


2024-04-03 5.6.4 Function

 [현행 문서: 5.6.4 Function](#)

- 1 Function
 - 1.1 매개변수
 - 1.1.1 필수 매개변수(required parameter)
 - 1.1.2 선택적 매개변수 (optinal parameter)
 - 1.1.3 기본 매개변수
 - 1.1.4 나머지 매개변수 (rest parameter)
 - 1.2 반환타입
 - 1.2.1 명시적 반환타입
 - 1.3 함수타입 (function type)
 - 1.4 그 외 반환 타입
 - 1.4.1 void
 - 1.4.2 Never 반환 타입

1 Function

1.1 매개변수

1.1.1 필수 매개변수(required parameter)

단어	번역	의미
Parameter	매개변수	함수와 메서드 입력 변수(Variable)명
Argument	전달인자,인자,인수	함수와 메서드 입력 값(Value)

자바 스크립트에서는 인수의 수와 상관없이 함수를 호출할 수 있습니다.

하지만 타입스크립트는 함수에 선언된 모든 매개변수가 필수라고 가정합니다.

함수가 잘못된 수의 인수로 호출되면, 타입스크립트는 타입 오류의 형태로 이의를 제기합니다.

함수가 너무 적거나 많은 인수로 호출되면 타입스크립트는 인수의 개수를 계산합니다.

```
function singTwo(first: string, second: string) {
  console.log(`${first} / ${second}`);
}

singTwo("Ball and Chain");
singTwo("I W ill Survive", "Higher Love"); // Ok
singTwo("Go Your Own Way", "The Chain", "Dreams");
```

[blocked URL](#)

1.1.2 선택적 매개변수 (optinal parameter)

자바스크립트에서 함수 매개변수가 제공되지 않으면 함수 내부의 인숫값은 undefined으로 기본값이 설정된다는 것을 떠올려보세요.

타입스크립트에서는 선택적 객체 타입 속성과 유사하게 타입 애너테이션의 : 앞에 ?를 추가해 매개변수가 선택적이라고 표시합니다.

선택적 매개변수에는 항상 | undefined가 유니언 타입으로 추가되어 있습니다.

```
function announceSong(song: string, singer?: string) {
  console.log(`Song: ${song}`);

  if (singer) {
    console.log(`Singer: ${singer}`);
  }
}
announceSong("Greensleeves");
announceSong("Greensleeves", undefined);
announceSong("Greensleeves", "sia");
```

[blocked URL](#)

함수에서 사용되는 모든 선택적 매개변수는 마지막 매개변수여야 합니다.

필수 매개변수 전에 선택적 매개변수를 위치시키면 다음과 같이 타입스크립트 구문 오류가 발생합니다.

```
function announceSinger(singer?: string, song: string) {}
```

[blocked URL](#)

1.1.3 기본 매개변수

자바스크립트에서 선택적 매개변수를 선언할 때 =와 값이 포함된 기본값을 제공할 수 있습니다.

타입스크립트의 타입 추론은 초기 변수값과 마찬가지로 기본 함수 매개변수에 대해서도 유사하게 작동합니다.

타입스크립트는 함수의 매개변수에 대해 인수를 누락하거나 undefined 인수를 사용해서 호출하는 것을 여전히 허용합니다.

매개변수에 기본값이 있고 타입 애너테이션이 없는 경우, 타입스크립트는 해당 기본값을 기반으로 매개변수 타입을 유추합니다.

```
function rateSong(song: string, rating = 0) {
  console.log(`${song} gets ${rating}/5 stars!`);
}

rateSong("Photograph");
rateSong("Set Fire to the Rain", 5);
rateSong("Set Fire to the Rain", undefined);

rateSong("At Last!", "100");
```

[blocked URL](#)

1.1.4 나머지 매개변수 (rest parameter)

자바스크립트의 일부 함수는 임의의 수의 인수로 호출할 수 있도록 만들어집니다.

스프레드 연산자(...)는 함수 선언의 마지막 매개변수에 위치합니다.

해당 매개변수에서 시작해 함수에 전달 된 '나머지' 인수가 모두 단일 배열에 저장되어야 함을 나타냅니다.

타입스크립트는 이러한 나머지 매개변수의 타입을 일반 매개변수와 유사하게 선언할 수 있습니다.

단, 인수 배열을 나타내기 위해 끝에 [] 구문이 추가된다는 점만 다릅니다.

```
function singAllTheSongs(singer: string, ...songs: string[]) {
  for (const song of songs) {
    console.log(`${song}, by ${singer}`);
  }
}

singAllTheSongs("Alicia Keys");
singAllTheSongs("Lady Gaga", "Bad Romance", "faker");

singAllTheSongs("Ella Fitzgerald", 2000);
```

[blocked URL](#)

1.2 반환타입

1.2.1 명시적 반환타입

변수와 마찬가지로 타입 애너테이션을 사용해 함수의 반환 타입을 명시적으로 선언하지 않는 것이 좋습니다.

그러나 특히 함수에서 반환 타입을 명시적으로 선언하는 방식이 매우 유용할 때가 종종 있습니다.

- 가능한 반환값이 많은 함수가 항상 동일한 타입의 값을 반환하도록 강제합니다.
- 타입스크립트는 재귀 함수의 반환 타입을 통해 타입을 유추하는 것을 거부합니다.
- 수백 개 이상의 타입스크립트 파일이 있는 매우 큰 프로젝트에서 타입스크립트 타입 검사 속도를 높일 수 있습니다.

```
function singSongsRecursive(songs: string[], count = 0): number {
  return songs.length
    ? singSongsRecursive(songs.slice(1), count + 1)
    : count;
}

const singSongsRecursiveLambda = (songs: string[], count = 0): number =>
  songs.length ? singSongsRecursive(songs.slice(1), count + 1) : count;
```

[blocked URL](#)

1.3 함수타입 (function type)

자바스크립트에서는 함수를 값으로 전달할 수 있습니다.

즉, 함수를 가지기 위한 매개변수 또는 변수의 타입을 선언하는 방법이 필요합니다.

함수 타입 구문은 화살표 함수와 유사하지만 함수 본문 대신 타입이 있습니다.

다음 `nothingInGivesString` 변수 타입은 매개변수가 없고 `string` 타입을 반환하는 함수임을 설명합니다.

다음 `inputAndOutput` 변수 타입은 `string[]` 매개변수와 `count` 선택적 매개변수 및 `number` 값을 반환하는 함수임을 설명합니다.

- ```
let nothingInGivesString: () => string;
let inputAndOutput: (songs: string[], count?: number) => number;
```

[blocked URL](#)

## 1.4 그 외 반환 타입

### 1.4.1 void

반환 타입이 `void`인 함수는 값을 반환하지 않을 수 있습니다.

`void` 타입은 자바스크립트가 아닌 함수의 반환 타입을 선언하는 데 사용하는 타입스크립트 키워드입니다.

void 타입은 함수의 반환값이 자체적으로 반환될 수 있는 값도 아니고, 사용하기 위한 것도 아니라는 표시임을 기억하세요.

다음 logSong 함수는 void를 반환하도록 선언되었으므로 값 반환을 허용하지 않습니다.

```
function logSong(song: string | undefined) : void {
 if (!song) {
 return;
 }
 console.log(`${song}`);

 return true;
}
```

[blocked URL](#)

### 1.4.2 Never 반환 타입

일부 함수는 값을 반환하지 않을 뿐만 아니라 반환할 생각도 전혀 없습니다.

never 반환 함수는 (의도적으로) 항상 오류를 발생시키거나 무한 루프를 실행하는 함수입니다.

함수가 절대 반환하지 않도록 의도하려면 명시적 : never 타입 애너테이션을 추가해 해당 함수를 호출한 후 모든 코드가 실행되지 않음을 나타냅니다.

다음 fail 함수는 오류만 발생시키므로 param의 타입을 String으로 좁혀서 타입스크립트의 제어 흐름 분석(control flow analysis)을 도와줍니다.

```
function fail(message: string): never {
 throw new Error(`Invariant failure: ${message}`);
}

function workWithUnsafeParam(param: unknown) {
 if (typeof param !== 'string') {
 fail(`param should be a string, not ${typeof param}`);
 }
 param.toUpperCase();
}
```

[blocked URL](#)