01 자바스크립트 기본 용어와 출력 방법

- **자바스크립트 기본 용어**
  - 표현식 - 값을 만들어 내는 간단한 코드
  - 문장 - 프로그래밍 언어에 실행할 수 있는 코드의 최소 단위
  - 종결 - 문장 마지막에 세미콜론(;) 또는 줄 바꿈

```
273;
10 + 20 + 30 * 2;
var name = '홍' + '길' + '동';
alert('Hello JavaScript');
```

(a) 표현식 예

```
273
10 + 20 + 30 * 2
'JavaScript'
```

(b) 문장 예

표현식과 문장 예

01 자바스크립트 기본 용어와 출력 방법

▪ 자바스크립트 기본 용어

- 키워드 - 자바스크립트를 처음 만들 때 정해진 특별한 의미가 부여된 단어

자바스크립트 키워드

break	else	instanceof	true	case	false
new	try	catch	finally	null	typeof
continue	for	return	var	default	function
switch	void	delete	if	this	while
do	in	throw	with	const	class

- 식별자 - 자바스크립트에서 변수나 함수 등에 이름을 붙일 때 사용하는 단어

사용할 수 있는 예
```
alpha
alpha10
_alpha
$alpha
AlPha
ALPHA
```

사용할 수 없는 예
```
break
273alpha
has space
```

(a) 식별자 예

- 키워드를 사용하면 안 됩니다.
- 특수 문자는 _과 $만 허용합니다.
- 숫자로 시작하면 안 됩니다.
- 공백은 입력하면 안 됩니다.

(b) 식별자 생성 규칙

식별자 예와 식별자 생성 규칙

01 자바스크립트 기본 용어와 출력 방법

▪ 자바스크립트 기본 용어

▪ 식별자 생성 규칙

```
i love you → iLoveYou
i am a boy → iAmABoy
create server → createServer
```

글자가 모두 붙어 있지만 대·소문자로 구분했으므로 쉽게 끊어서 읽을 수 있습니다.

식별자 생성 관례

❶ 생성자 함수 이름은 항상 대문자로 시작합니다.

❷ 변수, 인스턴스, 함수, 메서드의 이름은 항상 소문자로 시작합니다.

❸ 여러 단어로 된 식별자는 각 단어의 첫 글자를 대문자로 합니다.

자바스크립트의 식별자 종류

구분	단독으로 사용	다른 식별자와 함께 사용
식별자 뒤에 괄호 없음	변수	속성
식별자 뒤에 괄호 있음	함수	메서드

01 자바스크립트 기본 용어와 출력 방법

- **자바스크립트 기본 용어**
  - 주석 - 프로그램 진행에 전혀 영향을 주지 않는 코드로, 프로그램을 설명하는 데 사용

주석 처리 방법

방법	형태
❶ 한 행 주석 처리	// 주석문
❷ 여러 행 주석 처리	/* 주석문 주석문 */

```
<script>
    // 주석은 코드 실행에 영향을 주지 않습니다.
    /*
    alert('Hello JavaScript .. !');
    alert('Hello JavaScript .. !');
    alert('Hello JavaScript .. !');
    */
</script>
```

01 자바스크립트 기본 용어와 출력 방법

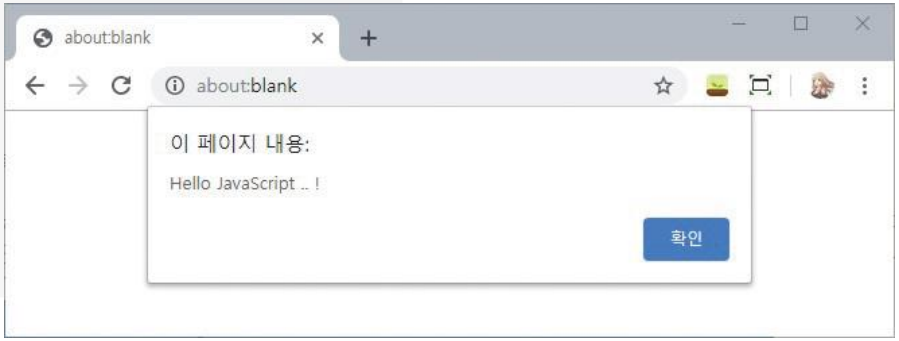
- **자바스크립트 출력**
  - 가장 기본적인 출력 방법 - alert( ) 함수를 사용해 웹 브라우저에 경고 창을 띄우기

alert() 함수 형태

- **자바스크립트를 이용한 메시지 출력**

output_alert.html

```
<!DOCTYPE html>
<html>
<head>
    <title>JavaScript Basic</title>
    <script>
        alert('Hello JavaScript .. !');
    </script>
</head>
<body>

</body>
</html>
```



02 자료형과 변수

- **자료형**
  - 숫자
    - 가장 기본적인 자료형
    - 정수와 실수 구분하지 않음

사칙 연산자

연산자	설명	예
+	덧셈	> 52 + 273 325 > 52.273 + 103.57 155.843
-	뺄셈	> 52 - 273 -221 > 52.273 - 103.57 -51.29699999999999

자바스크립트는 부동소수점(소수점을 포함한 숫자)을 계산할 때 약간의 오차를 발생시킵니다.

연산자	설명	예
*	곱셈	> 52 * 273 14196 > 52.273 * 103.57 5413.91461
/	나눗셈	> 52 / 273 0.19047619047619047 > 52.273 / 103.57 0.504711789128126 > 1 / 0 Infinity

자바스크립트는 어떤 숫자를 0으로 나누었을 때, 무한을 나타내는 값Infinity이 됩니다.

나머지 연산자

연산자	설명	예
%	나머지	> 10 % 5 0 > 7 % 3 1

02 자료형과 변수

- **자료형**
  - 문자열
    - 문자 집합
    - 'abcdefg', 'Hello World', '안녕하세요.'

문자열 생성

방법	예
작은따옴표 사용	> 'Hello JavaScript .. !' "Hello JavaScript .. !" > '"문자열"입니다.' ""문자열"입니다."
큰따옴표 사용	> "Hello JavaScript .. !" "Hello JavaScript .. !" > "'문자열'입니다." "'문자열'입니다."

이스케이프 문자

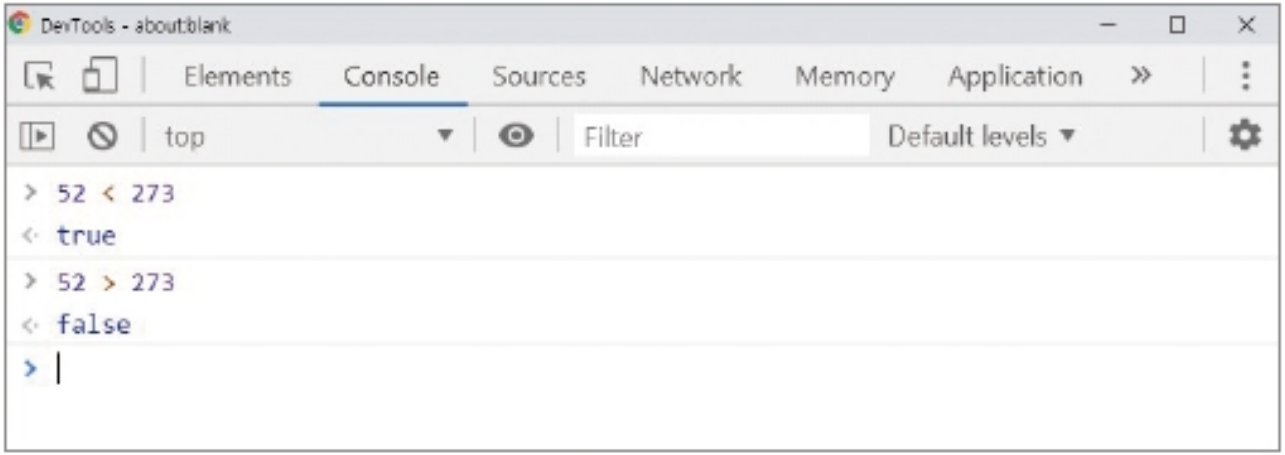
이스케이프 문자	설명	예
₩t	수평 탭	> '한빛\t아카데미' "한빛 아카데미"
₩n	행 바꿈	> '한빛\n아카데미' "한빛 아카데미"
₩₩	역 슬래시	> '\\\\' \\
₩'	작은따옴표	> '\'\'\'' "'''"
₩"	큰따옴표	> "\"\"\"" """"

문자열 연결 연산자

연산자	설명	예
+	문자열 연결	> '가나다' + '라마' + '바사아' + '자차카타' + '파하' "가나다라마바사아자차카타파하"

02 자료형과 변수

- **자료형**
  - 불(bool)
    - 참과 거짓을 표현할 때 사용하는 자료



불 사용 예

02 자료형과 변수

- **자료형**
  - 비교 연산자
    - 두 대상을 비교할 수 있는 연산자

비교 연산자

연산자	설명	예
>=	좌변이 우변보다 크거나 같음	> 10 >= 20 false > '가방' > '하마' false
<=	우변이 좌변보다 크거나 같음	> 10 <= 20 true
>	좌변이 우변보다 큼	> 10 > 20 false
<	우변이 좌변보다 큼	> 10 < 20 true
==	좌변과 우변이 같음	> 10 == 20 false
!=	좌변과 우변이 다름	> 10 != 20 true

문자열 순서도 비교 가능합니다.

02 자료형과 변수

- **자료형**
  - 논리 연산자

논리 연산자

연산자	설명	예
!	논리 부정(참이면 거짓, 거짓이면 참)	> !true false > !(10 == 10) false
&&	논리곱(둘 다 참이어야 참)	> true && true true — 둘 다 참일 때만 참입니다. > true && false false > false && true false > false && false false > (10 < 20) && (20 < 30) true — 20이 10~30 사이에 있다는 것을 나타낼 때는 이러한 형태로 표현해야 합니다.
||	논리합(둘 중 하나만 참이어도 참)	> true || true true > true || false true > false || true true — 둘 중 하나만 참이어도 참입니다. > false || false false

02 자료형과 변수

- **변수**
  - 값을 저장할 때 사용하는 식별자
  - ❶ 변수를 선언합니다.
  - ❷ 변수를 초기화합니다.

변수 선언과 초기화

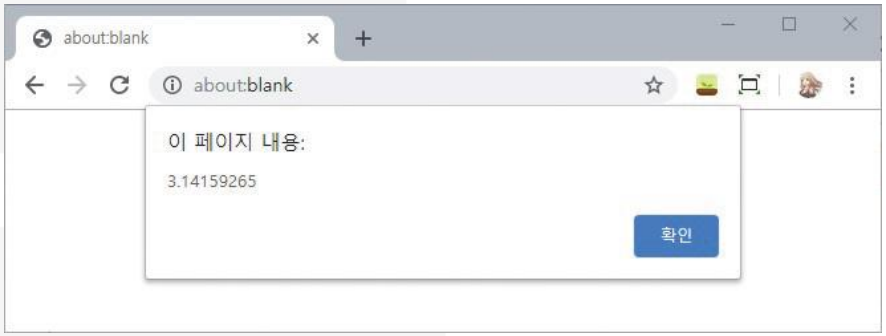
```
> var pi = 3.14159265;
undefined
```

변수 선언과 값 할당

```
❶ > var pi;                  // 변수 선언
   undefined
❷ > pi = 3.14159265;         // 값 할당
   undefined
```

변수에 저장된 값 출력

```
> var pi = 3.14159265;
undefined
> alert(pi);
undefined
```



02 자료형과 변수

- **자바스크립트를 이용한 메시지 출력(1)**
  - **1. HTML 페이지 만들기**

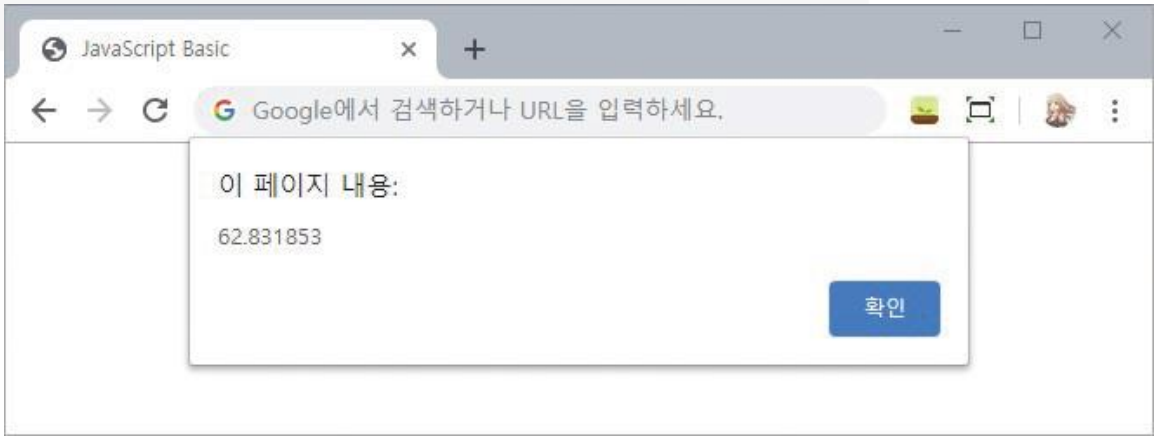
variable_use.html

```
<!DOCTYPE html>
<html>
<head>
    <title>JavaScript Basic</title>
    <script>

    </script>
</head>
<body>

</body>
</html>
```

- **자바스크립트를 이용한 메시지 출력(2)**
  - **2. 변수 사용하기**

variable_use.html

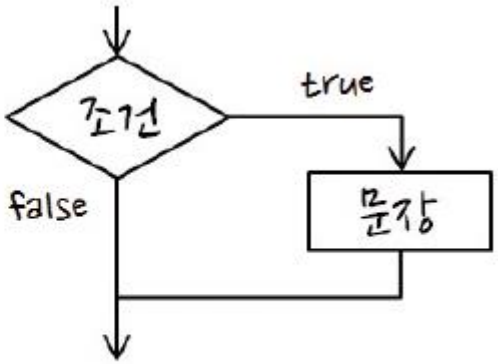
```
<script>
    // 변수를 선언 및 초기화합니다.
    var radius = 10;
    var pi = 3.14159265;

    // 출력합니다.
    alert(2 * radius * pi);
</script>
```



03 조건문과 반복문

- **조건문**
  - If 조건문
    - 조건이 true이면 문장을 실행하고 false이면 문장 무시
    - 실행 문장이 한 행이면 중괄호 생략 가능함
    - 실행 문장이 여러 행이라면 중괄호 필요함

```
if (조건) {
    문장
}
```



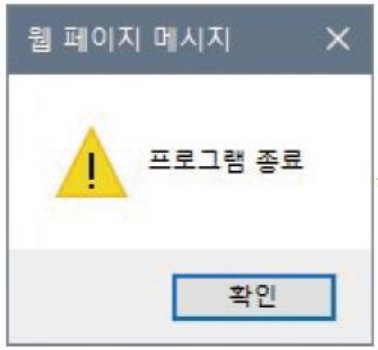
if 조건문 기본 형태와 순서도

03 조건문과 반복문

- **if 조건문으로 참과 거짓 판별**

condition_basic.html

```
<script>
    // 조건문
    if (273 < 52) {
        // 표현식 "273 < 52"가 참일 때 실행합니다.
        alert('273 < 52 => true');
    }
    // 프로그램 종료
    alert('프로그램 종료');
</script>
```



크롬은 경고 창이 너무 넓게 출력되므로 인터넷 익스플로러의 경고 창을 스크린샷으로 넣었습니다. 크롬에서 이러한 모양의 경고 창이 출력되지 않는다고 당황하지 마세요.

03 조건문과 반복문

- **if 조건문으로 오전과 오후 판별**
  - **1. 현재 시간 구하기**

condition_getDate.html

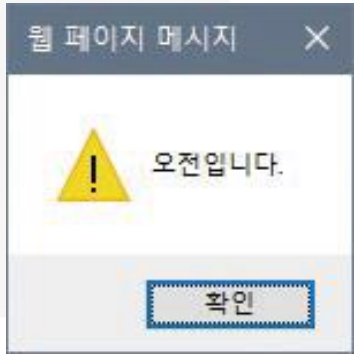
```
<script>
    // Date 객체를 선언합니다: 현재 시간 측정
    var date = new Date();

    // 요소를 추출합니다.
    var year = date.getFullYear();
    var month = date.getMonth() + 1;
    var day = date.getDay();
    var hours = date.getHours();
    var minutes = date.getMinutes();
    var seconds = date.getSeconds();
</script>
```

03 조건문과 반복문

- **if 조건문으로 오전과 오후 판별**
  - **2. 오전과 오후 구분하기**

condition_date.html

```
<script>
    // 변수를 선언합니다.
    var date = new Date();
    var hours = date.getHours();

    // 조건문
    if (hours < 12) {
        // 표현식 "hours < 12"가 참일 때 실행합니다.
        alert('오전입니다.');
    }

    if (12 <= hours) {
        // 표현식 "12 <= hours"가 참일 때 실행합니다.
        alert('오후입니다.');
    }
</script>
```



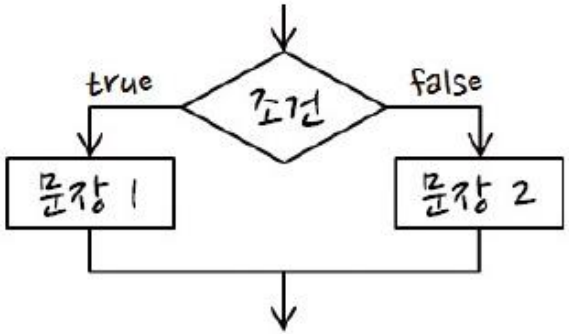
03 조건문과 반복문

조건문

If else 조건문

- 두 가지로 분명하게 나뉘는 상황에서 편리하게 사용 가능
- 실행 문장이 한 행이면 중괄호 생략 가능함
- 실행 문장이 여러 행이라면 중괄호 필요함

```
if (조건) {
    문장 1
} else {
    문장 2
}
```



if else 조건문 기본 형태와 순서도

03 조건문과 반복문

- **if else 조건문으로 오전과 오후 판별**

condition_else.html

```
<script>
    // 변수를 선언합니다.
    var date = new Date();
    var hours = date.getHours();

    // 조건문
    if (hours < 12) {
        // 표현식 "hours < 12"가 참일 때 실행합니다.
        alert('오전입니다.');
    } else {
        // 표현식 "hours < 12"가 거짓일 때 실행합니다.
        alert('오후입니다.');
    }
</script>
```

03 조건문과 반복문

- **조건문**
  - 중첩 조건문과 if else if 조건문

```
if (조건) {
    if (조건) {
        문장
    } else {
        문장
    }
} else {
    if (조건) {
        문장
    } else {
        문장
    }
}
```

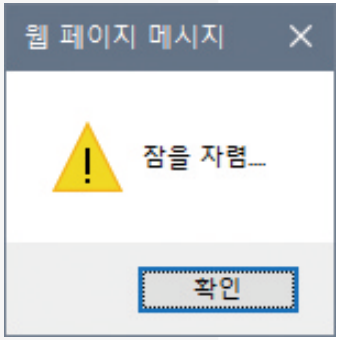
중첩 조건문 형태

03 조건문과 반복문

- **중첩 조건문으로 하루 일정 표현**

condition_ifelseif.html

```
<script>
    // Date 객체를 선언합니다: 현재 시간 측정
    var date = new Date();
    var hours = date.getHours();

    // 조건문
    if (hours < 5) {
        alert('잠을 자렴....');
    } else if (hours < 7) {
        alert('준비');
    } else if (hours < 9) {
        alert('출근');
    } else if (hours < 12) {
        alert('빈둥빈둥');
    } else if (hours < 14) {
        alert('식사');
    } else {
        // 여러 가지 업무를 수행합니다.
    }
</script>
```



03 조건문과 반복문

- **조건문**
  - 중첩 조건문의 중괄호를 생략했을 때 만드는 조건문
    - if else if 조건문

```
if (조건) {
    문장
} else if (조건) {
    문장
} else if (조건) {
    문장
} else {
    문장
}
```

if else if 조건문 형태

03 조건문과 반복문

- **if else if 조건문으로 하루 일정 표현**

condition_duplication.html

```
<script>
    // Date 객체를 선언합니다: 현재 시간 측정
    var date = new Date();
    var hours = date.getHours();

    // 조건문
    if (hours < 5) {
        alert('잠을 자렴....');
    } else {
        if (hours < 7) {
            alert('준비');
        } else {
            if (hours < 9) {
                alert('출근');
            } else {
                if (hours < 12) {
                    alert('빈둥빈둥');
                } else {
                    if (hours < 14) {
                        alert('식사');
                    } else {
                        // 여러 가지 업무를 수행합니다.
                    }
                }
            }
        }
    }
</script>
```

03 조건문과 반복문

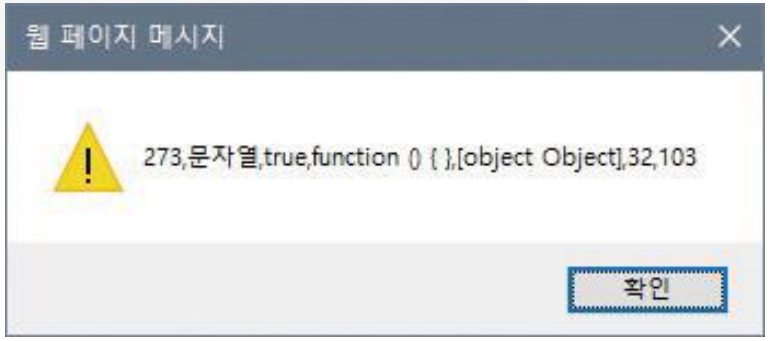
▪ 반복문

▪ 배열

- 변수 여러 개를 한꺼번에 다룰 수 있는 자료형
- 요소 - 배열 내부에 입력된 자료 하나하나
- 배열 내부에 다양한 자료형을 입력 가능
- 배열 전체를 출력하면 요소가 순서대로 표시

array_basic.html

```
<script>
    // 변수를 선언합니다.
    var array = [273, '문자열', true, function () { }, {}, [32, 103]];
    alert(array);
</script>
```

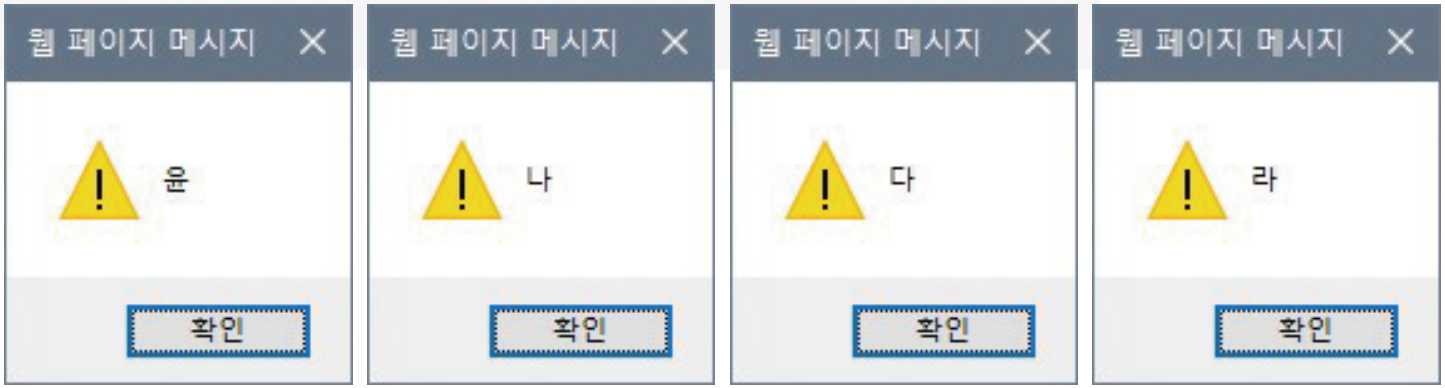


03 조건문과 반복문

- **배열 생성과 배열 요소 접근**

loop_array.html

```
<script>
    // 변수를 선언합니다.
    var array = ['가', '나', '다', '라'];

    // 배열 요소를 변경합니다.
    array[0] = '윤';
    // 요소를 출력합니다.
    alert(array[0]);
    alert(array[1]);
    alert(array[2]);
    alert(array[3]);
</script>
```



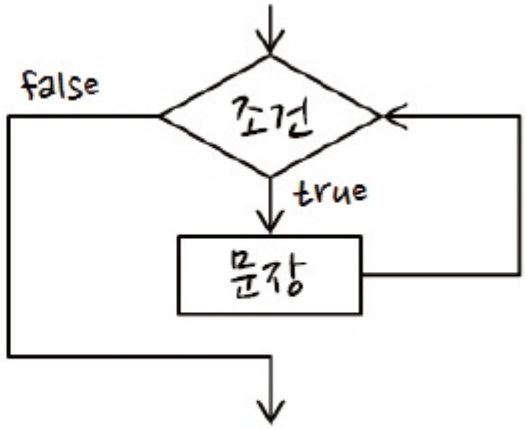
03 조건문과 반복문

▪ 반복문

▪ while 반복문

- 가장 기본적인 반복문
- if 조건문과 형식이 비슷하지만,
  if 조건문과 달리 불(bool) 표현식이 참이면 중괄호 안 문장을 계속 실행

```
while (조건) {
    문장
}
```



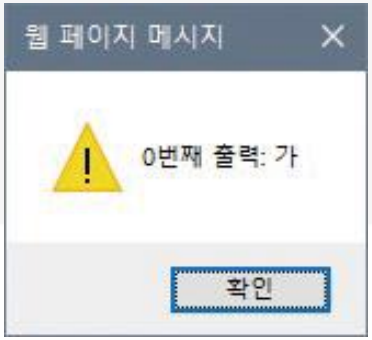
while 반복문 기본 형태와 순서도

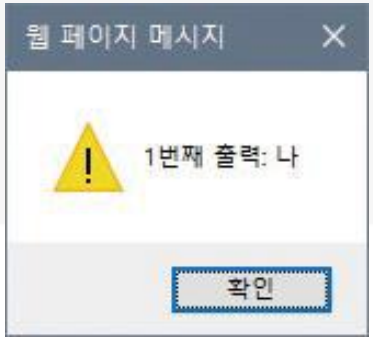
03 조건문과 반복문

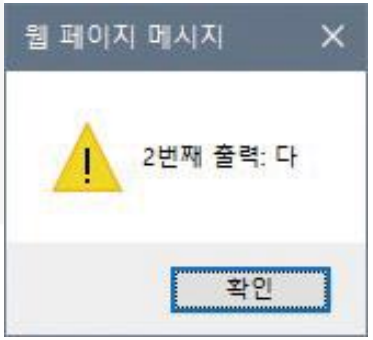
- **while 반복문**

loop_while.html

```
<script>
    // 변수를 선언합니다.
    var i = 0;
    var array = ['가', '나', '다'];

    // 반복을 수행합니다. i가 배열 원소 개수인 3보다 작을 때 반복합니다.
    while (i < array.length) {
        // 출력합니다.
        alert(i + '번째 출력: ' + array[i]);

        // 탈출하려고 변수를 더합니다.
        i++;
    }
</script>
```







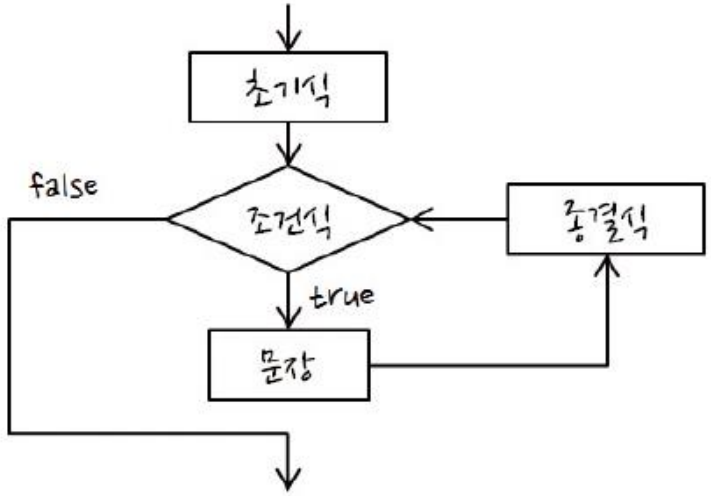
03 조건문과 반복문

▪ 반복문

▪ for 반복문

- for 반복문은 원하는 횟수만큼 반복하고 싶을 때 사용
- ❶ 초기식을 비교합니다.
- ❷ 조건식을 비교합니다. 조건이 거짓이면 반복문을 종료합니다.
- ❸ 문장을 실행합니다.
- ❹ 종결식을 실행합니다.
- ❺ 앞의 ❷ 단계로 이동합니다.

```
for (초기식; 조건식; 종결식) {
    문장
}
```



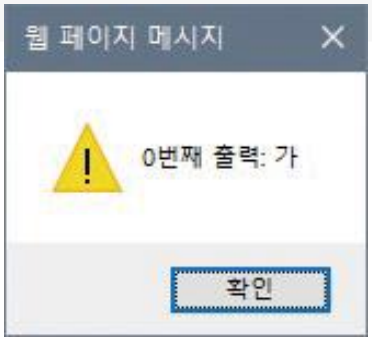
for 반복문 기본 형태와 순서도

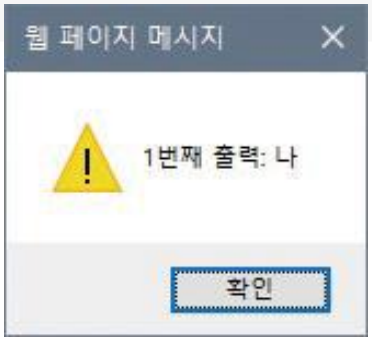
03 조건문과 반복문

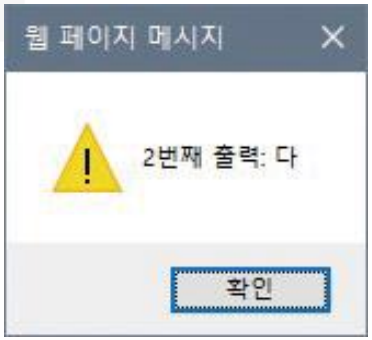
- **for 반복문**

loop_for.html

```
<script>
    // 변수를 선언합니다.
    var array = ['가', '나', '다'];

    // 반복을 수행합니다.
    for (var i = 0; i < 3; i++) {
        // 출력합니다.
        alert(i + '번째 출력: ' + array[i]);
    }
</script>
```







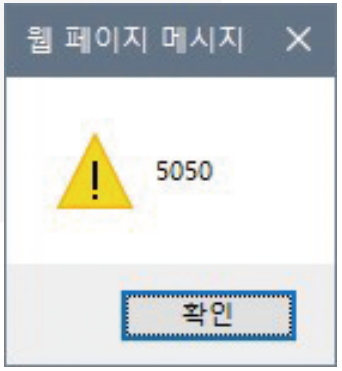
03 조건문과 반복문

- **for 반복문을 사용한 0부터 100까지 합 계산**

loop_forPlus.html

```
<script>
    // 변수를 선언합니다.
    var output = 0;

    // 반복을 수행합니다.
    for (var i = 0; i <= 100; i++) {
        output += i;
    }

    // 출력합니다.
    alert(output);
</script>
```

100까지 더해야 하므로 <= 연산자를 사용합니다.



04 함수

선언과 호출, 실행 우선순위

선언과 호출

- 함수 - 코드 집합을 나타내는 자료형
- 익명 함수 생성 - 함수 이름을 입력하지 않고 만들기
- 선언적 함수 생성 - 함수 이름을 입력해서 만들기

함수 생성 방법

방법	표현
익명 함수	`function () {  }`

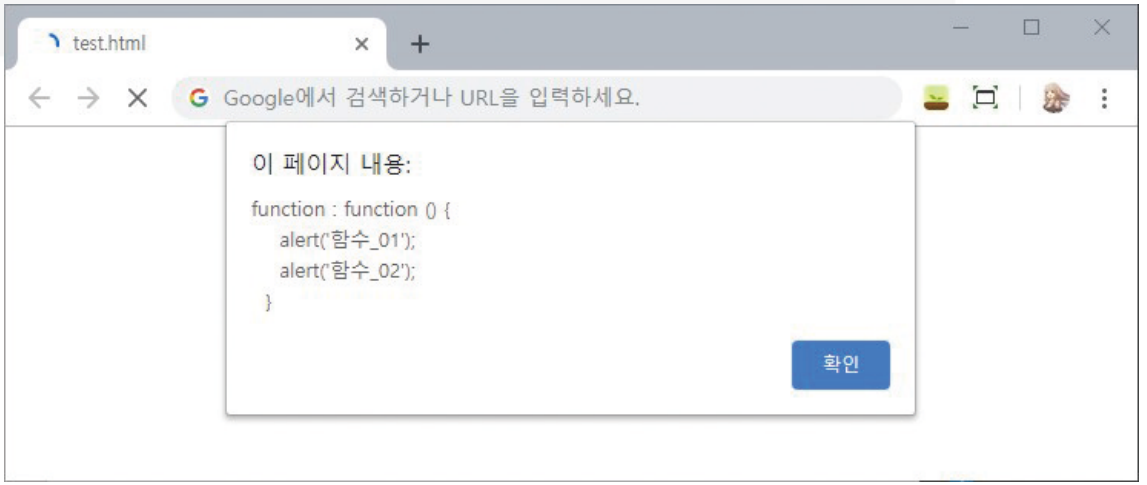
선언적 함수	`function 함수() {`
	
	`}`

03 조건문과 반복문

- **함수 선언**
  - **1. 익명 함수 선언하기**

function_noname.html

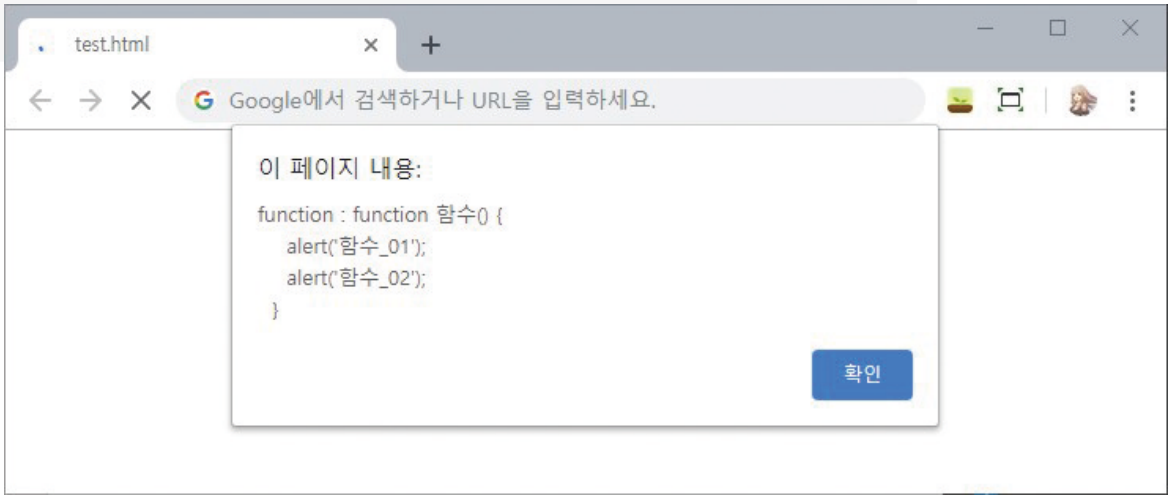
```
<script>
    // 함수를 선언합니다.
    var 함수 = function () {
        alert('함수_01');
        alert('함수_02');
    };
    // 출력합니다.
    alert(typeof (함수) + ' : ' + 함수);
</script>
```



- **함수 선언**
  - **2. 선언적 함수 선언하기**

function_name.html

```
<script>
    // 함수를 선언합니다.
    function 함수() {
        alert('함수_01');
        alert('함수_02');
    };

    // 출력합니다.
    alert(typeof (함수) + ' : ' + 함수);
</script>
```



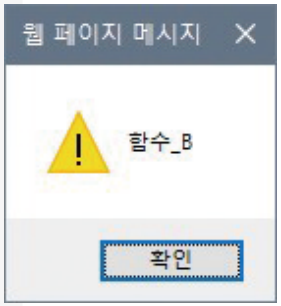
선언과 호출, 실행 우선순위

실행 우선순위

- 가장 마지막에 입력된 값이 저장

function_priorityBetweenNoname.html

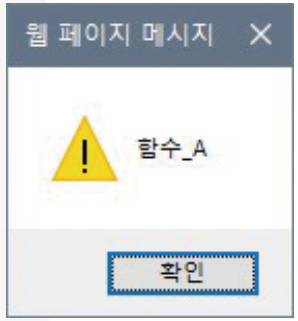
```
<script>
    // 함수를 선언합니다.
    var 함수 = function () { alert('함수_A'); };
    var 함수 = function () { alert('함수_B'); };

    // 함수를 호출합니다.
    함수();
</script>
```



- ※주의 ※ 선언적 함수와 익명 함수를 함께 사용할 때
  모든 코드를 읽기 전에 선언적 함수를 먼저 읽음

unction_priority.html

```
<script>
    // 함수를 선언합니다.
    var 함수 = function () { alert('함수_A'); };
    function 함수() { alert('함수_B'); };

    // 함수를 호출합니다.
    함수();
</script>
```

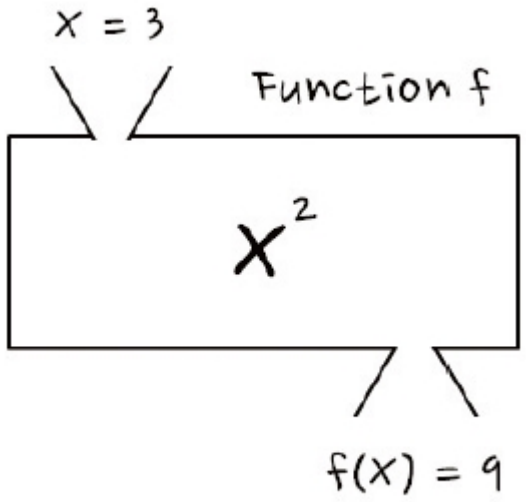


매개변수와 반환 값

- 매개변수
  - 함수의 괄호 안에 집어넣어 함수 쪽에 추가적인 정보를 전달하는 것
- 리턴 값
  - 함수를 실행한 결과를 반환 값

```
function 함수 이름(매개변수, 매개변수, 매개변수) {
    // 함수 코드
    // 함수 코드
    // 함수 코드
    return 반환 값;
}
```

매개변수와 반환 값을 가지는 함수 형식



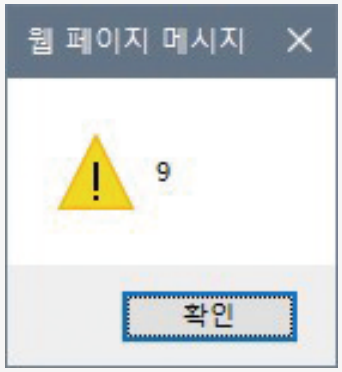
함수

03 조건문과 반복문

- **매개변수와 반환 값이 있는 함수**

function_return.html

```
<script>
    // 함수를 선언합니다.
    function f(x) {
        return x * x;
    }

    // 함수를 호출합니다.
    alert(f(3));
</script>
```



04 함수

- **콜백 함수**
  - 매개변수로 전달되는 함수
- **콜백 함수**

function_callback.html

```
<script>
    // 함수를 선언합니다.
    function callTenTimes(callback) {
        // 10회 반복합니다.
        for (var i = 0; i < 10; i++) {
            callback(); // 매개변수로 전달된 함수를 호출합니다.
        }
    }

    // 변수를 선언합니다.
    var callback = function () {
        alert('함수 호출');
    };
    callTenTimes(callback); // 함수를 호출합니다.
</script>
```

열 번 출력됩니다.

- **콜백 함수**
  - **익명 함수로 실행하기**

' reference_nonameCallback.html

```
<script>
    // 함수를 선언합니다.
    function callTenTimes(callback) {
        for (var i = 0; i < 10; i++) {
            callback();
        }
    }

    // 함수를 호출합니다.
    callTenTimes(function () {
        alert('함수 호출');
    });
</script>
```

05 객체

- **객체 개요**
  - 객체는 자료형 여러 개를 한 번에 저장
  - 배열은 요소에 접근할 때 인덱스를 사용하지만, 객체는 키를 사용

object_create.html

```
<script>
  // 객체를 선언합니다.
  var product = {
     제품명: '7D 건조 망고',
     유형: '당절임',
     성분: '망고, 설탕, 메타중아황산나트륨, 치자황색소',
     원산지: '필리핀'
  };
</script>
```

키	속성
제품명	7D 건조 망고
유형	당절임
성분	망고, 설탕, 메타중아황산나트륨, 치자황색소
원산지	필리핀

객체 생성 예

05 객체

객체 개요

- 객체 뒤의 대괄호를 사용해 키를 입력하면 객체 속성에 접근

```
product['제품명'] → '7D 건조 망고'
product['유형']   → '당절임'
product['성분']   → '망고, 설탕, 메타중아황산나트륨, 치자황색소'
product['원산지'] → '필리핀'
```

생성한 객체 속성에 접근하는 예 1

- 객체 뒤의 점(.)을 찍어 객체 속성에 접근

```
product.제품명   → '7D 건조 망고'
product.유형     → '당절임'
product.성분     → '망고, 설탕, 메타중아황산나트륨, 치자황색소'
product.원산지   → '필리핀'
```

생성한 객체 속성에 접근하는 예 2

객체 개요

- for in 반복문
  - 객체 요소를 하나씩 살펴볼 수 있음

```
for (var 키 in 객체) {
    문장
}
```

for in 반복문 형태

object_withForIn.html

```
<script>
    // 객체를 선언합니다.
    var product = {
        제품명: '7D 건조 망고',
        유형: '당절임',
        성분: '망고, 설탕, 메타중아황산나트륨, 치자황색소',
        원산지: '필리핀'
    };

    // 출력합니다.
    for (var i in product) {
        alert(i + ':' + product[i]);
    }
</script>
```

05 객체

- **속성과 메서드**
  - 요소(element)
    - 배열에 있는 값 하나하나
  - 속성(property)
    - 객체에 있는 값 하나하나

```
// 객체를 선언합니다.
var object = {
    number: 273,
    string: 'rintiantta',
    boolean: true,
    array: [52, 273, 103, 32],
    method: function () {
    }
};
```

다양한 자료형의 객체 속성 예

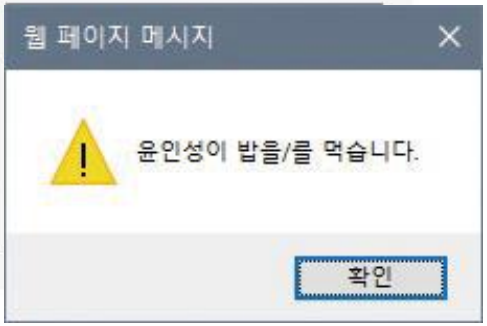
속성과 메서드

this 키워드

- 객체에 있는 속성을 메서드에서 사용하고 싶을 때는 자신이 가진 속성임을 분명하게 표시해야 함

object_this.html

```
<script>
    // 객체를 선언합니다.
    var person = {
        name: '윤인성',
        eat: function (food) {
            alert(this.name + '이 ' + food + '을/를 먹습니다.');
        }
    };

    // 메서드를 호출합니다.
    person.eat('밥');
</script>
```



Thank You