25장. 클래스

25.1 클래스는 프로토타입의 문법적 설탕인가?

▼ 클래스란?

함수이며 기존 프로토타입 기반 패턴을 클래스 기반 패턴처럼 사용할 수 있도록 하는 문법적 설탱이라 볼 수 있다.

⇒ 새로운 객체 생성 메커니즘

▼ 클래스와 생성자 함수의 차이점

1. new 연산자 없이 호출하면 에러 발생
2. 상속을 지원하는 extends와 super 키워드를 제공
3. 호이스팅이 발생하지 않는 것처럼 동작
4. 암묵적으로 strict mode가 지정되어 실행되며 strict mode 해제 안됨
5. 정적 메서드는 모두 프로퍼티 어트리뷰트 [[Enumberable]]의 값이 false → 열거되지 않음

25.2 클래스 정의

▼ 클래스를 표현식으로 정의할 수 있다는 것

클래스가 값으로 사용할 수 있는 일급 객체라는 것을 의미

▼ 클래스의 특징

일급 객체로서의 특징

1. 무명의 리터럴로 생성할 수 있다
2. 변수나 자료구조에 저장할 수 있다
3. 함수의 매개변수에게 전달할 수 있다
4. 함수의 반환값으로 사용할 수 있다

▼ 클래스 메서드

constructor(생성자), 프로토타입 메서드, 정적 메서드

25.3 클래스 호이스팅

▼ 클래스 호이스팅 특징

클래스는 함수로 평가되기에 함수 객체를 생성하는 시점에 프로토 타입도 더불어 생성된다.

단, 클래스는 클래스 정의 이전에 참조 불가능

→ 클래스는 let, const 키워드로 선언한 변수처럼 호이스팅 된다. 일시적 사각지대에 빠짐

25.4 인스턴스 생성

▼ 클래스의 인스턴스 특징

클래스는 인스턴스를 생성하는 것이 유일한 존재 이유이므로 반드시 new 연산자와 함께 호출해야한다

25.5 메서드

▼ constuctor란?

인스턴스를 생성하고 초기화하기 위한 특수한 메서드, 이름 변경 불가능

메서드로 해석되는 것이 아닌 클래스가 평가되어 생성한 함수 객체 코드의 일부가 된다.

별도의 반환문을 갖지 않는다. → 암묵적으로 this(인스턴스)를 반환하기 때문

⇒ return문 반드시 생략

▼ 프로토타입 메서드란?

인스턴스는 프로토타입 메서드를 상속받아 사용할 수 있다.

▼ 정적 메서드란?

정적 메서드는 인스턴스를 생성하지 않아도 호출할 수 있는 메서드

메서드에 static 키워드를 붙이면 정적 메서드가 된다.

▼ 정적 메서드와 프로토타입 메서드의 차이점

1. 자신이 속해 있는 프로토타입 체인이 다르다.
2. 정적 = 클래스로 호출, 프로토타입 = 인스턴스로 호출
3. 정적 = 인스턴스 프로퍼티 참조 불가능, 프로토타입은 가능

25.6 클래스의 인스턴스 생성 과정

▼ 과정

1. 인스턴스 생성과 this 바인딩
2. 인스턴스 초기화
3. 인스턴스 반환

25.7 프로퍼티

▼ 인스턴스 프로퍼티

constructor 내부에서 정의

▼ 접근자 프로퍼티

자체적으로 값을 갖지 않고 다른 데이터 프로퍼티의 값을 읽거나 저장할 때 사용하는 접근자 함수로 구성된 프로퍼티

▼ 클래스 필드란?

클래스 기반 객체지향 언어에서 클래스가 생성할 인스턴스의 프로퍼티를 가리키는 용어

→ 클래스 필드에 초기값을 할당하지 않으면 undefined를 갖는다.

▼ private 필드 특징

클래스 내부에서만 참조 가능

25.6 상속에 의한 클래스 확장

▼ 상속에 의한 클래스 확장이란?

기존 클래스를 상속받아 새로운 클래스를 확장하여 정의하는 것

▼ 상속에 의한 클래스 확장 특징

코드 재사용 관점에서 매우 유용, 자신만의 고유한 속성만 추가