# INTERACTION PROGRAMMING 1

인터랙션 프로그래밍 1

*2Week.*

*2018. 3. 15.*

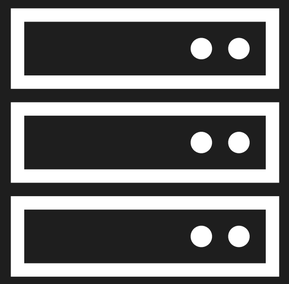
# JavaScript


# JavaScript

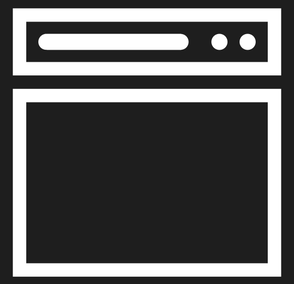
# WEB

# WEB

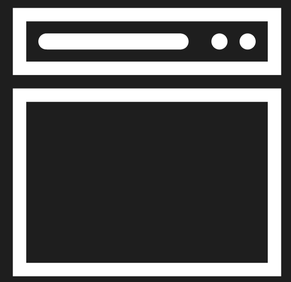
Web Server



Web Browser



# Web Browser



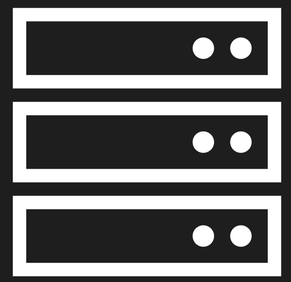
# JavaScript 작성법

## JavaScript 작성법

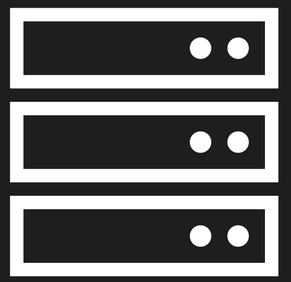
```
<!-- JavaScript -->
<script type="text/javascript">
    //code.
</script>
```

# JavaScript 의 탈 브라우저화

Web Server

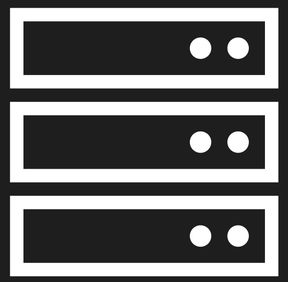


Web Server



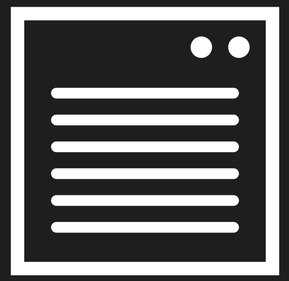
- PHP
- JAVA
- Python
- Ruby
- C

Web Server




Node.js

# Google Apps Script

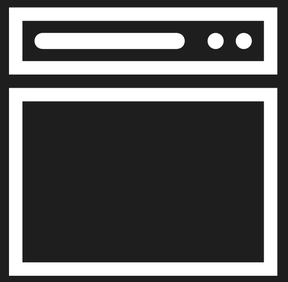


# Langage

# Environment

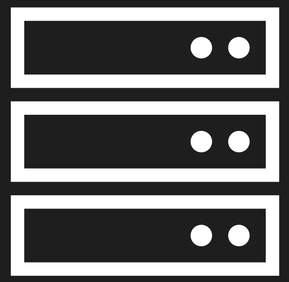
## Environment 환경

### Web Browser



alert();

### Web Server



write();

### Google Apps Script

msgBox();

User — UI

Developer — CODE

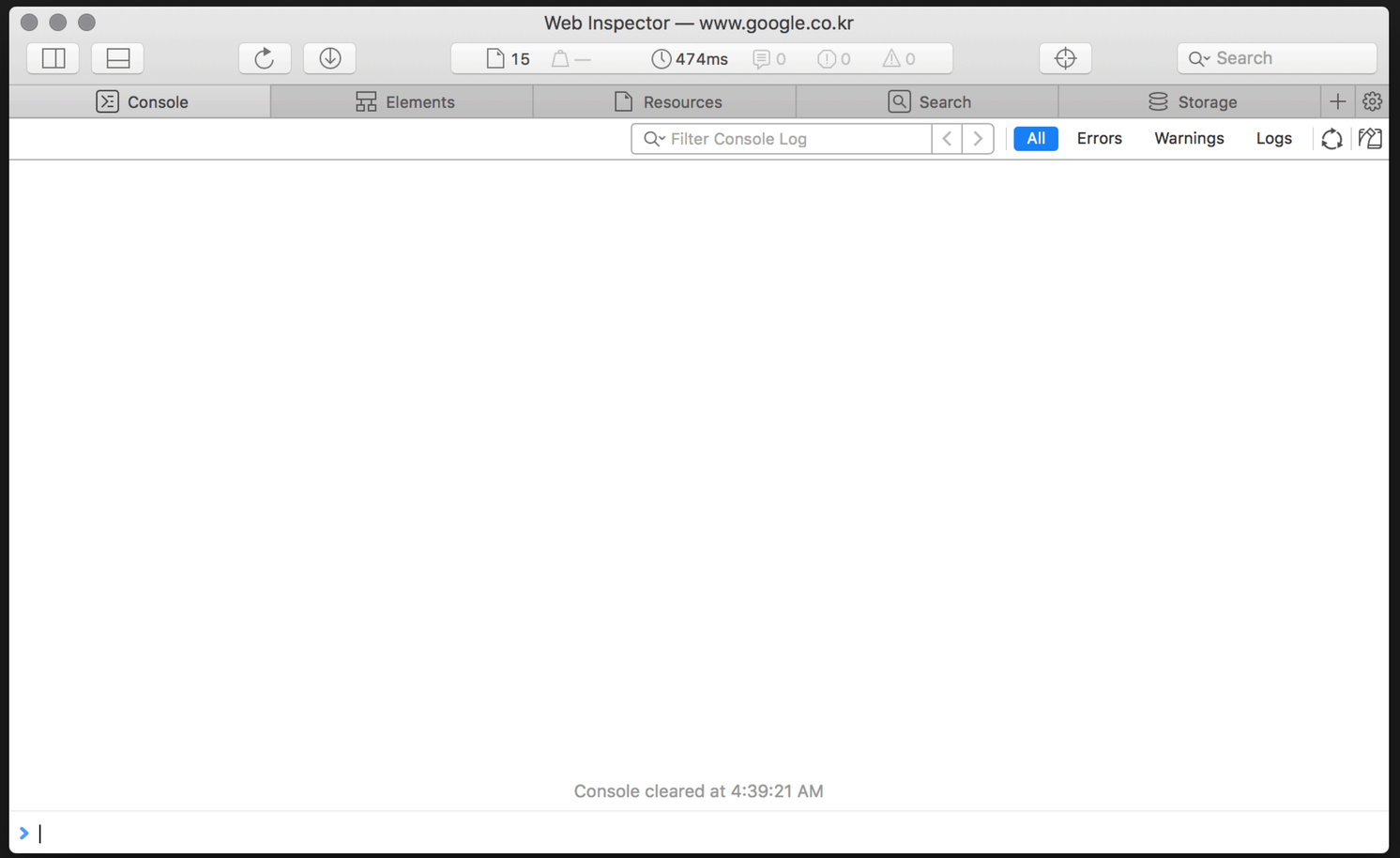
# JavaScript

## JavaScript 작성법

```
<!-- JavaScript -->
<script type="text/javascript">
    //code.
</script>
```

```
<script type="text/javascript">
    console.log('안녕하세요.');
    alert('김용원입니다.')
</script>
```

# Web Inspector



alert

## alert 경고창

```
alert('    ');
```

# console.log

## console.log 콘솔창의 로그 기록

```
console.log('    ');
```

comment

## comment 주석

```
//주석.

/*
주석(여러줄).
*/
```

# comment 주석

```
//주석.

/*
주석(여러줄).
*/
```

브라우저에서 실행시켜도 아무런 반응이 없다.
명령을 읽지 않고 무시한다.

## HTML 의 주석

```
<!--  -->
```

;

; 줄바꿈

```
console.log('    ');
alert('    ');
```

; 줄바꿈

```
console.log('    ');
alert('    ');
```

# ; 줄바꿈

```
console.log('    ');
alert('    ');
```

줄바꿈, ; (세미콜론)
명령이 끝났다는 것을 명시적으로 사용하는 기호

# Number

## Number 숫자

```
//정수.
1+1;
```

딱 떨어지는 숫자.

```
//실수.
1.5+1.5;
```

소수점이 있는 수, 현실을 반영한 수

## Number 숫자

```
//정수.
1+1;
```

딱 떨어지는 숫자.

```
//실수.
1.5+1.5;
```

소수점이 있는 수, 현실을 반영한 수

**JavaScript** 에서는 정수 / 실수 구분이 중요하지 않음.
다른 언어 (**C**, **JAVA**...) 에서는 중요함.

# Operator

# Operator 연산자

+ − * / %

# Operator 연산자

```
+ - * / %

1 + 1;
10 - 1;
2 * 2;
9 / 3;
8 % 2;
```

# Math

# Math 수학

```
Math
Math.pow          제곱
Math.round        반올림
Math.ceil         올림
Math.floor        내림
Math.sqrt         제곱근
Math.random       랜덤
```

## Math 수학

```
Math.pow(3, 2);
Math.round(1.4);
Math.ceil(1.2);
Math.floor(1.2);
Math.sqrt(9);
Math.random();
```

# Math 수학

```
Math.pow(3, 2);      3 의 2 제곱
Math.round(1.4);     1.4 의 반올림
Math.ceil(1.2);      1.2 의 올림
Math.floor(1.2);     1.2 의 내림
Math.sqrt(9);        9 의 제곱근
Math.random();       0~1.0 사이의 랜덤한 숫자
```

## Math 수학

```
Math.round(100 * Math.random());
```

# String

# String 문자

## String 문자

```
"사이에 작성";
'사이에 작성';
"사이에 작성';
```

```
//escape.
"사이에 \'작성\'";
'사이에 \"작성\"';
```

원래 가지고 있던 임무에서 탈출

```
//문자열 줄바꿈.
"hello\nworld";
```

## String 문자

```
1
"1"
```

## String 문자

```
typeof 1
typeof "1"
typeof []
typeof {}
```

## String 문자

```
typeof 1      Number
typeof "1"    String
typeof []     Array
typeof {}     Object
```

# String 의 연산

## String 의 연산

```
"hello" + "world";
"hello" + " world";
```

## String 의 연산

```
1 + 1
"1" + "1"
1 + "1"
```

## String 의 연산

```
"hello world".length;    문자의 길이
"hello".indexOf('h');    문자의 순번
```

# Variable

## Variable 변수

```
var variable;
```

## Variable 변수

```
var variable = value;
```

변수명

## Variable 변수

```
var variable = value;
```

변수명 변수값

## Variable 변수

```
var a = 1;
var b = 2;
a + b
```

## Variable 변수

```
var a = 1;
var b = "2";
a + b
```

## Variable 변수

```
var a = 1;
var b = "2";
a + b
```

## Variable 변수

```
var a = "hello";
var b = "world";
a + b
```

## Variable 변수

```
var a = "hello";
a = "HELLO~";
```

var 선언 이후에는 var 를 사용하지 않아도 된다.

## Variable 변수의 재활용성

```
var a = "hello";
a = "HELLO~";
a = a + " WORLD!";
a += ":^)";
```

변수는 무수히 많은 재활용이 가능하다.

100 에 10 을 더한 후, 10 으로 나누고, 10 을 뺀 후, 10 을 곱한다.

100 에 10 을 더한 후, 10 으로 나누고, 10 을 뺀 후, 10 을 곱한다.

```
var sum = ( ( (100 + 10) / 10 ) - 10 ) * 10;
```

100 에 10 을 더한 후, 10 으로 나누고, 10 을 뺀 후, 10 을 곱한다.
200 에 20 을 더한 후, 20 으로 나누고, 20 을 뺀 후, 20 을 곱한다.
300 에 20 을 더한 후, 10 으로 나누고, 30 을 뺀 후, 40 을 곱한다.

100 에 10 을 더한 후, 10 으로 나누고, 10 을 뺀 후, 10 을 곱한다.
200 에 20 을 더한 후, 20 으로 나누고, 20 을 뺀 후, 20 을 곱한다.
300 에 20 을 더한 후, 10 으로 나누고, 30 을 뺀 후, 40 을 곱한다.

```
var a = 100;
var b = 10;
var sum = ( ( (a + b) / b ) - b ) * b;
```

100 에 10 을 더한 후, 10 으로 나누고, 10 을 뺀 후, 10 을 곱한다.
200 에 20 을 더한 후, 20 으로 나누고, 20 을 뺀 후, 20 을 곱한다.
300 에 20 을 더한 후, 10 으로 나누고, 30 을 뺀 후, 40 을 곱한다.

```
var a = 200;
var b = 20;
var sum = ( ( (a + b) / b ) - b ) * b;
```

100 에 10 을 더한 후, 10 으로 나누고, 10 을 뺀 후, 10 을 곱한다.
200 에 20 을 더한 후, 20 으로 나누고, 20 을 뺀 후, 20 을 곱한다.
300 에 20 을 더한 후, 10 으로 나누고, 30 을 뺀 후, 40 을 곱한다.

```
var a = 300, b = 20, c = 10, d = 30, e = 40;
var sum = ( ( (a + b) / c ) - d ) * e;
```

100 에 10 을 더한 후, 10 으로 나누고, 10 을 뺀 후, 10 을 곱한다.
200 에 20 을 더한 후, 20 으로 나누고, 20 을 뺀 후, 20 을 곱한다.
300 에 20 을 더한 후, 10 으로 나누고, 30 을 뺀 후, 40 을 곱한다.

```
var a = 1000, b = 300, c = 5, d = 10, e = 60;
var sum = ( ( (a + b) / c ) - d ) * e;
```

변할 수 있는 영역과 변하지 않는 영역으로 구분할 수 있다.
(유지보수)

# 비교 연산자

## 비교 연산자

```
//연산자
- + * / %
```

## 비교 연산자

```
//연산자
- + * / %

//대입 연산자(이항 연산자)
var variable = value;
```

# 비교 연산자

```
//연산자
- + * / %

//대입 연산자(이항 연산자)
var variable = value;

//비교 연산자
Boolean
true / false;
1 / 0
```

# 비교 연산자

==

> <

>= <=

## 비교 연산자

==

> <

>= <=

값이 같은지 큰지 작은지를 비교

## 비교 연산자

==

> <

>= <=

=>

주의합니다. 다른 명령어입니다.

## 비교 연산자

```
var a = 1;
var b = 1;
a == b
```

## 비교 연산자

```
var a = 2;
var b = 1;
a > b
```

## 비교 연산자

```
var a = 2;
var b = 1;
a < b
```

## 비교 연산자

```
var a = 2;
var b = 2;
a >= b
```

## 비교 연산자

```
var a = 1;
var b = 2;
a <= b
```

# 동등 연산자

# 동등 연산자

```
//대입 연산자(이항 연산자)
=

//동등 연산자
==
```

## 동등 연산자

```
var a = 1;
var b = 2;
a == b
```

## 동등 연산자

```
var a = 1;
var b = 1;
a == b
```

## 동등 연산자

```
var a = "one";
var b = "하나";
a == b
```

## 동등 연산자

```
var a = "one";
var b = "하나";
a == b
```

## 동등 연산자

```
var a = "one";
var b = "one";
a == b
```

## 동등 연산자

```
var a = 1;
var b = "1";
a == b
```

# 동등 연산자

```
var a = 1;
var b = "1";
a == b
```

?

# 동등 연산자

```
var a = 1;
var b = "1";
a === b
```

# 일치 연산자

# 일치 연산자

```
//대입 연산자(이항 연산자)
=

//동등 연산자
==

//일치연산자
===
```

# 일치 연산자

```
var a = 1;
var b = "1";
a == b
```

```
var a = 1;
var b = "1";
a === b
```

정확히 일치 하는지를 비교, **Strict** (엄격한)

동등 연산자는 버그를 발생시킬 위험이 있다.

# Data Type

# Data Type

```
//Data type.
Boolean
Number
String
undefined
null
1
1 이 아닌 수
NaN
```

```
//Data type.
Boolean      true | false
Number       -1 0 1 2 3 4 5...
String       "a" "b" "c"...
undefined    undefined
null         null
1            Boolean 의 true 로 간주
1 이 아닌 수   Boolean 의 false 로 간주
NaN          성립이 되지 않는 수, 계산할 수 없음을 의미함.
```

## Data Type

```
var a = null;
var a;
```

값이 없는 상태, 의도해서 값이 없는 상태로 만든 것

값이 정의되지 않은 상태

## Data Type

```
var a = null;
var b;
a == b
```

## Data Type

```
var a = null;
var b;
a == b
```

```
var a = null;
var b;
a === b
```

# Data Type

```
0 === -0
```

## Data Type

```
true == 1
```

## Data Type

```
true == 1      true === 1
```

## Data Type

```
NaN === NaN
```

```
NaN === NaN
```

둘 다 **NaN** 이라도 **false** 가 된다.

# 부정

# 부정

```
//부정
!=
!==
```

## 부정

```
var a = 1;
var b = 2;
a == b
```

## 부정

```
var a = 1;
var b = 2;
a == b

a != b
```

## 부정

```
var a = 1;
var b = 1;
a != b
```

## 부정

```
var a = "a";
var b = "b";
a != b
```

## 부정

```
var a = "a";
var b = "a";
a != b
```

# Object

# Object

```
var object = {};
```

# Object

```
var object = { key : value };
```

## Object

```
var person = {
    name : "김용원",
    job : "교수",
    phone : "010-9137-8688",
    email : "rh@102labs.com"
};
```

## Object

```
var person = {
    "name" : "김용원",
    "job" : "교수",
    "phone" : "010-9137-8688",
    "email" : "rh@102labs.com"
};
```

## Object

```
person.name;
person.job;
person.phone;
person.email;
```

# Object

```
person["name"];
person["job"];
person["phone"];
person["email"];
```

**과제**입니다.

# HOMEWORK

## 1. JavaScript 연습하기

- 숫자(Number), 문자(String), 수학(Math), 변수(variable), 연산자(산술, 대입, 비교, 동등, 일치, 부정), 데이터 타입(Data Type), 오브젝트(Object) 에 대해 익히고 연습합니다.
- 값은 console.log() 를 이용하여 출력합니다.
- 주석을 이용하여 JavaScript 각 문법에 대한 예상 답안과 질문을 합니다.

## 2. 자기소개하기

- 숫자(Number), 문자(String), 수학(Math), 변수(variable), 연산자(산술, 대입, 비교, 동등, 일치, 부정), 데이터 타입(Data Type), 오브젝트(Object) 를 이용하여 장문의 자기소개를 합니다.
- 값은 console.log() 를 이용하여 문서 실행시 자기소개가 한꺼번에 출력되도록 합니다.
- 데이터 형식, 연산 등에서 잘 모르는 부분은 주석을 이용하여 질문을 합니다.

### 제출

- github에 업로드

### 기한

- 2018. 3. 21. 23:00 까지