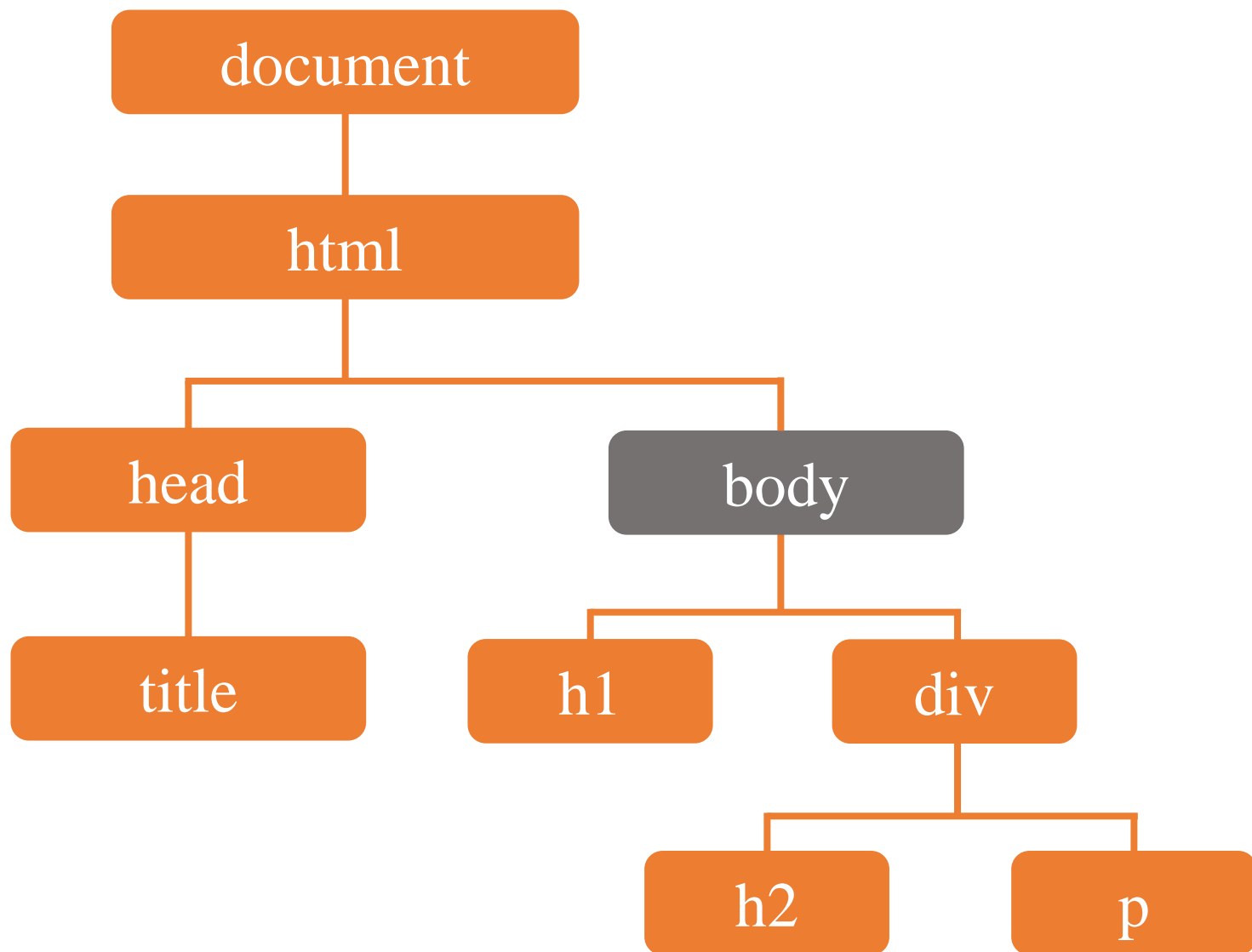


Работа с DOM-деревом страницы

10-11 класс



Что такое DOM-дерево?





Создание, добавление и удаление узлов

document.**querySelector**(...)

аналог **.getElementById**, выбор элемента по id

document.**createElement**(...)

создание узла с определенным тегом

anyTag.**appendChild**(...)

добавление узла в родительский узел

document.**createTextNode**(...)

создание текстового узла

anyTag.**removeChild**(...)

удаление дочернего узла

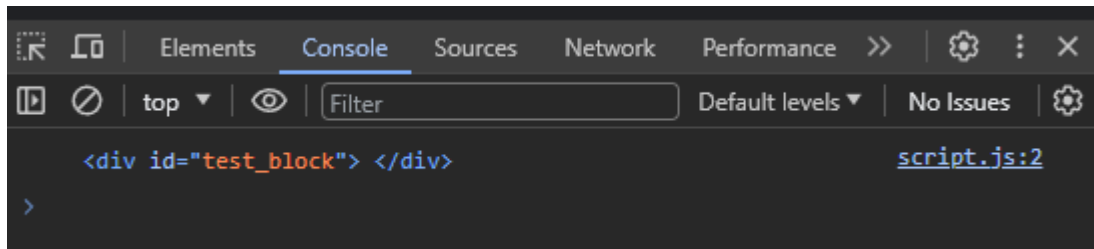


Примеры использования

```
13 <body>
14   <div id="test_block">
15
16   </div>
17 </body>
18
```

Сохранение узла в
переменную для
дальнейших действий

```
JS script.js > ...
1 let block = document.querySelector("#test_block");
2 console.log(block);
```





Примеры использования

JS script.js > ...

```
1 let block = document.querySelector("#test_block");
2 console.log(block);
3
4
5 let newElem = document.createElement("div");
6
7
8 newElem.innerHTML = "Новый блок";
9
10
11 newElem.setAttribute("id", "new_block");
12
13
14 block.appendChild(newElem);
15
```

Выбор узла по id

Создание блока div

Вставка текста в блок

Установка значения атрибуту

Добавление дочернего узла



Примеры использования

JS script.js > ...

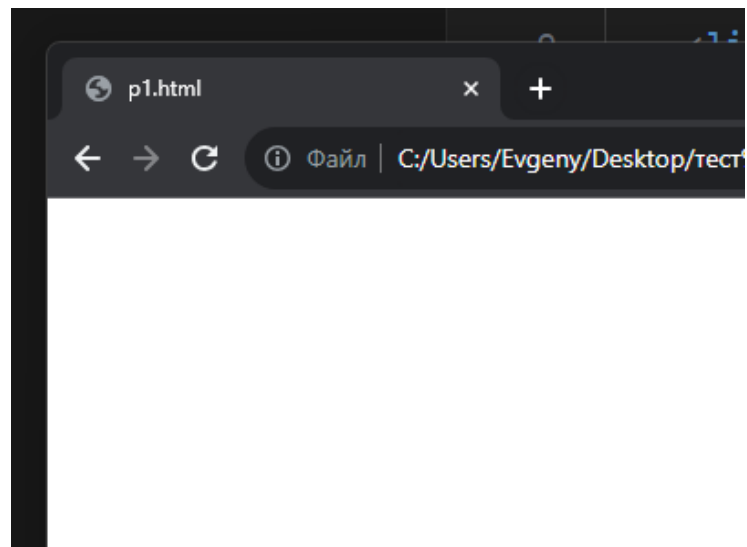
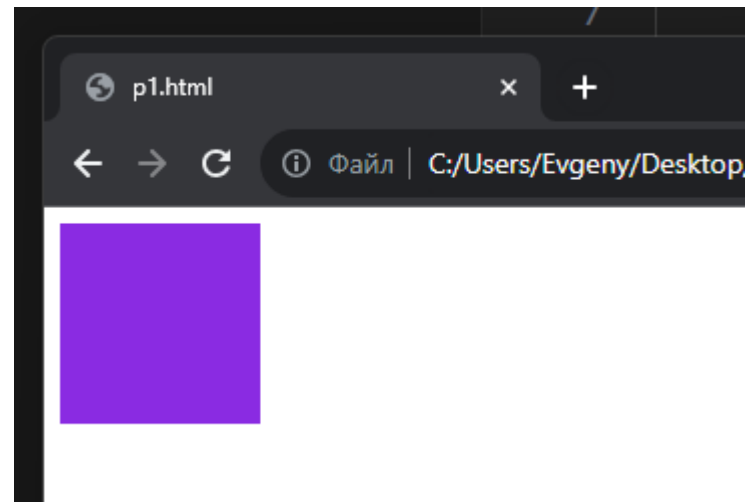
```
1 let block = document.querySelector("#test_block");
2 console.log(block);
3
4
5 let newElem = document.createElement("div");
6
7
8 newElem.innerHTML = "Новый блок";
9
10
11 newElem.setAttribute("id", "new_block");
12
13
14 block.appendChild(newElem);
15
```

```
<!DOCTYPE html>
<html lang="eng">
  <head> ... </head>
  <body>
    ... <div id="test_block"> == $0
      <div id="new_block">Новый блок</div>
    </div>
  </body>
</html>
```



Примеры использования

```
p1.html style.css script.js
JS script.js > ...
1 let block = document.querySelector("#test_block");
2 console.log(block);
3
4 block.addEventListener("click", () => {
5     block.parentNode.removeChild(block);
6 })
7
8
9
```





Документация



developer.mozilla.org/ru/docs/Web/API/Node

Node

`Node` это интерфейс, от которого наследуют несколько типов DOM, он так же позволяет различным типам быть обработанными(или протестированными).

Следующие интерфейсы полностью наследуют от `Node` его методы и свойства: `Document`, `Element`, `CharacterData` (which `Text`, `Comment`, и `CDataSection` inherit), `ProcessingInstruction` (en-US), `DocumentFragment`, `DocumentType` (en-US), `Notation`, `Entity`, `EntityReference`

Эти интерфейсы могут возвращать null в особых случаях, когда методы и свойства не уместны. Они могут сбросить исключение - например, когда добавляются дети к типу узла, у которого не может их существовать.

Свойства

Наследует свойства от родителей `EventTarget` .[1]

`Node.baseURI` Только для чтения

Возвращает `DOMString` показывающие основной URL. Понятие основного URL изменяется из одного языка в другой; В HTML, это соответствует протоколу, доменному имени и структуре каталогов, все до последнего `'/'`.

`Node.baseURIObject` (en-US) ⚠

(Не доступно для веб-контента.) Только для чтения. Объект `nsIURI`, представляющий базовый URI элемента.

Задание 1. Создайте активный список покупок в который можно добавлять новые элементы и удалять существующие:

```
<h1>Покупочки ^_^</h1>
<div id="container">
  <div id="add_item_menu">
    <div class="add_item_menu_text">

    </div>
    <input id="add_item_name" type="text">
    <button id="add_item_btn">Добавить</button>
  </div>
  <div id="item_list">


  </div>
</div>
```


Покупочки ^_^

Новый элемент:

Добавить

style.css > ...

```
1  body {
2      width: 100%;
3      height: 100%;
4      font-family: "Roboto";
5  }
6
7  #container {
8      padding: 10px;
9      display: grid;
10     grid-template-rows: 1fr 3fr;
11     width: 500px;
12     height: 500px;
13     background-color:  #c0c0c0;
14 }
15
```

```
16 .add_item_menu_text {
17     font-size: large;
18     font-weight: 500;
19 }
20
21 #add_item_menu {
22     display: flex;
23     flex-direction: row;
24     align-items: center;
25     justify-content: center;
26     gap: 30px;
27 }
28
29 #item_list {
30     width: 500px;
31     height: 400px;
32     background-color:  #8f8f8f;
33 }
```

Подсказки к заданию 1:

```
29 #item_list {
30     box-sizing: border-box;
31     padding: 10px 20px;
32     width: 500px;
33     height: 400px;
34     background-color: #8f8f8f;
35 }
36
37 .item {
38     background-color: white;
39     padding-left: 50px;
40     display: flex;
41     flex-direction: row;
42     align-items: center;
43     justify-content: left;
44     gap: 30px;
45 }
```

```
<div id="item_list">
  <div class="item">
    <p class="item_text">Молоко</p>
    <button id="item_btn">Удалить</button>
  </div>
</div>
```

Настройте элемент списка как он будет
выглядеть. CSS оставляем, html
удаляем.

Подсказки к заданию 1:

JS script.js > add_btn.addEventListener("click") callback

```
1 let add_btn = document.querySelector("#add_item_btn");
2
3 add_btn.addEventListener("click", () => {
4     let new_text = document.querySelector("#add_item_name").value; // сохраняем текст
5     document.querySelector("#add_item_name").value = ""; // очищаем поле
6
7     let item_list = document.querySelector("#item_list"); // сохраняем узел списка
8
9     let new_item = document.createElement("div"); // создаем запись в списке
10    new_item.setAttribute("class", "item"); // присваиваем записи класс
11
12    let item_text = document.createElement("p"); // создаем текст записи
13    item_text.setAttribute("class", "item_text"); // присваиваем тексту класс
14    console.log(new_text);
15    item_text.innerHTML = new_text; // кладем текст из поля ввода внутрь узла
16
17    let item_btn = document.createElement("button"); // создаем кнопку удаления
18    item_btn.setAttribute("id", "item_btn"); // присваиваем кнопке id
19    item_btn.innerHTML = "Удалить"; // добавляем текст кнопке
20
21
22    new_item.appendChild(item_text); // Собираем нашу структуру
23    new_item.appendChild(item_btn);
24    item_list.appendChild(new_item);
25
```

При нажатии на кнопку
«добавить» должна
добавиться новая запись.
На скриншоте
представлено её
добавление и настройка.

Подсказки к заданию 1:

```
/*  
    Добавляем обработчик событий на кнопку.  
    Сложность в том, что удалить нужно не только кнопку,  
    поэтому получаем данные о родителе, а затем из  
    родителя родителя удаляем сам узел.  
*/  
  
item_btn.addEventListener("click", () => {  
    let parent = item_btn.parentNode;  
    parent.parentNode.removeChild(parent);  
});
```

Покупочки ^_^

Новый элемент:

вапвапвап

3534

23523

ывпу45еуеу

Домашнее задание. Кнопки со стрелками перекидывают содержимое из одной рамки в другую. Также можно удалять каждого ученика и добавлять новых. В поле ввода отличник/двоечник пользователь может ввести лишь два варианта: «отличник» или «двоечник».

Двоечники

Петя Петров

Удалить

Иван Иванов

Удалить

Коля Николаев

Удалить

>>

<<

Отличники

Алексей Алексеев

Удалить

Добавить нового ученика:

фамилия и имя

отличник/двоечник

Добавить