SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

ECMAScript 6
타입과 연산자

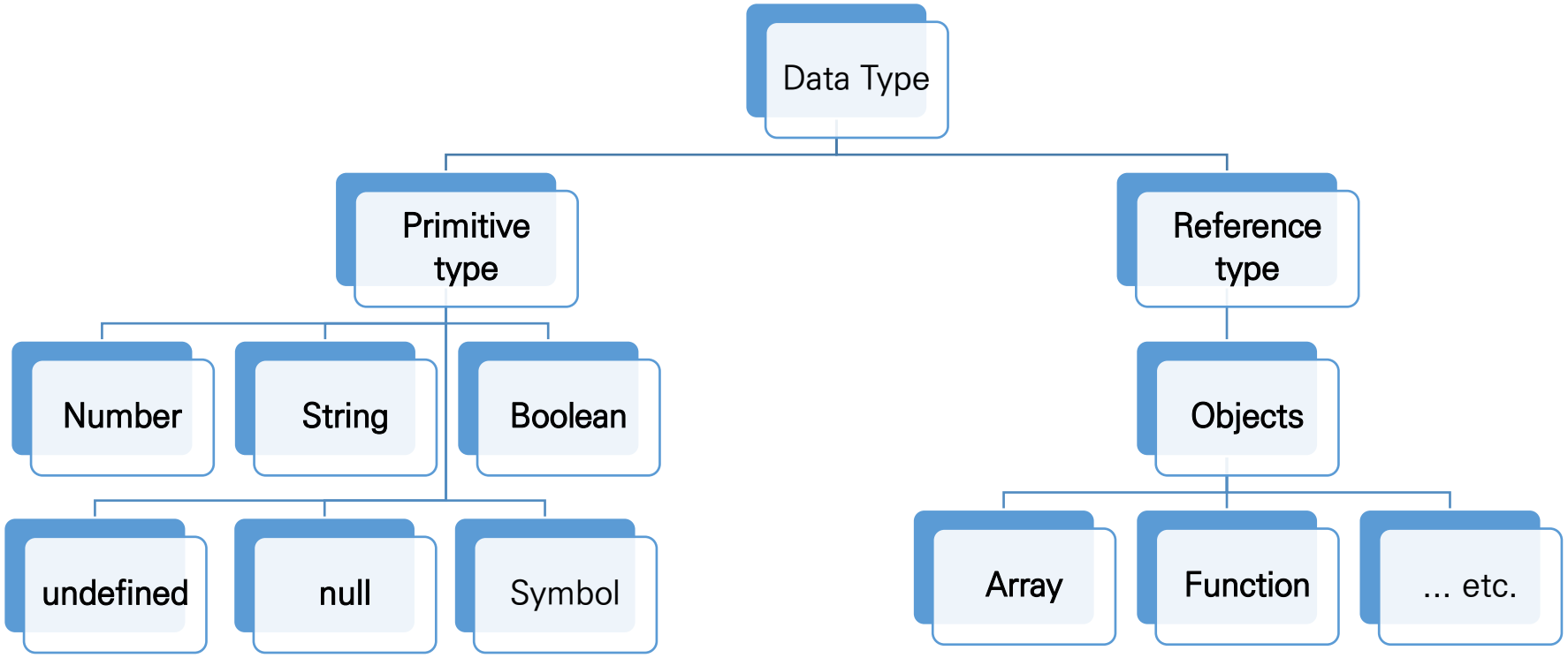
SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

데이터 타입

데이터 타입 종류

- 자바스크립트의 모든 값은 **특정한 데이터 타입을 가짐**
- 크게 **원시 타입*** (Primitive type)과 **참조 타입*** (Reference type)으로 분류됨



(참고) 원시 타입과 참조 타입 비교

원시 타입 (Primitive type)

- 객체 (object)가 아닌 기본 타입들을 말함
- 변수에 **해당 타입의 값이 담김**
- 다른 변수에 복사할 때 **실제 값이 복사됨**

```
let message = '안녕하세요!' // 1. message 선언 및 할당

let greeting = message    // 2. greeting에 message 복사
console.log(greeting)     // 3. '안녕하세요!' 출력

message = 'Helo, world!'  // 4. message 재할당
console.log(greeting)     // 5. '안녕하세요!' 출력

//=> 즉, 원시 타입은 실제 해당 타입의 값을 변수에 저장한다.
```

참조 타입 (Reference type)

- 객체 (object) 타입의 자료형들을 말함
- 변수에 해당 **객체의 참조 값이 담김**
- 다른 변수에 복사할 때 **참조 값이 복사됨**

```
const message = ['안녕하세요']  // 1. message 선언 및 할당

const greeting = message      // 2. greeting에 message 복사
console.log(greeting)         // 3. ['안녕하세요!'] 출력

message[0] = 'Hello, world!'  // 4. message 재할당
console.log(greeting)         // 5. ['Hello, world!'] 출력

//=> 즉, 참조 타입은 해당 객체를 참조할 수 있는 참조 값을 저장한다.
```

원시 타입 (Primitive type)

```
const a = 13          // 양의 정수
const b = -5          // 음의 정수
const c = 3.14        // 실수
const d = 2.998e8     // 거듭제곱
const e = Infinity    // 양의 무한대
const f = -Infinity   // 음의 무한대
const g = NaN         // 산술 연산 불가
```

- ❑ 숫자 (Number) 타입
  - ○ **정수, 실수 구분 없는 하나의 숫자 타입**
  - ○ **부동소수점 형식**을 따름
  - ○ (참고) NaN (Not-A-Number)
    - ○ **계산 불가능한 경우** 반환되는 값
      - ○ ex) 'Angel' / 1004 => NaN

원시 타입 (Primitive type)

```
const firstName = 'Brendan'
const lastName = 'Eich'
const fullName = firstName + lastName

console.log(fullName) // BrandenEich
```

```
const firstName = 'Brendan'
const lastName = 'Eich'
const fullName = `${firstName} ${lastName}`

console.log(fullName) // Branden Eich
```

- **문자열 (String) 타입**
  - **텍스트 데이터**를 나타내는 타입
  - 16비트 유니코드 문자의 집합
  - **작은따옴표** 또는 큰따옴표 모두 가능
  - 템플릿 리터럴 (Template Literal)
    - ES6부터 지원
    - 따옴표 대신 backtick(` `)으로 표현
    - ${ expression } 형태로 표현식 삽입 가능

원시 타입 (Primitive type)

```
let firstName
console.log(firstName) // undefined
```

- undefined
  - 변수의 **값이 없음**을 나타내는 데이터 타입
  - 변수 선언 이후 **직접 값을 할당하지 않으면 자동으로 undefined가 할당됨**

원시 타입 (Primitive type)

```
let firstName = null
console.log(firstName) // null

typeof null // object
```

❑ null

- 변수의 **값이 없음**을 **의도적으로 표현**할 때 사용하는 데이터 타입
- (참고) null 타입과 **typeof 연산자***
  - **typeof 연산자*: 자료형 평가를 위한 연산자**
  - **null 타입은** ECMA 명세의 원시 타입의 정의에 따라 원시 타입에 속하지만, **typeof 연산자의 결과는 객체(object)로 표현됨 (참고 자료)**

(참고) undefined 타입과 null 타입 비교

undefined

- **빈 값을 표현**하기 위한 데이터 타입
- 변수 선언 시 아무 값도 할당하지 않으면 자바스크립트가 자동으로 할당
- typeof 연산자의 결과는 undefined

```
typeof undefined // undefined
```

null

- **빈 값을 표현**하기 위한 데이터 타입
- 개발자가 의도적으로 필요에 의해 할당
- typeof 연산자의 결과는 object

```
typeof null // object
```

원시 타입 (Primitive type)

```
let isAdmin = true
console.log(isAdmin) // true

isAdmin = false
console.log(isAdmin) // false
```

❑ **불리언 (Boolean) 타입**

- **논리적 참 또는 거짓**을 나타내는 타입
- **true** 또는 **false로 표현**
- **조건문 또는 반복문***에서 유용하게 사용
  - (참고) 조건문 또는 반복문에서 **불리언이 아닌 데이터 타입**은 자동 형변환 규칙에 따라 true 또는 false로 변환됨

(참고) ToBoolean Conversions (자동 형변환) 정리

❑ 조건문 또는 반복문에서 표현식의 결과가 참/거짓으로 판별되는 경우 발생

데이터 타입	거짓	참
Undefined	항상 거짓	X
Null	항상 거짓	X
Number	0, -0, NaN	나머지 모든 경우
String	빈 문자열	나머지 모든 경우
Object	X	항상 참

참조 **타입** (Reference type)

- **자세한 내용은 해당 타입의 챕터 참고**
- 타입별 챕터 링크 목록
  - 함수 (Functions)
  - 배열 (Arrays)
  - 객체 (Objects)

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

연산자

할당 연산자

```
let x = 0

x += 10
console.log(x) // 10

x -= 3
console.log(x) // 7

x *= 10
console.log(x) // 70

x /= 10
console.log(x) // 7

x++             // += 연산자와 동일
console.log(x) // 8

x--             // -= 연산자와 동일
console.log(x) // 7
```

- 오른쪽에 있는 **피연산자의 평가 결과**를

  왼쪽 피연산자에 **할당하는 연산자**

- 다양한 연산에 대한 단축 연산자 지원

- (참고) Increment 및 Decrement 연산자*
  - Increment(++): 피연산자의 값을 1 **증가시키는** 연산자
  - Decrement(--): 피연산자의 값을 1 **감소시키는** 연산자
  - Airbnb Style Guide에서는 '+=' 또는 '-='와 같이

    더 분명한 표현으로 적을 것을 권장

비교 연산자

```
const numOne = 1
const numTwo = 100
console.log(numOne < numTwo)   // true

const charOne = 'a'
const charTwo = 'z'
console.log(charOne > charTwo) // false
```

- 피연산자들(숫자, 문자, Boolean 등)을 비교하고 **비교의 결과값을 불리언으로 반환하는 연산자**
- 문자열은 유니코드 값을 사용하며 표준 사전순서를 기반으로 비교
  - ex) 알파벳끼리 비교할 경우
    - 알파벳 오름차순으로 우선순위를 지님
    - 소문자가 대문자보다 우선순위를 지님

동등 비교 연산자 (==)

```
const a = 1004
const b = '1004'
console.log(a == b) // true

const c = 1
const d = true
console.log(c == d) // true

// 자동 타입 변환 예시
console.log(a + b) // 10041004
console.log(c + d) // 2
```

- 두 피연산자가 같은 값으로 평가되는지 비교 후 불리언 값을 반환
- 비교할 때 암묵적 타입 변환을 통해 타입을 일치시킨 후 같은 값인지 비교
- 두 피연산자가 모두 객체일 경우 메모리의 같은 객체를 바라보는지 판별
- 예상치 못한 결과가 발생할 수 있으므로 특별한 경우를 제외하고 사용하지 않음

일치 비교 연산자 (===)

```
const a = 1004
const b = '1004'
console.log(a === b) // false

const c = 1
const d = true
console.log(c === d) // false
```

- 두 피연산자가 같은 값으로 평가되는지 비교 후 불리언 값을 반환
- 엄격한 비교*가 이뤄지며 암묵적 타입 변환이 발생하지 않음
  - 엄격한 비교*: 두 비교 대상의 타입과 값 모두 같은지 비교하는 방식
- 두 피연산자가 모두 객체일 경우 메모리의 같은 객체를 바라보는지 판별

논리 연산자

```
/*
  and 연산
*/
console.log(true && false) // false
console.log(true && true)  // true
console.log(1 && 0)        // 0
console.log(4 && 7)        // 7
console.log('' && 5)       // ''

/*
  or 연산
*/
console.log(true || false)  // true
console.log(false || false) // false
console.log(1 || 0)         // 1
console.log(4 || 7)         // 4
console.log('' || 5)        // 5

/*
  not 연산
*/
console.log(!true)          // false
console.log(!'Bonjour!')    // false
```

- 세 가지 논리 연산자로 구성
  - and 연산은 && 연산자를 이용
  - or 연산은 || 연산자를 이용
  - not 연산은 ! 연산자를 이용
- 단축 평가 지원
  - ex) false && true => false
  - ex) true || false => true

삼항 연산자 (Ternary Operator)

```
console.log(true ? 1 : 2)  // 1
console.log(false ? 1 : 2) // 2

const result = Math.PI > 4 ? 'Yes' : 'No'
console.log(result) // Nope
```

- **세 개의 피연산자를 사용하여 조건에 따라 값을 반환하는 연산자**
- 가장 왼쪽의 **조건식이 참이면 콜론(:) 앞의 값을 사용**하고 **그렇지 않으면 콜론(:) 뒤의 값을 사용**
- 삼항 연산자의 결과는 변수에 할당 가능
- (참고) 한 줄에 표기하는 것을 권장

타입과 연산자 실습

- 목표: 자바스크립트 타입과 연산자 연습 (02-types-and-operators.js)
- 문제: 파일에 작성된 주석 참고

타입과 연산자 Quiz

Q1. 자바스크립트의 데이터 타입은 크게 원시 타입과 참조 타입으로 분류된다. T/F

Q2. Number 타입은 0을 제외한 모든 경우 참으로 자동 형변환이 이뤄진다. T/F

Q3. 일치 비교 연산자(===)는 자동 형변환을 통해 타입과 값이 같은지 판별한다. T/F

타입과 연산자 Quiz

Q1. 자바스크립트의 데이터 타입은 크게 원시 타입과 참조 타입으로 분류된다. T

Q2. Number 타입은 0을 제외한 모든 경우 참으로 자동 형변환이 이뤄진다. F

A2. Number 타입은 0, -0, NaN을 제외한 모든 경우 참으로 형변환 된다.

Q3. 일치 비교 연산자(===)는 자동 형변환을 통해 타입과 값이 같은지 판별한다. F

A3. 일치 비교 연산자는 자동 형변환이 일어나지 않는다.