# 26장. ES6 함수의 추가 기능

## 26.1 함수의 구분

▼ ES6 이전의 함수

일반 함수로서 호출할 수 있는 것은 물론 생성자 함수로서 호출 가능

→ callable[호출할 수 있는 함수 객체]이면서 constructor[인스턴스를 생성할 수 있는 함수 객체]

▼ ES6 함수의 구분

| | constructor | prototype | super | arguments |
|---|---|---|---|---|
| **일반함수** | O | O | X | O |
| **메서드** | X | X | O | O |
| **화살표 함수** | X | X | X | X |

## 26.2 메서드

▼ 메서드란?

ES6 사양에서 메서드는 메서드 축약 표현으로 정의된 함수만을 의미

ES6 사양에서 정의한 메서드는 인스턴스를 생성할 수 없는 non-constructor이다.

→ 인스턴스를 생성할 수 없으므로 prototype 프로퍼티가 없고 프로토타입도 생성하지 않음

▼ 메서드 특징

자신을 바인딩한 객체를 가리키는 내부슬롯 [[HomeObject]]를 갖는다.

→ 내부 슬롯 [[HomeObject]]을 갖는 메서드는 super키워드 사용가능

메서드가 아닌 함수는 super 키워드 사용 불가능 → [[HomeObject]]를 갖지 않기 때문

## 23.3 화살표 함수

▼ 화살표 함수란?

function 키워드 대신 화살표 (⇒)를 사용하여 기존의 함수 정의 방식보다 간략하게 함수를 정의할 수 있다.

→ 콜백 함수 내부에서 this가 전역 객체를 가리키는 문제를 해결하기 위한 대안으로 유용

▼ 화살표 함수 정의 순서

1. 함수 정의 : 함수 표현식으로 정의해야 함
2. 매개변수 선언 : 매개변수가 여러 개인 경우 소괄호 안에 선언
3. 함수 몸체 정의 : 함수 몸체 내부의 문이 값으로 평가될 수 있는 문이라면 암묵적으로 반환

▼ 화살표 함수와 일반 함수의 차이

1. 화살표 함수는 인스턴스를 생성할 수 없는 non-constructor이다.
2. 중복된 매개변수 이름을 선언할 수 없다.
3. 화살표 함수는 함수 자체의 this, arguments. suepr. new.tartget  바인딩을 갖지 않는다.

▼  화살표 함수의 this

콜백 함수 내부의 this가 외부 함수의 this와 다르기 때문에 발생하는 문제를 해결하기 위해 의도적으로 설계된 것

▼ Array.prototype.map 메서드

베열을 순회하며 배열의 각 요소에 대하여 인수로 전달된 콜백 함수를 호출

→ 콜백 함수의 반환 값들로 구성된 새로운 배열을 반환

▼ lexical this란?

화살표 함수 내부에서 this를 참조하면 상위 스코프의 this를 그대로 참조

## 26.4 Rest 파라미터

▼ Rest 파라미터란?

매개변수 이름 앞에 세개의 점 ...을 붙여서 정의한 매개변수를 의미

→ Rest 파라미터는 함수에 전달된 인수들의 목록을 배열로 전달받는다.

▼ Rest 파라미터 특징

1. Rest 파라미터는 먼저 선언된 매개변수에 할당된 인수를 제외한 나머지 인수들로 구성된 배열

   → Rest 파라미터는 반드시 마지막 파라미터여야 함

2. Rest 파라미터는 단 하나만 선언할 수 있음
3. Rest 파라미터는 함수 객체의 length 프로퍼티에 영향을 주지 않음

## 26.5 매개변수 기본값

▼ 인수가 전달되지 않아 undefined인 경우

기본값을 할당

→ 매개변수 기본값은 매개변수에 인수를 전달하지 않은 경우와 undefined 전달한 경우에만 유효

→ Rest 파라미터에서는 기본 값 지정 불가능