**일일교육내용**

**일자 : 2019.05.27**

|  |  |
| --- | --- |
| **교육과목** | 자바스크립트 |
| **교육내용** | 상속  1. 객체생성자의 재정의는 재정의 전 객체생성자를 통해 생성된 객체에 영향을 미치지 못함  2. prototype으로 기능 추가  3. 하지만 prototype으로 생성자의 속성 또는 메소드를 역으로 변경할 수 없음 |
| **실습 내용** | **문제1. 아래와 같은 Car 객체를 정의하고 메소드를 추가합니다**  **1) startEngine**  **=> isDriving true로 변경**  **=> fuel -5**  **document.writeln 으로 시동후 연료상태와 isDriving 상태를 풀력합니다**  **2) accelerate**  **=> velocity 5 증가;**  **=> fuel 5 감소;**  **=> 속도와 연료 상태 출력**  **3) decelerate**  **=> velocity 5 감소;**  **=> fuel 1 감소;**  **=> 속도와 연료 상태 출력**  **4) stopEngine**  **=> 주행상태 변경 false**  **=> 연료 감소 5**  **=> 주행상태와 연료상태 출력**  var Car = function(f){  this.fuel = f;  this.velocity = 0;  this.isDriving = false;  };  Car.prototype.startEngine = function () {  this.isDriving = true;  this.fuel -= 5;  document.writeln('startEngine<br/>');  document.writeln('isDriving: '+ this.isDriving+'<br/>');  document.writeln('fuel: '+this.fuel+'<br/><br/>');  }  var mCar1 = new Car(30);  document.writeln('fuel of my car: '+mCar1.fuel+'<br/>');  mCar1.startEngine();  Car.prototype.accelerate = function () {  this.velocity +=5;  this.fuel -= 5;  document.writeln('accelerate<br/>');  document.writeln('velocity: '+this.velocity+'<br/>');  document.writeln('fuel: '+this.fuel+'<br/><br/>');  }  mCar1.accelerate();  Car.prototype.decelerate = function () {  this.velocity -=5;  this.fuel -= 1;  document.writeln('decelerate<br/>');  document.writeln('velocity: '+this.velocity+'<br/>');  document.writeln('fuel: '+this.fuel+'<br/><br/>');  }  mCar1.decelerate();  Car.prototype.stopEngine = function () {  this.isDriving = false;  this.fuel -= 1;  document.writeln('stopEngine<br/>');  document.writeln('isDriving: '+this.isDriving+'<br/>');  document.writeln('fuel: '+this.fuel+'<br/><br/>');  }  mCar1.stopEngine();  **문제2.**  **Car 객체를 이용해 상속을 진행해 K5를 만들고 property추가, 메소드의 변경과 추가를 진행합니다.**  **property추가: model**  **메소드 변경내역: 가속과 감속시 속도의 변화를 5에서 10으로 변경합니다.**  **추가내역: 네비게이션의 시작과 종료 기능을 추가합니다. (호출되면 네비게이션의 시작과 종료을 알리는 기능만 탑재)**  **상속한 K5를 생성자로 객체 k5를 생성하고 ==> var k5 = new K5(1000, '2013년형');**  var Car = function (f) {  this.fuel = f; // 연료  this.velocity = 0; // 속도  this.isDriving = false; // 주행 상태  this.startEngine = function () {  this.isDriving = true;  this.fuel -= 5;  document.writeln('Car startEngine<br/>');  };  this.accelerate = function () {  this.velocity += 5;  this.fuel -= 5;  document.writeln('Car: 속도를 올립니다++++ (velocity: ' + this.velocity + ', fuel: ' + this.fuel + ')<br/>');  };  this.decelerate = function () {  this.velocity -= 5;  this.fuel -= 1;  document.writeln('Car: 속도를 줄입니다---- (velocity: ' + this.velocity + ', fuel: ' + this.fuel + ')<br/>');  };  this.stopEngine = function () {  this.isDriving = false;  this.fuel -= 1;  document.writeln('k5.fuel: ' + this.fuel + '<br/>');  document.writeln('Car stop Engine<br/>');  document.writeln('k5.fuel: ' + this.fuel + '<br/>');  };  };  Car.prototype.startNavigation = function () {  console.log("Car start Navigation");  };  Car.prototype.stopNavigation = function () {  console.log("Car stop Navigation");  };  var K5 = function (fuel, model) {  Car.apply(this, [fuel]);  this.model = model;  };  K5.prototype = new Car();  //K5.prototype 프로퍼티가 new Car()를 참조하면  //Car() 생성자 함수의 프로토타입 멤버를 상속하게 됩니다.  K5.prototype.constructor = K5;  delete K5.prototype.fuel;  delete K5.prototype.velocity;  delete K5.prototype.isDriving;  K5.prototype.accelerate = function () {  this.velocity += 10;  this.fuel -= 5;  document.writeln('K5 ' + this.model + ': 속도를 올립니다++++ (velocity: ' + this.velocity + ', fuel: ' + this.fuel + ')<br/>');  };  K5.prototype.decelerate = function () {  this.velocity -= 10;  this.fuel -= 1;  document.writeln('K5 ' + this.model + ': 속도를 줄입니다---- (velocity: ' + this.velocity + ', fuel: ' + this.fuel + ')<br/>');  };  K5.prototype.startNavigation = function () {  document.writeln('K5 ' + this.model + ': 네비게이션 안내를 시작합니다...<br/>');  };  K5.prototype.stopNavigation = function () {  document.writeln('K5 ' + this.model + ': 네비게이션 안내를 종료합니다...<br/>');  };  var k5 = new K5(1000, '2013년형');  console.log("model: "+k5.model);  k5.startNavigation();  k5.stopEngine();  **문제3. 생성한 k5를 이용해**  **기능을 테스트하여 화면과 같은 결과를 출력합니다.**  var k5 = new K5(1000, '2013년형');  document.writeln('>>> k5.fuel: ' + k5.fuel + '<br/>');  k5.startEngine();  document.writeln('>>> k5.fuel: ' + k5.fuel + '<br/>');  k5.startNavigation();  for (var i = 0; i < 5; i++)  k5.accelerate();  document.writeln('>>> k5.fuel: ' + k5.fuel + '<br/>');  for (var i = 0; i < 5; i++)  k5.decelerate();  document.writeln('>>> k5.fuel: ' + k5.fuel + '<br/>');  k5.stopNavigation();  document.writeln('>>> k5.fuel: ' + k5.fuel + '<br/>');  k5.stopEngine();  document.writeln('>>> k5.fuel: ' + k5.fuel + '<br/>');  **4. 빈 Car 객체 생성자를 정의합니다.**  **startEngine, accelerate, decelerate, stopEngine 의 4개 메소드를 추가합니다.**  **이때 각 메소드는 자신의 역할만 출력합니다 (예: 시동을 켭니다. 가속합니다...)**  **빈 객체생성자 K5를 만듭니다. 상속을 진행합니다**  **1. 프로토타입상속**  **2. prototype.constructor 변경**  **3. startNavigation 추가**  **4. stopNavigation 추가**  var Car = function(){ };  Car.prototype.startEngine = function () {  document.writeln('startEngine<br/>');  };  Car.prototype.startEngine();  Car.prototype.accelerate = function () {  document.writeln('accelerate<br/>');  };  Car.prototype.accelerate();  Car.prototype.decelerate = function () {  document.writeln('decelerate<br/>');  };  Car.prototype.decelerate();  Car.prototype.stopEngine = function () {  document.writeln('stopEngine<br/>');  };  Car.prototype.stopEngine();  var K5 = function () { };  K5.prototype = new Car();  K5.prototype.constructor = K5;  K5.prototype.startNavigation = function () {  document.writeln("K5 startNavigation<br/>");  };  K5.prototype.stopNavigation = function () {  document.writeln("K5 stopNavigation<br/>");  };    document.writeln("<br/>K5<br/>");  var k5 = new K5();  k5.startEngine(); // startEngine  k5.startNavigation(); // startNavigation  k5.accelerate(); // accelerate  k5.decelerate(); // decelerate  k5.stopNavigation(); // v  k5.stopEngine(); // stopEngine  **5. 각 조마다 상속 예제 만들기**  var Coffee = function (\_name, \_ih) { // 커피  this.name = \_name;  this.price = 4200;  this.ih = \_ih;  this.shot = 2;  this.option = undefined;  }  Coffee.prototype.addShot = function (\_shots) { // 샷추가  this.shot += \_shots;  this.price += (\_shots \* 200);  }  Coffee.prototype.printMenu = function () { // 현재 메뉴 출력  document.writeln(this.name + "(" + this.price + ")<br/>");  }  var Americano = function (\_ih) { // 아메  Coffee.call(this, "Americano", \_ih);  }  Americano.prototype = Object.create(Coffee.prototype);  Americano.prototype.constructor = Americano;  var Latte = function (\_ih) { // 라떼  Coffee.call(this, "Latte", \_ih);  this.price += 500;  this.sylup = false;  }  Latte.prototype = Object.create(Coffee.prototype);  Latte.prototype.constructor = Latte;  Latte.prototype.addSylup = function () { // 라떼 시럽추가  this.sylup = true;  this.price += 200;  }  var Moka = function (\_ih) { // 모카  Coffee.call(this, "Moka", \_ih);  this.price += 1000;  this.sylup = true;  this.whipping = false;  }  Moka.prototype = Object.create(Coffee.prototype);  Moka.prototype.constructor = Moka;  Moka.prototype.addWhipping = function () { // 모카 휘핑추가  this.whipping = true;  this.price += 300;  };  var cafeMoka = new Moka('ICE');  cafeMoka.printMenu();  cafeMoka.addWhipping();  cafeMoka.printMenu();  var cafeLatte = new Latte('HOT');  cafeLatte.printMenu();  cafeLatte.addSylup();  cafeLatte.printMenu();  var ame = new Americano('ICE');  ame.printMenu();  ame.addShot(1);  ame.printMenu(); |
| **기타** |  |