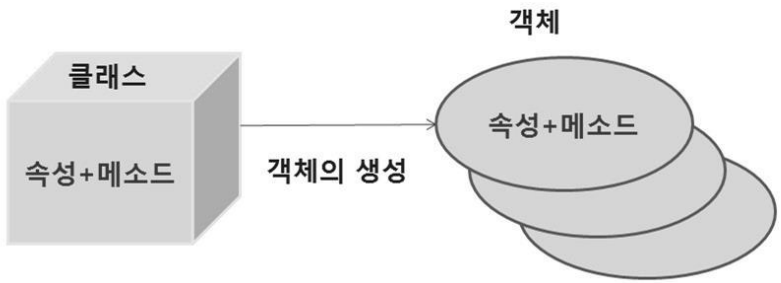
JavaScript 객체

사용자정의객체

객체의 정의

- 자바스크립트에서는 윈도우즈와 버튼, 폼 그리고 이미지 등이 객체와 연결되어 있음
- 완전한 객체 지향 언어는 아니므로 객체 기반 언어라고 부르지만 다양하게 객체 지향언어의 기능을 많이 수행
- 자바스크립트에서 객체의 종류
  - 1. 사용자 정의 객체
  - 2. 내장 객체 (미리 작성되어 있는 객체)
    - Date, String, Number 등
  - 3. BOM관련 객체 (Browser 객체)
  - 4. DOM관련 객체 (document 객체)
- 객체(object): 실세계를 표현하고 프로그램 내에서 작업을 용이하도록 만들어 주는 프로그래밍 구조
  - 구성:속성(Property)- 고유한 식별자(identifier), 메소드(Method)- 행위(behavior)
- 클래스(Class): 객체를 생성해 주는 틀(Frame)



객체의 정의

- 속성
  - 변수, 데이터 멤버, 혹은 필드라고도 부르며, 객체의 특성(상태)
  - 객체명.속성 형식으로 사용됨, 객체명['속성']으로도 사용 가능

```
Worker.ShowEmloyee = 200;
```

객체명 속성 데이터 값

- 메소드: 클래스 행위 정의
  - 메소드명 뒤에 ()를 반드시 붙여서 나타내야 하며, 괄호 안에 인수를 포함
  - function이라는 예약어로 정의

```
function 함수명(가인수1, --- ,가인수n) {
  함수몸체;
}
```

메소드 호출할 때 인수: 실인수(Actual argument)
function으로 메소드 정의할 때 인수: 가인수(Dummy argument)

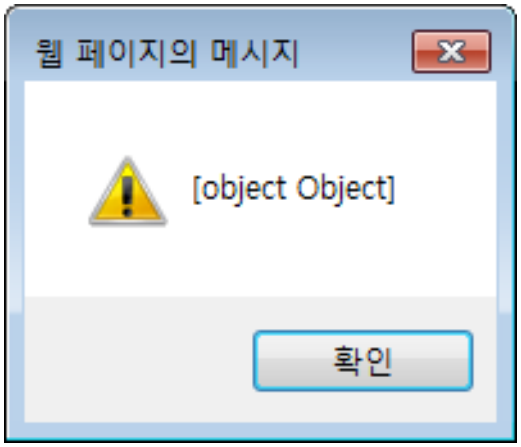
객체의 작성

- 1. Object 함수를 이용한 객체의 작성
  - 하나의 생성자로 여러 개의 객체를 생성할 수 있음
  - 생성자는 특별한 기능 객체를 작성의 객체의 설계도를 의미
  - new 키워드는 객체를 작성하는 생성자 앞에 나타냄

var 객체명 = new Object(인수1, 인수2, ...)

var cat = new Object(); // var cat = {};

```
<html>
  <head>
    <title>Object() 생성자</title>
    <script>
            var cat = new Object();
            alert(cat);
    </script>
  </head>
  <body>
  </body>
</html>
```



객체의 작성

- 1. Object 함수를 이용한 객체의 작성
  - 객체 속성 정의

cat.name = "야옹이";
cat.color="노랑";

```
<html>
  <head>
    <title>사용자 정의 객체</title>
    <script>
      var toy = new Object(); // var toy = {};
      toy.name = "레고";
      toy.color = "빨강";
      toy.shape = "직사각형";
     </script>
  </head>
  <body bgcolor="lightblue">
    <script>
      document.write("<br/>장난감은  " + toy.name + " 입니다.");
      document.write("<br/>색상은 " + toy.color
                    + "이고,  모양은  " + toy.shape+ "입니다.");
    </script>
  </body>
</html>
```

장난감은 레고 입니다.
색상은 빨강이고, 모양은 직사각형입니다.

객체의 작성

- 1. Object 함수를 이용한 객체의 작성
  - 객체의 메소드 정의

```
cat.play();
dog.fetch("ball");

window.close();
document.write("Hello\n");
```

```
<html>
  <head>
    <title>사용자 정의 객체</title>
    <script>
      var toy = new Object();   // 객체의 생성
      toy.name = "레고";        // 객체에 속성을 할당
      toy.color = "빨강";
      toy.shape = "직사각형";
      toy.display=printObject;
      function printObject() {
          document.write("<br/>장난감은  " + toy.name + "입니다.<br/>");
          document.write("색상은  " + toy.color + " 이고, 모양은  "
                                + toy.shape+ "입니다.<br/><br/>");
      }
     </script>
  </head>
```

객체의 작성

- 1. Object 함수를 이용한 객체의 작성
  - 객체의 메소드 정의

```
<body>
    <script>
        toy.display();  // 객체의 메소드 호출
        toy.color="파랑";
        toy.display();
    </script>
</body>
</html>
```

```
장난감은 레고입니다.
색상은 빨강 이고, 모양은 직사각형입니다.

장난감은 레고입니다.
색상은 파랑 이고, 모양은 직사각형입니다.
```

객체의 작성

- 2. function 내부에서 this 사용하여 객체 작성
  - this: 현재의 객체를 참조하는 예약어/축약어

```
<script>
  function Book(){   // Book 클래스의 생성
    this.title = "자바스크립트 프로그래밍";   // 속성
    this.author = "홍길동";
  }
  var bookObj = new Book();   // Book 객체의 작성
  alert(bookObj.title + " 와 " + bookObj.author);
</script>
```

객체의 작성

- 2. function 내부에서 this 사용하여 객체 작성

```
<HTML>
  <HEAD>
    <TITLE> 사용자 정의객체  선언 및 이용  </TITLE>
    <SCRIPT>
       function Display() {
        document.writeln("이름:"+ this.name
                                    + "***전공:***"+this.major);
        document.writeln("학번:"+ this.hakbun
                                     + "***나이:***"+this.age+"<BR/>");
       }
       function Pinfor(pname,pmajor,phakbun,page) {
         this.name=pname;
         this.major=pmajor;
         this.hakbun=phakbun;
         this.age=page;
         this.print=Display;
       }
    </SCRIPT>
  </HEAD>
```

객체의 작성

- 2. function 내부에서 this 사용하여 객체 작성

```
<BODY>
   <h3>사용자 정의객체  선언 </h3>
   <hr/>
   <SCRIPT>
      p1=new Pinfor("홍길동","전산학",20111234,33);
      p2=new Pinfor("이순신","통계학",20113456,20);
      p3=new Pinfor("강감찬","물리학",20117890,25);
      p1.print();
      p2.print();
      p3.print();
   </SCRIPT>
 </BODY>
</HTML>
```

사용자 정의객체 선언

이름:홍길동***전공:***전산학 학번:20111234***나이:***33
이름:이순신***전공:***통계학 학번:20113456***나이:***20
이름:강감찬***전공:***물리학 학번:20117890***나이:***25

객체의 작성

- 2. function 내부에서 this 사용하여 객체 작성

```
<html>
  <head>
    <title>사용자 정의객체  선언 및 이용</title>
     <script>
          function book(title, author, publisher){
             this.pagenumber = 0;
             this.title = title;
             this.author = author;
             this.publisher = publisher;
             this.pageForward = pageForward;
             this.pageBackward = pageBackward;
          }
          function pageForward(){
             this.pagenumber++;
             return this.pagenumber;
          }
          function pageBackward(){
             this.pagenumber--;
             return this.pagenumber;
          }
      </script>
```

객체의 작성

- 2. function 내부에서 this 사용하여 객체 작성

```
</head>
<body>
  <script>
      var myBook = new book("자바스크립트 프로그래밍",  " 강양원, ",  " 조범석" );
      myBook.pagenumber = 5;
      document.write( "<br/>"+ myBook.title + "<br/>" + myBook.author +
                          myBook.publisher +  "<br/>" +
                          "<br/>현재 페이지는 " + myBook.pagenumber );
      document.write("<br>다음 페이지 : " );
      for(i=0;i<3;i++){
         document.write("<br/>" + myBook.pageForward());
      }
      document.write("<br/>이전 페이지 : ");
      for(;i>0; i--){
         document.write("<br/>" + myBook.pageBackward());
      }
  </script>
</body>
</html>
```

JavaScript & jQuery
강양원, 조범석

현재 페이지는 5
다음 페이지 :
6
7
8
이전 페이지 :
7
6
5

객체의 작성

- 2. function 내부에서 this 사용하여 객체 작성
  - 이름없는 function의 메소드이용

```
this.calculate = function() { return r * t; }
// 이름없는 function의 할당
```

```
<html> <head><title>functions</title>
  <script>
    function Distance(r, t){ //Constructor function
      this.rate = r;    this.time = t;
      this.calculate = function(){ return r * t; } // anonymous
    }
  </script> </head>
  <body>
    <script>
      var trip1 = new Distance(50, 1.5);
      var trip2 = new Distance(75, 3.2);
      document.write("첫번째 여행 결과 : " + trip1.calculate() + "</p>");
      document.write("두번째 여행 결과 : " + trip2.calculate() + "</p>");
    </script>
  </body>
</html>
```

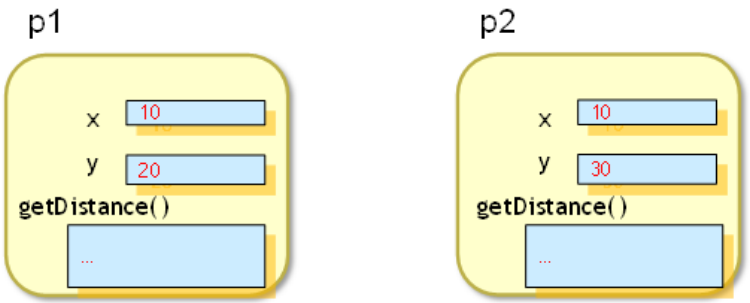
첫번째 여행 결과 : 75

두번째 여행 결과 : 240

프로토타입

- 자바스크립트에서 메소드를 여러 객체가 공유하려면 어떻게 해야 하는가?
- 현재는 메소드를 공유할 수 없다.

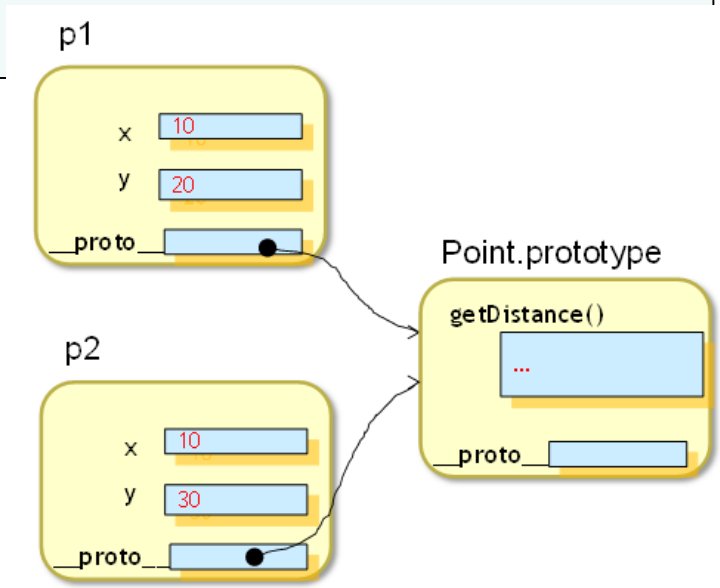
```
function Point(xpos, ypos) {
   this.x = xpos;
   this.y = ypos;
   this.getDistance = function () {
            return Math.sqrt(this.x * this.x + this.y * this.y);
   };
}
var p1 = new Point(10, 20);
var p2 = new Point(10, 30);
```



프로토타입

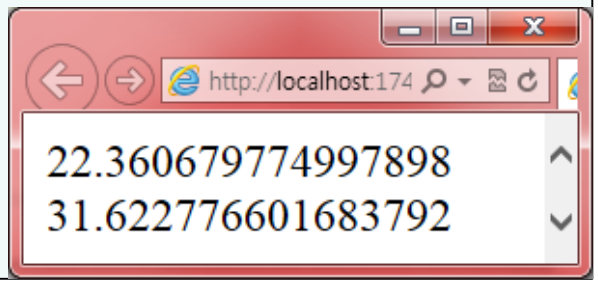
- 자바스크립트의 모든 객체들은 prototype이라는 숨겨진 객체를 가지고 있으며 이 객체를 이용하여서 공유되는 메소드를 작성할 수 있다.

```
function Point(xpos, ypos) {
   this.x = xpos;
   this.y = ypos;
}

Point.prototype.getDistance = function () {
   return Math.sqrt(this.x * this.x + this.y * this.y);
};
```



프로토타입 예제

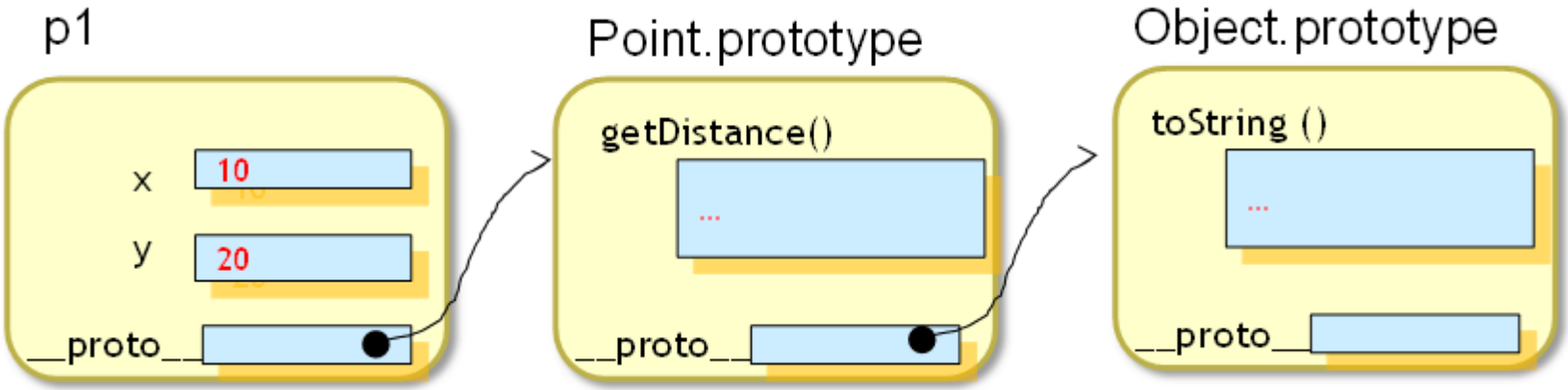
```
<!DOCTYPE html>
<html>
<body>
   <script>
      function Point(xpos, ypos) {
         this.x = xpos;
         this.y = ypos;
      }
      Point.prototype.getDistance = function (p) {
         return Math.sqrt(this.x * this.x + this.y * this.y);
      }
      var p1 = new Point(10, 20);
      var d1 = p1.getDistance();
      var p2 = new Point(10, 30);
      var d2 = p2.getDistance();
      document.writeln(d1 + "<br />");
      document.writeln(d2 + "<br />");

   </script>
</body>
</html>
```



프로토타입 체인

- 자바스크립트에서 속성이나 메소드를 참조하게 되면 다음과 같은 순서대로 찾는다.

1. 객체 안에 속성이나 메소드가 정의되어 있는지 체크한다.
2. 객체 안에 정의되어 있지 않으면 객체의 prototype이 속성이나 메소드를 가지고 있는지 체크한다.
3. 원하는 속성/메소드를 찾을 때까지 프로토타입 체인(chain)을 따라서 올라간다.



객체의 작성

- 3. JSON을 이용한 객체 작성

```
클래스이름 = function(파라미터) {   // 생성자 정의
   …
}
```

```
클래스이름.prototype ={
   메소드명1:function(파라미터1){
       …
   },
   메소드명2:function(파라미터2){
      …
   }
}
```

객체의 작성

- 3. JSON을 이용한 객체 작성

```
Member = function(name, id, securityNo) {
  this.name = name;
  this.id = id;
  this.securityNo = securityNo;
}
Member.prototype = {
  setValue: function(newName, newId, newSecurityNo) {
    this.name = newName;
    this.id = newId;
    this.securityNo = newSecurityNo;
  },
  getAge: function() {
    var birthYear = parseInt(this.securityNo.substring(0, 2));
    var code = this.securityNo.substring(6,7);
    if (code == '1' || code == '2') {
      birthYear += 1900;
    } else if (code == '3' || code == '4') {
      birthYear += 2000;
    }
    var today = new Date();
    return today.getFullYear() - birthYear;
  },
  toString: function() {
    return this.name + "[" + this.id + "]";
  }
}
```

객체의 작성

- 3. JSON을 이용한 객체 작성
  - 이름 충돌 문제 발생: 패키지로 해결

```
var 변수명 = new Object(); // 변수명을 패키지로 인식
변수명.클래스명 = function(파라미터) {   // 생성자 정의
   …
}
변수명.클래스명.prototype ={
   메소드명1:function(파라미터1){
      …
   },
   메소드명2:function(파라미터2){
      …
   }
}
```

```
var 변수명= {};                    //JSON을 이용하여 정의
```

객체의 작성

- 3. JSON을 이용한 객체 작성
  - 중첩패키지 정의

```
var 변수명1= new Object(); // 변수명을 패키지로 인식
var 변수명2= new Object();
변수명1.변수명 2.클래스명 = function(파라미터) {   // 생성자 정의
   …
}
변수명1.변수명 2.클래스명.prototype ={
   메소드명1:function(파라미터1){
        …
   },
   메소드명2:function(파라미터2){
       …
   }
}
```

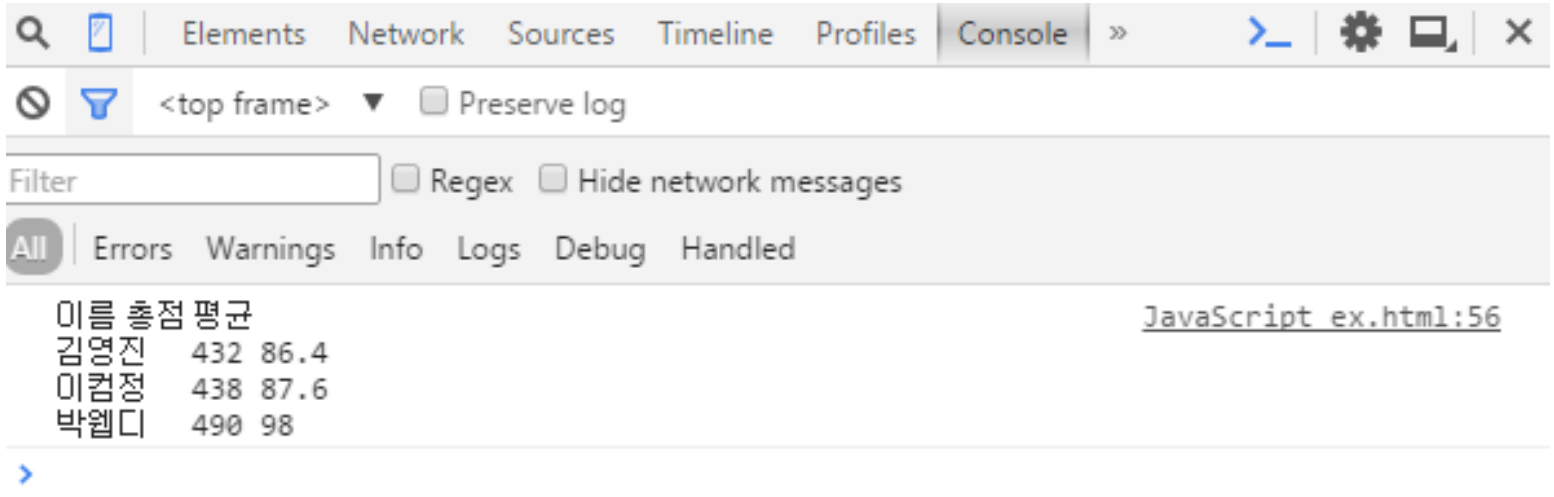
```
var 변수명1= {};              //JSON을 이용하여 정의
변수명1.변수명2 ={};
```

객체의 작성

```
<script>
  Student =function(name, korean, math, english, society, science){
    this.이름=name; this.언어= korean; this.수리= math;
    this.외국어= english; this.사탐= society; this.과탐= science;
  }
  Student.prototype={
    getSum:function(){
      return this.언어+this.수리+this.외국어+this.사탐+this.과탐;
    },
    getAverage:function(){
      return this.getSum()/5;
    },
    toString:function(){
      return this.이름+ '₩t' + this.getSum() + '₩t'+ this.getAverage();
    }
  };
  var students=[];
  students.push(new Student('김영진',85,92,82,78,95));
  students.push(new Student('이컴정',82,84,90,92,90));
  students.push(new Student('박웹디',96,98,100,96,100));
```

객체의 작성

```
  var output = '이름₩t총점₩t평균₩n';
  for(var i in students){
      output+=students[i].toString()+'₩n';
  }
  console.log(output);
</script>
```



객체의 응용

- with 문
  - 객체의 속성 또는 메소드를 참조할 때 줄여 입력할 수 있도록 하기 위해 사용

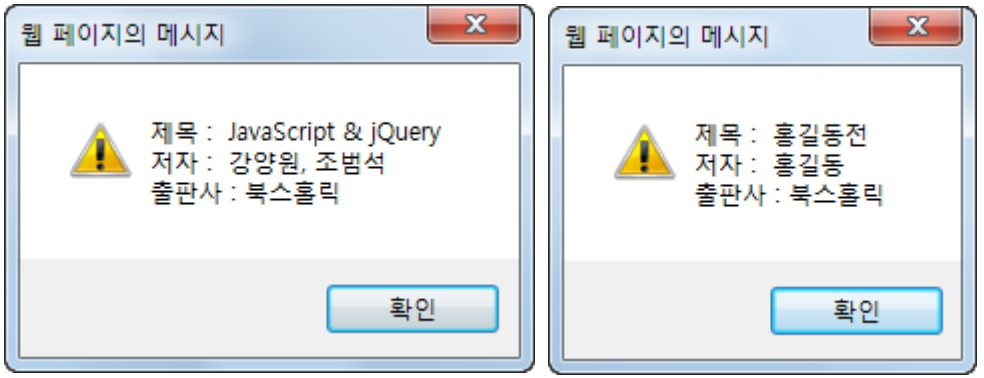
```
with (객체){
  < 객체명 또는 점이 없는 속성명>
}

with(employee){
  document.write(name, ssn, address); // employee.name
}
```

```
<html>
  <head>
    <title>The with Keyword</title>
    <script>
      function book(title, author, publisher){
        this.title = title;   // Properties
        this.author = author;
        this.publisher = publisher;
        this.show = show;  // Define a method
      }
```

객체의 응용

- with 문

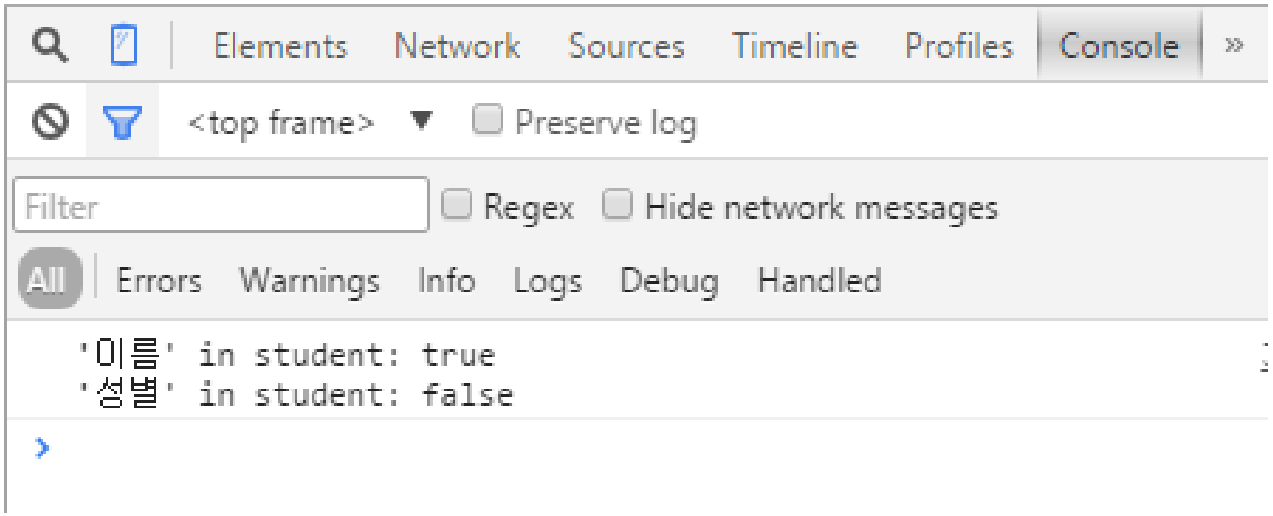
```
        function show() {
            with(this) {    // The with keyword with this
                var info = "제목 :  " + title;
                info += "₩n저자 :  " +  author;
                info += "₩n출판사 : " + publisher;
                alert(info);
            }
        }
    </script>
  </head>
<body bgcolor="lightblue">
    <script>
        var childbook = new book("홍길동전","홍길동", "북스홀릭");
        var adultbook =new book("JavaScript & jQuery",
                                "강양원, 조범석", "북스홀릭");
        childbook.show();
        adultbook.show();
    </script>
  </body>
</html>
```



객체의 응용

- in문

```
<script>
   var output = '';
   var student = {
         이름: '김영진',
         국어: 94, 영어: 85, 수학: 90, 사회: 85
   };
   output += "'이름' in student: " +('이름' in student) + "₩n";
   output += "'성별' in student: " +('성별' in student);
   console.log(output);
</script>
```



객체의 응용

- for/in 반복문
  - 객체 속성 또는 배열 요소를 반복하여 내용을 참조할 수 있도록 하는 반복문

```
for(var 속성변수 in 객체){
   문장;
}
```

```
<html>
  <head> <title>for/in 문</title>
    <script>
      function book(title, author, publisher) {
            this.title = title;
            this.author = author;
            this.publisher = publisher;
            this.show = show;
      }
      function show(obj, name) {
            var result = "";
            for(var prop in obj) {
               result += name + "." + prop + " = " + obj[prop] + "<br>";
            }
            return result;
      }
```

객체의 응용

- for/in 반복문

```
 </script>
  </head>
  <body bgcolor="lightblue">
    <script>
       myBook =new book("자바스크립트 프로그래밍", "강양원, 조범석", "북스홀릭");
       document.write("<br/><br/>" + myBook.show(myBook, "myBook"));
    </script>
  </body>
</html>
```

```
myBook.title = 자바스크립트 프로그래밍
myBook.author = 강양원, 조범석
myBook.publisher = 북스홀릭
myBook.show = function show(obj, name) { //
Function to show the object's properties var result =
""; for (var prop in obj) { result += name + "." + prop + "
= " + obj[prop] + "
"; } return result; }
```