# Prototypes

**Прототип** - это способ наследования свойств в объектах JavaScript.

Каждый объект при создании может унаследовать какие-то свойства (поля) от другого объекта.

Унаследованные свойства будут (должны) находиться в поле с ключом **\_\_proto\_\_**.

Рассмотрим на примере:

const car = { wheels: 4, doors: 4 }

const yellowCar = {color: 'yellow', \_\_proto\_\_: car}

console.log(yellowCar) //отобразит {color: 'yellow'}

console.log(yellowCar.wheels) //отобразит 4, унаследованное от объекта car

Если мы просто отобразим содержимое объекта yellowCar, то увидим лишь поле **color: 'yellow'**, но мы всё равно сможем обратиться к полям wheels и doors, так как они были унаследованы от объекта car.

Кроме того, мы можем добавлять в прототип новые свойства:

const car = { wheels: 4, doors: 4 }

const yellowCar = {color: 'yellow', \_\_proto\_\_: car}

yellowCar.\_\_proto\_\_.driver = 1

console.log(yellowCar) //отобразит {color: 'yellow'}

console.log(yellowCar.driver) //отобразит 1

console.log(car)

/\*

отобразит

{

wheels:4,

doors:4,

driver:1

}

\*/

Каждый созданный объект, включая массивы и функции, по-умолчанию наследует стандартный прототип.

В зависимости от типа объекта, ему присваивается протитип со своим набором свойств.

Именно так, массивы получают свойства map, filter и т.д.  
  
Объекты не являются единственными, кто имеет прототипы в JS.

Например, мы уже знаем, что у строки есть свойства length, indexOf() и другие. Мы можем добавить строкам в прототип новые свойства.

String.prototype.foo = function(){ return `--- ${this} ---`}

console.log('Hello'.foo()) //отобразит '--- Hello ---'

В примере выше мы добавили всем строкам метод foo(), который будет добавлять строке обрамление.