

SOBESEDOVANIE#1

Самые нужные.

created by: webDev(канал на YouTube).

written by: s2trv(Aziz Sattorov).

.№1.

#1

Question:

Что такое doctype?
Для чего он используется?

Answer:

```
1 <!-- doctype for HTML5 -->
2 <!DOCTYPE html>
```

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:

```
1 <!-- doctype -->
2 <!DOCTYPE html>
3 <!-- html -->
4 <html lang="en">
5   <!-- head -->
6   <head>
7     <meta charset="UTF-8">
8     <title>Document</title>
9   </head>
10  <!-- body -->
11  <body>
12    <h1>Hello world!</h1>
13  </body>
14 </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 семантические ТЭГИ ВЫ ЗНАЕТЕ?

Answer:

```
1 <!-- global -->
2 <header></header>
3 <footer></footer>
4 <aside></aside>
5 <nav></nav>
6 <main></main>
7 <!-- locals -->
8 <p></p>
9 <h1></h1>
10 <em></em>
11 <strong></strong>
12 <ul></ul>
```

Семантика - это использование правильных тегов, описывающих содержимых контекста

#4

Question:

Какая разница
между тэгами
 и <i>?

Answer:

```
1 <!-- semantic meaning -->
2 <strong></strong>
3 <em></em>
4 <!-- only visual changes -->
5 <b></b>
6 <i></i>
```

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 предназначены 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:

**Что такое CSS?
Для чего он используется?**

Answer:

Расшифровывается как - Cascading Style Sheets.

Предназначены они для добавления стилей на HTML страницу.

#6

Question:

**Варианты добавление
CSS стилей на страницу?**

Answer:

```
1  /* inline styles */
2  <div style="background-color:red;"></div>
3
4  /* globa styles */
5  <head>
6    <style>
7      div { background-color: red; }
8    </style>
9  </head>
10
11 /* external file */
12 <head>
13   <link rel="stylesheet" href="css/styles.css" />
14 </head>
15
16 /* import inside CSS */
17 @import "style/media.css";
18 @import "style/footer.css";
```

Существует 4 основных способа:

(показано выше)

#7

Question:

ТИПЫ ПОЗИЦИОНИРОВАНИЯ В CSS?

Answer:

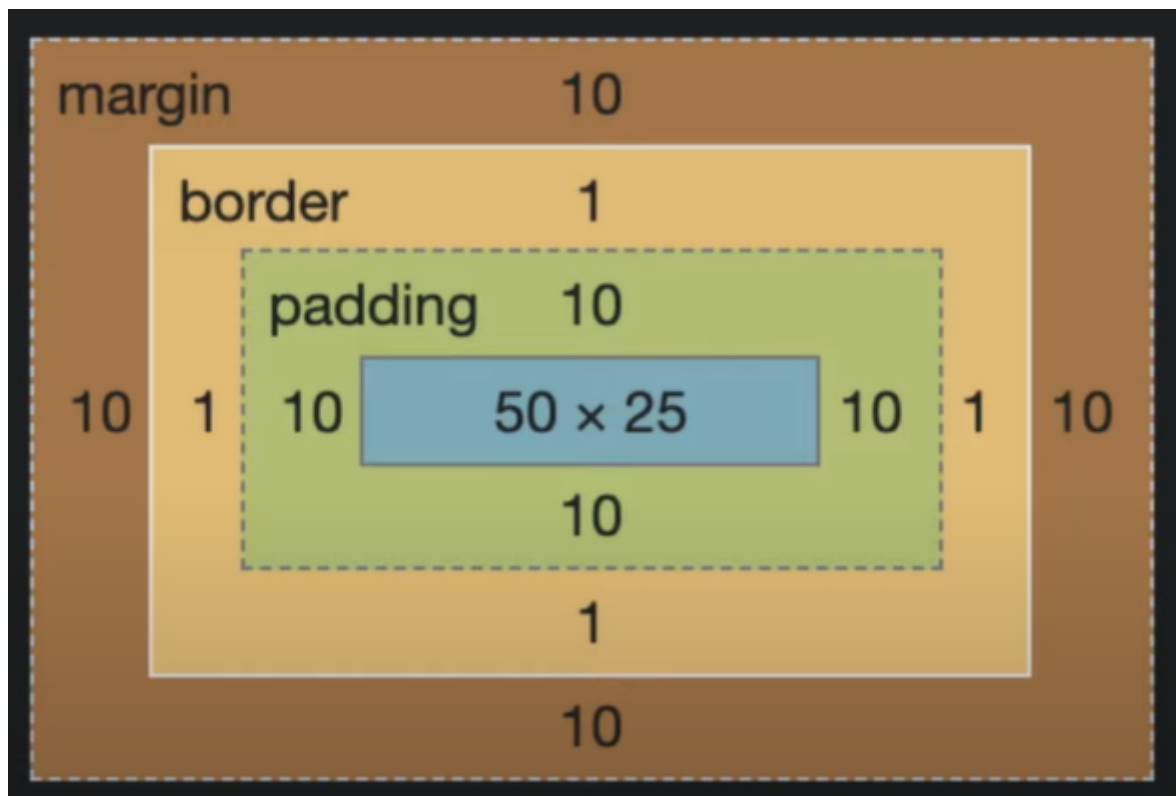
Static	Дефолтный тип позиционирования
Relative	Элемент позиционируется относительно своего текущего положения
Absolute	Позиционирование относительно другого элемента у которого позиционирование не static. Если такого нет, то относительно окна браузера
Fixed	Позиционирование только относительно окна браузера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

#8

Question:

Блочная модель CSS?

Answer:



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

В неё входит:

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

#9

Question:

Типы данных в JavaScript?

Answer:


```
1 // 1 - string
2 typeof "str" // "string"
3 // 2 - number
4 typeof 0 // "number"
5 typeof 12.345 // "number"
6 // 3 - bigint
7 typeof 1n // "bigint"
8 // 4 - boolean
9 typeof true // "boolean"
10 typeof false // "boolean"
11 // 5 - symbol
12 typeof Symbol() // "symbol"
13 // 6 - object
14 typeof {} // "object"
15 typeof [] // "object"
16 // 7 - null
17 typeof null // "object"
18 // 8 - undefined
19 typeof undefined // "undefined"
```

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:

Разница между == и ===?

Answer:

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

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

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

#11

Question:

Что такое Strict mode в JavaScript?

Answer:

```
1 // document.js
2 "use strict";
3
4 // function
5 function func() {
6     "use strict";
7 }
8
9 // by default in ES6 modules
10 function module() {
11
12 }
13 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:

```
1 // Execution
2 sum(1, 2) // 3
3 multipl(1, 2) // error
4
5 // function declaration
6 function sum(a, b) {
7     return a + b;
8 }
9
10 // function expression
11 var multipl = function(a, b) {
12     return a * b;
13 }
```

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'у).

#13

Question:

Функция проверки палиндрома?

```
isPalindrome('тест');    // false  
isPalindrome('шалаш');  // true
```

Answer:

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

```
1 // Base
2 function isPalindrome(string) {
3     var arr = string.split('');
4     var reversArr = arr.reverse();
5     var resString = reversArr.join('');
6     var res = string === resString;
7     return res;
8 }
9
10 // Advanced
11 function isPalindrome(str) {
12     return str === str.split('').reverse().join('');
13 }
14
15 // ES6
16 const isPalindrome = str =>
17     str === str.split('').reverse().join('');
```