Модуль - JavaScript

10/15 занятие

Объект Date

Объект для работы с датой и временем - “коробочка с отметкой во времени”

Всегда используется как новый экземпляр `new Date()`, так как в момент создания фиксирует в свойствах дату и время своей генерации

```
const currentDate = new Date('November 14, 2023 14:15:30')
```

`new Date(`год, **месяц**, день, часы, минуты, секунды, миллисекунды`)`

```
new Date(2023, 10, 14, 14, 15, 30, 0)
```

-> Tue Nov 14 2023 14:15:30 GMT+0300 (Москва, стандартное время)

Методы Date - геттеры

Date.now() - возвращает время и дату текущего момента в миллисекундах

Получение отформатированной даты:

```
const today = new Date()
today.getDay()          // 6 | порядковый номер дня от 1 до 7
today.getHours()        // 14 | числовое значение часа 0 - 24
today.getTime()         // 1660475730000 | время объекта в миллисекундах
today.getFullYear()     // 2023 | год
today.toISOString()     // 2023-...:30.000Z | строку в формате ISO
```

Количество миллисекунд измеряется с 1 января 1970 года UTC

Методы Date - сеттеры

Установка даты:

```
const moment = new Date()
moment.setFullYear(год,?месяц,?день)  // устанавливает год
moment.setMonth(месяц,?день)          // устанавливает месяц
moment.setDate(день)                  // устанавливает день(число)
moment.setHours(часы,?мин,?сек,?мс)   // устанавливает часы
moment.setMinutes(минуты,?сек,?мс)    // устанавливает минуты
moment.setSeconds(сек,?мс)            // устанавливает секунды
moment.setMilliseconds(мс)            // устанавливает миллисекунды
```

Вычисления даты

Добавление, ***вычитание*** и ***сравнение*** даты и времени производится со значениями приведенными в **миллисекунды**

```
const moment1 = new Date(Date.UTC(2022, 0, 1, 0, 0, 0))

const moment2 = new Date(Date.UTC(2022, 11, 31, 0, 0, 0))


let diffInMs = moment1 - moment2          //-31449600000
```

Задание 1

Посчитать сколько дней между датами

```
const date1 = new Date(Date.UTC(2023, 4, 4, 0, 0, 0))

const date2 = new Date(Date.UTC(2021, 7, 24, 0, 0, 0))
```

Объект Date - локальный формат

```
const today = new Date()
const options = { timeZone:'Europe/Moscow'}
today.toLocaleString('ru-Ru', options ))
```

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Intl/DateTimeFormat/DateTimeFormat#parameters

Задание 2

Написать функцию dateFormat(date), которая принимает **объект даты** и возвращает **строку** в формате “18.11.2023”

Практика: Калькулятор возраста

Реализовать компонент с интерфейсом

- поле для ввода даты и времени рождения(можно использовать input datepicker или несколько текстовых инпутов)
- кнопка “Рассчитать”

При нажатии на Рассчитать на экране должен появиться результат в виде:

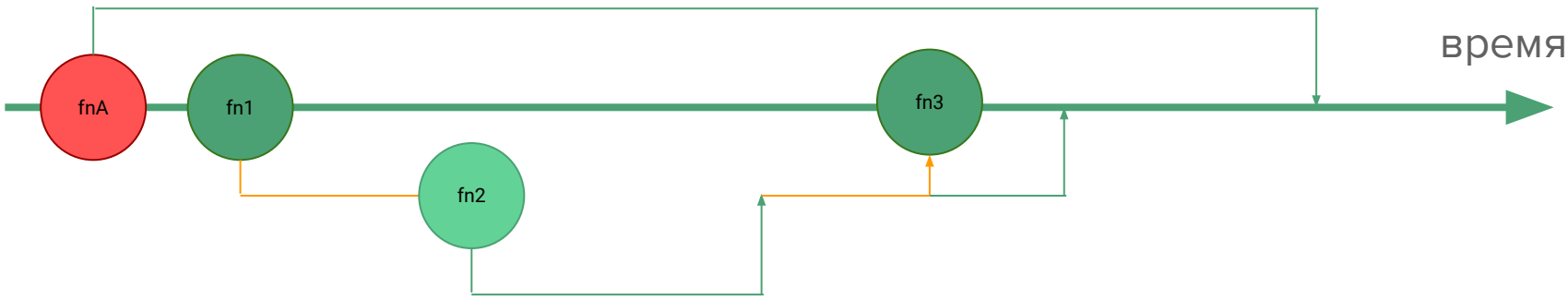
- Ваш возраст: 33 года(25 лет), 12 дней и 6 часов
- Ваш знак по гороскопу: Водолей

перерыв

Асинхронные функции

Обычный код JS выполняется синхронно в 1 потоке, то есть последовательно. И когда выполняется синхронная функция, остальные функции ждут, когда она завершится.

Асинхронная функция выполняется вне зависимости от других, но так же в 1 потоке, то есть ее результат может вернуться спонтанно с течением времени, пока выполняется другой код, но если код заблокирован, то и эта функция будет ждать “окошко” для выполнения



SetTimeout

Функция для отложенного вызова переданной в аргумент функции

```
< JS >
setTimeout( ()=> {          // 1 аргумент - функция
        … код функции …
} , 1000)                   // 2 аргумент - время до вызова в миллисекундах(0.001 сек)



// какое значение a мы увидим в консоли?
let a = 10

setTimeout(() => {
        console.log( 'Значение a равно ' + a )
}, 10000)

a = 11
```

SetInterval

Вызывает функцию с заданным интервалом(бесконечно)

```
< JS >
setInterval( ()=> {             // 1 аргумент - функция
        … код функции …
} , 1000)                       // 2 аргумент - время интервала в миллисекундах(0.001 сек)



// какие значение i мы увидим в консоли?
let i = 1

setTimeout(() => {
        console.log( 'Значение a равно ' + i++)
}, 10000)

i = 2
```

остановка таймера

Обе функции (setTimeout и setInterval) при вызове возвращают идентификатор, при помощи которого их можно остановить

< JS >

```
let interva1_id = setInterval( func1, 1000)          // например 10

let timeout1_id = setInterval( func2, 1000)          // например 3
let timeout2_id = setInterval( func3, 1000)          // например 5
let timeout3_id = setInterval( func4, 1000)          // например 19



clearInterval(10)                                    // отменит setInterval( func1, 1000)

clearTimeout(timeout2_id)                            // отменит setInterval( func3, 1000)
```

Практика: Обратный отсчет

- Обновление каждую секунду
- Подсчет секунд
- Подсчет дней
- Подсчет месяцев
- Подсчет лет

https://codepen.io/jnweb/pen/Rwvxvjm

Дополнительно

- Учитывать летнее время

Конец занятия