

OOP

Constructor Fun Introduction

148

```
function User(id, username, salary) {
```

```
    this.id = id;
```

```
    this.username = username;
```

```
    this.salary = salary;
```

3

```
let userOne = new User(100, "Elzero", 5000);
```

```
-- Two -- (101, "Sayed", 6000);
```

الآن نريد أن نطبّق كل هذه الأشياء على المثلثات
لذلك سنكتب لها مثلاً مثلثاً له كل طرف يساوي 50
وكل زاوية تساوي 60 درجة

User userOne.i \Rightarrow 100

User userTwo.s \Rightarrow 6000

Constructor Fun New Syntax

149

درس ١٤٨ دلیل Fun new چیزی که باید بدل چیزی

ES2015 ای Fun new چیزی که باید بدل چیزی

اگر این فورم ای Fun new بخواهد

Class User {

constructor(id, username, salary) {

this.i = id;

this.u = username;

this.s = salary;

}

3

let userOne = new User(100, "Elzero", 5000);

console.log(userOne instanceof User) => true

console.log(userOne.constructor === User) => true

Deal with Properties & Methods

150

class User {

constructor(id, username, salary) {

this.i = id; // Properties

this.u = username || "Unknown";

this.s = salary < 6000 ? salary + 500 : salary;

this.msg = function () {

return `Hello \${this.u} Your Salary Is \${this.s}`;

}

}

// Methods

writeMsg() {

return `Hello \${this.u} Your Salary Is \${this.s}`;

}

3

let userOne = new User(100, "Elzero", 5000);

console.log(userOne.msg())

console.log(userOne.writeMsg())

3

OOP #2

Update Properties And Build In Constructors

151

ابدأ method `updateName` في class `User` بـ "this" بدل من `user`

```
class User {  
    // properties  
    constructor(u, n, b) {  
        this.u = u;  
        this.name = n;  
        this.balance = b;  
    }  
    // methods  
    updateName(newName) {  
        this.u = newName;  
    }  
}
```

```
let userOne = new User(100, "Elzero", 5000);
```

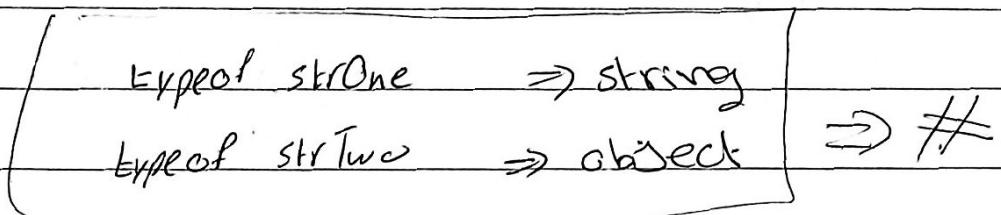
```
console.log(userOne.u) => Elzero
```

```
userOne.updateName("osama");
```

```
console.log(userOne.u) => osama
```

```
let strOne = "Elzero";
```

```
let strTwo = new String("Elzero");
```



strOne instanceof String False

strTwo instanceof String True

strOne.constructor === String True

strTwo.constructor === String True

Class Static Properties And Methods

152

Class میں کوئی class var, fun, static variable نہیں
state assignment کیں, obj → static members کے

class User {

static count = 0;

constructor(id, username, salary) {

 this.i = id;

 this.u = username;

 this.s = salary;

 User.count++;

}

static countMembers() {

 return ` \${this.count} Members Created`;

}

}

let userOne = new User(100, "osama", 50000);

اندیشہ کیا? static یعنی class count ::

userOne.count | User.count

↓

| user.countMembers()

OOP ٣

Class / Inheritance

153

وهي class ونوعها Admin, username, id هي user ، و هي class لـ Admin
وهي class ونوعها Admin, username, id هي user ، و هي class لـ Admin
Admin ي継承 user . Admin ي継承 user . Admin ي継承 user . Admin ي継承 user .

class User {

 |

 |

 3

class Admin extend User {

 constructor(id, username, per) {

 super(id,username);

 this.p = per;

 username, id Admin

 3

User ← class ← Method ← ... ← Admin class لـ Admin class لـ Admin

Class Encapsulation

154

بنادرو کیف تینی متنگ بکون خاص ب class و یعنی اگر بنادر ~~فراز~~ ^{فراز} ~~obj~~ ^{obj} مانند ~~تینی~~ ^{تینی} تواند فرالانگیم:

(٤) عن طريق متغير تاز و موجودة في class Fun

Class User E

```
#e    // private property  
constructor (id, username, eSalary) {  
    this.i = id;  
    this.u = username;  
    this.#e = eSalary;  
}  
  
getSalary () {  
    return parseInt (this.#e);  
}
```

let user

`let userOne = new User(100, "osama", "5000 Ghns");`

~~`user.n`~~ \Rightarrow osama

~~`user.e`~~ \Rightarrow Error

Class \rightarrow no to private like \underline{e} \in غير موصولة

~~`user.getSalary() * 0.3`~~ \Rightarrow 1500

Prototype Introduction

155

لما تكون Prototyoe شبيه Class خواص Class مساعدة في إنشاء Class

OOP ~~4~~

Add ~~Prop~~ Prototype Chain and Extend

Constructors Features

156

ـ دلائل على تسلسله من class حاتورة

ـ نسخة المفاصل يتغير، وحاتورة خواص زعم لا وجود لها

ـ تقدّر اضافة من مثبط Class لـ method

User.prototype.sayHello = function () {

 |

 return "welcome \${this.u}";

};

ـ تقدّر اضافة من مثبط properties

User.prototype.love = "Elzero Web School";

ـ built-in ~~is Local~~ obj واحتورة كل مفاصيلها هي مفاصيلها.

Object.prototype.love = _____;

لأن function String هي Prototype في المعيار العربي
ـ "أداة إضافة".

```
String.prototype.addDot = function(val){  
    return '.' + {this};};
```

3

```
let myStr = "Elzero";
```

[myStr.addDot] \Rightarrow .Elzero.

Object Meta & Descriptor Part 1

157

Let's see properties of my obj this

```
const myObj = {
```

```
  a: 1,
```

```
  b: 2
```

```
};
```

```
Object.defineProperty(myObj, "c", {
```

```
  writable: true, ← ①
```

```
  enumerable: true, ← ②
```

```
  configurable: true, ← ③
```

```
  value: 3, ← ④
```

```
});
```

```
for (let prop in myObj) {
```

```
  console.log(prop, myObj[prop])
```

```
  }
```

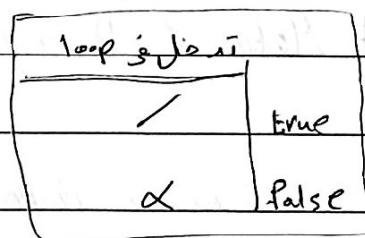
a 1

b 2

c 3

C: انواع

✓	true	لئے : ①	↳ false پریسی دلٹ، ③، ②، ① تکمیل
✗	false	لئے : ①	



defineProperty

✓	true	لئے : ③
✗	false	

Object Meta Data & Descriptor Part-2

158

والفروضی کے قابل محتوا کی طرف سے اس کا انتشار کیا جائے کہ اس کا انتشار کیا جائے

Object.defineProperties (myObj, {

c: {

configurable: true,

value: 3,

},

d: {

configurable: true

value: 4,

},

e:

Object.getOwnPropertyDescriptor (myObj, "c")

Descriptors (myObj);