Inheritance in JavaScript

Inheritance is an important concept in object oriented programming. In the classical inheritance, methods from base class get copied into derived class.

In JavaScript, inheritance is supported by using prototype object. Some people call it "Prototypal Inheriatance" and some people call it "Behaviour Delegation".

function Person(firstName, lastName) {

this.FirstName = firstName || "unknown";

this.LastName = lastName || "unknown";

};

Person.prototype.getFullName = function () {

return this.FirstName + " " + this.LastName;

}

In the above example, we have defined Person class (function) with FirstName & LastName properties and also added getFullName method to its [prototype](https://www.tutorialsteacher.com/javascript/prototype-in-javascript) object.

Now, we want to create Student class that inherits from Person class so that we don't have to redefine FirstName, LastName and getFullName() method in Student class. The following is a Student class that inherits Person class.

function Student(firstName, lastName, schoolName, grade)

{

Person.call(this, firstName, lastName);

this.SchoolName = schoolName || "unknown";

this.Grade = grade || 0;

}

//Student.prototype = Person.prototype;

Student.prototype = new Person();

Student.prototype.constructor = Student;

Please note that we have set Student.prototype to newly created person object. The new keyword creates an object of Person class and also assigns Person.prototype to new object's prototype object and then finally assigns newly created object to Student.prototype object. Optionally, you can also assign Person.prototype to Student.prototype object.

Now, we can create an object of Student that uses properties and methods of the Person as shown below.

function Person(firstName, lastName) {

this.FirstName = firstName || "unknown";

this.LastName = lastName || "unknown";

}

Person.prototype.getFullName = function () {

return this.FirstName + " " + this.LastName;

}

function Student(firstName, lastName, schoolName, grade)

{

Person.call(this, firstName, lastName);

this.SchoolName = schoolName || "unknown";

this.Grade = grade || 0;

}

//Student.prototype = Person.prototype;

Student.prototype = new Person();

Student.prototype.constructor = Student;

var std = new Student("James","Bond", "XYZ", 10);

alert(std.getFullName()); // James Bond

alert(std instanceof Student); // true

alert(std instanceof Person); // true