객체
- class -

class 정의

❖ class 키워드

- 객체의 원형을 정의하는 키워드

```
class 클래스명 {

}
```

class 정의

❖ 생성자

- constructor() 함수로 정의
  - this 객체가 새로 생성됨
  - 매개변수 지정 가능

```
class 클래스명 {
   constructor([매개변수]) {
      // this.를 이용하여 객체의 프로퍼티 정의
   }

}
```

class 정의

❖ **student.js**

```
class Student {
   constructor(name) {
      this.name = name;
   }
}

var s1 = new Student("홍길동");
console.log(s1.name);
```

class 정의

❖ 프로토타입 메서드

- class 블럭안에 정의되는 함수
- function 키워드 없이 정의

```
class 클래스명 {
   constructor([매개변수]) {
      // this.를 이용하여 객체의 프로퍼티 정의
   }

   함수명([매개변수]) {    // 프로토타입 메서드

   }
}
```

class 정의

❖ **student.js**

```
class Student {
   constructor(name, age) {
      this.name = name;
      this.age = age;
   }

   printProfile(){
      console.log(`이름 : ${this.name}, 나이 : ${this.age}`)
   }
}

var s1 = new Student("홍길동", 20);
s1.printProfile();

console.log("printProfile" in s1.__proto__)
console.log("printProfile" in Student.prototype)
```

class 정의

❖ Getter/Setter 메서드

- get, set 키워드로 설정

```
class 클래스명 {
   constructor([매개변수]) {
      // this.를 이용하여 객체의 프로퍼티 정의
   }

   함수명([매개변수]) {     // 프로토타입 메서드

   }

   get 프로퍼티명() {

   }
   set 프로퍼티명(매개변수) {

   }
}
```

class 정의

❖ **student.js**

```
class Student {
   constructor(name, age) {
      this._name = name;
      this.age = age;
   }

   printProfile(){
      console.log(`이름 : ${this.name}, 나이 : ${this.age}`)
   }

   get name() {
      return this._name;
   }

   set name(name) {
      this._name = name;
   }

}
```

class 정의

❖ **student.js**

```
var s1 = new Student("홍길동", 20);

console.log(s1.name);
s1.name = '고길동'
console.log(s1.name);
console.log(s1);


[실행결과]
홍길동
고길동
Student { _name: '고길동', age: 20 }
```

class 정의

❖ 정적 메서드

- static 키워드로 메서드 정의
- 자체 인스턴스 (this) 없이 구현

```
class 클래스명 {
   :

   static 메서드명(매개변수) {

   }
}
```

상속

❖ extends 키워드와 super 키워드 도입

```
class 자식클래스 extends 부모클래스 {

   constructor([매개변수]) {
      super([매개변수]);      // 부모의 생성자 호출, 이 후에 this 생성

   }


}
```

- 부모의 생성자는 반드시 호출

상속

❖ Inherit.js

```
class Parent {
   constructor(name) {
      this.name = name;
   }

   print() {
      console.log("이름 : " + this.name);
   }
}
```

상속

❖ Inherit.js

```
class Child extends Parent {
   constructor(name, age) {
      super(name);
      this.age = age;
   }

   print() {
      super.print();
      console.log("나이 : " + this.age);
   }

   static sayHello() {
      console.log('Hello~');
   }
}
```

상속

❖ Inherit.js

```
class GrandChild extends Child {
   constructor(name, age, address) {
      super(name, age);
      this.address = address;
   }

   print() {
      super.print();
      console.log("주소 : " + this.address);
   }
}


var person = new GrandChild("홍길동", 20, "서울");
person.print();
GrandChild.sayHello();
```