Undefined

변수를 선언하고 값을 할당하기 전의 형태(값)라고 볼 수 있다. (*변수에 값이 할당되어 있지 않음.)
존재하지 않는 객체나 없는 값을 읽으려 할때 undefined가 나온다.

Null

NULL의 심볼이며, 의도를 갖고 변수에 null을 할당하여 값이 없다는 것을 나타낸다.
null이 할당된 변수의 타입을 확인해 보면 object인걸 확인할 수 있다.

```
let a = null;
console.log(a); // null
console.log(typeof a); // object
```

Literal

값을 할당할때 사용된다
문자열 literal, 숫자형 literal 등등이 있다고 한다.

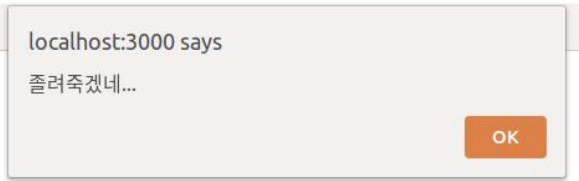
Template Literal

ES6부터 새로 도입된 문자열 표기법이다.
문자열 생성시 따옴표 대신,백틱(`)을 사용한다.
-표현식 삽입 ${}사이에 변수나 연산 등을 삽입할 수 있게 되었다.

```
var name = `교통비`
var price = 2350
var num = 2;
console.log(`${name}는 왕복 ${price * num}원 입니다.`)  // 교통비는 4700원입니다.
console.log(`${name}는 왕복 ${2350*2}원 입니다.`) --- {연산값}이 문자열로 자동 변환되어 사용된다.
```

alert

이 함수를 사용하여 경고창을 띄울수 있다.



함수(function)

특별한 목적의 작업을 수행하도록 설계된 독립적인 블록
javascript에서 코드 집합을 나타내는 자료형)
작업을 수행하거나 값을 계산하는 문장 집합 같은 자바스크립트 절차
함수도 일반 객체처럼 값으로 취급된다. 즉 하나의 데이터 타입 이다.
이러한 특징때문에 자바스크립트의 함수는 일급 객체라고 하며, 변수에도 할당 가능하다.

EmptyObject

객체는 빈 객체더라도 if문에서 true로 처리된다.
객체가 비어 있더라도 객체 자체가 존재하는 것이기에 true로 인식한다.

```
let emptyObj ={}
if (emptyObj){
   console.log("true")   //----emptyObj는 {} 비어있는 obfect로 true!!
} else {
   console.log("false")
}
```

Spread

Spread

전개 구문을 사용하면 배열이나 문자열과 같이 반복 가능한 문자를 0개 이상의 인수 (함수로 호출할 경우)
또는 요소 (배열 리터럴의 경우)로 확장하여, 0개 이상의 키-값의 쌍으로 객체로 확장시킬 수 있다.

```
const arr = [1, 2, 3, 4, 5];
const a = [...arr];

console.log(a) // [1, 2, 3, 4, 5];
```

```
if(Programming) {
    console.log("FunctionObjectTest2: Programming true")   //------programming 이라는 함수가 있다
} else {
    console.log("FunctionObjectTest2: Programming false")
}

console.log("FunctionObjectTest2: " + prog)   // ------ [object Object]

if(prog) {
    console.log("FunctionObjectTest2: prog true") //------ prog 라는 programming 함수선언
} else {
    console.log("FunctionObjectTest2: prog false")
}

console.log("FunctionObjectTest2: " + prog.name) //------JavaScript prog(programming) 함수에 name호출

if(prog.name) {
    console.log("FunctionObjectTest2: prog.name true")   //------ name값이 있으니 true
} else {
    console.log("FunctionObjectTest2: prog.name false")
}
JSON.stringify
console.log("FunctionObjectTest2: " + prog.students)   //------ 17 JavaScript prog(programming) 함수에 students호출

if(prog.students) {
    console.log("FunctionObjectTest2: prog.students true") //------students 값이 있으니 true
} else {
    console.log("FunctionObjectTest2: prog.students false")
}

console.log("FunctionObjectTest2: prog.getName - " + prog.getName) // Programming.prototype.getName 호출
                                                                   // function () {return this.name;}
console.log(prog.getName())   //-----현재 얻은 getname은 JavaScript

if(prog.getName) {
    console.log("FunctionObjectTest2: prog.getName true") //------ name값이 있으니 true
} else {
    console.log("FunctionObjectTest2: prog.getName false")
}
```

```
console.log("FunctionObjectTest2: prog.toString - " + prog.toString)//------??function toString() { [native code]}
console.log(prog.toString())//------[object Object]

if(prog.toString) {
    console.log("FunctionObjectTest2: prog.toString true") //------ 문자열이 있다??
} else {
    console.log("FunctionObjectTest2: prog.toString false")
}

console.log("FunctionObjectTest2: prog.name2 - " + prog.name2) //------undefined 이건 없는 값이다.

// 자바스크립트에서 없는 것은 결국 undefined로 false 처리됨
if(prog.name2) {
    console.log("FunctionObjectTest2: prog.name2 true")
} else {
    console.log("FunctionObjectTest2: prog.name2 false")//------
}

console.log("FunctionObjectTest2: prog.getName2 - " + prog.getName2) //------undefined 이것도 없는값이다.

if(prog.getName2) {
    console.log("FunctionObjectTest2: prog.getName2 true")
} else {
    console.log("FunctionObjectTest2: prog.getName2 false")//------
}
```

```
// ~~ in 객체에서 잡히고
// hasOwnProperty에서 거짓이 나오면 필요에 의해 새로 추가한 객체
// 양쪽 모두에서 잡히면 원래부터 가지고 있던 순수한 속성
console.log("FunctionObjectTest2: " + ("name" in prog)) //------true prog안에 name은 있다
console.log("FunctionObjectTest2: " + ("students" in prog))//------true prog안에 students은 있다
console.log("FunctionObjectTest2: " + ("getName" in prog)) //------true prog(programming)프로토타입 내포되getName이 있다.
console.log("FunctionObjectTest2: " + ("name2" in prog)) //------false name2는 어디에도 없다....

//hasOwnProperty 메소드가 하는 일은 객체가 특정 프로퍼티에 대한 소유 여부를 반환한다.
console.log("FunctionObjectTest2: " +
    (prog.hasOwnProperty( v: "students")))
console.log("FunctionObjectTest2: " +
    (prog.hasOwnProperty( v: "getName"))) //---prog(programming)의 순수한 속성이 아니므로 false
console.log("FunctionObjectTest2: " +
    ("toString" in prog))
console.log("FunctionObjectTest2: " +
    (prog.hasOwnProperty( v: "toString")))//---prog(programming)의 순수한 속성이 아니므로 false

//JSON.stringify  object에 내용을 확인!
console.log("FunctionObjectTest2: " + JSON.stringify(prog))
```