Прототипное ООП

Занятие 14

GO IT

goit.com.ua

План занятия

- Объект прототип
- F.prototype и new
- Встроенные «классы»
- Делаем свои «классы»
- Наследование
- Примеси
- instanceof

Объект прототип

__proto__

```
var Vehicle = {
    canRoll: true
};

var Moto = {
    places: 2
};

Moto.__proto__ = Vehicle;

console.log(Moto.canRoll); //=> true
console.log(Moto.places); //=> 2
```

hasOwnProperty

```
var Vehicle = {
    canRoll: true
};

var Moto = {
    __proto__: Vehicle,
    places: 2
};

for (var key in Moto) {
    console.log(key + " = " + Moto[key]); //=> true, 2
}

for (var key in Moto) {
    if (Moto.hasOwnProperty(key)) console.log(key + " = " +
Moto[key]); //=> 2
}
```

Object.create(null)

```
var fuel = {};

fuel.gas = 22.6;
fuel.diesel = 20.5;

console.log(fuel.toString)
      //=> function
toString()
```

```
var fuel =
Object.create(null);

fuel.gas = 22.6;
fuel.diesel = 20.5;

console.log(fuel.toString)
      //=> undefined
```

Методы для __proto__

- `Object.getPrototypeOf(obj)` ⇔ `obj.__proto__`
- `Object.setPrototypeOf(obj, proto)` ⇔ `obj.__proto__ = proto`
- `Object.create(proto [, propertiesObj])`

“F.prototype” и “new”

F.prototype и new

```
var Lexus = {
    isCool = “Hell yeah!”
};

function ModelF(type) {
    this.type = type;
    this.__proto__ = Lexus;
}

var car = new ModelF(“Sport
coupe”);

console.log(car.isCool); //=>
“Hell yeah!”
```

```
var Lexus = {
    isCool = “Hell yeah!”
};

function ModelF(type) {
    this.type = type;
}

ModelF.prototype = Lexus;

var car = new ModelF(“Sport
coupe”);

console.log(car.isCool); //=>
“Hell yeah!”
```

Свойство constructor

```
function Vehicle(name) {
    this.name = name;
}

// Vehicle.prototype = {
//  constructor: Vehicle
// }

var mazda = new Vehicle(“Mazda”);
var toyota = new
mazda.constructor(“Toyota”);

console.log(mazda.name); //=> Mazda
console.log(toyota.name); //=>
Toyota
```

```
function Vehicle(name) {
    this.name = name;
}

// Vehicle.prototype = {
//  sounds: “Beeep!”
//
// };

Vehicle.prototype.rolls = true;
Vehicle.prototype.sounds =
“Beeep!”;

var mazda = new Vehicle(“Mazda”);
var toyota = new
mazda.constructor(“Toyota”)

console.log(mazda.name); //=> Mazda
console.log(toyota.name); //=>
Toyota
```

IE8 — Помним, не любим, не скорбим

```
function inherits(proto) {
    function F() {};
    F.prototype = proto;
    return new F;
}

var Vehicle = { rolls: true };
var Moto = inherits(Vehicle);

console.log(Moto.rolls); //=> true

// Полифил
// if (!Object.create) Object.create = inherits;
// var Moto = Object.create(Vehicle);
```

Встроенные классы

Object.prototype и методы у {}

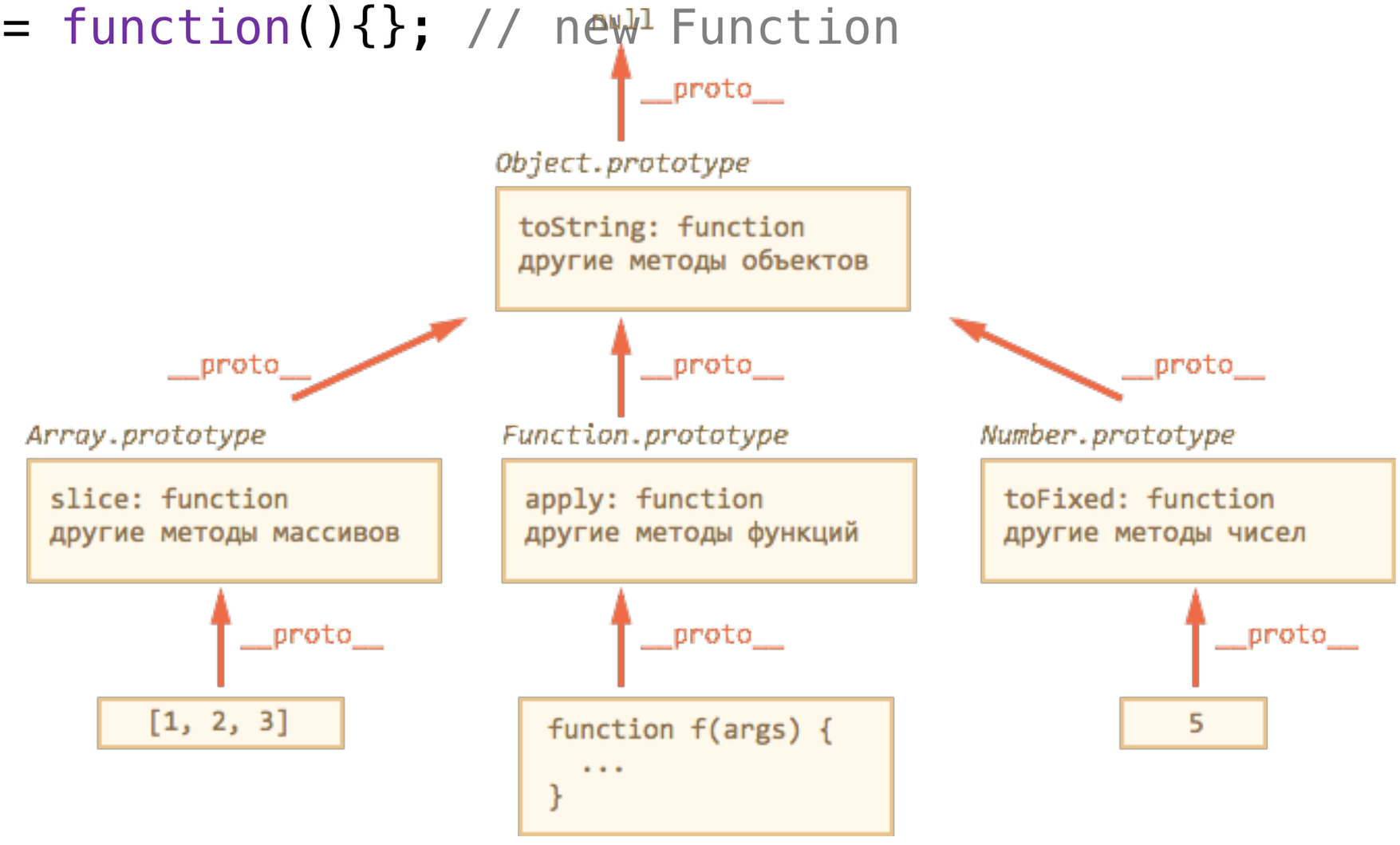
```
var Vehicle = {};

console.log( Vehicle ); //=> [object Object]
console.log( Vehicle.hasOwnProperty(“toString”) ); //
=> false

console.log( Vehicle.__proto__.hasOwnProperty(“toStri
ng”) ); //=> true

console.log( Vehicle.toString ==
Object.prototype.toString ); //=> true
console.log( Vehicle.__proto__ == Object.prototype );
//=> true
console.log( Vehicle.__proto__.__proto__ ); //=> null
```

Встроенные классы

```
var Vehicle = {}; // new Object
var tankStations = []; // new Array
var fillTank = function(){}; // new Function
```

Иерархия:

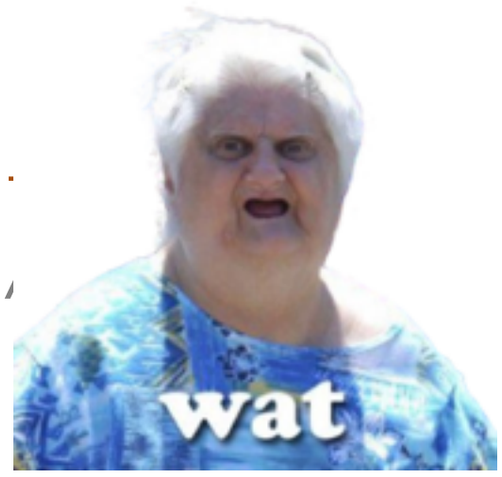
- null
  - Object
    - Array
    - Function
    - Number
    - String
    - Boolean
    - …



Примитивы

```
// Number.prototype
// Boolean.prototype
// String.prototype

var foo = “Banana”;
foo.color = “yellow”;
console.log( foo.color ); //=> undefined
console.log( foo.toUpperCase() ) //=> YELLOW
```

```
var isFalse = new Boolean(fa
console.log( !! isFalse ); /
```



Изменение встроенных прототипов

```
String.prototype.isAwesome = function() {
    return this + “ is awesome, dude!”;
};

var beer = “Guinness”;
console.log( beer.isAwesome() ); //=> Guinness is awesome,
dude!
```

Делаем свои «классы»

Конструктор и класс через прототип

```
function Vehicle () {
    this.doNoize = function() { console.log(“WooWoo!”) };
}

var Mazda = new Vehicle, Toyota = new Vehicle;

console.log( Mazda.hasOwnProperty(“doNoize”) ); //=> true
console.log( Mazda.doNoize == Toyota.doNoize ); //=> false


//==========================

function Vehicle () {}

Vehicle.prototype.doNoize = function() {console.log(“WooWoo!”) }

var Mazda = new Vehicle, Toyota = new Vehicle;

console.log( Mazda.hasOwnProperty(“doNoize”) ); //=> false
console.log( Mazda.doNoize == Toyota.doNoize ); //=> true
```

Наследование

GO IT

goit.com.ua

Наследование в «классах»

```
function Vehicle() {
    this.canRoll = true;
}
Vehicle.prototype.doNoize = function() {
    console.log( “WooWoo!” );
};

function Moto(make) {
    this.make = make;
}

// Moto.prototype.__proto__ = Vehicle.prototype; IE10- fail =(

Moto.prototype = Object.create( new Vehicle );
Moto.prototype.constructor = Moto; // Good practice

var ktm = new Moto( “KTM” );
console.log( Moto.canRoll ); //=> true
console.log( Moto.doNoize ); //=> WooWoo!
```

Вызов конструктора родителя

```
function Vehicle(make, model) {
    this.make = make;
    this.model = model;
}
Vehicle.prototype.doNoize = function() {
    console.log( this.make + “ ” + this.model  + “ sounds
dangerous!”)
}

function Moto(make, model) {
    Vehicle.call(this, make, model);
}

Moto.prototype = Object.create(Vehicle.prototype);
Moto.prototype.constructor = Moto;

var moto = new Moto(“Kawasaki”, “Ninja”);
console.log( moto.doNoize() ); //=> Kawasaki Ninja sounds
dangerous!
```

Переопределение методов

```
function Vehicle(make) {
    this.make = make;
}
Vehicle.prototype.doNoize = function() {
    console.log( this.make + “ sounds dangerous!” );
}
function Moto(make) {
    Vehicle.call(this, make);
}
Moto.prototype = Object.create(Vehicle.prototype);
Moto.prototype.constructor = Moto;

Moto.prototype.doNoize = function() {
    Vehicle.prototype.doNoize.apply(this, arguments);
    console.log( “Wzzzzzaaaammm!” );
};

var moto = new Moto( “Kawasaki” );
console.log( moto.doNoize() ); //=> Kawasaki sounds dangerous! \n
Wzzzzzaaaammm!
```

Примеси

Примеси

```
function Moto(make) {
    this.make = make;
}

var stuntsMixins = {
    doBurnout: function() { console.log( this.make + “ does a burnout!” ) },
    doWheelie: function() { console.log( this.make + “ does a wheelie!” ) }
}

for ( var mixin in stuntsMixins ) {
    Moto.prototype[mixin] = stuntMixins[mixin];
}

var moto = new Moto( “Jawa” );

console.log( moto.doBurnout() ); //=> Jawa does a burnout!
```

instanceof

instanceof

```
function Vehicle(){
    // …
}
function Moto(){
    // …
}

Moto.prototype = Object.create(new Vehicle);
Moto.prototype.constructor = Moto;

var mt = new Moto;

console.log( mt instanceof Moto ); //=> true
console.log( mt instanceof Vehicle ); //=> true
console.log( mt instanceof Object ); //=> true
```

План занятия

- Объект прототип
- F.prototype и new
- Встроенные «классы»
- Делаем свои «классы»
- Наследование
- Примеси
- instanceof

2015