

## 14 자바스크립트 기본 문법

## 14-1. 변수 알아보기

## ■ 변수와 상수

### • 변수

- 변할 수 있는 값을 가지며 `let` 키워드를 사용해 선언

```
let 변수
```

자료형을 지정하지 않음

### • 상수

- 변하지 않는 값을 가지며 `const` 키워드를 사용해 선언함

```
const 상수
```

자료형을 지정하지 않음

- `let`과 `const`에 대해서는 다음 chapter에서 자세히 학습함

## ■ 변수와 상수 선언 규칙

### • 변수 이름

- 영어 문자, 언더스코어(`_`), 숫자를 사용
- 첫 글자는 영문자, `_` 기호, `$` 기호를 사용한다
- 띄어쓰기나 기호는 허용하지 않음

[ 예 ] `now`, `_now`, `now25` ( 사용 가능 )

[ 예 ] `25now`, `now 25`, `*now` ( 사용 불가능 )

- 영어 대소문자를 구별하며 예약어는 사용할 수 없음

### • 여러 단어를 연결할 경우

- 하이픈이나 언더스코어를 사용
- 중간에 대문자를 섞어 쓸 수도 있음

[ 예 ] `total-area`, `total_area`, `totalArea` 등

chapter14 &gt; &lt;&gt; age-result.html &gt; ...

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5  </head>
6  <body>
7
8    <script>
9      let today = new Date();
10     let currentYear = today.getFullYear();
11
12     let birthYear = prompt("태어난 연도를 입력하세요. (YYYY)", "");
13     let age = currentYear - birthYear + 1;
14
15     document.write(currentYear + "년 현재<br>");
16     document.write(birthYear + "년에 태어난 사람의 나이는 " + age + "세입니다.");
17   </script>
18
19 </body>
20 </html>

```

이 페이지 내용:

태어난 연도를 입력하세요. (YYYY)




2023년 현재

2000년에 태어난 사람의 나이는 24세입니다.

## 14-2 자료형 이해하기

## ■ 문자형 자료

- 한글, 한자, 영문자, HTML string 등의 데이터
  - 반드시 인용 부호 내에 표현해야 함
  - 숫자형 자료를 이중 인용 부호 내에 표현할 경우 문자열로 인식 연산에 참여 불가능

```
let 변수명 = "문자형 자료";
```

```
let str="javascript";  
let num="100";  
let tag="<h1> String </h1>";
```

## ■ 숫자형 자료

- 정수나 실수 값을 가지는 자료형
  - 10진수, 8진수, 16진수 표현이 가능
  - 인용 부호 내에 표현하지 않음

```
let 변수명 = 숫자형 자료;
```

```
let num1 = 10 ;      // 숫자 10을 10진수로 표현  
let num1 = 012 ;     // 숫자 10을 8진수로 표현  
let num1 = 0Xa ;     // 숫자 10을 16진수로 표현
```

- 인용부호 내의 숫자형 데이터(문자형 자료)를 숫자형 데이터로 지정하는 방법

```
let num1 = "10";      // 문자열  
let num2 = Number(num1); // 숫자형으로 변환  
let num3 = parseInt(num1); // 숫자형으로 변환
```

## ■ 논리형 데이터

- 값으로 참(true)이나 거짓(false)을 가지는 데이터

```
let 변수명 = true | false
```

```
let s=true;           // 논리형 변수 s에 true가 저장됨  
let t=10>=100;        // (10>=100)이 거짓이므로 논리형 변수 t에 false가 저장됨
```

- Boolean 함수의 사용

- Boolean() 함수의 인자로 0, null, undefined가 지정되는 경우 false를 반환 ( 그 외의 값은 true를 반환 )

```
let 변수명 = Boolean(인자)
```

```
let s = Boolean(0);           // Boolean 함수가 false를 반환하므로 변수 s에 false가 저장됨  
let t = Boolean("홍길동");    // Boolean 함수가 true를 반환하므로 변수 t에 true가 저장됨
```

## ■ null 자료형과 undefined 자료형

### • undefined 자료형

- var 키워드를 사용해 변수는 선언되어 있지만 아무런 값도 지정되지 않은 경우
- 차후에 변수가 어떤 자료형의 데이터가 저장될 지 몰라 변수를 선언하기만 할 경우 사용

```
let num2;
```

### • null 자료형

- 변수의 값으로 null이 지정된 경우
- 주로 변수에 저장된 데이터를 비우고자 할 때 사용

```
let num2 = null;
```



## ■ 묵시적 자료형 변환

- 선언된 자료형은 대입 되는 값의 성격에 따라 언제든지 다른 자료형의 변환 가능
  - 별도의 데이터형 변환 과정이 필요 없음
- 묵시적 형변환의 예
  - data라는 변수를 선언하고 200이라는 값으로 초기화
  - 변수 data에 [한국교통대학교] 라는 문자열을 다시 지정할 경우

```
let data = 200;  
data = "한국교통대학교";
```

변수 data는 숫자형으로 선언되고 200이라는 값을 가짐

기존 200이라는 값은 "한국교통대학교"라는 값으로 변경되고, data는 문자열 형으로 변환됨

## ■ 숫자와 문자의 자동 변환

- 사칙 연산자 중 + 연산자의 경우, 모든 숫자 자료형을 문자 자료형으로 자동 변환

```
<script>
  alert('52 + 273');           // 문자열 '52+273'를 출력
  alert(52 + 273);             // 숫자 325를 출력
  alert('52' + 273);           // 문자열 52273을 출력
  alert('52' + '273');         // 문자열 52273을 출력
</script>
```

- +연산자를 제외한 모든 사칙 연산자의 경우, 모든 문자 자료형을 숫자 자료형으로 자동 변환

```
<script>
  alert('52 * 273');           // 문자열 '52*273'를 출력
  alert(52 * 273);             // 숫자 14196을 출력
  alert('52' * 273);           // 숫자 14196을 출력
  alert('52' * '273');         // 숫자 14196을 출력
</script>
```

## 14-3. 연산자 알아보기

## ■ 산술 연산자

### • 수학 계산을 할 때 사용하는 연산자

종류	설명
+	두 피연산자의 값을 더합니다.
-	첫 번째 피연산자 값에서 두 번째 피연산자 값을 뺍니다.
*	두 피연산자의 값을 곱합니다.
/	첫 번째 피연산자 값을 두 번째 피연산자 값으로 나눕니다.
%	첫 번째 피연산자 값을 두 번째 피연산자 값으로 나눈 나머지를 구합니다.
++	피연산자를 1 증가시킵니다.
--	피연산자를 1 감소시킵니다.

19  
11  
60  
3.75  
3  
16  
3

chapter14 > <> operator-arth.html > ...

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |
9  |       let num1 = 15;
10 |       let num2 = 4;
11 |
12 |       document.write(num1 + num2, "<br>");
13 |       document.write(num1 - num2, "<br>");
14 |       document.write(num1 * num2, "<br>");
15 |       document.write(num1 / num2, "<br>");
16 |       document.write(num1 % num2, "<br>");
17 |       num1++;
18 |       document.write(num1, "<br>");
19 |       num2--;
20 |       document.write(num2, "<br>");
21 |
22 |   </script>
23 </body>
24 </html>
```

## ■ 할당(대입) 연산자

- 연산자 오른쪽의 실행 결과를 왼쪽 변수에 할당하는 연산자

종류	설명
=	연산자 오른쪽의 값을 왼쪽 변수에 할당합니다.
+=	y = y + x를 의미합니다.
-=	y = y - x를 의미합니다.
*=	y = y * x를 의미합니다.
/=	y = y / x를 의미합니다.
%=	y = y % x를 의미합니다.

13  
10  
30  
0

```
chapter14 > <> operator-alloc.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |       |
9  |       |   let num1 = 10;
10 |       |   let num2 = 3;
11 |       |
12 |       |   num1 += num2;
13 |       |   document.write(num1, "<br>");
14 |       |   num1 -= num2;
15 |       |   document.write(num1, "<br>");
16 |       |   num1 *= num2;
17 |       |   document.write(num1, "<br>");
18 |       |   num1 %= num2;
19 |       |   document.write(num1, "<br>");
20 |       |
21 |   </script>
22 </body>
23 </html>
```

chapter14 &gt; &lt;&gt; operator-alloc2.html &gt; ...

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |
9  |       let str="<table border='1'>";
10 |       str+="<tr>";
11 |       str+="<td>100</td><td>200</td><td>300</td>";
12 |       str+="</tr>";
13 |       str+="</table>";
14 |
15 |       document.write(str);
16 |
17 |   </script>
18 </body>
19 </html>
```

100	200	300
-----	-----	-----

## ■ 연결 연산자

- 둘 이상의 문자열을 하나의 문자열로 결합하는 연산자
  - 변수와 문자열을 결합해 출력할 때도 사용

한국교통대학교 소프트웨어전공  
충주시 한국교통대학교 소프트웨어전공  
충주시 한국교통대학교 소프트웨어전공

chapter14 > <> operator-connect.html > ...

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5  </head>
6  <body>
7    <script>
8
9      let str1 = "한국교통대학교 ";
10     let str2 = "소프트웨어전공"
11
12     let str3 = str1 + str2;
13     document.write(str3 + "<br>");
14
15     document.write("충주시 " + str1 + str2 + "<br>");
16
17     document.write("충주시 ", str1 , str2 , "<br>");
18
19   </script>
20 </body>
21 </html>
```

## ■ 비교 연산자

- 피연산자 2개의 값을 비교해서 true나 false로 결과 반환

종류	설명	예시	
		조건식	결과값
==	피연산자가 서로 같으면 true입니다.	3 == "3"	true
===	피연산자도 같고 자료형도 같으면 true입니다.	a === "3"	false
!=	피연산자가 서로 같지 않으면 true입니다.	3 != "3"	false
!==	피연산자가 같지 않거나 자료형이 같지 않으면 true입니다.	3 !== "3"	true
<	왼쪽 피연산자가 오른쪽 피연산자보다 작으면 true입니다.	3 < 4	true
<=	왼쪽 피연산자가 오른쪽 피연산자보다 작거나 같으면 true입니다.	3 <= 4	true
>	왼쪽 피연산자가 오른쪽 피연산자보다 크면 true입니다.	3 > 4	false
>=	왼쪽 피연산자가 오른쪽 피연산자보다 크거나 같으면 true입니다.	3 >= 4	false



chapter14 > <> operator-compare.html > ...

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |
9  |       const a = 10;
10 |       const b = 20;
11 |       const c = 10;
12 |       const f = "20";
13 |
14 |       document.write(a > b, "<br>");
15 |       document.write(a < b, "<br>");
16 |       document.write(a <= b, "<br>");
17 |       document.write(b == f, "<br>");
18 |       document.write(a != b, "<br>");
19 |       document.write(b === f, "<br>");
20 |
21 |   </script>
22 </body>
23 </html>
```

false  
true  
true  
true  
true  
false

## ■ 논리 연산자

- true와 false가 피연산자인 연산자 조건을 처리할 때 사용

종류	기호	설명
OR 연산자		피연산자 중 하나만 true여도 true가 됩니다.
AND 연산자	&&	피연산자가 모두 true일 경우에만 true가 됩니다.
NOT 연산자	!	피연산자의 반댓값을 지정합니다.

```
chapter14 > <> operator-logical.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |   |
9  |   |   alert( 30>20 && 20>10 );
10  |   |
11  |   </script>
12 </body>
13 </html>
```

이 페이지 내용:

true

확인

## ■ typeof 연산자

- 특정 데이터 또는 변수의 자료형을 출력하기 위해 사용

typeof 변수 ( 또는 데이터 )

```
chapter14 > <> operator-typeof.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |       const num = 100;
9  |       const str = "자바스크립트";
10 |       document.write("num의 데이터형 : ", typeof num, "<br>");
11 |       document.write("str의 데이터형 : ", typeof str);
12 |
13 |   </script>
14 </body>
15 </html>
```

num의 데이터형 : number  
str의 데이터형 : string

## ■ 템플릿 문자열

### • 문자열과 변수(또는 수식)를 함께 출력할 때 기존 방법보다 간단하게 구현할 수 있는 방법

#### - 기존 방법

문자열 내의 변수나 수식은 문자열 연산자(+) 와 괄호 또는 콤마(,)와 괄호를 사용해 서로 구분해 표현

#### - 템플릿 문자열 사용

기호와 표현식을 사용

기호는 키보드 숫자 키 1의 왼쪽 키를 의미

#### - 표현식(expression)

\$로 시작하고 중괄호 내에 표현되는 변수나 수식 형태의 구조

[예] \${10+20}

[예] \${result} //result가 변수라고 가정

27+52의 값은 79 입니다.  
변수 data의 값은 korea 입니다.

```
chapter14 > <> template.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |
9  |       document.write(`27+52의 값은 ${27+52} 입니다.<br>`);
10 |       const data = "korea";
11 |       document.write(`변수 data의 값은 ${data} 입니다.`);
12 |
13 |   </script>
14 </body>
15 </html>
```

## 14-4. 조건문 알아보기

## ■ if 문

- 조건에 만족하는 경우에만 특정 스크립트를 수행

```
if( 조건식 ){  
    조건식이 true 일 때 실행되는 문장  
}
```

- 조건식이 false이면 실행되는 문장이 없음
- 조건식에 논리형이 아닌 다른 자료형의 데이터가 오는 경우
  - 자료형에 관계 없이 true와 false로 인식됨

0, null, ""(빈 문자), undefined를 제외한 모든 데이터를 true로 인식

```
let num = 3;  
if ( num ) {                // true로 인식됨  
}
```

```
let num = 0;  
if ( num ) {                // false로 인식됨  
}
```

```
chapter14 > <> if.html > ...  
1  <!DOCTYPE html>  
2  <html lang="en">  
3  <head>  
4      <meta charset="UTF-8">  
5  </head>  
6  <body>  
7      <script>  
8  
9          const userName = prompt("방문자의 이름은?", "");  
10  
11          if(userName) {  
12              document.write(userName+"님 반갑습니다!");  
13          }  
14      </script>  
15  
16  </body>  
17  </html>
```

이 페이지 내용:

방문자의 이름은?

홍길동

확인

취소

홍길동님 반갑습니다!

## ■ if-else 문

- 조건에 만족 여부에 따라 서로 다른 스크립트가 수행됨

```
if( 조건식 ){
    조건식이 true 일 때 실행되는 문장
} else {
    조건식이 false 일 때 실행되는 문장
}
```

- if 구문 내에 또 다른 if 구문이 입력해 사용
  - 여러 조건들이 모두 만족하는지 조사하는데 사용

```
if( 조건식 ){
    if( 조건식 ){
    } else {
    }
} else {
}
```

chapter14 > <> if-esle.html > ...

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |
9  |       const userNumber = prompt("숫자를 입력하세요.");
10 |
11 |       if (userNumber % 3 === 0)
12 |           alert("3의 배수입니다.");
13 |       else
14 |           alert("3의 배수가 아닙니다.");
15 |
16 |   </script>
17 </body>
18 </html>
```

```
chapter14 > <> nested-if.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |
9  |       const id = "kim";
10 |       const pw = "kimpass";
11 |       const user_id = prompt("아이디는?", "");
12 |       const user_pw = prompt("비밀번호는?", "");
13 |
14 |       if(id == user_id) {
15 |           if(pw == user_pw) {
16 |               document.write(user_id+"님 반갑습니다!");
17 |           } else {
18 |               alert("비밀번호가 일치하지 않습니다.");
19 |               location.reload();
20 |           }
21 |       } else {
22 |           alert("아이디가 일치하지 않습니다.");
23 |           location.reload();
24 |       }
25 |
26 |   </script>
27 </body>
28 </html>
```

이 페이지 내용:

아이디가 일치하지 않습니다.

확인

이 페이지 내용:

비밀번호가 일치하지 않습니다.

확인

kim님 반갑습니다!



## ■ 문자 데이터를 숫자 데이터로 변환

- prompt()로 받아들인 데이터는 문자형 데이터로 간주됨
  - 연산에 참여시키면 묵시적 형변환이 발생하지만 연산에 사용하려면 가급적 숫자 데이터로 변환하는 것이 좋음
- parseInt() 함수
  - 인자로 지정된 값을 정수형 데이터(10진수)로 변환
- parseFloat() 함수
  - 인자로 지정한 값을 실수형 데이터로 변환
    - 인자가 정수인 경우 정수로 표현
    - 인자가 실수인 경우 실수로 표현
- Number() 함수
  - 인자가 정수나 실수에 관계없이 숫자형 데이터로 변경
- isNaN() 함수
  - 인자로 가지는 값이 숫자인지 조사하는 함수
    - 인자가 숫자이면 FALSE를 반환, 문자이면 TRUE를 반환
  - 주로 조건문에 사용

chapter14 > <> chatToNum1.html > ...

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5 </head>
6 <body>
7   <script>
8
9     const userNumber = prompt("숫자를 입력하세요.");
10
11     if( isNaN(userNumber)) {
12       alert('숫자를 입력해야합니다. ');
13       exit;
14     }
15
16     const myNum1 = parseInt(userNumber);
17     document.write(myNum1,"<br>");
18
19     const myNum2 = parseFloat(userNumber);
20     document.write(myNum2,"<br>");
21
22     const myNum3 = Number(userNumber);
23     document.write(myNum3);
24
25   </script>
26 </body>
27 </html>
```

이 페이지 내용:

숫자를 입력하세요.

abcd

확인

취소

이 페이지 내용:

숫자를 입력해야합니다.

확인

이 페이지 내용:

숫자를 입력하세요.

7.5

확인

취소

7  
7.5  
7.5

```
chapter14 > <> if-esle2.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |
9  |       const userNumber = prompt("숫자를 입력하세요.");
10 |
11 |       if( isNaN(userNumber)) {
12 |           alert('숫자를 입력해야합니다. ');
13 |           exit;
14 |       }
15 |
16 |       userNumber1 = parseInt(userNumber);
17 |
18 |       if (userNumber1 % 3 === 0)
19 |           alert("3의 배수입니다.");
20 |       else
21 |           alert("3의 배수가 아닙니다.");
22 |
23 |   </script>
24 </body>
25 </html>
```

이 페이지 내용:

숫자를 입력해야합니다.

확인

이 페이지 내용:

3의 배수입니다.

확인

## ■ 삼항 조건 연산자

- 연산의 수행 결과에 따라 실행되는 스크립트 코드가 다를 때 사용

조건식 ? 실행문1 : 실행문2 ;

- 조건식의 결과가 true이면 실행문1을 수행하고, 거짓인 경우 실행문2를 수행

```
chapter14 > <> operator-threestate.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |       const userNumber = prompt("숫자를 입력하세요.");
9  |       if( !isNaN(userNumber)) {
10 |           userNumber1 = parseInt(userNumber);
11 |           userNumber1 % 3 == 0 ? alert("3의 배수입니다") : alert("3의 배수가 아닙니다.");
12 |       } else {
13 |           alert('숫자를 입력해야합니다. ');
14 |           exit;
15 |       }
16 |   </script>
17 </body>
18 </html>
```

## ■ if 문과 논리 연산자의 사용

### • 걱정 체중

걱정 체중 = ( 키 - 100 ) \* 0.9

홍길동님은 걱정 체중입니다.

chapter14 > <> weight.html > ...

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5  </head>
6  <body>
7    <script>
8
9      const name = prompt("당신의 이름은?", "");
10     const height = prompt("당신의 키는?", "0");
11     const weight = prompt("당신의 몸무게는?", "0");
12
13     const normal = (height - 100) * 0.9;
14
15     let result = weight >= normal - 2 && weight <= normal + 2;
16     const message = result ? "걱정 체중입니다." : "걱정 체중이 아닙니다.";
17
18     document.write(name + "님은 " + message);
19
20   </script>
21 </body>
22 </html>
```

## switch 문

- 조건이 하나 이상일 때 사용

- 조건에 따른 분기 아닌 값에 의한 분기 수행
- 조건을 값으로 표현해야 함
- 모든 CASE 구문은 break 구문을 사용함  
인위적으로 사용하지 않는 경우도 있음

```
switch ( 변수 )
{
    case 값1 : 실행문1;
    break;
    case 값2 : 실행문2;
    break;
    ....

    case 값n : 실행문n;
    break;
    default : 실행문           // 변수와 같은 값이 없을 경우 실행
}
```

```
chapter14 > <> switch1.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5  </head>
6  <body>
7      <script>
8
9          const site = prompt("네이버, 다음, 구글 중 선택", "");
10         let url;
11
12         switch(site){
13             case "구글": url = "http://www.google.com";
14             break;
15             case "다음": url = "http://www.daum.net";
16             break;
17             case "네이버": url = "https://www.naver.com";
18             break;
19             default: alert("보기 중에 없는 사이트입니다.");
20         }
21
22         if(url) location.href = url;
23
24     </script>
25 </body>
26 </html>
```

chapter14 > <> switch2.html > ...

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5 </head>
6 <body>
7   <script>
8     const input_score = prompt("점수를 입력하세요.");
9     if( isNaN(input_score)) {
10       alert('숫자를 입력해야합니다. ');
11       exit;
12     }
13     const score = parseInt(input_score/10);
14
15     switch (score)
16     {
17       case 10:
18       case 9:
19         alert("합격입니다.");
20         break;
21       case 8:
22       case 7:
23         alert("조건부 합격입니다.");
24         break;
25       default:
26         alert("불합격입니다.");
27     }
28   </script>
29 </body>
30 </html>
```

127.0.0.1:5500 내용:  
합격입니다.

확인

127.0.0.1:5500 내용:  
조건부 합격입니다.

확인

127.0.0.1:5500 내용:  
불합격입니다.

확인

## ■ 짧은 조건문

- 논리 연산자의 특성을 조건문으로 사용

- 논리합 연산자 사용

A || B

- A가 true을 경우 B는 수행되지 않음  
논리합은 A와 B중 하나가 true이면 true이기 때문
- A가 false일 경우 B가 수행됨

- 논리합 연산자 사용

A && B

- A가 false을 경우 B는 수행되지 않음  
논리곱은 A와 B가 모두가 true여야 true이기 때문
- A가 true일 경우 B가 수행됨

```
chapter14 > <> short.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |       const input = Number( prompt('숫자를 입력하세요', '숫자'));
9  |
10 |       input % 2 == 0 || alert('홀수입니다.');
11 |       input % 2 == 0 && alert('짝수입니다');
12 |
13 |   </script>
14 </body>
15 </html>
```

127.0.0.1:5500 내용:

숫자를 입력하세요

6

확인

취소

127.0.0.1:5500 내용:

짝수입니다

확인

127.0.0.1:5500 내용:

숫자를 입력하세요

7

확인

취소

127.0.0.1:5500 내용:

홀수입니다.

확인



## 14-5 반복문 알아보기

## ■ for 반복문

- 초기값, 조건식, 증감값으로 구성되며 각각은 세미콜론(;)으로 구분
  - 주로 정확한 반복의 횟수를 알고 있을 때 사용

```
for ( 초기값; 조건식; 증감값; ){  
    실행할 문장들;  
}
```

```
for ( 초기값; 조건식; 증감값; ){  
    for ( 초기값; 조건식; 증감값; ){  
        실행할 문장들;  
    }  
}
```

### • for 반복문의 실행 순서

- ① 초기값을 설정
- ② 조건을 검사
- ③ 스크립트 실행
- ④ 증감 실행
- ⑤ ②~④의 과정을 반복 수행(조건을 만족할 때까지)

```
chapter14 > <> for1.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |   |
9  |   |   let i, j;
10 |   |
11 |   |   for (i = 1; i <= 3; i++) {
12 |   |   |   document.write("<h3>" + i + "단</h3>");
13 |   |   |   for (j = 1; j <= 9; j++) {
14 |   |   |   |   document.write(i + " X " + j + " = " + i*j + "<br>");
15 |   |   |   |   }
16 |   |   |   }
17 |   |
18 |   </script>
19 </body>
20 </html>
```

**1단**

1 X 1 = 1  
1 X 2 = 2  
1 X 3 = 3  
1 X 4 = 4  
1 X 5 = 5  
1 X 6 = 6  
1 X 7 = 7  
1 X 8 = 8  
1 X 9 = 9

**2단**

2 X 1 = 2  
2 X 2 = 4  
2 X 3 = 6  
2 X 4 = 8  
2 X 5 = 10  
2 X 6 = 12  
2 X 7 = 14  
2 X 8 = 16  
2 X 9 = 18

**3단**

3 X 1 = 3  
3 X 2 = 6  
3 X 3 = 9  
3 X 4 = 12  
3 X 5 = 15  
3 X 6 = 18  
3 X 7 = 21  
3 X 8 = 24  
3 X 9 = 27

chapter14 &gt; &lt;&gt; for2.html &gt; ...

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <style>
6     div {
7       display:inline-block;
8       padding:0 20px 30px 20px; margin:15px;
9       border:1px solid #ccc; line-height:2;
10    }
11    div h3 {
12      text-align:center; font-weight:bold;
13    }
14  </style>
15 </head>
16 <body>
17   <script>
18     let i, j;
19
20     for (i = 1; i <= 3; i++) {
21       document.write("<div>");
22       document.write("<h3>" + i + "단</h3>");
23       for (j = 1; j <= 9; j++) {
24         document.write(i + " X " + j + " = " + i*j + "<br>");
25       }
26       document.write("</div>");
27     }
28   </script>
29 </body>
30 </html>
```

1단

1 X 1 = 1  
1 X 2 = 2  
1 X 3 = 3  
1 X 4 = 4  
1 X 5 = 5  
1 X 6 = 6  
1 X 7 = 7  
1 X 8 = 8  
1 X 9 = 9

2단

2 X 1 = 2  
2 X 2 = 4  
2 X 3 = 6  
2 X 4 = 8  
2 X 5 = 10  
2 X 6 = 12  
2 X 7 = 14  
2 X 8 = 16  
2 X 9 = 18

3단

3 X 1 = 3  
3 X 2 = 6  
3 X 3 = 9  
3 X 4 = 12  
3 X 5 = 15  
3 X 6 = 18  
3 X 7 = 21  
3 X 8 = 24  
3 X 9 = 27

## ■ for-in 반복문

- 배열의 내용을 출력하기 위해 사용되는 반복문

```
for ( const 변수 in 배열명 ) {  
    실행 문장;  
}
```

- 변수는 배열을 다루기 위한 제어 변수 역할 수행
- 변수는 const 키워드로 지정하며, let 키워드로도 지정할 수 있음  
반복문 내부의 실행문장에서 변수의 값을 인위적으로 변경하는 구문이 있다면 let 키워드로 지정해야 함
- 다음의 두 구문은 동일한 동작을 수행

```
for ( let i in myArray ) {  
    실행 문장;  
}
```

```
for ( let i=0; i < myArray.length; i++ ) {  
    실행 문장;  
}
```

chapter14 > <> for-in.html > ...

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <meta charset="UTF-8">
5  </head>
6  <body>
7  |   <script>
8  |       const array = ['포도', '사과', '바나나', '망고'];
9  |
10 |
11 |       document.write("<h3>for 문을 이용한 출력</h3>");
12 |       for (let i=0; i<array.length; i++) {
13 |           document.write(array[i], "<br>");
14 |       }
15 |
16 |       document.write("<h3>for-in 문을 이용한 출력</h3>");
17 |       for (const j in array) {
18 |           document.write(array[j], "<br>");
19 |       }
20 |
21 |   </script>
22 </body>
23 </html>
```

## for 문을 이용한 출력

포도  
사과  
바나나  
망고

## for-in 문을 이용한 출력

포도  
사과  
바나나  
망고

## for-of 문

- 배열의 요소에 바로 접근해 처리할 수 있는 반복문

```
for ( const 변수 of 배열명 ) {  
    실행 문장;  
}
```

- 변수는 배열의 요소를 의미
- [참고] for-in 반복문에서의 변수는 배열 요소의 인덱스를 의미함

### for-in 문을 이용한 출력

포도  
사과  
바나나  
망고

### for-of 문을 이용한 출력

포도  
사과  
바나나  
망고

```
chapter14 > <> for-of.html > ...  
1  <!DOCTYPE html>  
2  <html lang="en">  
3  <head>  
4    <meta charset="UTF-8">  
5  </head>  
6  <body>  
7    <script>  
8      const array = ['포도', '사과', '바나나', '망고'];  
9  
10     document.write("<h3>for-in 문을 이용한 출력</h3>");  
11     for (const i in array) {  
12       document.write(array[i], "<br>");  
13     }  
14  
15     document.write("<h3>for-of 문을 이용한 출력</h3>");  
16     for (const element of array) {  
17       document.write(element, "<br>");  
18     }  
19  
20   </script>  
21 </body>  
22 </html>
```

## ■ while 문

- 조건식이 참(true)일 동안 스크립트를 반복 수행
  - 주로 반복할 횟수를 정확하게 알지 못할 때 사용

```
초기값;  
while ( 조건식 ) {  
    실행할 문장들;  
    증감값;  
}
```

- 실행 순서 ( ① ~ ③의 과정을 반복 수행 )
  - ① 조건식을 검사
  - ② 조건식이 참인 경우 문장과 증감식을 수행 (거짓이면 순환 탈출)
  - ③ 다시 조건식을 검사

## ■ do-while 문

```
초기값;  
do {  
    실행할 문장들;  
    증감값;  
} while ( 조건식 )
```

- while 반복문과 do-while 반복문의 차이
  - while 반복문
    - 조건을 비교한 후 실행
    - 조건에 따라 실행되지 않을 수도 있음 )
  - do-while 반복문
    - 실행한 다음 조건을 비교함
    - 조건에 관계 없이 무조건 1회 실행됨 )



chapter14 > <> while.html > html > body > script

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5 </head>
6 <body>
7   <script>
8     let i, dan, result;
9     const num = prompt("원하는 단을 입력하세요.", 0);
10
11     if( isNaN(num) ) {
12       alert("숫자를 입력하세요");
13       exit;
14     }
15     dan = parseInt(num);
16
17     document.write("<h3>" + dan + "단</h3>");
18     i=1;
19     while ( i < 10 )
20     {
21       result = dan*i;
22       document.write(dan + " * " + i + " = " + result + "<BR>");
23       i++;
24     }
25   </script>
26 </body>
27 </html>
```

127.0.0.1:5500 내용:

원하는 단을 입력하세요.

확인

취소

## 7단

7 \* 1 = 7  
7 \* 2 = 14  
7 \* 3 = 21  
7 \* 4 = 28  
7 \* 5 = 35  
7 \* 6 = 42  
7 \* 7 = 49  
7 \* 8 = 56  
7 \* 9 = 63

## ■ break 구문

- 특정 조건에 만족하면 현재의 순환문을 완전히 탈출함
  - 조건에 따라 순환문의 수행을 중단하고자 할 경우 사용

```
for ( 초기값; 조건식; 증감값; ){  
    if ( 조건 ) break;    // 반복문 탈출  
    실행할 문장들;  
}
```

```
초기값;  
while ( 조건식 ) {  
    if ( 조건 ) break;    // 반복문 탈출  
    실행할 문장들;  
    증감값;  
}
```

```
chapter14 > <> break.html > ...  
1  <!DOCTYPE html>  
2  <html lang="en">  
3  <head>  
4      <meta charset="UTF-8">  
5  </head>  
6  <body>  
7      <script>  
8            
9          for(let i = 1; i <= 10; i++){  
10             if(i == 6) break;  
11             document.write(i, "<br>");  
12         }  
13           
14         document.write("=== The End ===");  
15     }  
16 </script>  
17 </body>  
18 </html>
```

```
1  
2  
3  
4  
5  
=== The End ===
```

## ■ continue 문

- 특정 조건에 만족해도 현재의 순환문을 완전히 탈출하지 않음
  - continue문 하단의 실행 문장만을 수행하지 않고 순환문을 계속 수행
  - 조건에 따라 순환문 내의 일부 실행 문장을 수행하지 않을 경우 사용

```
for ( 초기값; 조건식; 증감값; ) {  
    if ( 조건 ) continue; // 현재 순환만 탈출  
    실행할 문장들;  
}
```

```
초기값;  
while ( 조건식 ) {  
    if ( 조건 ) continue; // 현재 순환만 탈출  
    실행할 문장들;  
    증감값;  
}
```

```
chapter14 > <> continue.html > ...  
1  <!DOCTYPE html>  
2  <html lang="en">  
3  <head>  
4      <meta charset="UTF-8">  
5  </head>  
6  <body>  
7      <script>  
8  
9          for(let i = 1; i <= 10; i++){  
10             if(i == 6) continue;  
11             document.write(i, "<br>");  
12         }  
13  
14         document.write("=== The End ===");  
15     </script>  
16 </body>  
17 </html>  
18
```

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
=== The End ===
```

수고하셨습니다