삼육대학교 SW융합교육원 김진호

자바스크립트 기초교육

<class>
</body>
<head>
<title>

1-2. Class inheritance (클래스 상속)

1-2-1. inheritance basic syntax

1-2-2. method overriding

1-2-3. constructor overriding

1. Class

1-2. class inheritance (클래스 상속)

클래스 상속을 사용하면 클래스를 다른 클래스로 확장할 수 있다.

1-2-1. inheritance basic syntax (상속 기본 문법)

- Animal 클래스 선언

```
class Animal {

    constructor(name, weight) {
        this.name = name;
        this.weight = weight;
    }

    eat(foodWeight) {
        this.weight += foodWeight;
        console.log(`${this.name}(은)는 ${foodWeight}kg의 식사를 하고
${this.weight}kg이 되었습니다.`);
    }

    move(lostWeight) {
        if (this.weight > lostWeight)
            this.weight -= lostWeight;
        console.log(`${this.name}(은)는 움직임으로 인해 ${lostWeight}kg 감량되어
${this.weight}kg이 되었습니다.`)
    }

}

let animal = new Animal("동물", 30);

animal.eat(1);
animal.move(0.5);
```

1. Class

1-2. class inheritance (클래스 상속)

1-2-1. inheritance basic syntax (상속 기본 문법)

- Animal을 상속 받는 Human 클래스 선언
- extends 키워드를 사용한다.

```
class Human extends Animal {

    develop(language) {
        console.log(`${this.name}(은)는 ${language}로 개발을 합니다. 정말 즐겁습니다^.^`);
    }

}

let human = new Human("수강생", 70);

// Animal에 정의 된 메서드 접근 가능 - 인간도 동물이므로 동물이 할 수 있는 일반적인 동작을 수행할 수 있다.
human.eat(2);
human.move(1);

// Human에 정의 된 메서드 접근 가능
human.develop("JavaScript");
```

- 키워드 extends는 프로토타입을 기반으로 동작한다.
- extends는 Human.prototype.[[Prototype]]을 Animal.prototype으로 설정한다.
- 그렇기 때문에 Human.prototype에서 메서드를 찾지 못하면 Animal.prototype에서 메서드를 가져온다.

1. Class

1-2. class inheritance (클래스 상속)

1-2-2. method overriding (메서드 오버라이딩)

부모 메서드 전체를 교체하지 않고, 부모 메서드를 토대로 일부 기능만 변경하고 싶을 때,
부모 메서드의 기능을 확장하고 싶을 때 메서드 오버라이딩을 사용한다.

- Animal 클래스 선언

```
class Animal {

    constructor(name, weight) {
        this.name = name;
        this.weight = weight;
    }

    eat(foodWeight) {
        this.weight += foodWeight;
        console.log(`${this.name}(은)는 ${foodWeight}kg의 식사를 하고
${this.weight}kg이 되었습니다.`);
    }

    move(lostWeight) {
        if (this.weight > lostWeight)
            this.weight -= lostWeight;
        console.log(`${this.name}(은)는 움직임으로 인해 ${lostWeight}kg 감량되어
${this.weight}kg이 되었습니다.`)
    }

}
```

1. Class

1-2. class inheritance (클래스 상속)

1-2-2. method overriding (메서드 오버라이딩)

- Animal을 상속 받는 Tiger 클래스 선언

```
class Tiger extends Animal {

    attack(target) {
        console.log(`${this.name}(은)는 ${target}을 공격합니다.`);
    }

    // Animal의 move를 확장한 Tiger의 move
    move(target) {
        // super.을 통해 부모 클래스에 메서드를 참조한다.
        super.move(0.1);
        this.attack(target);
    }
}

let tiger = new Tiger("백두산 호랭이", 90);
tiger.move("슬픈 눈망울의 사슴");
```

1. Class

1-2. class inheritance (클래스 상속)

1-2-3. constructor overriding (생성자 오버라이딩)

- 클래스가 다른 클래스를 상속받고 constructor가 없는 경우에는 비어 있는 constructor가 만들어진다.

```
class Tiger extends Animal {
  constructor(...args) {
    super(...args);
  }
}
```

- 생성자는 기본적으로 부모 constructor를 호출한다.
- 이때 부모 constructor에도 인수를 모두 전달하는데 클래스에 자체 생성자가 없는 경우엔 이런 일이 모두 자동으로 일어난다.

1. Class

1-2. class inheritance (클래스 상속)

1-2-3. constructor overriding (생성자 오버라이딩)

- Animal 클래스 선언

```
class Animal {

    constructor(name, weight) {
        this.name = name;
        this.weight = weight;
    }

    eat(foodWeight) {
        this.weight += foodWeight;
        console.log(`${this.name}(은)는 ${foodWeight}kg의 식사를 하고
${this.weight}kg이 되었습니다.`);
    }

    move(lostWeight) {
        if (this.weight > lostWeight)
            this.weight -= lostWeight;
        console.log(`${this.name}(은)는 움직임으로 인해 ${lostWeight}kg 감량되어
${this.weight}kg이 되었습니다.`)
    }

}
```

1. Class

1-2. class inheritance (클래스 상속)

1-2-3. constructor overriding (생성자 오버라이딩)

- Animal 클래스를 상속하는 Deer 클래스 선언

```
class Deer extends Animal {

    constructor(name, weight, legLength) {
        // this.name = name;
        // this.weight = weight;
        // ReferenceError: Must call super constructor in derived class
before accessing 'this' or returning from derived constructor
        // 상속 클래스의 생성자에선 반드시 super(...)를 호출해야 하는데, super(...)
를 호출하지 않아 에러가 발생한다.
        // super(...)는 this를 사용하기 전에 반드시 호출해야한다.
        super(name, weight);
        this.legLength = legLength;
    }

    hide(place) {
        console.log(`${this.name}(은)는 ${place}에 숨습니다.`);
    }
}

let deer = new Deer('슬픈 눈망울의 사슴', 40, 1);
deer.hide('동굴 안');
```

1. Class

1-2. class inheritance (클래스 상속)

1-2-3. constructor overriding (생성자 오버라이딩)

- 자바스크립트는 '상속 클래스의 생성자 함수(derived constructor)'와
  그렇지 않은 생성자 함수를 구분한다.
- 상속 클래스의 생성자 함수엔 특수 내부 프로퍼티인 [[ConstructorKind]]:"derived"가 붙는다.
- 일반 클래스는 new와 함께 실행되면 빈 객체가 만들어지고 this에 이 객체를 할당하지만,
  상속 클래스의 생성자 함수가 실행되면 빈 객체를 만들고 this에 이 객체를 할당하는 일을
  부모 클래스의 생성자가 처리해 주길 기대한다.
- 이런 차이 때문에 상속 클래스의 생성자에선 super를 호출해 부모 생성자를 실행해 주어야 하고
  그렇지 않으면 this가 될 객체가 만들어지지 않아 에러가 발생하는 것이다.