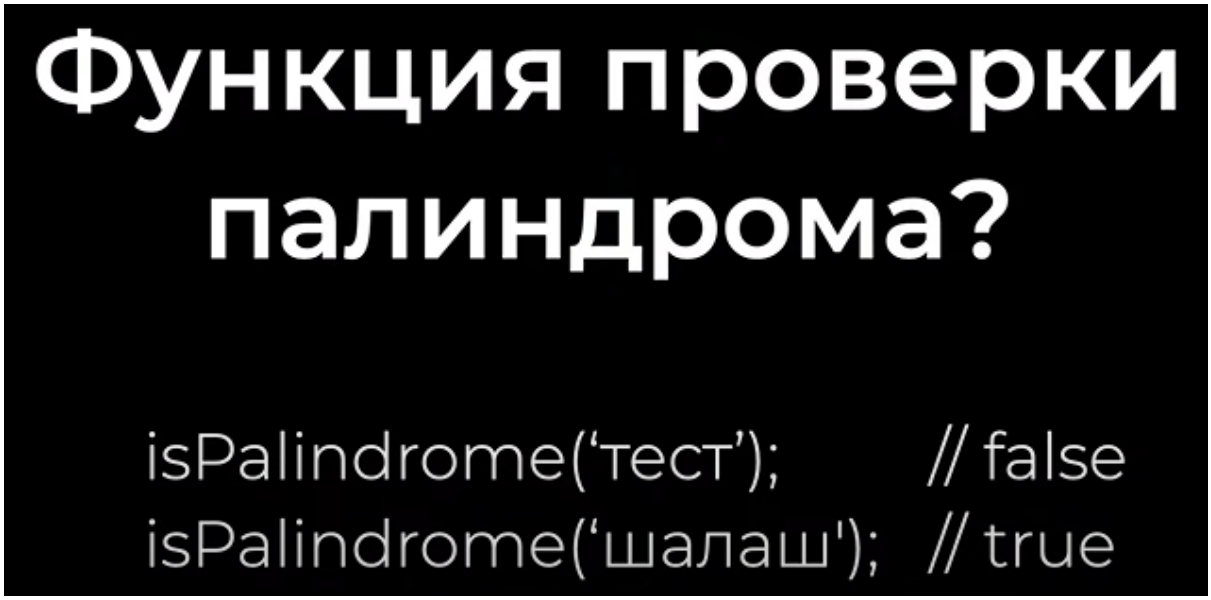
Function expression - это когда функция присваивается в переменную.

Разница:

— func.declaration создается интерпритатором до выполнения кода (всё благодаря hoisting’y).

#13

Question:



Answer:

Палиндром - это слово которое читается, слева направо и справа налева одинакого.

```
// Base
function isPalindrome(string) {
    var arr = string.split('');
    var reversArr = arr.reverse();
    var resString = reversArr.join('');
    var res = string === resString;
    return res;
}

// Advanced
function isPalindrome(str) {
    return str === str.split('').reverse().join('');
}

// ES6
const isPalindrome = str =>
    str === str.split('').reverse().join('');
```