**Объекты**

Мы уже немного познакомились с объектами и узнали, что их можно как сериализовать (преобразовать в строку), так и десериализовать (строку сделать объектом).

Объекты нужно сериализовать для удобного хранения и передачи. Так, сериализованный объект мы можем спокойно хранить в базе данных в виде строки.

**Функции работы с объектами**

**Object.assign()**

Метод копирования полей из нескольких объектов в один.

**В Object.assign()** первым аргументом всегда должен идти пустой объект {}, это избавит нас от нежелательной мутации. Начиная со второго аргумента мы записываем объекты, все поля которых хотим скопировать в один новый объект.

const obj = {

name: 'Igor',

job: 'Developer'

}

const newObj = Object.assign({}, obj, { id: 0, age: 24 })

console.log(newObj)

/\* В консоли увидим новый объект

{

name:"Igor",

job:"Developer",

id:0,

age:24

}

\*/

Так как JavaScript постоянно развивается, функция Object.assign() уже считается немного устаревшей, так как была выпущена 2015 году. Ей на смену пришел элегантный **Spread оператор.**

**Spread оператор**

Записывается тремя точками: ...

const obj = {

name: 'Igor',

job: 'Developer'

}

const newObj = { ...obj }

console.log(newObj) //получим новый объект с полями из старого

Кроме того, при копировании объектов мы можем обновлять поля с помощью тех же Object.assign() и spread оператора.

Давайте заменим в объекте значение поля **name**

const obj = {

name: 'Igor',

job: 'Developer'

}

const newObj = { ...obj, name: 'Dastan' }

/\* Это равносильно записи Object.assign({}, obj, { name: 'Dastan' }) \*/

console.log(newObj)

/\* В консоли увидим новый объект

{

name:"Dastan",

job:"Developer"

}

\*/

И, конечно же, мы можем чередовать объекты и изменение полей в **spread**.

const newObj = { ...obj, name: 'Dastan', ...obj2, ...obj3, id: 1 }

Но не забывайте, что код читается не только сверху вниз, но и слева направо, поэтому, если вы несколько раз пропишите поле name, в итоговом объекте будет самое последнее значение.

const newObj = { ...obj, name: 'Dastan', ...obj2, name: 'Alena' }

// В итоговом объекте поле name будет содержать значение Alena.

**Object.freeze()**

Запрещает изменение объекта переданного в аргумент.

const obj = {

name: 'Igor'

}

Object.freeze(obj)

obj.name = 'Dastan'

console.log(obj) //Получим изначальный объект, без изменений

**Удаление полей в объекте**

Поля в объектах удаляются с помощью ключевого слова **delete**, после которого нужно прописать объект и поле.

const obj = {

name: 'Igor',

age: 24

}

delete obj.age

**Динамические ключи**

Ключи в объектах могут изменяться динамически. Для этого нужно передавать на место ключа в объекте переменную в квадратных скобках.

const obj = {

name: 'Igor',

age: 24

}

const newObj = { [obj.name]: obj.age } //Плучим объект { Igor: 24 }