

**01**

**자바스크립트 객체  
이해하기**

# 01 자바스크립트 객체 이해하기

## ■ 자바스크립트 객체

- 자바스크립트를 이루는 거의 모든 것이 객체(object)

```
getTime() { . . . }  
setTime(h, m, s) { . . . }  
  
let h, m, s;
```

(a) 일반 자바스크립트로 작성한 시계 코드

Clock 객체

```
class Clock() {  
  let h, m, s;  
  
  getTime() { . . . }  
  setTime(h, m, s) { . . . }  
}
```

(b) Clock 객체를 사용하는 코드

서로 관련 있는 변수와 함수를  
객체로 묶어놓으니 편리하겠어.



그림 7-1 일반 코드와 객체를 사용하는 코드

# 01 자바스크립트 객체 이해하기

## ■ 자바스크립트 객체

### 1. 객체의 장점

- 코드를 상속하거나 재사용하여 간단하게 구현

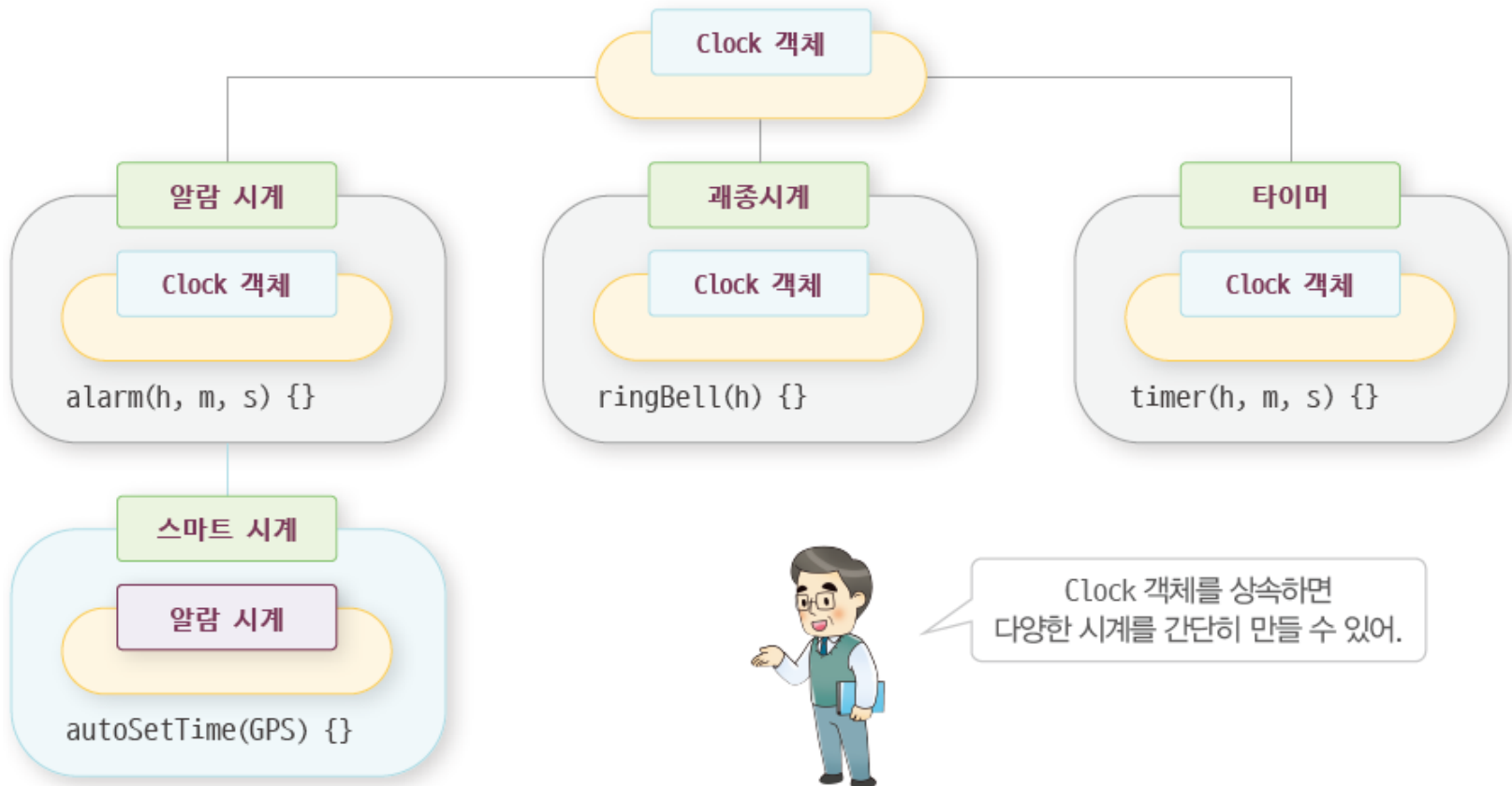


그림 7-2 Clock 객체를 상속하는 다양한 시계 만들기

# 01 자바스크립트 객체 이해하기

## ■ 자바스크립트 객체

### 2. 자바스크립트 객체의 구성

- 자바스크립트 객체는 데이터를 의미하는 속성(property)과 데이터를 조작하는 메서드로 구성된 집합



객체.속성에  
값을 담으면 돼.

`object.property = value`

(a) 속성



괄호가 있으면  
메서드야.

`object.method(value)`

(b) 메서드

그림 7-3 자바스크립트 객체의 구성

# 01 자바스크립트 객체 이해하기

## ■ 자바스크립트 객체

### 2. 자바스크립트 객체의 구성

표 7-1 자바스크립트 객체의 메서드

메서드	내용
<code>document.write()</code>	document 객체의 write() 메서드
<code>Math.floor()</code>	Math 객체의 floor() 메서드
<code>Math.random()</code>	Math 객체의 random() 메서드

표 7-2 객체의 속성값 변경

<code>style.color = "red"</code>	style 객체 color 속성을 red로 변경
<code>style.background = "aqua"</code>	style 객체 background 속성을 aqua로 변경

# 01 자바스크립트 객체 이해하기

## ■ 자바스크립트 객체

### 3. 객체 선언과 객체 생성

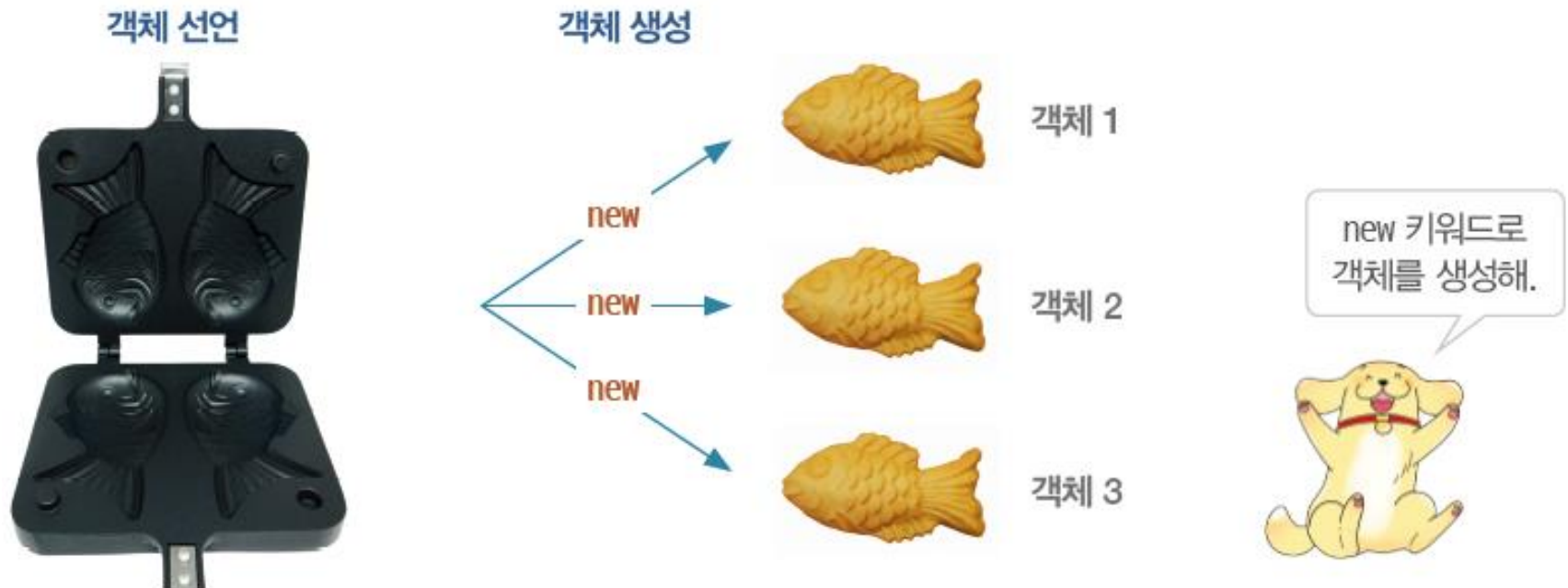


그림 7-4 객체 선언과 객체 생성

# 01 자바스크립트 객체 이해하기

## ■ 자바스크립트 객체

### 4. 전역객체

- HTML에서 최상위 객체는 window

- 전역객체 메서드

표 7-3 전역객체 메서드

전역객체 메서드	객체 이름 생략	내용
<code>window.alert()</code>	<code>alert()</code>	알림 메시지
<code>window.prompt()</code>	<code>prompt()</code>	사용자로부터 입력
<code>window.confirm()</code>	<code>confirm()</code>	확인/취소 버튼(확인=true, 취소=false)
<code>window.eval()</code>	<code>eval()</code>	수식을 계산하여 결과를 반환
<code>window.parseInt()</code>	<code>parseInt()</code>	정수로 변환
<code>window.parseFloat()</code>	<code>parseFloat()</code>	실수로 변환
<code>window.isNaN()</code>	<code>isNaN()</code>	숫자이면 false, 숫자가 아니면 true

# 01 자바스크립트 객체 이해하기

## ■ 자바스크립트 객체

### 4. 전역객체

- isNaN() 메서드

표 7-4 isNaN() 메서드

입력값	반환값	설명
isNaN(1)	false	숫자이면 false
isNaN("0.5")	false	숫자로 변환할 수 있으면 false
isNaN("str")	true	숫자로 변환할 수 없으면 true



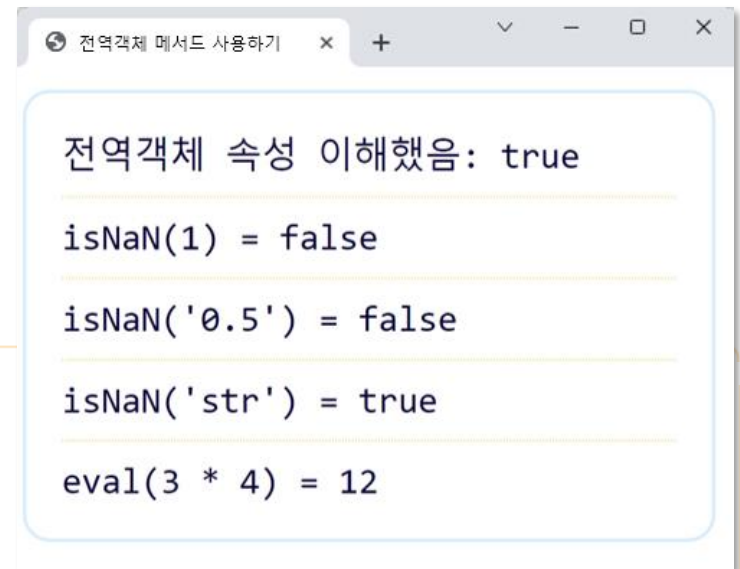
# 01 자바스크립트 객체 이해하기

- 전역객체 메서드 사용하기

## 예제 7-1

### 전역객체 메서드 사용하기

```
</head>
<body>
  <script>
    const jud = window.confirm("confirm()에서 window.가 생략된 것을 이해했나요?");
    document.write("전역객체 속성 이해했음: " + jud + "<hr>");
    document.write("isNaN(1) = " + isNaN(1) + "<hr>");
    document.write("isNaN('0.5') = " + isNaN('0.5') + "<hr>");
    document.write("isNaN('str') = " + isNaN('str') + "<hr>");
    document.write("eval(3 * 4) = " + eval(3 * 4));
  </script>
</body>
</html>
```



# 01 자바스크립트 객체 이해하기

## ■ 배열과 Array 객체

### 1. 배열

- 배열 생성하기

```
let arr = [10, 20, 30, 40, 50];  
document.write(arr);
```



10,20,30,40,50

```
let arr = [1, "첫번째", 2.34, true];  
document.write(arr);
```



1,첫번째,2.34,true

# 01 자바스크립트 객체 이해하기

## ■ 배열과 Array 객체

### 1. 배열

- 배열 생성하기

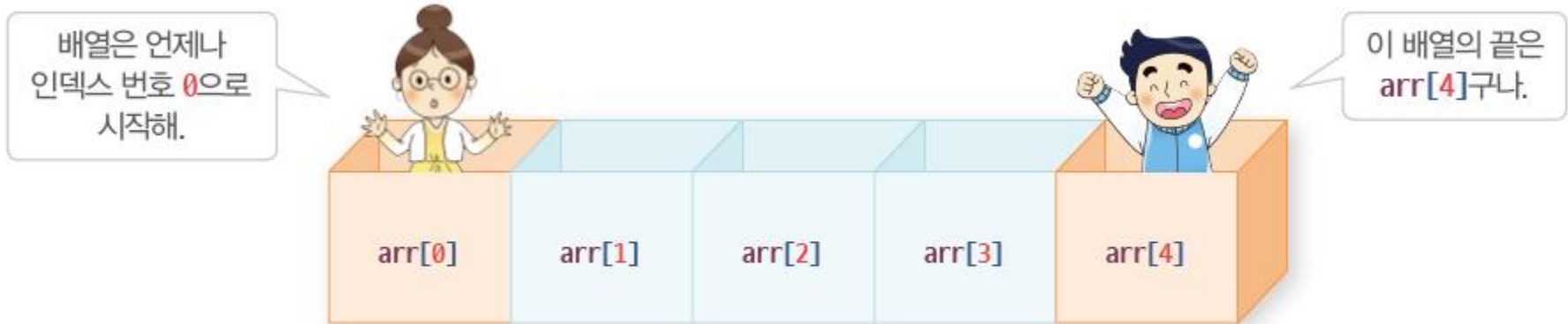


그림 7-5 원소가 5개인 배열

# 01 자바스크립트 객체 이해하기

## ■ 배열과 Array 객체

### 1. 배열

- 배열 원소에 접근하기

```
let arr = [10, 20, 30, 40, 50];  
for (let i = 0; i < 5; i++)  
  document.write "[" + i + "]=" + arr[i] + " ");
```



[0]=10 [1]=20 [2]=30 [3]=40 [4]=50

# 01 자바스크립트 객체 이해하기

## ■ 배열과 Array 객체

### 1. 배열

- 배열 원소에 접근하기

```
let arr = [10, 20, 30, 40, 50];  
for (let i = 0; i < 5; i++){  
  arr[i] = arr[i] + 5;  
  document.write "[" + i + "]=" + arr[i] + " ");  
}
```



[0]=15 [1]=25 [2]=35 [3]=45 [4]=55

# 01 자바스크립트 객체 이해하기

## ■ 배열과 Array 객체

### 1. 배열

- 빈 배열 선언하기

```
let arr = []           // 빈 배열을 선언
arr[0] = 23;
arr[1] = 12;
arr[2] = 34;
for (let i = 0; i < 3; i++)
  document.write(`arr[${i}]=${arr[i]} `);
```



```
arr[0]=23 arr[1]=12 arr[2]=34
```

# 01 자바스크립트 객체 이해하기

## ■ 배열과 Array 객체

### 1. 배열

- 빈 배열 선언하기

```
let arr = []  
arr[0] = 23;  
arr[1] = 12;  
for (let i = 0; i < 3; i++)  
  document.write(`arr[${i}]=${arr[i]} `);
```



```
arr[0]=23 arr[1]=12 arr[2]=undefined
```

# 01 자바스크립트 객체 이해하기

## ■ 배열과 Array 객체

### 2. 배열 객체

```
let arr = new Array(10, 20, 30, 40, 50);  
for (let i = 0; i < arr.length; i++)  
    document.write("arr[" + i + "]= " + arr[i] + " ");
```



```
arr[0]=10 arr[1]=20 arr[2]=30 arr[3]=40 arr[4]=50
```

```
let arr = new Array();           // 빈 배열 객체 선언, arr = []와 같음
```



# 01 자바스크립트 객체 이해하기

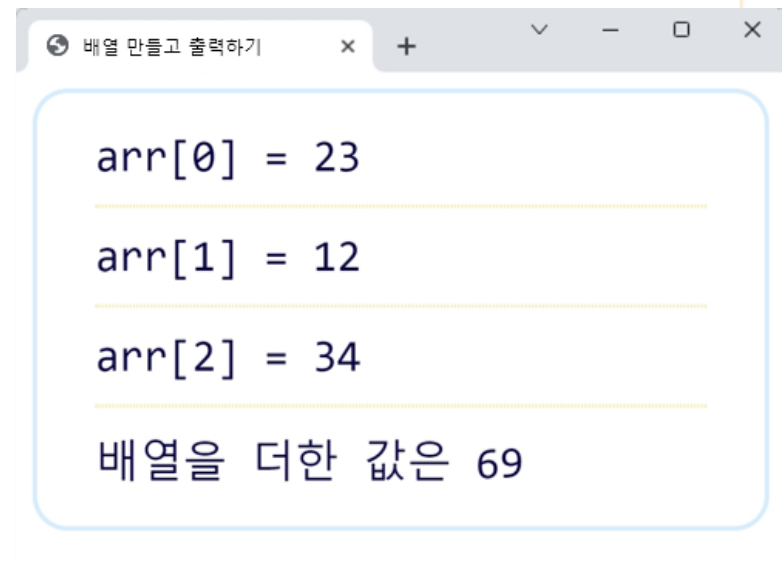
- 배열 객체로 배열 만들고 출력하기

예제 7-2

배열 만들고 출력하기

ex7-2.html

```
</head>
<body>
  <script>
    let arr = new Array();    // arr = []와 같음
    let sum = 0;
    arr[0] = 23;
    arr[1] = 12;
    arr[2] = 34;
    for (let i = 0; i < arr.length; i++){
      document.write(`arr[${i}] = ${arr[i]}<hr>`);
      sum = sum + arr[i];
    }
    document.write(`배열을 더한 값은 ${sum}`);
  </script>
</body>
</html>
```



# 01 자바스크립트 객체 이해하기

## ■ 배열과 Array 객체

### 2. 배열 객체

- 배열 객체의 메서드

표 7-5 Array 객체 메서드

메서드	내용
<code>arr1.concat(arr2)</code>	arr1 배열 뒤에 arr2 배열을 합치고 그 복사본을 반환함
<code>filter(function())</code>	function()의 조건에 맞는 배열의 복사본을 반환함
<code>indexOf(x, i)</code>	i번째 인덱스 원소부터 시작하여 문자열 x가 처음 나타나는 위치를 찾아 반환하고 x가 없으면 -1을 반환함(i를 생략하면 처음부터 찾음)
<code>join(s)</code>	배열의 모든 원소를 합친 문자열을 반환(괄호 안에 s가 있으면 원소 사이에 s를 삽입하여 합친 문자열을 반환)
<code>pop()</code>	배열 맨 뒤의 값을 삭제함
<code>push(x)</code>	배열 맨 뒤에 x를 삽입함
<code>reverse()</code>	배열의 원소 순서를 거꾸로 바꿈

# 01 자바스크립트 객체 이해하기

## ■ 배열과 Array 객체

### 2. 배열 객체

- 배열 객체의 메서드

표 7-5 Array 객체 메서드

메서드	내용
<code>shift()</code>	배열 맨 앞의 원소를 제거하고 그 값을 반환
<code>slice(ix, iy)</code>	ix부터 iy 직전까지의 복사본을 반환함(iy를 생략하면 ix부터 끝까지의 복사본을 반환)
<code>sort()</code>	모든 원소를 정렬함
<code>toString()</code>	배열의 각 원소를 문자열로 바꾸어 반환함
<code>unshift(x)</code>	배열 앞부분에 x값을 삽입함

# 01 자바스크립트 객체 이해하기

## ■ 배열과 Array 객체

### 2. 배열 객체

- 배열 객체의 메서드

#### 예제 7-3

#### Array 객체의 메서드 사용하기

```
</head>
```

```
<body>
```

```
<script>
```

```
let a1 = [0, 1, 2, 3, 4];
```

```
let a2 = new Array(9, 8, 7, 6, 5);
```

```
document.write(`a1 = [${a1}]<br>`);
```

```
document.write(`a2 = [${a2}]<hr>`);
```

```
document.write(`a1.concat(a2) : ${a1.concat(a2)}<hr>`);
```

```
function gt(e) { return(e >= 7) };
```

```
document.write(`a2.filter(gt) : ${a2.filter(gt)}<hr>`);
```

```
document.write(`a2.indexOf(6) : ${a2.indexOf(6)}<hr>`);
```

Array 객체의 메서드 사용하기 x + - □ ×

```
a1 = [0,1,2,3,4]
```

```
a2 = [9,8,7,6,5]
```

```
a1.concat(a2) : 0,1,2,3,4,9,8,7,6,5
```

```
a2.filter(gt) : 9,8,7
```

```
a2.indexOf(6) : 3
```

```
a2.push(4) : 9,8,7,6,5,4
```

```
a2.pop() : 9,8,7,6,5
```

```
a1.reverse() : 4,3,2,1,0
```

```
a2.shift() : 8,7,6,5
```

```
a2.unshift(9) : 9,8,7,6,5
```

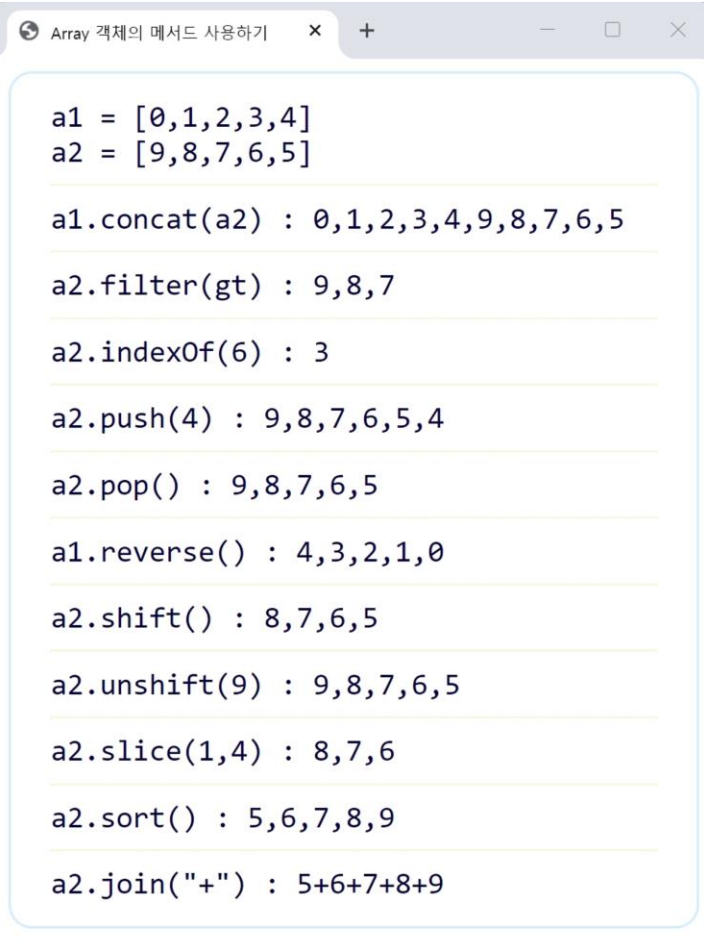
```
a2.slice(1,4) : 8,7,6
```

```
a2.sort() : 5,6,7,8,9
```

```
a2.join("+") : 5+6+7+8+9
```

# 01 자바스크립트 객체 이해하기

```
a2.push(4);
document.write(`a2.push(4) : ${a2}<hr>`);
a2.pop();
document.write(`a2.pop() : ${a2}<hr>`);
document.write(`a1.reverse() : ${a1.reverse()}<hr>`);
a2.shift();
document.write(`a2.shift() : ${a2}<hr>`);
a2.unshift(9);
document.write(`a2.unshift(9) : ${a2}<hr>`);
document.write(`a2.slice(1, 4) : ${a2.slice(1, 4)}<hr>`);
document.write(`a2.sort() : ${a2.sort()}<hr>`);
document.write(`a2.join("+") : ${a2.join("+")}<hr>`);
</script>
</body>
</html>
```



# 01 자바스크립트 객체 이해하기

## ■ 문자열과 Array 객체

### 1. String 객체

String 객체를 생성	문자열 변수를 선언
<code>let str = new String("문자열");</code>	<code>let str = "문자열";</code>

```
let str = "문자열";  
document.write("str = '" + str + "', length = " + str.length);
```



```
str = '문자열', length = 3
```

# 01 자바스크립트 객체 이해하기

## ■ 문자열과 Array 객체

### 1. String 객체

- String 객체의 인덱스 활용하기

```
let str = "문자열";  
document.write(`str[0]=${str[0]}, str[1]=${str[1]}, str[2]=${str[2]}`);
```



```
str[0]=문, str[1]=자, str[2]=열
```

# 01 자바스크립트 객체 이해하기

## ■ 문자열과 Array 객체

### 1. String 객체

- String 객체의 메서드

표 7-6 String 객체 메서드

메서드	내용
<code>charAt(i)</code>	i번째 인덱스의 문자를 반환함
<code>str1.concat(str2)</code>	str1에 str2 문자열을 합침
<code>indexOf(s, i)</code>	i번째부터 시작하여 문자 s가 처음 나타나는 위치를 찾아 반환하고 s가 없으면 -1을 반환함(i를 생략하면 처음부터 찾음)
<code>replace(s, t)</code>	문자 s를 문자 t로 변경함
<code>search(s)</code>	문자 s가 처음 나타나는 위치를 찾아 반환함. <code>indexOf()</code> 와 기능이 같지만 시작 위치를 지정할 수 없다는 점이 다름



# 01 자바스크립트 객체 이해하기

표 7-6 String 객체 메서드

메서드	내용
<code>slice(ix, iy)</code>	<code>ix</code> 부터 <code>iy</code> 직전까지의 문자열을 반환함( <code>iy</code> 를 생략하면 <code>ix</code> 부터 끝까지 반환). <code>substring()</code> 과 기능이 같지만 <code>iy</code> 가 음수일 수 있다는 점이 다름
<code>split(s, limit)</code>	<code>s</code> 를 분리자로 하여 문자열을 분리하고 배열을 반환함. <code>limit</code> 는 반환되는 배열 크기를 제한함("를 입력하면 한 문자씩 분리)
<code>substr(i, len)</code>	<code>i</code> 번째 인덱스부터 <code>len</code> 만큼의 문자열을 배열로 반환함( <code>len</code> 을 생략하면 끝까지 반환)
<code>substring(ix, iy)</code>	<code>ix</code> 부터 <code>iy</code> 직전까지의 배열을 반환함( <code>iy</code> 를 생략하면 <code>ix</code> 부터 끝까지 반환). <code>slice()</code> 와 비슷하지만 <code>iy</code> 는 언제나 양수라는 점이 다름
<code>toLowerCase()</code>	소문자로 변환함
<code>toUpperCase()</code>	대문자로 변환함
<code>trim()</code>	양끝의 공백 문자를 제거함

# 01 자바스크립트 객체 이해하기

## 예제 7-4 String 객체의 메서드 사용하기

</head>

<body>

<script>

```
const s1 = new String("Web ");
```

```
const s2 = "Programming";
```

```
document.write(`s1 = "${s1}"<br>`);
```

```
document.write(`s2 = "${s2}"<hr>`);
```

```
document.write(`s1.charAt(0) : ${s1.charAt(0)}<hr>`);
```

```
document.write(`s1.concat(s2) : ${s1.concat(s2)}<hr>`);
```

```
document.write(`s2.indexOf("i") : ${s2.indexOf("i")}<hr>`);
```

```
document.write(`s2.replace("m", "M") : ${s2.replace("m", "M")}<hr>`);
```

```
document.write(`s2.search("m") : ${s2.search("i")}<hr>`);
```

```
document.write(`s2.split("a") : ${s2.split("a")}<hr>`);
```

```
document.write(`s2.substr(0, 4) : ${s2.substr(0, 4)}<hr>`);
```

String 객체의 메서드 사용하기 × +

```
s1 = "Web "
```

```
s2 = "Programming"
```

```
s1.charAt(0) : W
```

```
s1.concat(s2) : Web Programming
```

```
s2.indexOf("i") : 8
```

```
s2.replace("m", "M") : PrograMming
```

```
s2.search("m") : 8
```

```
s2.split("a") : Progr,mming
```

```
s2.substr(0, 4) : Prog
```

# 01 자바스크립트 객체 이해하기

## 예제 7-4

## String 객체의 메서드 사용하기

ex7-4.html

```
document.write(`s2.substring(3, 7) : ${s2.substring(3, 7)}<hr>`);  
document.write(`s2.slice(3, -1) : ${s2.slice(3, -1)}<hr>`);  
document.write(`s2.toLowerCase() : ${s2.toLowerCase()}<hr>`);  
document.write(`s2.toUpperCase() : ${s2.toUpperCase()}<hr>`);  
document.write(`s1.trim() : +${s1.trim()}+`);  
</script>  
</body>  
</html>
```

```
s2.substring(3, 7) : gram  
s2.slice(3, -1) : grammin  
s2.toLowerCase() : programming  
s2.toUpperCase() : PROGRAMMING  
s1.trim() : +Web+
```

# 01 자바스크립트 객체 이해하기

## ■ 문자열과 Array 객체

### 1. String 객체

- String 객체의 메서드

```
arr = s2.split("a")    // arr은 Array 객체, arr[0]: "Progr", arr[1]: "mming"  
arr = s2.split("")    // arr[0]: "P", arr[1]: "r", ...  
arr = s2.split("a", 1) // arr[0]: "Progr" 1개만 반환
```

표 7-7 문자열의 음수 인덱스

인덱스	0	1	2	3	4	5	6	7	8	9	10
문자	P	r	o	g	r	a	m	m	i	n	g
음수 인덱스	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

# 01 자바스크립트 객체 이해하기

## ■ 문자열과 Array 객체

### 1. String 객체

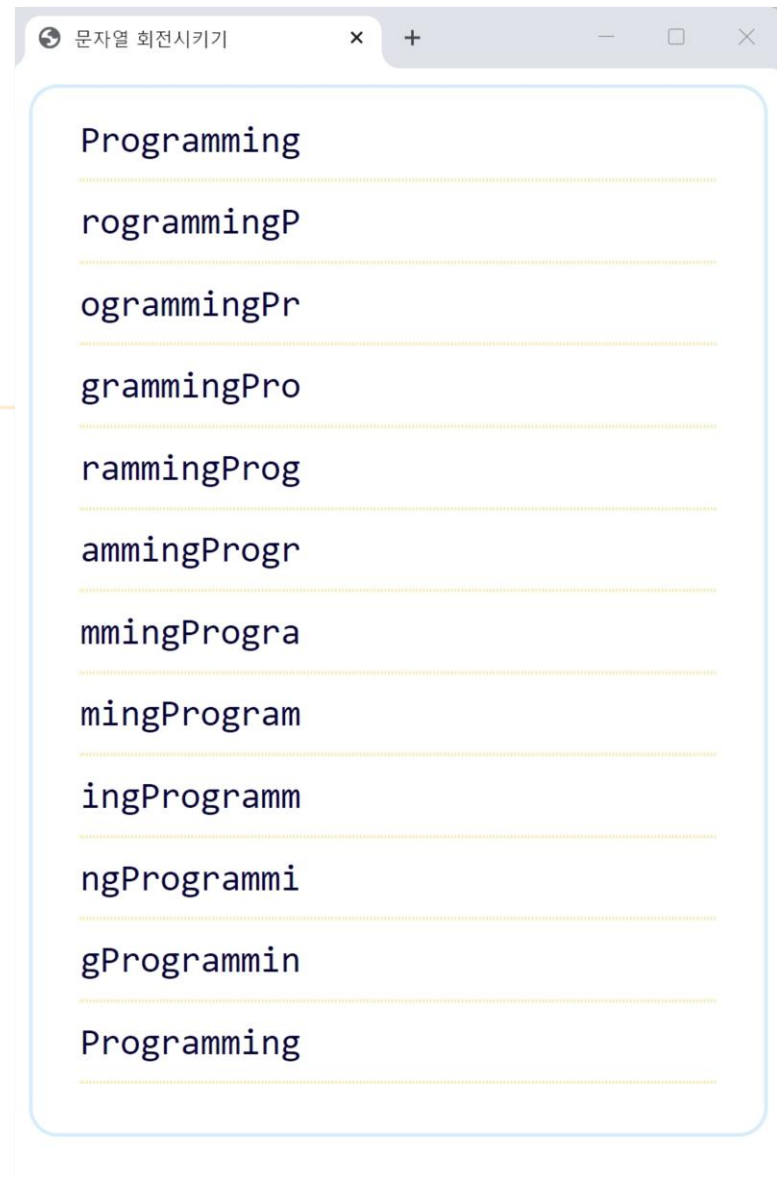
- String 메서드 활용하기

예제 7-5

문자열 회전시키기

```
<script>
  let text = new String("Programming");
  let firstChar;

  for (let i = 0; i <= text.length; i++) {
    document.write(text + "<hr>");
    firstChar = text[0];
    text = text.slice(1) + firstChar;
  };
</script>
```



# 01 자바스크립트 객체 이해하기

## ■ 문자열과 Array 객체

### 2. for-of 반복문

```
const dayString = "일월화수목금토";  
  
for (let day of dayString)  
  document.write(day + "요일, ");
```



일요일, 월요일, 화요일, 수요일, 목요일, 금요일, 토요일,

# 01 자바스크립트 객체 이해하기

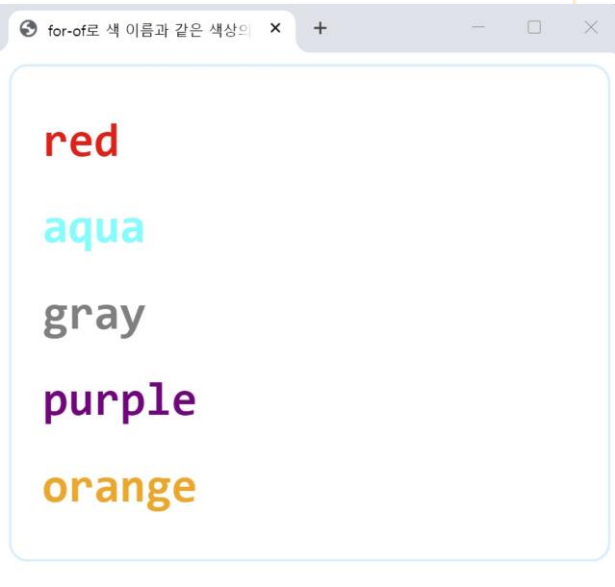
## 예제 7-6

## for-of로 색 이름과 같은 색상의 문자 만들기

ex7-6.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>for-of로 색 이름과 같은 색상의 문자 만들기</title>
  <link rel="stylesheet" href="style_js.css">
</head>
<body>
  <script>
    const colorArray = ["red", "aqua", "gray", "purple", "orange"];

    for(let color of colorArray)
      document.write("<h1 style='color:" + color + "'>" + color + "</h1>");
  </script>
</body>
</html>
```



# 01 자바스크립트 객체 이해하기

## ■ Date 객체

```
today = new Date()           // 현재 기준의 날짜 관련 객체 생성  
document.write(today + "<hr>");
```



Tue Oct 31 2023 10:54:51 GMT+0900 (한국 표준시)

```
today = new Date(2022, 3, 24) // 2022년 4월 24일의 Date 객체 생성
```



# 01 자바스크립트 객체 이해하기

## ■ Date 객체

- Date 객체의 메서드

표 7-8 Date 객체 메서드

메서드		내용
getFullYear()	setFullYear()	4자리 연도
getMonth()	setMonth()	월(0 ~ 11), 출력 시 + 1, 입력 시 - 1
getDate()	setDate()	일(1~31)
getDay()	setDay()	요일(0~6)
getHours()	setHours()	시간(0~23)
getMinutes()	setMinutes()	분(0~59)
getSeconds()	setSeconds()	초(0~59)
getMilliseconds()	setMilliseconds()	밀리초(0~999)
toLocaleString()		연, 월, 일, 시간 표시

# 01 자바스크립트 객체 이해하기

## ■ Date 객체

- Date 객체의 메서드

```
let today = new Date();  
const day = "일월화수목금토";  
document.write(day[today.getDay()] + "요일<br>");
```

# 01 자바스크립트 객체 이해하기

## ■ Date 객체

- Date 객체의 메서드

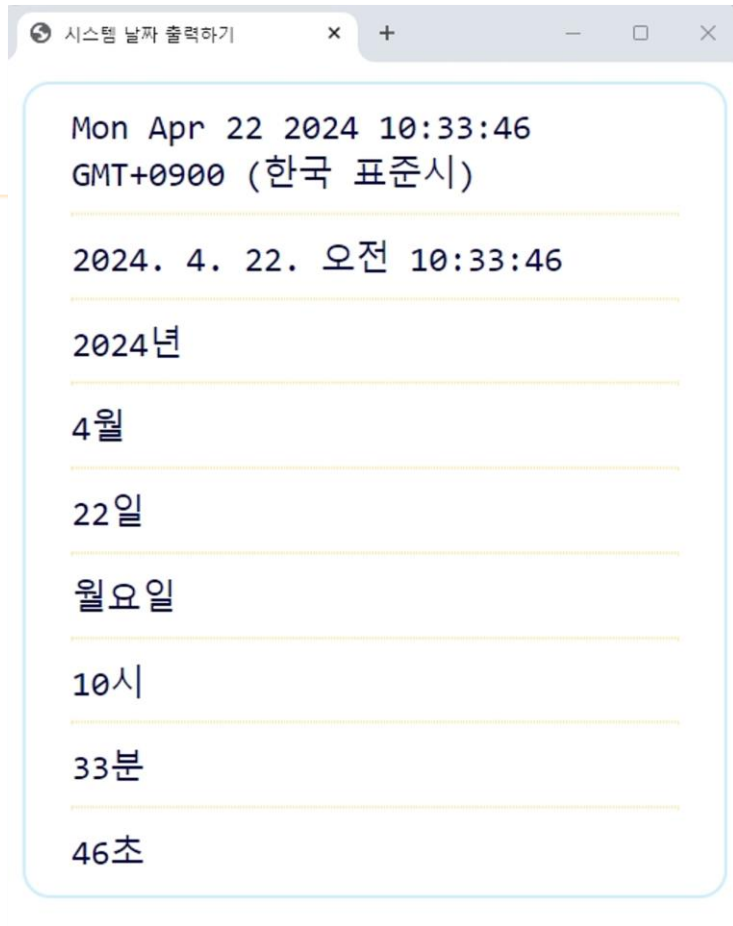
예제 7-7

시스템 날짜 출력하기

```
<script>
```

```
let today = new Date();
const day = "일월화수목금토";
document.write(today + "<hr>");
document.write(today.toLocaleString() + "<hr>");
document.write(today.getFullYear() + "년<hr>");
document.write(today.getMonth() + 1 + "월<hr>");
document.write(today.getDate() + "일<hr>");
document.write(day[today.getDay()] + "요일<hr>");
document.write(today.getHours() + "시<hr>");
document.write(today.getMinutes() + "분<hr>");
document.write(today.getSeconds() + "초");
```

```
</script>
```



# 01 자바스크립트 객체 이해하기

## ■ Date 객체

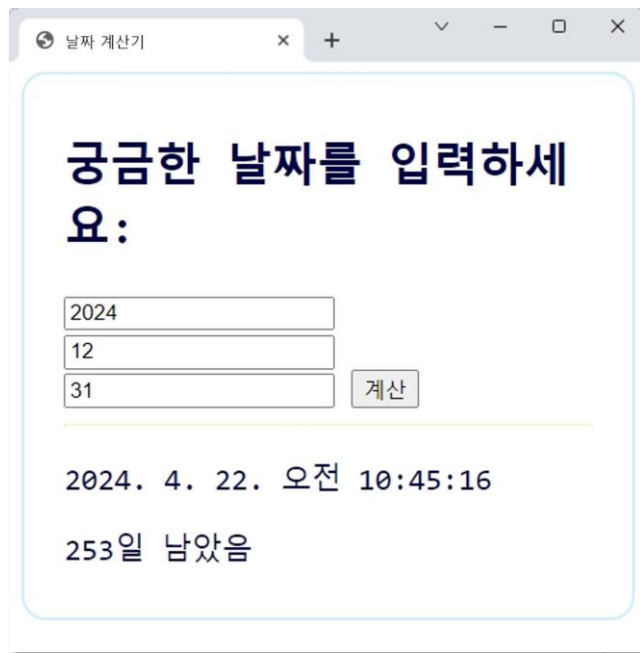
- Date 객체 활용하기

예제 7-8

날짜 계산기

ex7-8.html

```
</head>
<body>
  <h2>궁금한 날짜를 입력하세요:</h2>
  <input type="number" id="year" placeholder="년">
  <input type="number" id="month" placeholder="월">
  <input type="number" id="day" placeholder="일">
  <button onclick="calculateDays()">계산</button><hr>
  <p id="today"></p>
  <p id="result"></p>
  <script>
    function calculateDays() {
      const year = parseInt(document.getElementById("year").value);
      const month = parseInt(document.getElementById("month").value);
      const day = parseInt(document.getElementById("day").value);
```



날짜 계산기

궁금한 날짜를 입력하세요:

2024  
12  
31

계산

2024. 4. 22. 오전 10:45:16

253일 남았음

# 01 자바스크립트 객체 이해하기

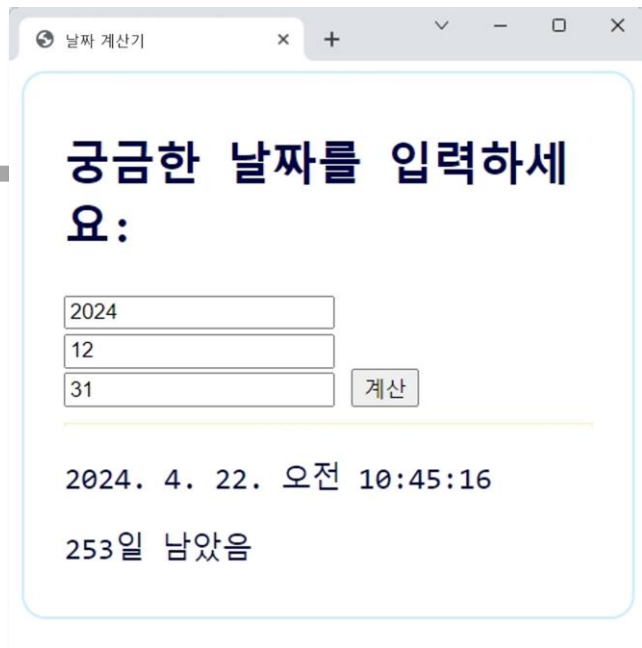
## ■ Date 객체

- Date 객체 활용하기

```
const today = new Date();
const inputDate = new Date(year, month - 1, day);
const timeDiff = today - inputDate;
const daysDiff = Math.floor(timeDiff / (1000 * 60 * 60 * 24));

document.getElementById("today").innerHTML=`${today.toLocaleString()}`;
if(daysDiff >= 0)
    document.getElementById("result").innerHTML=`${daysDiff}일 전`;
else
    document.getElementById("result").innerHTML=`${-(daysDiff)}일 남았음`;
}
```

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



# 01 자바스크립트 객체 이해하기

## ■ Math 객체

- Math 객체의 메서드

표 7-9 Math 객체 메서드

메서드	내용
<code>abs(x)</code>	절댓값
<code>cos(x)</code> , <code>sin(x)</code> , <code>tan(x)</code>	코사인, 사인, 탄젠트
<code>exp(x)</code>	지수 $e^x$
<code>pow(x, y)</code>	지수 $x^y$
<code>random()</code>	무작위 수
<code>floor(x)</code> , <code>round(x)</code> , <code>ceil(x)</code>	버림, 반올림, 올림
<code>log()</code>	로그
<code>sqrt(x)</code>	제곱근
<code>max(a, b, c)</code>	최댓값
<code>min(a, b, c)</code>	최솟값

# 01 자바스크립트 객체 이해하기

## 예제 7-9

## 무작위로 배경색 바꾸기

```
<script>
  const letters = "0123456789ABCDEF";
  let color = "#";
  let count = 1;
  let intervalID = setInterval(changeColor, 1000);
  function changeColor() {
    if (++count > 25)
      clearInterval(intervalID);
    for (let k = 0; k < 6; k++)
      color += letters[Math.floor(Math.random()*16)];
    document.write("<div style='background-color:"
      + color + "'" + color + "</div>");
    color = "#";
    // 16진수 문자열 초기화
  }
</script>
</body>
</html>
```



**02**

# **사용자 객체 만들기**



## 02 사용자 객체 만들기

### ■ 사용자 객체

#### 1. 사용자 객체 만들기

- 객체의 속성과 메서드 나열하기

```
let person = {  
  name: "cho",           // 속성: 속성값, 속성: 속성값,  
  age: 23,  
  intro: function () {   // 속성: 함수는 메서드  
    document.write("My name: " + this.name + ", age: " + this.age + "<br>");  
  }  
};  
  
person.intro();
```

My name: cho, age: 23

## 02 사용자 객체 만들기

### ■ 사용자 객체

#### 1. 사용자 객체 만들기

- 빈 객체를 먼저 생성하기

```
let person = new Object();
person.name = "cho";
person.age = 23;
person.intro = function () {
    document.write("My name: " + this.name + ", age: " + this.age + "<br>");
};

person.intro();
```

My name: cho, age: 23

## 02 사용자 객체 만들기

### ■ 사용자 객체

#### 1. 사용자 객체 만들기

- **function**으로 객체 선언하기

```
function Person (name, age) {  
  this.name = name;  
  this.age = age;  
  this.intro = function () {  
    document.write("My name: " + this.name + ", age: " + this.age + "<hr>");  
  }  
};  
  
let person1 = new Person("cho", 23);  
let person2 = new Person("kim", 24);  
person1.intro();  
person2.intro();
```

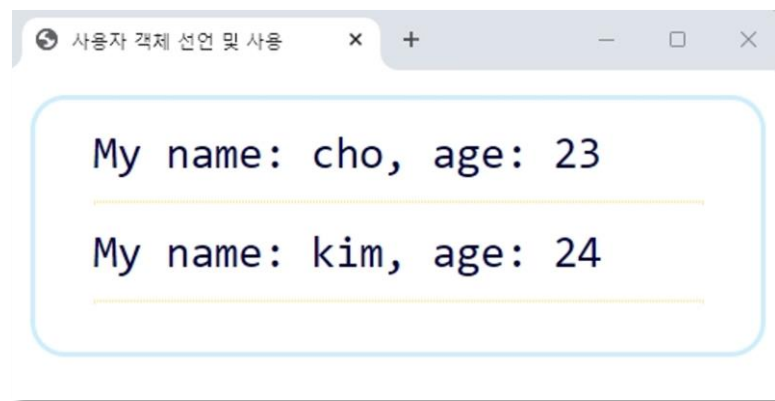
## 02 사용자 객체 만들기

예제 7-10

사용자 객체 선언 및 사용하기

ex7-10.html

```
<script>
function Person (name, age) {
  this.name = name;
  this.age = age;
  this.intro = function () {
    document.write("My name: " + this.name + ", age: " + this.age + "<hr>");
  }
};
let person1 = new Person("cho", 23);
let person2 = new Person("kim", 24);
person1.intro();
person2.intro();
</script>
</body>
</html>
```



## 02 사용자 객체 만들기

### ■ 사용자 객체

#### 2. 사용자 객체의 속성 다루기

- 속성값을 변경하거나 속성 추가하기

```
let person = new Object();
person.name = "cho";
person.age = 23;
person.intro = function () {
    document.write("My name: " + this.name + ", age: " + this.age + "<br>");
};
person.age = 33;           // 객체 속성 age 값 변경
person.weight = 70;        // 객체 속성 weight와 값 추가
document.write("age: " + person.age + ", weight: " + person.weight);
```



age: 33, weight: 70

## 02 사용자 객체 만들기

### ■ 사용자 객체

#### 2. 사용자 객체의 속성 다루기

- 속성 삭제하기

```
let person = new Object();
person.name = "cho";
person.age = 23;
person.intro = function () {
    document.write("My name: " + this.name + ", age: " + this.age + "<br>");
};
delete person.age;
document.write("age: " + person.age + "<br>");
```



age: undefined

## 02 사용자 객체 만들기

### ■ 사용자 객체

#### 2. 사용자 객체의 속성 다루기

- 대괄호로 속성에 접근하기

```
let person = new Object();
person.name = "cho";
person.age = 23;
person.intro = function () {
    document.write("My name: " + this.name + ", age: " + this.age + "<br>");
};
const know = "age";
document.write("age: " + person[know] + "<br>");
```



age: 23

## 02 사용자 객체 만들기

### ■ 사용자 객체

#### 3. for-in 반복문

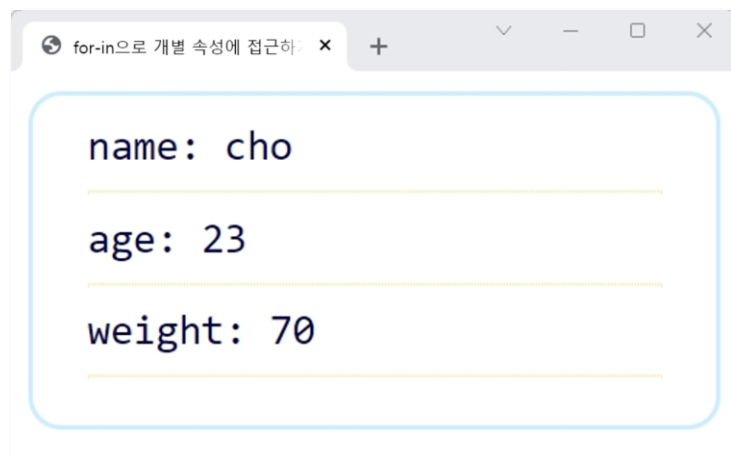
예제 7-11

for-in으로 개별 속성에 접근하기

ex7-11.html

```
<script>
  let person = new Object();
  person.name = "cho";
  person.age = 23;
  person.weight = 70;

  for(let prop in person) {
    document.write(prop + ": " + person[prop] + "<hr>");
  }
</script>
</body>
</html>
```





## 02 사용자 객체 만들기

### ■ class를 이용하는 객체 상속

```
class Person {  
  constructor(name, age) {  
    this.name = name;  
    this.age = age;  
  }  
  intro() {  
    document.write("My name: " + this.name + ", age: " + this.age + "<br>");  
  }  
}  
  
let person1 = new Person("cho", 23);  
person1.intro();
```

My name: cho, age: 23

## 02 사용자 객체 만들기

### ■ class를 이용하는 객체 상속

예제 7-12 class를 이용한 객체 상속하기

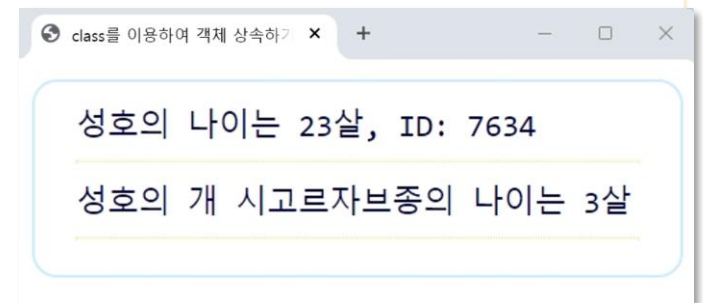
ex7-12.html

```
<script>
class Animal {
  constructor(name, age) {
    this.name = name;
    this.age = age;
  }
}

class Person extends Animal {
  constructor(name, age, id) {
    super(name, age);      // Animal 객체의 속성에 접근
    this.id = id;
  }
}
```

## 02 사용자 객체 만들기

```
class Person extends Animal {  
  constructor(name, age, id) {  
    super(name, age);           // Animal 객체의 속성에 접근  
    this.id = id;  
  }  
}  
  
class Dog extends Animal {  
  constructor(name, age, owner) {  
    super(name, age);           // Animal 객체의 속성에 접근  
    this.owner = owner;  
  }  
}  
  
let ps = new Person("성호", 23, 7634);  
let dog = new Dog("시고르자브종", 3, "성호");  
document.write(`${ps.name}의 나이는 ${ps.age}살, ID: ${ps.id}<hr>`);  
document.write(`${dog.owner}의 개 ${dog.name}의 나이는 ${dog.age}살<hr>`);  
</script>
```



## 02 사용자 객체 만들기

### ■ class를 이용하는 객체 상속

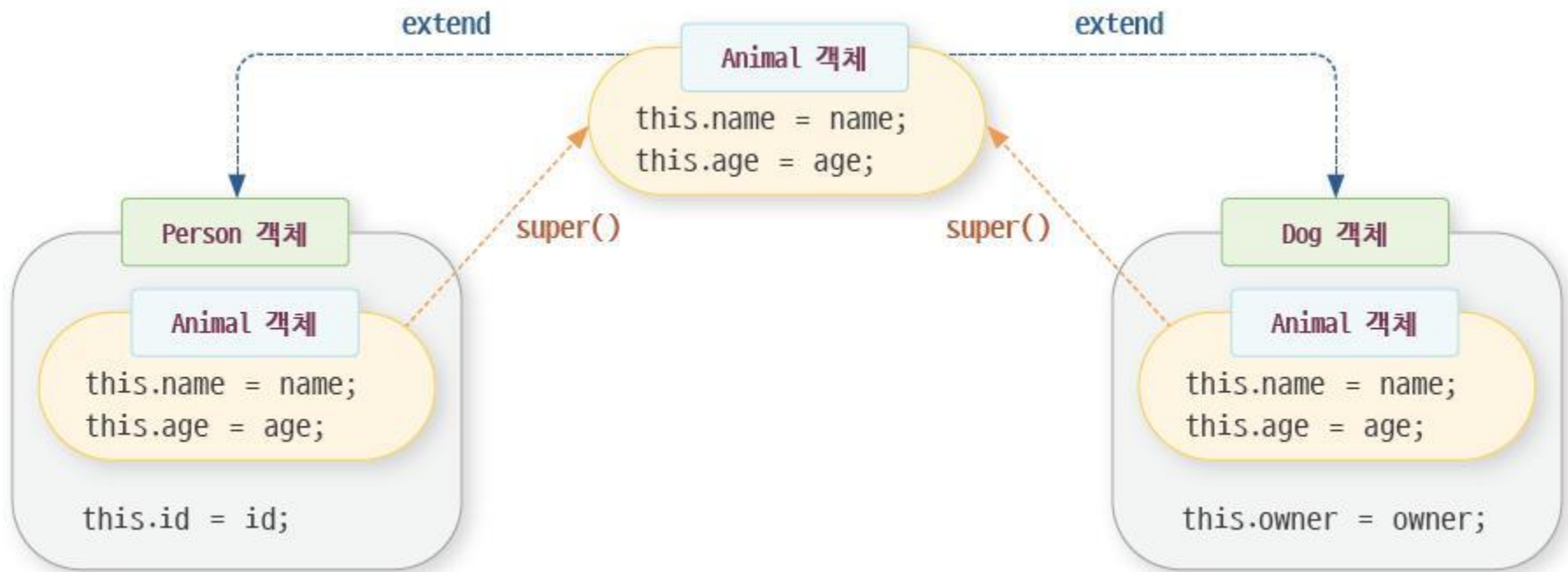


그림 7-6 Animal 객체 상속 후 객체의 구조

## 03 this 란 무엇인가

- 객체 자체(객체를 가르키는 참조)
- 사용되는 위치와 상황에 따라 다르게 바인딩된다.

```
function showThis() {  
  console.log(this);  
}  
showThis();
```

```
console.log(this);
```

```
function Person(){  
  this.func=function(){  
    console.log(this);  
  }  
}  
let obj=new Person();  
obj.func();
```

## 03 this 란 무엇인가

- **function() 안에서의 this : 동적으로 결정**
  - 일반 함수 안에서의 this
    - 브라우저 : window,
    - Node.js : Global 객체
  - 생성자 함수 안에서의 this
    - 브라우저 : 생성된 객체
    - Node.js : 생성된 객체
  - 객체 리터럴 안에서의 this
    - 브라우저 : 현재 객체
    - Node.js : 현재 객체
  - 함수가 일반 함수이냐 생성자 함수이냐는 호출형태로 결정된다.
    - 일반함수 호출형태와 같이 호출 : `functionName();`
    - New 와 함께 호출 : `new FunctionName();`

## 03 this 란 무엇인가

```
function Person (name, age) {  
  this.name = name;  
  this.age = age;  
  console.log(this);  
};  
  
let obj1=new Person('홍창기', 30);  
Person('오지환', 33);
```

```
Person { name: '홍창기', age: 30 }  
<ref *1> Object [global] {  
  global: [Circular *1],  
  clearImmediate: [Function: clearImmediate],  
  ...  
  ...  
  crypto: [Getter],  
  name: '오지환',  
  age: 33  
}
```

## 03 this 란 무엇인가

- 화살표 함수안에서의 this : 정적으로 결정
  - 렉시컬 스코프(lexical scope)를 따른다.
  - 기본적으로 화살표 함수안에는 this가 없으며 상위 스코프의 This를 참조하는 것이다.
  - 일반적으로 화살표 함수안에서는 특별한 경우를 제외하고 this를 쓰지 않는다.



## 03 this 란 무엇인가

```
function Person(name, age){  
  this.name=name;  
  this.age=age;  
  this.intro=()=>{  
    console.log(this.name, this.age);  
  }  
}  
let obj1=new Person('홍창기', 30);  
obj1.intro();
```

홍창기 30

```
let obj2={  
  name: '홍창기',  
  age: 30,  
  intro: ()=>{  
    console.log(this.name, this.age);  
  }  
}  
obj2.intro();
```

undefined undefined

## 03 this 란 무엇인가

- 화살표함수에서 this를 쓰면 안되는 경우
  - 객체의 메소드에서 쓰면 안된다.
  - 생성자함수로 쓰면 안된다.
  - addEventListener 함수의 콜백 함수로 사용된 화살표함수에서 쓰면 안된다
    - addEventListener의 콜백함수를 화살표함수로 정의하면 this가 상위 스코프인 window객체를 가르키게 된다.

```
let button = document.getElementById('myButton');  
  
button.addEventListener('click', () => {  
  console.log(this === window); // => true  
  this.innerHTML = 'Clicked button';  
});
```

## 03 this 란 무엇인가

화살표 함수는 `this`를 사용하지 않는 간단한 기능을 구현할 때나 객체의 메소드 내에서 콜백함수를 정의할 때 메소드를 소유하고 있는 객체를 `this`로 사용하고자 할 때만 사용한다.

```
const person = {  
  name: 'Kim',  
  age: 30,  
  greet: function() {  
    console.log(`Hello, my name is ${this.name}`);  
  
    setTimeout(() => {  
      console.log(`I am ${this.age} years old`);  
    }, 1000);  
  }  
};  
  
person.greet();
```

```
Hello, my name is Kim  
I am 30 years old
```