Модуль - JavaScript

5/15 занятие

Объявление функции в виде выражения

Объявление функции в ES5 - Function Declaration:

```
function названиеФункции (параметр1, параметр2, ...) {
    console.log("что-то")
}
```

Объявление функции в ES5 - Function Expression:

```
const названиеФункции = function (параметр1, параметр2, ...) {
    console.log("что-то")
}
```

Объявление стрелочной функции в ES6 только Function Expression :

```
const названиеФункции = (параметр1, параметр2, ...) => {
    return
}
```

Вызов одинаковый для всех: **названиеФункции** (параметр1, параметр2, ...)

Задание 1

Написать функцию расчета ФОТ:

- имеется стартовый капитал **1000** рублей и **1** работник компании
- по плану, каждую неделю нанимается **1** новый сотрудник, а зарплата индексируется на **+1.5%**
- каждые **3 дня** выплачивается премия всем сотрудникам в размере **1000** рублей каждому сотруднику

В качестве входных параметров функция принимает **номер дня**(число), начиная с 0 дня работы(начало отсчета)

Для проверки результата функция должна вернуть корректные значения на 4 и 9 день

Методы document для работы с классами

const button = document.querySelector('button')

button.classList.add('active') // добавили класс

button.classList.remove('active') // удалили класс

button.classList.contains('active') // вернет false, так как такого класса у элемента нет

classList.toggle() // Метод можно использовать, чтобы включать или выключать класс

Методы document для перехода от parent к child

const redButtonsGroup = document.querySelector('.red.buttons-group')

redButtonsGroup.querySelector('div.button') // найден потомок с классом *button* внутри обертки *buttons-group*

document.querySelector('.red.buttons-group').querySelector('div.button') // то же

redButtonsGroup.parentElement // найдет родителя (обертку) группы кнопок

redButtonsGroup.parentNode // найдет родителя (обертку) группы кнопок

Методы document для работы с атрибутами

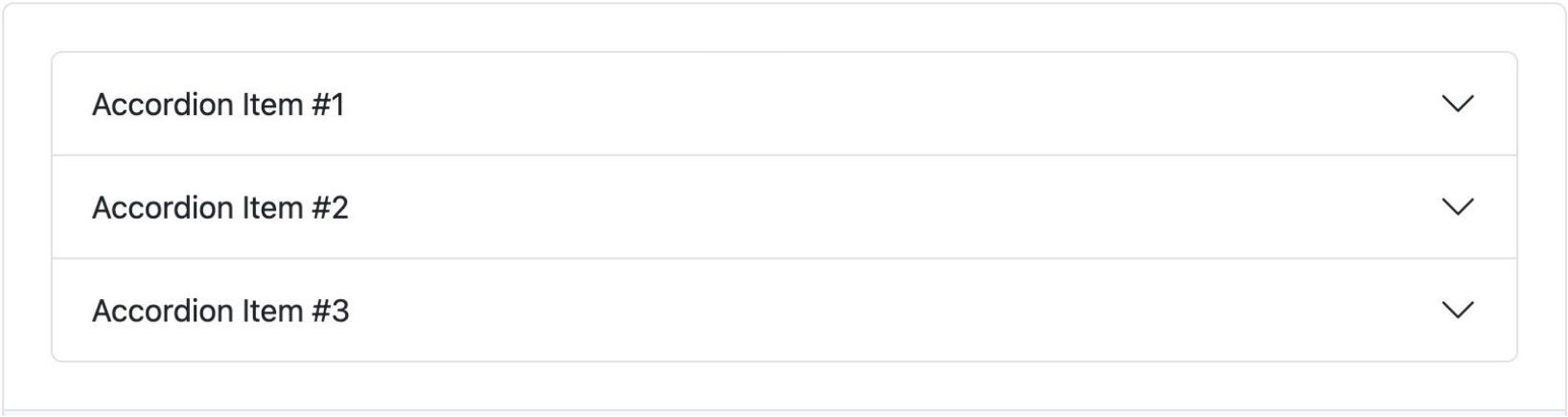
```
element.style.cssText = 'color: blue; border: 1px solid black'

element.setAttribute('style', 'color:red; border: 1px solid blue;')

element.style.color = 'blue'
```

Практика: Создание аккордеона

https://getbootstrap.com/docs/5.3/components/accordion/



перерыв

Практика: Создание компонента с вкладками

https://getbootstrap.com/docs/5.3/components/navs-tabs/#tabs

Практика: Создание карусели

https://getbootstrap.com/docs/5.3/components/carousel/

События

Событие это реакция браузера на взаимодействие пользователя с DOM-элементом, либо на заданные условия по времени или состоянию DOM. Например, когда DOM полностью готов, происходит событие **DOMContentLoaded**

Самые распространенные события:

click – клик мышью(кнопка нажата и отпущена) или тач по DOM элементу
contextmenu – клик правой кнопкой мыши по элемент
mouseover / mouseout – курсор пересекает границу элемента
mousedown / mouseup – только нажатие кнопки мышь или отпускание
mousemove – движении курсора в пределах границ элемента, раз в N миллисекунд

focus – когда устанавливается фокус для ввода текста в <input>.
keydown и **keyup** - нажата или отпущена клавиша

Обработка событий

Аттрибут onclick = ФункцияОбработчик

<button id='**btn**' onclick='**clickHandler(e)**'>

Чаще используется отслеживание событий:

```
btn.addEventListener("click", (e) => {
	//исполняемый код
});
```

Объект event

При возникновении любого события формируется объект **event**, который содержит всю возможную информацию о событии

Например:

event.target – тот элемент с которым взаимодействовали(кликнули, зацепили границу, ...).

event.currentTarget – элемент, на котором отслеживается событие

Конец занятия