Алгоритмы обработки данных на JavaScript

Пользовательские типы данных. Формат JSON

Примитивный тип данных

Переменная хранит само значение
Значение переменной не объект, массив, функция.

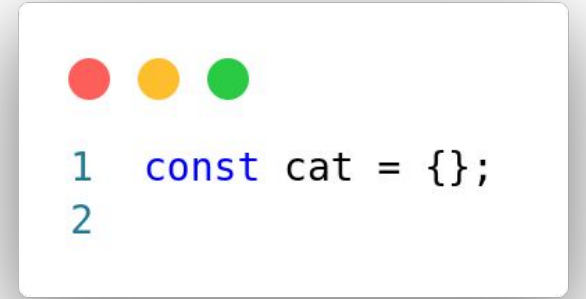
Ссылочный тип данных

Переменная хранит ссылку на место в памяти, где хранится сам массив/объект

Объекты

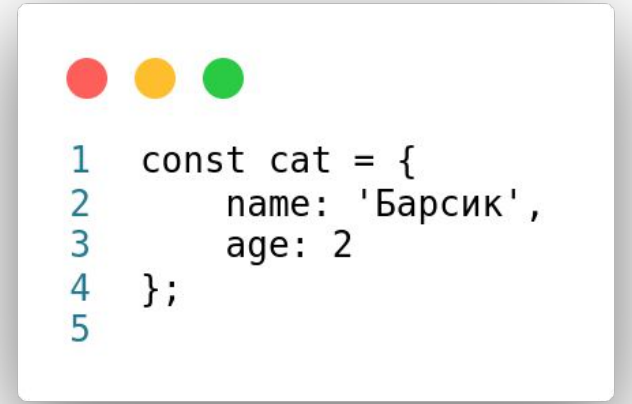
Объект — это сущность с набором свойств

Объекты создаются при помощи объектных литералов

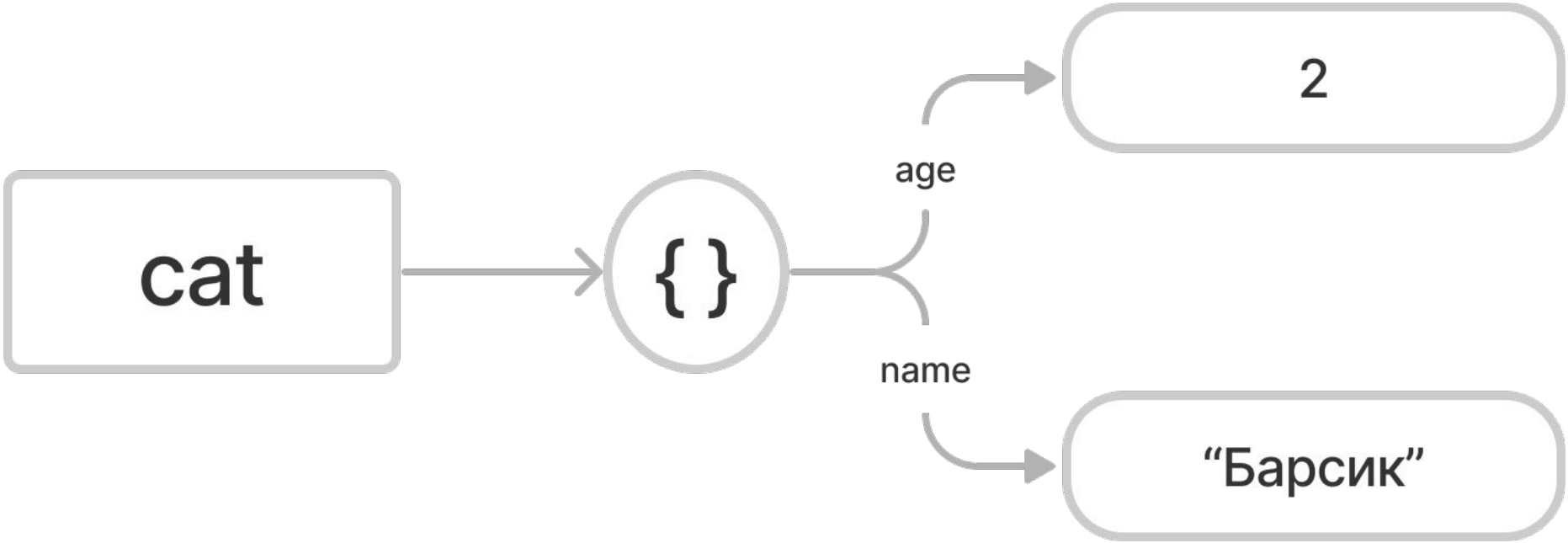


Объекты

При создании объектов мы можем задать ему список свойств в виде пар имя свойства:значение свойства



```
1  const cat = {
2      name: 'Барсик',
3      age: 2
4  };
5
```


cat
{ }
age
name
2
“Барсик”

Объект

- Свойство объекта состоит из имени свойства и значения.
- Имя свойства - уникальное значение строкового типа.
- Значение свойства может быть любого типа, включая вложенный объект, функцию.

Обращение к свойствам объекта

Доступ к свойствам объекта может осуществляться при помощи скобочной или точечной записи. Никаких ограничений на имена свойств нет. Скобочная запись позволяет обращаться и объявлять свойства, ключ которых не удовлетворяет правилам именования переменных.

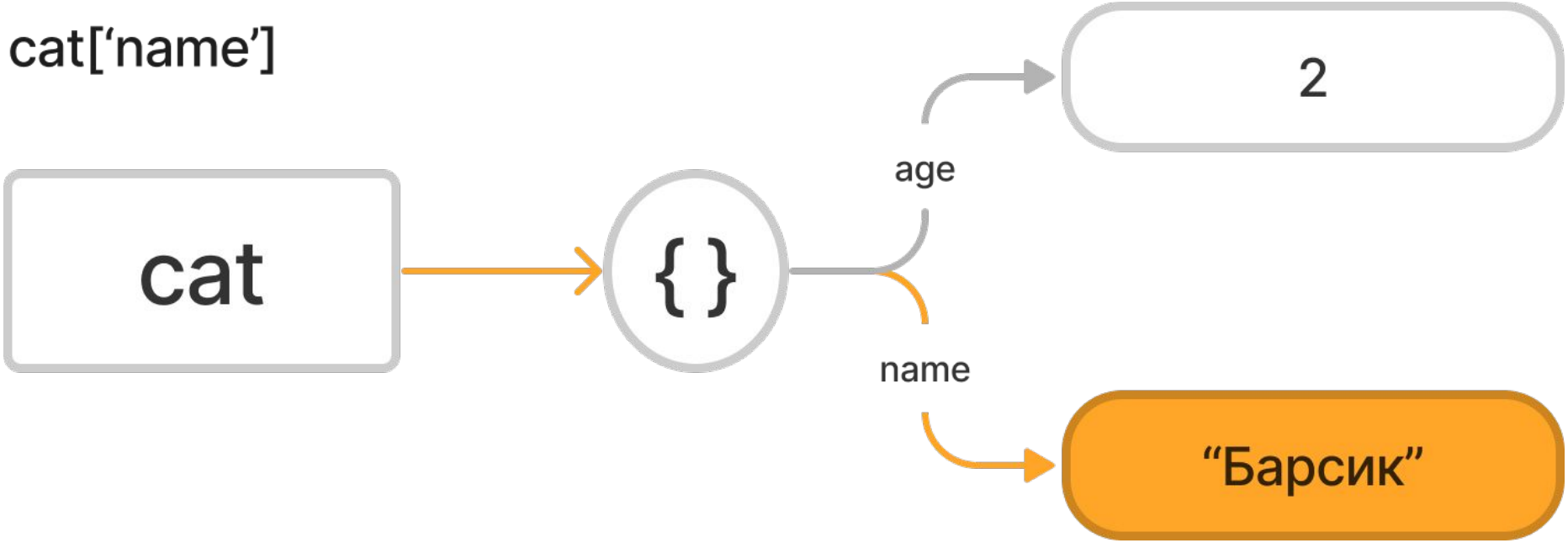
```
1  cat.age; // точечная запись
2  cat['name']; // скобочная запись
3
```

Обращение к свойствам объекта

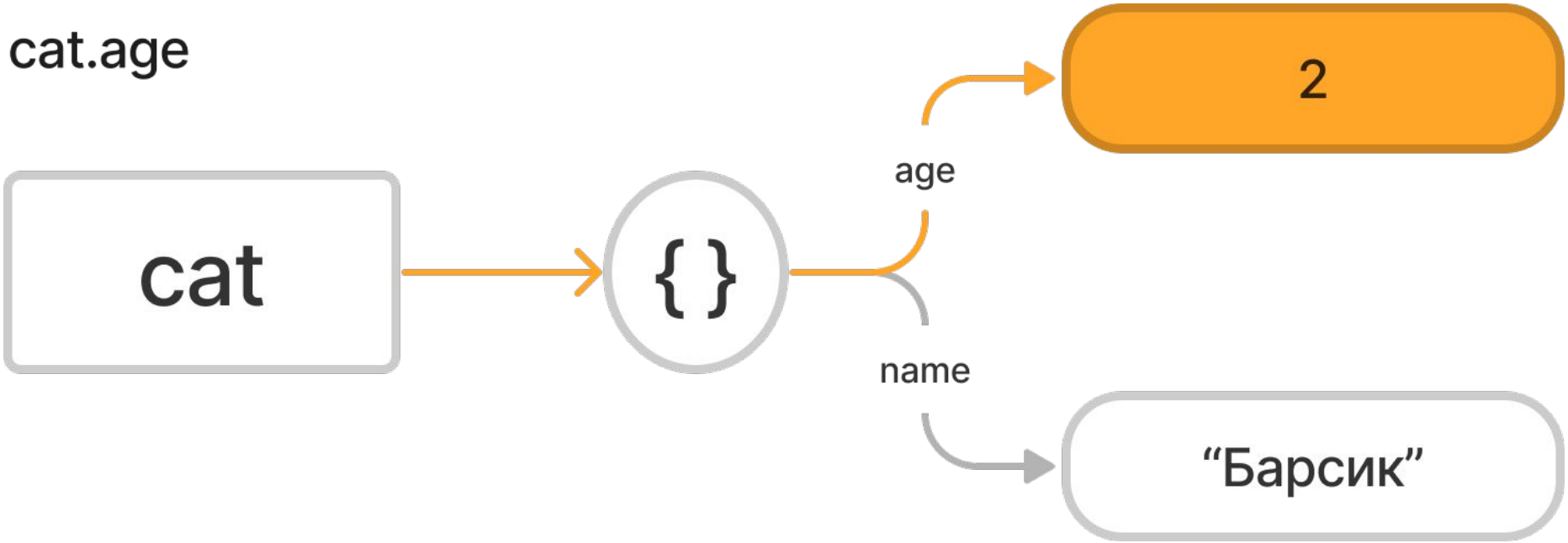
Обращение к свойствам объекта в виде obj.prop или obj[‘prop’], по сути, происходит в 3 шага:

1) Если объект null/undefined, то ошибка.
2) Если свойство существует, то возвращается его значение
3) Если свойство не существует, то возвращается undefined

cat['name']



cat.age



Скобочная запись

Скобочная запись позволяет обратиться к свойству, имя которого является результатом выражения. Также её следует использовать для создания вычисляемого свойства. В иных случаях следует использовать точечную запись.

```
1  const key = prompt("Интересуемое свойство");
2
3  alert(cat[key]);
```

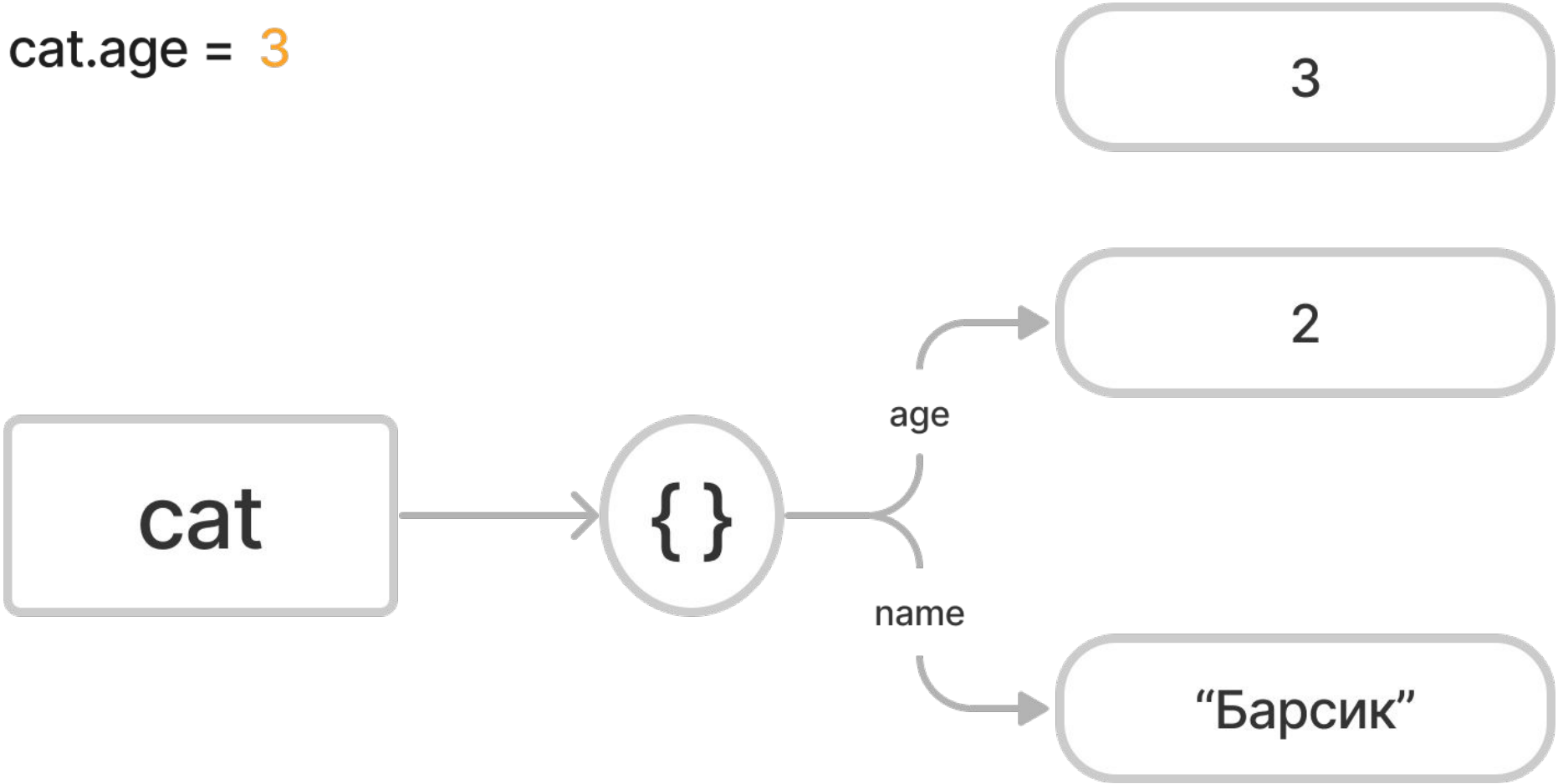
```
1  const basket = {};
2
3  const product = prompt("Товар?");
4  const count = +prompt("Количество?");
5
6  basket[product] = count;
```

Установка свойств

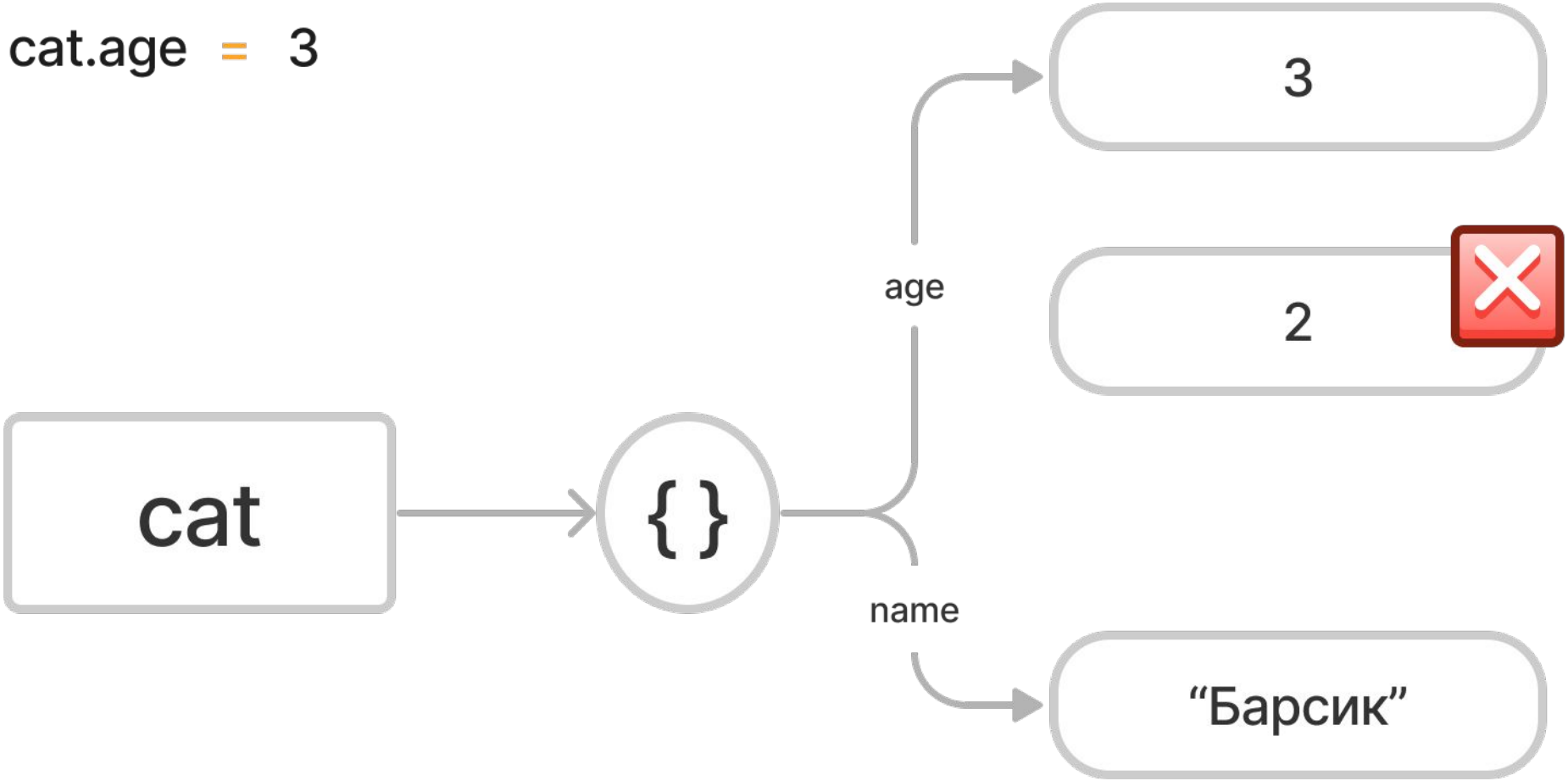
Установка свойств в объектах осуществляется при помощи точечной/скобочной записи и оператора присваивания

```
1  cat.age = 3; // изменение существующего свойства
2  cat.gender = 'm'; // добавление нового свойства
```

cat.age = 3



cat.age = 3

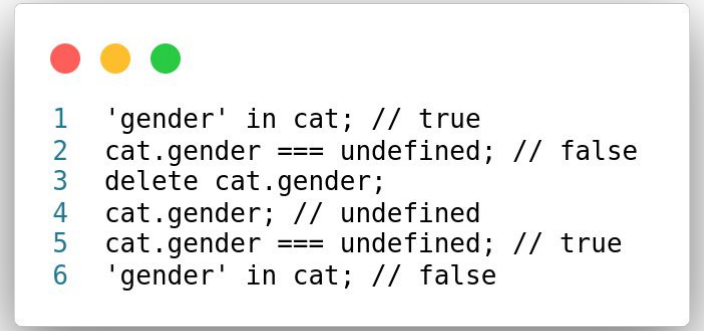


Удаление свойств

Для удаления свойств из объекта используется оператор delete

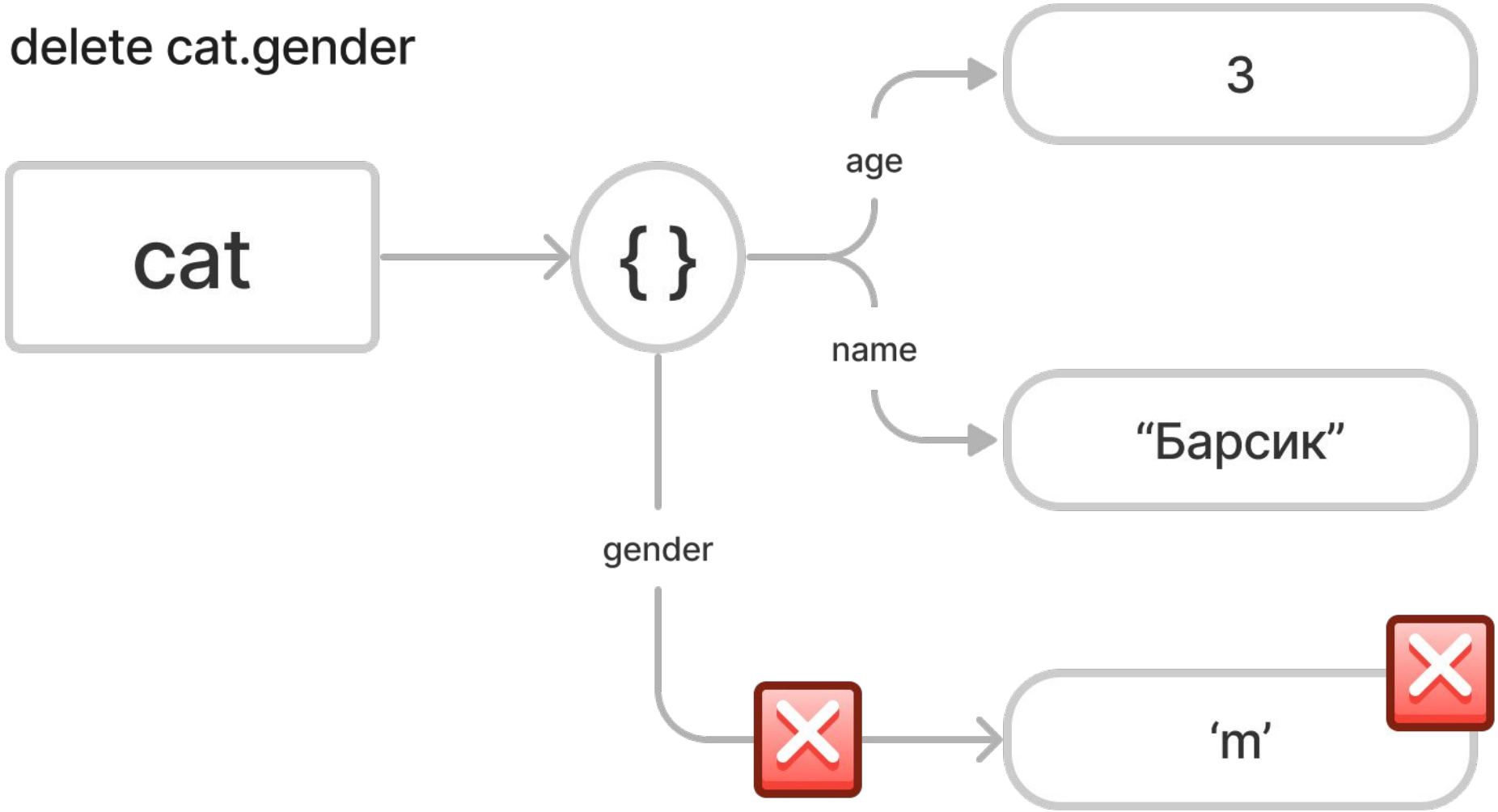
Для проверки существования свойства в объекте используется оператор in

Значение несуществующего свойства = undefined



```
1  'gender' in cat; // true
2  cat.gender === undefined; // false
3  delete cat.gender;
4  cat.gender; // undefined
5  cat.gender === undefined; // true
6  'gender' in cat; // false
```

delete cat.gender



Обход свойств объекта

Для перебора всех свойств объекта следует использовать цикл for .. in

В объявленной переменной хранится имя свойства во имя итерации

```
1  for (let property in cat) {
2      console.log(property); // age, name
3      console.log(cat[property]); // 3, "Барсик"
4  }
```

Object.keys, values

- Object.keys(obj) – возвращает массив имён свойств объекта
- Object.values(obj) – возвращает массив значений свойств объекта

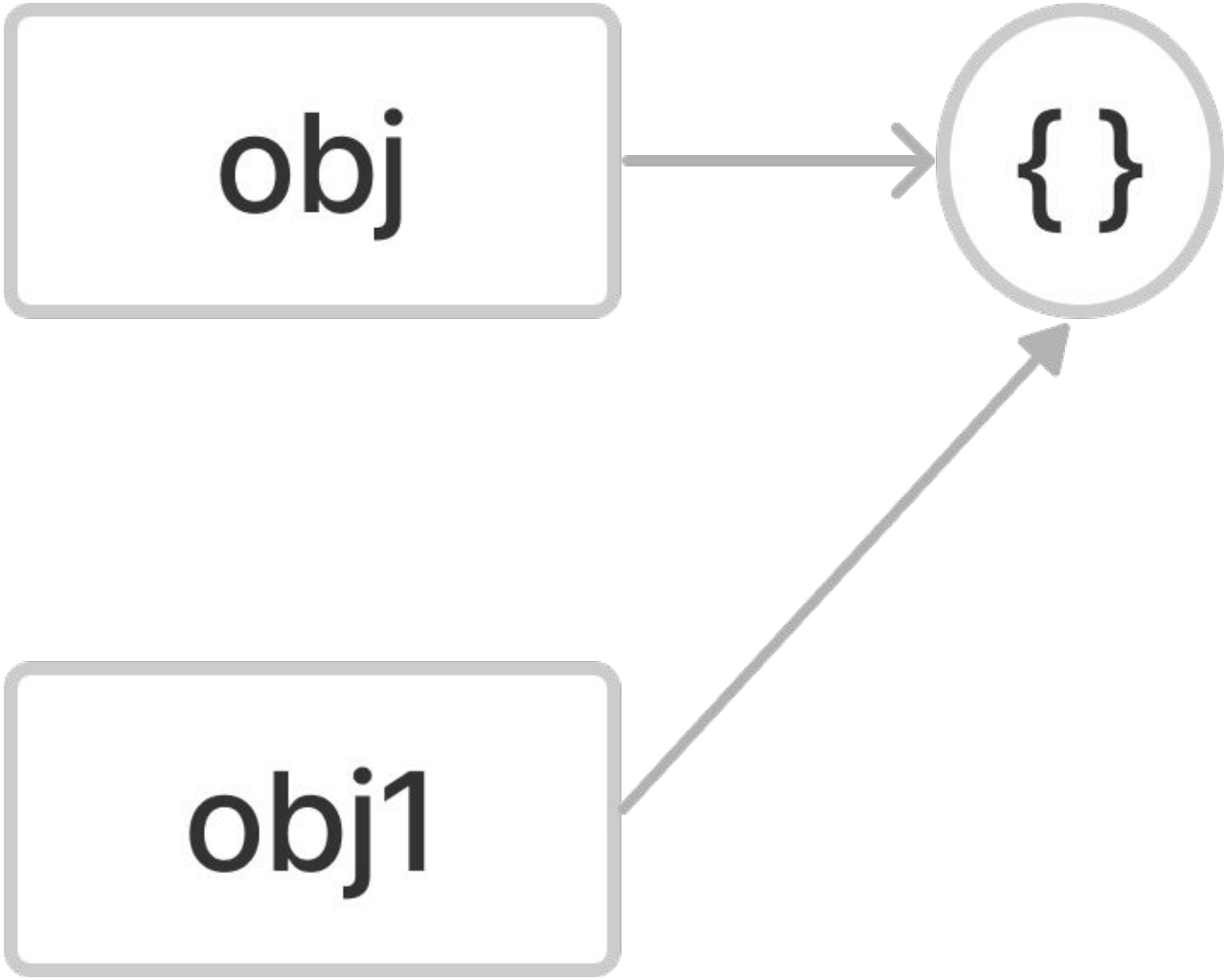
Сравнение объектов

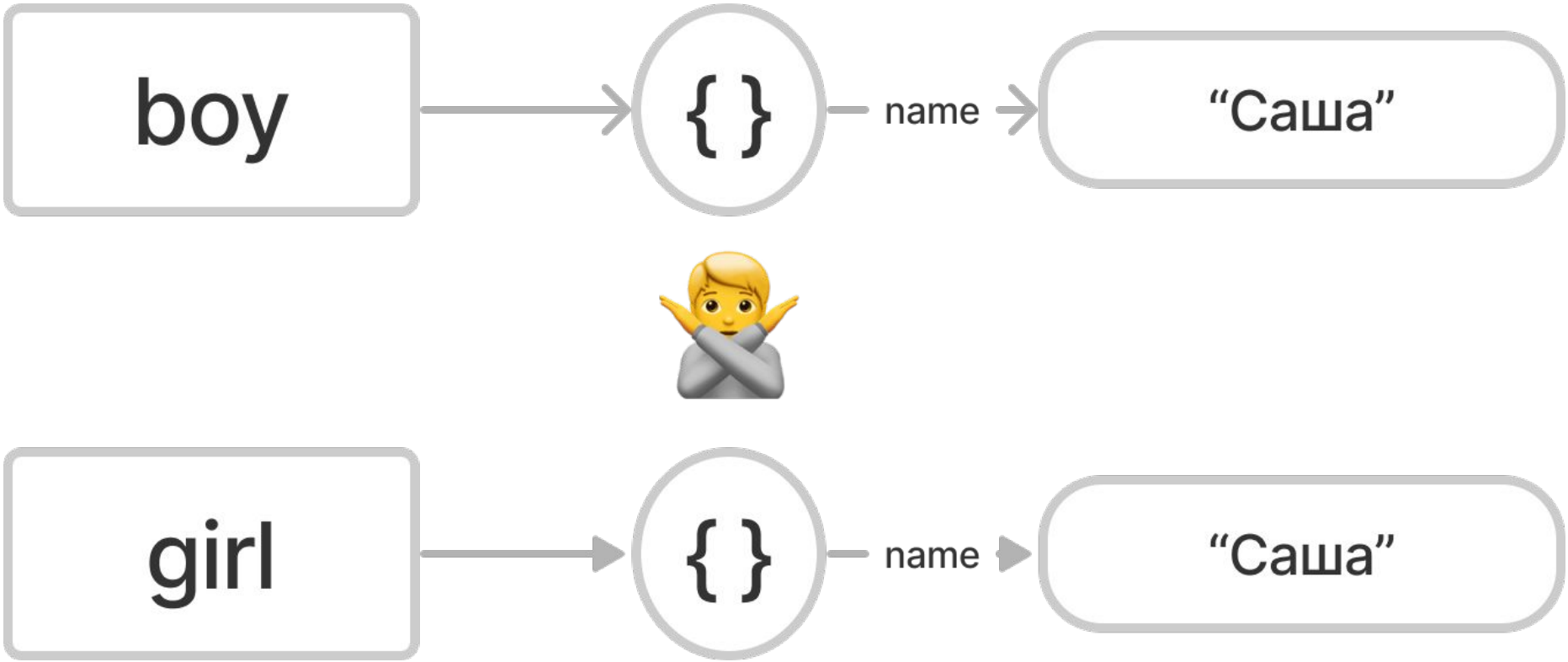
Два объекта равны тогда и только тогда, когда это один и тот же объект

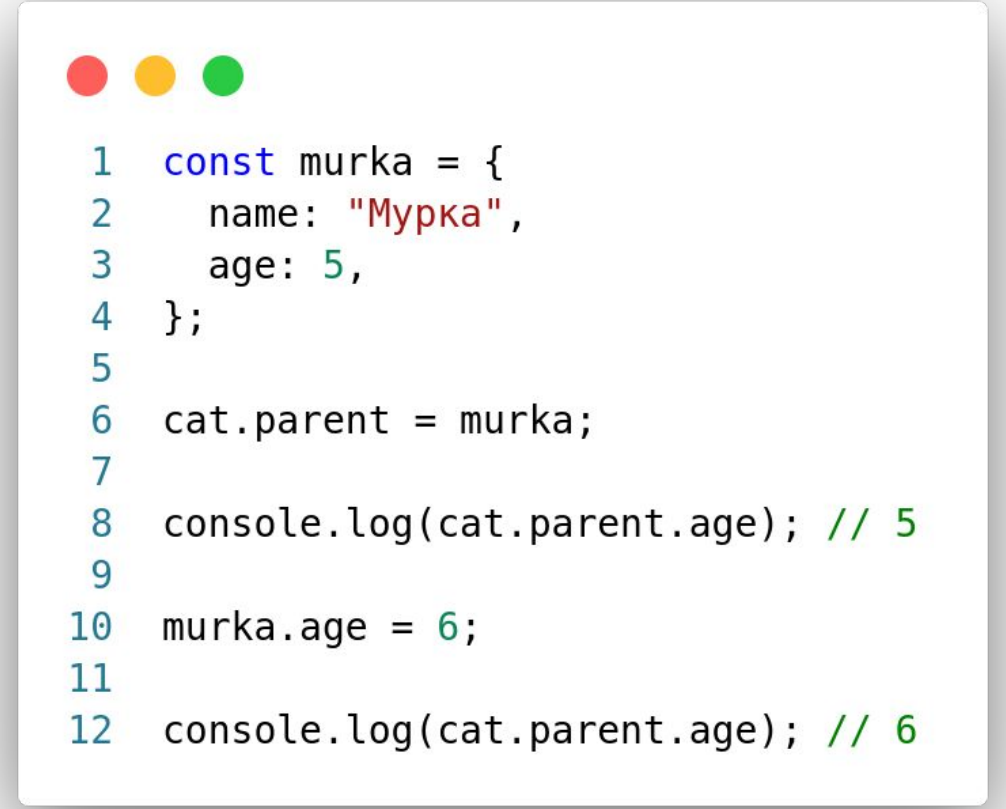
Одинаковые внешне, но независимые объекты не равны друг другу

```
1  const obj = {};
2  const obj1 = obj;
3
4  console.log(obj === obj1); // true
5  // ссылка на тот же объект!
```

```
1  const boy = {
2    name: "Саша",
3  };
4
5  const girl = {
6    name: "Саша",
7  };
8
9  console.log(boy === girl); // false
10
```


obj
{ }
obj1


boy
{ }
name
“Саша”
girl
{ }
name
“Саша”



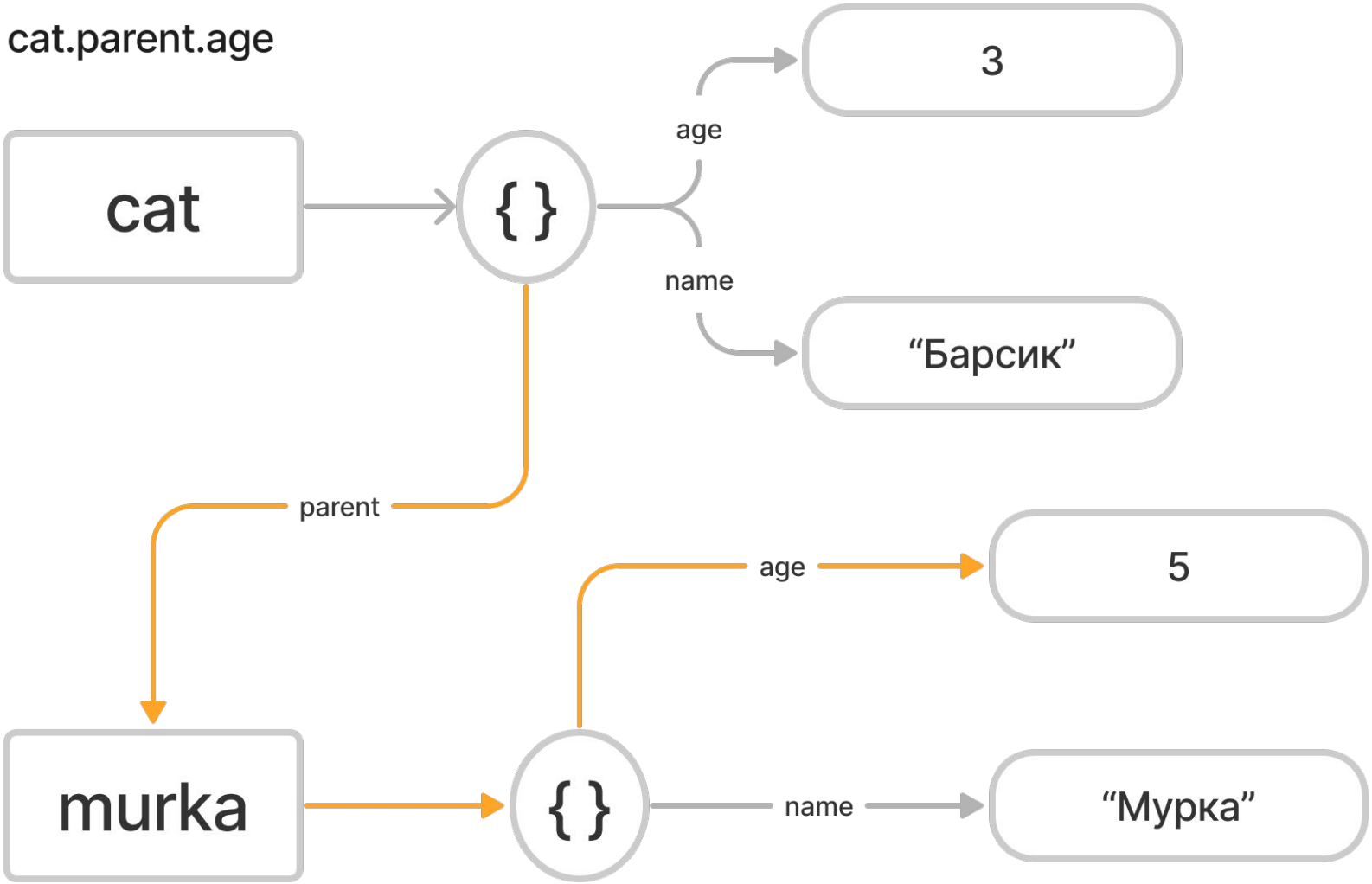
```
const murka = {
  name: "Мурка",
  age: 5,
};

cat.parent = murka;

console.log(cat.parent.age); // 5

murka.age = 6;

console.log(cat.parent.age); // 6
```


cat.parent.age
cat
{ }
age
3
name
“Барсик”
parent
murka
{ }
age
5
name
“Мурка”

Объявление объекта

```
const cat = {
  name: "Барсик",
  age: 2,
};
// Пустой объект
const empty = {};
```

Обращение к полям объекта

```
// Точечная нотация
cat.name; // Барсик
cat.age; // 2

// Скобочная нотация
cat['age']; //
```

Операции со свойствами

```
const murka = {
  name: "Мурка",
  "age": 3,
};

// Объявление свойства
cat.parent = murka;

// Удаление свойства
delete murka.age;

```

```
// Обращение к вложенному полю

cat.parent.name; // Мурка

// Обращение к несуществующему
полю
cat.randomKey; //undefined
cat.randomKey.random; //
TypeError!
```

Методы объектов

Методы задают поведение объекта

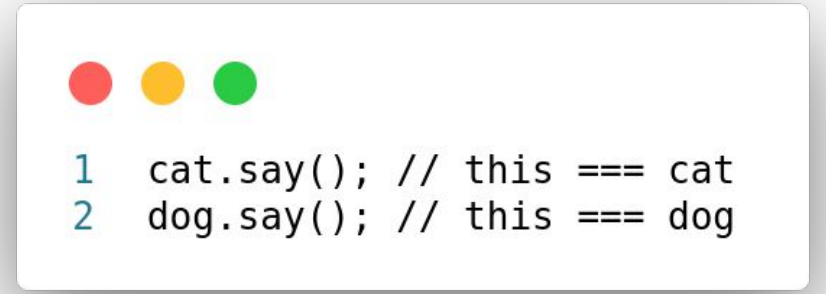
Метод - свойство объекта, являющееся функцией

```
1  cat.say = function () {
2    alert("Мяу Мяу я " + this.name);
3  }; // добавление метода объекту
4
5  const dog = {
6    name: "Шарик",
7    say() { // объявление метода при создании объекта
8      alert("Гав гав я  " + this.name);
9    },
10 };
11
```

Ключевое слово this

this указывает на объект, в контексте которого был вызван метод.

В простейшем и наиболее распространенном случае this указывает на объект, стоящий до точки.



```
1  cat.say(); // this === cat
2  dog.say(); // this === dog
```

Словарь

Объект можно рассматривать как словарь.

Словарь позволяет обращаться к значению по текстовому ключу (в нашем случае по имени свойства)

Словарь

значения	Portugues	English	French	Russian	Spanish
ключи	pt	en	fr	ru	es

```
1  const languages = {
2    pt: "Portugues",
3    en: "English",
4    fr: "French",
5    ru: "Russian",
6    es: "Spanish",
7  };
8
9  languages['pt'];
10 languages['en'];
```

Bedarfsforschung *ж. эк.* изуче́ние спро́са (ры́нка).

bedarfsgerecht I отвеча́ющий спро́су; **II** соответственно спро́су.

bedecken покрыва́ть, прикрыва́ть, укрыва́ть.

bedeckt о́блачный; па́смурный.

Bedenken *мн. ч.* сомне́ния.

bedenken обду́мывать *что-л.*, ду́мать, размышля́ть *о чём-л. или над чем-л.*

bedenklich 1. озабо́ченный, встрево́женный; **2.** трево́жный, опа́сный, внуша́ющий опасе́ние; **3.** сомни́тельный, риско́ванный.

bedeuten зна́чить; означа́ть; was soll das bedeuten? что э́то зна́чит?; das hat nichts zu bedeuten э́то не име́ет (никако́го) значе́ния, э́то нева́жно.

bedeutend I значи́тельный, кру́пный; ва́жный, выдающи́йся; **II** значи́тельно, гора́здо.

Bedeutung *ж.* -en значе́ние; von Bedeutung значи́тельный, ва́жный.

bedeutungslos не име́ющий (никако́го) значе́ния, незначи́тельный, мало ва́жный.

bedeutungsvoll 1. ва́жный, знамена́тельный; **2.** многозначи́тельный.

bedienen I обслу́живать; **II** sich bedienen: bitte, bedienen Sie sich! угоща́йтесь, пожа́луйста!

Bedienung *ж.* -en **1.** (*ед. ч.*) обслу́живание, *тех. тж.* управле́ние; **2.** (*ед. ч.*) официа́нт; продаве́ц; **3.** обслу́живающий персона́л.

Bedienungspult *ср. тех.* пульт управле́ния.

bedingen обусло́вливать.

bedrängen 1. пресле́довать, тесни́ть *врага и т. п.*; **2.** досажда́ть, докуча́ть, надоеда́ть *кому-л. чем-л.*; **3.** тяготи́ть, угнета́ть *кого-л. о чувстве и т. п.*

Bedrängnis *ж.* бе́дственное (тяжёлое) положе́ние.

bedrohen грози́ть, угрожа́ть *кому/чему-л.*, ста́вить под угро́зу *что-л.*; von bedroht sein находи́ться под угро́зой чего́-л.

bedrohlich угрожа́ющий, опа́сный.

Bedrohung *ж.* -en угро́за, опа́сность; Bedrohung des Friedens угро́за ми́ру; atomare Bedrohung угро́за я́дерного нападе́ния.

bedrücken угнета́ть, тяготи́ть.

bedrückend тя́жкий, тя́гостный.

bedruckt *текст.* набивно́й.

bedürfen нужда́ться *в чём-л.*; das bedarf der Erläuterung э́то тре́бует поясне́ния, э́то ну́жно (сле́дует) поясни́ть.

Bedürfnis *ср.* -ses, -se **1.** *мн. ч.* потре́бности, запро́сы; kulturelle Bedürfnisse культу́рные запро́сы; **2.** потре́бность, нужда́; Bedürfnis nach haben нужда́ться (име́ть потре́бность) в чём-л.

Beefsteak [ˊbi:fste:k] *ср.* -s, -s бифште́кс.

beeilen, sich торопи́ться, спеши́ть.

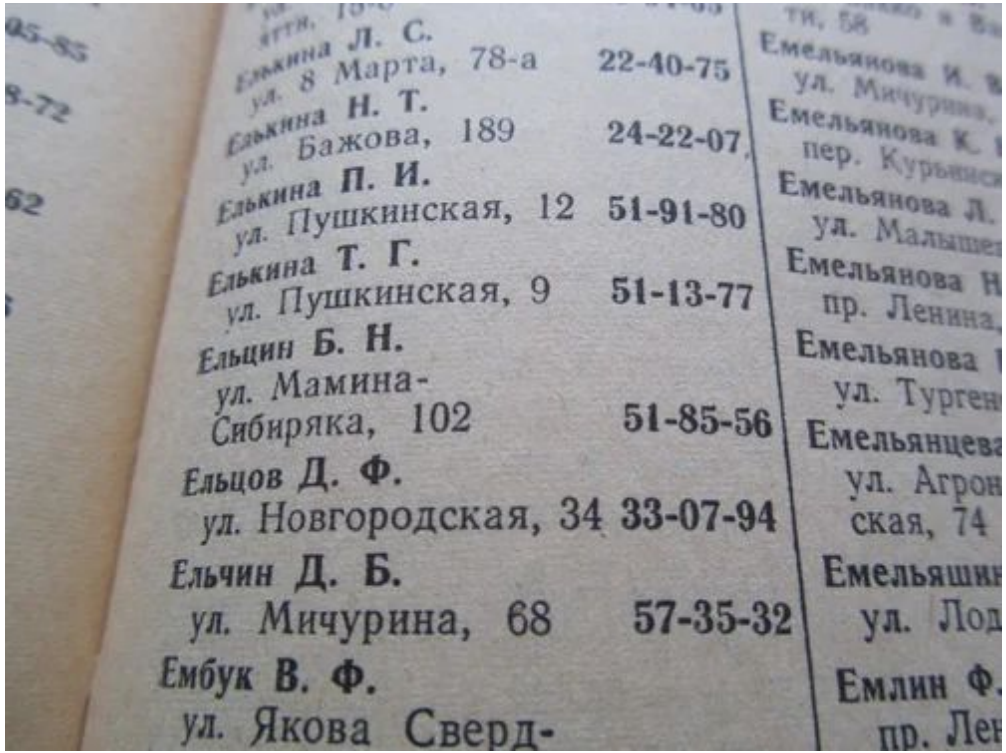
beeindrucken производи́ть (си́льное) впечатле́ние *на кого-л.*

beeindruckend впечатля́ющий, вуши́тельный.

beeinflussen ока́зывать влия́ние, влия́ть *на кого-л.*

beend(ig)en конча́ть, ока́нчивать; прекраща́ть; завершáть.

Beerdigung *ж.* -en погребе́ние, по́хороны.



```
const capitals = {
  Russia: "Moscow",
  USA: "Washington",
};
let country;

do {
  country = prompt("В какой стране вы живёте?");

  if (capitals[country]) {
    alert("Столица вашей страны " + capitals[country]);
  } else {
    const city = prompt("Какая столица вашей страны?");

    capitals[country] = city;
  }
} while (country);
```

JSON

JSON - простой формат обмена данными, удобный для чтения и написания как человеком, так и компьютером.

JSON - текстовый формат, полностью независимый от языка реализации. Это делает JSON идеальным языком обмена данными.

Создан и популяризирован Дугласом Крокфордом

JSON

JSON основан на двух структурах данных:

- Коллекция пар ключ/значение. JS объект
- Упорядоченный список значений. JS массив

JSON

JSON поддерживает следующие типы данных:

- объекты
- массивы
- строки
- числа
- null
- boolean

JSON

```
{
    "name": "303",
    "year": 2,
    "isPartTimeGroup": false,
    "students": [{
        "name": "John Doe",
        "age": 20
    }]
}
```

- имена свойств обязательно заключаются в двойные кавычки
- для строк разрешены только двойные кавычки
- наличие комментариев и висячих запятых не допускается

JSON.stringify

JSON.stringify - принимает объект как параметр и возвращает эквивалентную JSON строку
Специфические свойства языка JS не сериализуются. А именно:

- методы объектов
- undefined поля

```
> const cat = {
      name: "Барсик",
      age: 2,
      say() {
          alert('Meow');
      }
  }

  JSON.stringify(cat);
< '{"name":"Барсик","age":2}'
```

JSON.parse

JSON.parse - принимает строку JSON в качестве параметра и возвращает соответствующий объект JavaScript

```
> const str = `
  {
      "message": "HELLO!"
  }
  `;

  JSON.parse(str);
◂· ▼{message: 'HELLO!'} i
     message: "HELLO!"
```