

Язык JavaScript (основы: объекты
как ассоциативные массивы)

JS

ОБЪЕКТ

Объект - это набор свойств, каждое свойство состоит из имени и значения, ассоциированного с этим именем. Значением свойства может быть функция, которую можно назвать методом объекта. Объекты передаются по ссылке.

Применение объектов в JavaScript:

- ❖ как ассоциативный массив, для хранения данных в формате ключ-значение;
- ❖ как возможности языка для ООП.

Ассоциативный массив – структура данных, в которой можно хранить данные в формате ключ-значение.

Объявление объекта (2 варианта):
чаще используется первый

1) **let obj** = {};

2) **let obj** = **new Object**();

СВОЙСТВА ОБЪЕКТА

Объект может содержать в себе любые значения, которые называются свойствами объекта. Свойства можно понимать как переменную, закрепленную за объектом.

Доступ к свойствам осуществляется по имени:

❖ **имяОбъекта.имяСвойства** - запись «через точку»
или

❖ **объект[свойство]** - запись через квадратные скобки

Имена свойств объекта могут быть строками JavaScript, или тем, что может быть сконвертировано в строку. **Неопределенные свойства** объекта являются **undefined** (не null).

СВОЙСТВА ОБЪЕКТА (ПРИМЕР)

```
let user = {}; // создали объект user
user.id = 1; /* создали свойство с именем id,
присвоили ему значение 1 */
user.login = 'nickname';
user.email = 'nickname@email.ru';
console.log('Логин: ', user.login); /* прочитали
свойство name */
delete user.email; // удалили свойство email
```

ПРОВЕРКА НА НАЛИЧИЕ СВОЙСТВА ОБЪЕКТА

1. Сравнить свойство с `undefined` (такой способ не подойдет, если свойство есть, а его значением является `undefined`);

2. Использовать оператор `in`:

```
if ("prop" in obj) {  
    //Свойство "prop" существует  
}
```

При написании программ для значения, которое обозначает неизвестность, используется `null`.

3. `obj.hasOwnProperty("prop");`

ДОСТУП К СВОЙСТВУ ЧЕРЕЗ ПЕРЕМЕННУЮ

Квадратные скобки позволяют обратиться к свойству, имя которого хранится в переменной

```
let idKey = 'id'; /* обратились к свойству  
объекта через переменную */  
console.log(user[idKey]); // аналогично записи  
console.log(user.id);
```

Такой вариант подходит для случаев, когда свойство определяется по ходу выполнения программы.

ПЕРЕЧИСЛЕНИЕ ВСЕХ СВОЙСТВ ОБЪЕКТА

1. Для перебора всех свойств из объекта используется цикл по свойствам `for..in`.

```
for (let i in obj){  
    console.log(i); //вывод ключа  
    console.log(obj[i]); /*вывод значения по  
    ключу*/  
}
```

2. **Object.keys(obj)** - возвращает массив со всеми собственными (те, что в цепочке прототипов, не войдут в массив) именами перечисляемых свойств объекта `obj`;

3. **Object.getOwnPropertyNames(obj)** - возвращает массив содержащий все имена своих свойств (перечисляемых и неперечисляемых) объекта `obj`.

ЛИТЕРАЛЬНЫЙ СПОСОБ ОБЪЯВЛЕНИЯ ОБЪЕКТА

Литеральный способ объявления объекта объявление объекта со свойствами, указанными в фигурных скобках.

```
let имяОбъекта = {  
    свойство1: значение,  
    свойство2: значение,  
    свойство3: значение  
};
```

Пример:

```
let user = {  
    id: 3,  
    login: 'qwerty',  
    email: 'qw@email.com'  
};
```

ДЕСТРУКТУРИЗАЦИЯ

Деструктуризация особый синтаксис присваивания, при котором можно присвоить массив или объект сразу нескольким переменным, разбив его на части.

Деструктуризация массива

```
let [name, age] = ["Иван", 67];  
console.log(name); // Иван  
console.log(age); // 67
```

При таком присвоении **первое значение** массива пойдёт в **первую переменную**, **второе** – во **вторую**, а **последующие** (если есть) – **будут отброшены**.

Если нужны и последующие элементы, можно использовать оператор ...

```
let [name, age, ...otherElems] = ["Иван", 67, "ivan@gmail.com",  
    "+79991112233"];  
console.log(name); // Иван  
console.log(age); // 67  
console.log(otherElems); // ["ivan@gmail.com", "+79991112233"]
```

ДЕСТРУКТУРИЗАЦИЯ

Деструктуризация объекта

```
let {перем1, перемен2} = {перем1: "значение", перемен2:  
  "значение"};
```

Объект справа - объект, который мы хотим разбить, это уже существующий объект.

Слева - список переменных, в которые попадут соответствующие свойства.

Если необходимо присвоить свойство объекта в переменную с другим именем, нужно использовать двоеточие.

```
let {свойство1:перем1, свойство2:перем2} =  
{свойство1: "значение", свойство2: "значение"};
```

ДЕСТРУКТУРИЗАЦИЯ В ПАРАМЕТРАХ

Если функция получает объект, то она может его тут же разбить в переменные:

```
let userData = {  
  name: "Иван",  
  login: "qwe",  
  age: 44  
};  
  
function showUserData({name, login, age,  
  surname="значение по умолчанию"}) {  
  console.log(name + ' ' + login + ' ' + age + ' ' + surname);  
  /*объект будет разбит на переменные */  
}  
  
showUserData(userData); /* Иван qwe 44 значение по  
умолчанию */
```

ЧТО ПОЧИТАТЬ ДОПОЛНИТЕЛЬНО

<https://learn.javascript.ru/object> - Объекты

<https://learn.javascript.ru/keys-values-entries> - Object.keys, values, entries

<https://learn.javascript.ru/destructuring-assignment> -
Деструктурирующее присваивание