**[[Java] 열거 타입(Enum) 사용법 & 예제](https://coding-factory.tistory.com/522)**

* 코딩팩토리

* [Language / Java](https://coding-factory.tistory.com/category/Language/Java)

* 2020. 4. 20.

프로그래밍을 하다보면 몇가지의 한정된 데이터들을 가지는 경우가 있습니다. 대표적으로는 "요일"이나 "계절" 같은것이지요 요일은 월,화,수,목,금,토,일 이렇게 7가지밖고 계절도 봄,여름,가을,겨울 4가지 계절로 한정되어 있습니다. 이렇게 이와 같이 한정된 데이터들을 갖는 데이터들은 열거형으로 묶어주면 편합니다.

* [열거형(Enum) 선언 방법](https://coding-factory.tistory.com/522#h)
* [열거형(Enum) 사용법](https://coding-factory.tistory.com/522#h_1)
  + [실사용 예시](https://coding-factory.tistory.com/522#h_2)
* [열거객체 메소드](https://coding-factory.tistory.com/522#h_3)
  + [values()](https://coding-factory.tistory.com/522#h_4)
  + [ordinal()](https://coding-factory.tistory.com/522#h_5)
  + [valueOf()](https://coding-factory.tistory.com/522#h_6)
  + [name()](https://coding-factory.tistory.com/522#h_7)
  + [compareTo()](https://coding-factory.tistory.com/522#h_8)

**열거형(Enum) 선언 방법**

Copy

enum Season { *//class 외부에서 선언*

봄, 여름, 가을, 겨울

}

public class enum\_ex {

public enum Season { *//class 내부에서 선언*

봄, 여름, 가을, 겨울

}

}

enum 키워드는 열거 타입을 선언하기 위한 키워드입니다. 열거 타입(Season)이름을 지정하고 열거 상수(봄,여름,가을,겨울)를 나열하면 됩니다. 열거 타입 이름으로 소스파일(.java)를 생성하여도 됩니다.

**열거형(Enum) 사용법**

Copy

public class enum\_ex {

public enum Season {

봄, 여름, 가을, 겨울

}

public static void main(String[] args) {

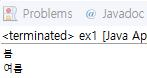
Season season = Season.봄;

System.out.println(season);

System.out.print(Season.여름);

}

}



열거형 변수 enum을 사용할때에는 위와 같이 마치 class에서 특정값을 땡겨갈때와 똑같이 사용하시면 됩니다.

**실사용 예시**

Copy

enum Season {

봄, 여름, 가을, 겨울

}

public class People {

public String name; *//이름*

public Season favorite\_session; *//좋아하는계절*

public static void main(String[] args) {

People people = new People();

people.name = "홍길동";

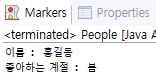
people.favorite\_session = Season.봄;

System.out.println("이름 : " + people.name);

System.out.println("좋아하는 계절 : " + people.favorite\_session);

}

}



enum이 사용되어질때는 위와 같이 사람별로 좋아하는 계절을 구해야하는 일이 생길경우 이름은 계속해서 변하지만 좋아하는계절의 계절의 경우에는 봄, 여름, 가을, 겨울 이렇게 4가지 계절로 고정입니다. 이럴때 계절에다가 enum을 활용하시면 좋습니다.

**열거객체 메소드**

**values()**

Copy

enum Season {

봄, 여름, 가을, 겨울

}

public class enum\_ex {

public static void main(String[] args) {

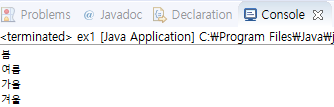
for(Season sea : Season.values()){

System.out.println(sea);

}

}

}



values()메소드는 열거 타입의 모든 열거객체들을 배열로 만들어 리턴합니다.

**ordinal()**

Copy

enum Season {

봄, 여름, 가을, 겨울

}

public class enum\_ex {

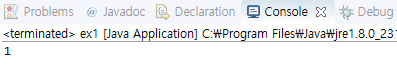
public static void main(String[] args) {

Season seson = Season.여름;

System.out.println(seson.ordinal());

}

}



Ordinal() 메소드는 전체 열거 객체 중 몇번째 열거 객체인지 알려줍니다. 열거 객체의 순번은 배열과 마찬가지로 인덱스가 0부터 시작합니다.

**valueOf()**

Copy

enum Season { *//class 외부에서 선언*

봄, 여름, 가을, 겨울

}

public class enum\_ex {

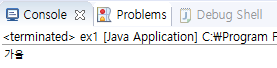
public static void main(String[] args) {

Season seson = Season.valueOf("가을");

System.out.println(seson);

}

}



values()메소드는 매개 값으로 주어지는 문자열과 동일한 문자열을 가지는 열거 객체를 리턴합니다.

**name()**

Copy

enum Season { *//class 외부에서 선언*

봄, 여름, 가을, 겨울

}

public class enum\_ex {

public static void main(String[] args) {

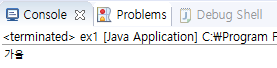
Season seson = Season.가을;

String name = seson.name();

System.out.println(name);

}

}



name() 메소드는 열거 객체가 가지고 있는 문자열을 리턴합니다. 이때 리턴되는 문자열은 열거 타입을 정의할때 사용한 상수 이름과 동일합니다.

**compareTo()**

Copy

enum Season { *//class 외부에서 선언*

봄, 여름, 가을, 겨울

}

public class enum\_ex {

public static void main(String[] args) {

Season seson1 = Season.가을;

Season seson2 = Season.겨울;

int result1 = seson1.compareTo(seson2);

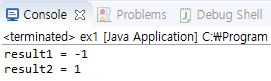
int result2 = seson2.compareTo(seson1);

System.out.println("result1 = " +result1);

System.out.println("result2 = " +result2);

}

}



compareTo()메소드는 매개값으로 주어진 열거 객체를 기준으로 전후로 몇번째 위치하는지 비교합니다. 만약 열거 객체가 매개값의 열거 객체보다 순번이 빠르다면 음수가, 순번이 늦다면 양수가 리턴됩니다.

# 4. [[Java] 자바 상수 final의 의미와 사용법](https://coding-factory.tistory.com/525)

* 코딩팩토리

* [Language / Java](https://coding-factory.tistory.com/category/Language/Java)

* 2020. 4. 23.

final의 의미는 최종적이란 뜻을 가지고 있습니다. final 필드는 초기값이 저장되면 최종적인 값이 되어 프로그램 실행 도중에 수정을 할 수 없습니다.

* [final 사용법](https://coding-factory.tistory.com/525#h)
  + [final 필드](https://coding-factory.tistory.com/525#h_1)
  + [final 객체](https://coding-factory.tistory.com/525#h_2)
  + [final 클래스](https://coding-factory.tistory.com/525#h_3)
  + [final 메서드](https://coding-factory.tistory.com/525#h_4)
  + [메서드의 인자값에 final을 사용하는 경우](https://coding-factory.tistory.com/525#h_5)

**final 사용법**

**final 필드**

Copy

final int number = 1; //final 타입 필드 [= 초기값];

final 필드는 위와 같이 선언하며 final 필드의 초기값을 줄 수 있는 방법은 딱 두가지 방법밖에 없습니다. 첫번째는 필드 선언시에 주는 방법이 있고, 두번째는 생성자를 통해서 주는 방법이 있습니다. 단순 값이라면 필드 선언시에 주는 것이 가장 간단하지만 복잡한 초기화 코드가 필요하거나 객체 생성 시에 외부 데이터로 초기화를 시켜야한다면 생성자를 통해서 초기값을 부여하는 방법을 써야 합니다. 생성자는 final 필드의 최종 초기화를 마쳐야 하는데 만약 초기화가 되지 않은 final 필드가 있다면 컴파일 에러가 발생합니다.

**final 객체**

Copy

class Company{

String name = "회사명";

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

public class Final\_ex {

public static void main(String[] args) {

final Company company = new Company();

*//company = new Company(); //객체를 한번 생성했다면 재생성 불가능*

company.setName("ex회사"); *//클래스의 필드는 변경가능*

}

}

객체 변수에 final로 선언하면 그 변수에 다른 참조 값을 지정할 수 없습니다. 즉 한번 생성된 final 객체는 같은 타입으로 재생성이 불가능합니다. 객체자체는 변경이 불가능하지만 객체 내부 변수는 변경 가능합니다.

**final 클래스**

Copy

*//final 클래스*

final class Company{

String name = "회사명";

}

*//상속 불가능*

class A\_Company extends Company{

}

클래스에 final을 사용하게되면 그 클래스는 최종상태가 되어 더이상 상속이 불가능합니다. final 클래스여도 필드는 Setter함수를 통하여 변경은 가능합니다.

**final 메서드**

Copy

class Company{

String name = "회사명";

public final void print() {

System.out.println("회사 이름은 :"+name+" 입니다.");

}

}

class A\_Company extends Company{

String name = "a회사";

*//메서드 오버라이드 불가능*

public void print() {

}

}

메서드에 final을 사용하게되면 상속받은 클래스에서 부모의 final 메서드를 재정의 할 수 없습니다. 자신이 만든 메서드를 변경할 수 없게끔 하고싶을때 사용되며 시스템의 코어부분에서 변경을 원치 않는 메서드에 많이 구현되어 있습니다.

**메서드의 인자값에 final을 사용하는 경우**

Copy

class Company{

String name = "회사명";

public void setName(final String name) {

*//name = "ex회사2"; //인자값으로 받은 final변수는 변경 불가능*

this.name = name;

}

}

public class Final\_ex {

public static void main(String[] args) {

final Company company = new Company();

company.setName("ex회사");

}

}

잘 사용하지는 않지만 코딩을 좀 더 명확하게 하고 싶은 경우 메서드의 인자값에 final을 사용하시는 분들이 종종 있습니다. final 필드와 마찬가지로 인자값에 final을 사용하는 경우 final 인자값의 변경이 불가능합니다.

# 12 [[Java] 자바 this의 의미와 사용법](https://coding-factory.tistory.com/523)

* 코딩팩토리

* [Language / Java](https://coding-factory.tistory.com/category/Language/Java)

* 2020. 4. 21.

People클래스에 Age필드와 setAge( ) 메소드가 다음과 같이 선언되어 있다고 가정해보겠습니다.

Copy

public class People{

*//필드*

int age;

*//메소드*

void setAge(int a){

age = a;

}

}

위와같이 age에 외부의 값을 입력받을 수 있는 setAge( )라는 Setter함수를 선언하였습니다. 위에서는 메서드의 인자 변수이름을 int a라 하였는데 여기서 인자값의 변수를 필드의 이름과 같이 하려면 어떻게 해야할까요?

this를 사용하면, 메서드의 인수나 변수에 필드와 같은 이름을 붙여도 그것들을 구분하여 사용할 수 있습니다. this는 주로 생성자와 메소드의 매개변수 이름이 필드와 동일한 경우, 인스턴스 멤버인 필드임을 명시하고자 할때 사용됩니다.

Copy

public class People{

*//필드*

int age;

*//메소드*

void setAge(int age){

this.age = age;

}

}

위와 같이 인자값의 변수명과 필드의 변수명이 같아도 this를 활용하면 같이 사용이 가능합니다. 위의 코드를 예로 들자면 this.age는 필드인 age를 의미합니다. 즉, this의 뒤의 변수명은 인수가 아닌 멤버변수(필드)를 의미하는 것이 됩니다.

좋아요34

공유하기

게시글 관리

구독하기

[저작자표시 비영리 변경금지](https://creativecommons.org/licenses/by-nc-nd/4.0/deed.ko)

응원하기

#### '[Language](https://coding-factory.tistory.com/category/Language) > [Java](https://coding-factory.tistory.com/category/Language/Java)' 카테고리의 다른 글

|  |  |
| --- | --- |
| [[Java] 자바 상수 final의 의미와 사용법](https://coding-factory.tistory.com/525)  (4) | 2020.04.23 |
| [[Java] 자바 static의 의미와 사용법](https://coding-factory.tistory.com/524)  (14) | 2020.04.22 |
| [[Java] 열거 타입(Enum) 사용법 & 예제](https://coding-factory.tistory.com/522)  (0) | 2020.04.20 |
| [[Java] 비트(Shift) 연산자 사용법 & 예제](https://coding-factory.tistory.com/521)  (2) | 2020.04.19 |

[댓글(12)](https://coding-factory.tistory.com/523#rp)

코딩팩토리님의  
글이 좋았다면 응원을 보내주세요!

응원하기

응원하기란?

[코딩팩토리](https://coding-factory.tistory.com/)

**[IT 분야 크리에이터](https://storyhome.kakao.com/topcreator/?category=it)**[my life is good](https://coding-factory.tistory.com/)구독하기

|  |  |
| --- | --- |
| https://img1.daumcdn.net/thumb/C100x100/?scode=mtistory2&fname=https%3A%2F%2Ftistory1.daumcdn.net%2Ftistory%2F3762699%2Fattach%2F87d5d31dfbfa4b2687baa913930229bc | [**하링아**](https://haaringa.tistory.com/)  2020.04.21 02:19[신고](https://coding-factory.tistory.com/toolbar/popup/abuseReport/?entryId=523&commentId=16250143) |
| 잘 보고 갑니다ㅎㅎㅎ 제 블로그도 놀러오세요~ | |
| [답글](https://coding-factory.tistory.com/523) [수정](https://coding-factory.tistory.com/523) | |

|  |  |
| --- | --- |
|  | [**코딩팩토리**](https://coding-factory.tistory.com/)  2020.04.21 08:53[신고](https://coding-factory.tistory.com/toolbar/popup/abuseReport/?entryId=523&commentId=16250358) |
| 방문 감사드립니다 | |
| [수정](https://coding-factory.tistory.com/523) | |

|  |  |
| --- | --- |
| https://img1.daumcdn.net/thumb/C100x100/?scode=mtistory2&fname=https%3A%2F%2Ftistory1.daumcdn.net%2Ftistory%2F3513443%2Fattach%2F7ab06abafdcc409ab8da3cfc9937a33d | [**바밸 : 균형-건강을 위한 시작**](https://bodynbalance.tistory.com/)  2020.04.21 08:55[신고](https://coding-factory.tistory.com/toolbar/popup/abuseReport/?entryId=523&commentId=16250368) |
| 잘보고 가요~^^ 구독합니다^^ | |
| [답글](https://coding-factory.tistory.com/523) [수정](https://coding-factory.tistory.com/523) | |

|  |  |
| --- | --- |
| https://img1.daumcdn.net/thumb/C100x100/?scode=mtistory2&fname=https%3A%2F%2Ftistory1.daumcdn.net%2Ftistory%2F3843774%2Fattach%2Fa04583e0868e4d379820406a29ad644e | [**JOHN TV**](https://muscle756.tistory.com/)  2020.04.21 18:00[신고](https://coding-factory.tistory.com/toolbar/popup/abuseReport/?entryId=523&commentId=16251683) |
| 포스팅 잘보고 갑니다~ 좋은하루 보내시고 자주 소통해요~ | |
| [답글](https://coding-factory.tistory.com/523) [수정](https://coding-factory.tistory.com/523) | |

|  |  |
| --- | --- |
| https://t1.daumcdn.net/tistory_admin/blog/admin/profile_default_04.png | **알 수 없는 사용자**  2020.04.21 19:04[신고](https://coding-factory.tistory.com/toolbar/popup/abuseReport/?entryId=523&commentId=16251867) |
| 개인적으로 코팅한지가 넘 오래 되어서 생각이 잘 않나지만 님의 블로그를 보니 다시 살아 있는 느낌이 납니다. 암튼 수고 하시고 대박 나세요 ^^ | |
| [답글](https://coding-factory.tistory.com/523) [수정](https://coding-factory.tistory.com/523) | |

|  |  |
| --- | --- |
| https://img1.daumcdn.net/thumb/C100x100/?scode=mtistory2&fname=https%3A%2F%2Ftistory1.daumcdn.net%2Ftistory%2F3596423%2Fattach%2F2cbecce0766d4ee6b5d4c5ec6cd48d89 | [**Flawagon**](https://flawagon119.tistory.com/)  2020.04.22 01:22[신고](https://coding-factory.tistory.com/toolbar/popup/abuseReport/?entryId=523&commentId=16252814) |
| 안녕하세요~ 좋은 정보글 감사드리며 일일방문자 25000명 축하드립니다! 구독하고갑니다:D | |
| [답글](https://coding-factory.tistory.com/523) [수정](https://coding-factory.tistory.com/523) | |

|  |  |
| --- | --- |
| https://img1.daumcdn.net/thumb/C100x100/?scode=mtistory2&fname=https%3A%2F%2Ftistory1.daumcdn.net%2Ftistory%2F675561%2Fattach%2Fe4d9ce3318814aeb9a9e9006f47796e5 | [**잉여토기**](https://dldduxhrl.tistory.com/)  2020.04.22 09:27[신고](https://coding-factory.tistory.com/toolbar/popup/abuseReport/?entryId=523&commentId=16253201) |
| this를 사용하면, 메서드의 인수나 변수에 필드와 같은 이름을 붙여도 구분할 수 있군요. this가 이것이라는 뜻의 영어 단어인지만 알았는데 자바에서 무척 필요한 디스네요. | |
| [답글](https://coding-factory.tistory.com/523) [수정](https://coding-factory.tistory.com/523) | |

|  |  |
| --- | --- |
| https://img1.daumcdn.net/thumb/C100x100/?scode=mtistory2&fname=https%3A%2F%2Ftistory1.daumcdn.net%2Ftistory%2F2984966%2Fattach%2F347dbe2948b745fc8a5e6f72f52564cd | [**돗개에요**](https://linuxiyka.tistory.com/)  2020.04.22 09:39[신고](https://coding-factory.tistory.com/toolbar/popup/abuseReport/?entryId=523&commentId=16253225) |
| 저도 코딩 공부를 다시 시작하려고 하는데 자주 올 것 같네요 ㅎㅎ 구독하겠습니다. 소통해요~ | |
| [답글](https://coding-factory.tistory.com/523) [수정](https://coding-factory.tistory.com/523) | |

|  |  |
| --- | --- |
| https://img1.daumcdn.net/thumb/C100x100/?scode=mtistory2&fname=https%3A%2F%2Ftistory1.daumcdn.net%2Ftistory%2F3764321%2Fattach%2Fc6046391ce8243b386aae9a1e9ae39c4 | [**대박리뷰**](https://socialcast.tistory.com/)  2020.04.22 11:30[신고](https://coding-factory.tistory.com/toolbar/popup/abuseReport/?entryId=523&commentId=16253402) |
| 좋은 정보 감사해요~~ 구독, 좋아요 꾸욱 누룹니다~~ | |
| [답글](https://coding-factory.tistory.com/523) [수정](https://coding-factory.tistory.com/523) | |

|  |  |
| --- | --- |
| https://img1.daumcdn.net/thumb/C100x100/?scode=mtistory2&fname=https%3A%2F%2Ft1.daumcdn.net%2Ftistory_admin%2Fstatic%2Fmanage%2Fimages%2Fr3%2Fdefault_L.png | [**진진자**](https://odjl.tistory.com/)  2020.04.22 15:33[신고](https://coding-factory.tistory.com/toolbar/popup/abuseReport/?entryId=523&commentId=16253808) |
| 좋은 정보 감사합니다!!! 구독 했어요 !! 피드로 자주 소통해요~~! | |
| [답글](https://coding-factory.tistory.com/523) [수정](https://coding-factory.tistory.com/523) | |

|  |  |
| --- | --- |
| https://t1.daumcdn.net/tistory_admin/blog/admin/profile_default_05.png | **익명**  2020.04.22 15:37 |
| 비밀댓글입니다. | |
| [답글](https://coding-factory.tistory.com/523) [수정](https://coding-factory.tistory.com/523) | |

|  |  |
| --- | --- |
| https://img1.daumcdn.net/thumb/C100x100/?scode=mtistory2&fname=https%3A%2F%2Ftistory1.daumcdn.net%2Ftistory%2F3006751%2Fattach%2F868cf0e9b5eb4f159bbf34771a4a0f37 | [**파라피아**](https://chinesemandarin.tistory.com/)  2020.04.22 18:39[신고](https://coding-factory.tistory.com/toolbar/popup/abuseReport/?entryId=523&commentId=16254215) |
| 너무 대단하시네요 자주 와서 많이 공부하고 갈께요 진짜로요 대단하십니다. | |
| [답글](https://coding-factory.tistory.com/523) [수정](https://coding-factory.tistory.com/523) | |

양식의 맨 위





비밀 글



등록

양식의 맨 아래



**코딩팩토리**

my life is good

### 카테고리

* + Language(246)
  + Web(202)
  + Mobile(47)
  + DB(133)
  + OS(111)
  + ETC. (131)N
  + 일상(100)
* [**브런치(brunch)**](https://brunch.co.kr/@e8362cab2f0c484#articles)

### 애드센스

### 최근 댓글

* [포스팅 해주셔서 감사드리고 오늘 하루도 좋은 ⋯](https://coding-factory.tistory.com/1040#comment21204482)
* [유익한 포스팅 잘 보았습니다. 편안한 저녁시간⋯](https://coding-factory.tistory.com/1040#comment21195350)
* [감사합니다. 보고 헷갈리는 것 정리됐네요. 참⋯](https://coding-factory.tistory.com/802#comment21014296)
* [좋은 글 보고가요~!](https://coding-factory.tistory.com/1033#comment21005700)
* [유용한 정보 잘 읽고 갑니다. 편안한 저녁 되⋯](https://coding-factory.tistory.com/1034#comment20980842)

### 애드센스

### 통계

* 전체 : 30,445,426
* 오늘 : 5,030
* 어제 : 13,525

### 달력

| [«](https://coding-factory.tistory.com/archive/202403)   [2024/04](https://coding-factory.tistory.com/archive/202404)   [»](https://coding-factory.tistory.com/archive/202405) | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **일** | **월** | **화** | **수** | **목** | **금** | **토** |
|  | 1 | 2 | 3 | [4](https://coding-factory.tistory.com/archive/20240404) | [5](https://coding-factory.tistory.com/archive/20240405) | [6](https://coding-factory.tistory.com/archive/20240406) |
| [7](https://coding-factory.tistory.com/archive/20240407) | [8](https://coding-factory.tistory.com/archive/20240408) | [9](https://coding-factory.tistory.com/archive/20240409) | [10](https://coding-factory.tistory.com/archive/20240410) | [11](https://coding-factory.tistory.com/archive/20240411) | [12](https://coding-factory.tistory.com/archive/20240412) | 13 |
| 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 |
| 28 | 29 | 30 |  |  |  |  |

### 보관함

* [2024/04(9)](https://coding-factory.tistory.com/archive/202404)
* [2024/03(4)](https://coding-factory.tistory.com/archive/202403)
* [2024/02(19)](https://coding-factory.tistory.com/archive/202402)
* [2024/01(8)](https://coding-factory.tistory.com/archive/202401)

## Copyright © 코딩팩토리 All Rights Reserved

## Designed by [JB FACTORY](https://www.jbfactory.net/)

**코딩팩토리**구독하기

# 14 [[Java] 자바 static의 의미와 사용법](https://coding-factory.tistory.com/524)

* 코딩팩토리

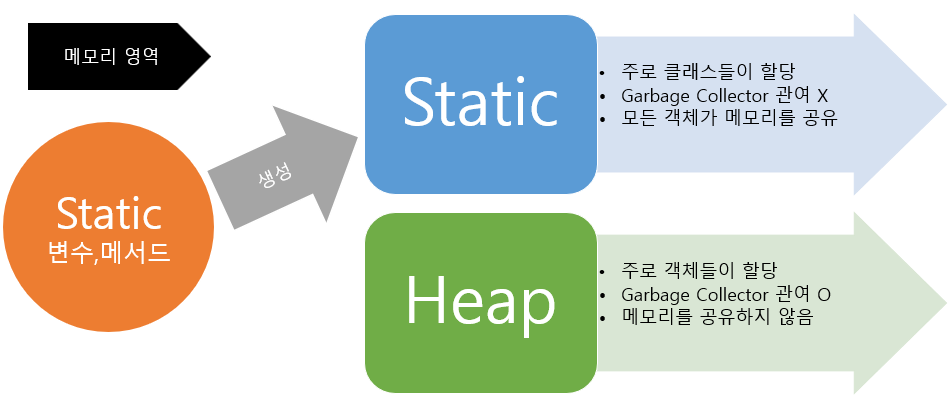
* [Language / Java](https://coding-factory.tistory.com/category/Language/Java)

* 2020. 4. 22.

**정적(Static)이란?**

정적(static)은 고정된이란 의미를 가지고 있습니다. Static이라는 키워드를 사용하여 Static변수와 Static메소드를 만들 수 있는데 다른말로 정적필드와 정적 메소드라고도 하며 이 둘을 합쳐 정적 멤버라고 합니다. (클래스 멤버라고도 합니다.) 정적 필드와 정적 메소드는 객체(인스턴스)에 소속된 멤버가 아니라 클래스에 고정된 멤버입니다. 그렇기에 클래스 로더가 클래스를 로딩해서 메소드 메모리 영역에 적재할때 클래스별로 관리됩니다. 따라서 클래스의 로딩이 끝나는 즉시 바로 사용할 수 있습니다.

**정적(Static) 멤버 생성**



Static 키워드를 통해 생성된 정적멤버들은 Heap영역이 아닌 Static영역에 할당됩니다. Static 영역에 할당된 메모리는 모든 객체가 공유하여 하나의 멤버를 어디서든지 참조할 수 있는 장점을 가지지만 Garbage Collector의 관리 영역 밖에 존재하기에 Static영역에 있는 멤버들은 프로그램의 종료시까지 메모리가 할당된 채로 존재하게 됩니다. 그렇기에 Static을 너무 남발하게 되면 만들고자 하는 시스템 성능에 악영향을 줄 수 있습니다.

**정적(Static) 멤버 선언**

필드나 메소드를 생성 시 인스턴스로 생성할것인지 정적으로 생성할것인지에 대한 판단 기준은 공용으로 사용하느냐 아니냐로 내리면 됩니다. 그냥 생성한다면 자동으로 인스턴스로 생성되며 정적으로 생성하려면 필드와 메소드 선언 시 static이라는 키워들를 추가적으로 붙이면 됩니다.

Copy

static int num = 0; *//타입 필드 = 초기값*

public static void static\_method(){} *//static 리턴 타입 메소드 {}*

**정적(Static) 필드 사용 예시**

Copy

class Number{

static int num = 0; *//클래스 필드*

int num2 = 0; *//인스턴스 필드*

}

public class Static\_ex {

public static void main(String[] args) {

Number number1 = new Number(); *//첫번째 number*

Number number2 = new Number(); *//두번쨰 number*

number1.num++; *//클래스 필드 num을 1증가시킴*

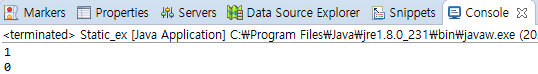
number1.num2++; *//인스턴스 필드 num을 1증가시킴*

System.out.println(number2.num); *//두번째 number의 클래스 필드 출력*

System.out.println(number2.num2); *//두번째 number의 인스턴스 필드 출력*

}

}



Number이라는 클래스안에 클래스 변수 num과 인스턴스 변수 num2를 생성하였고 두개의 Number인스턴스 number1과 number2를 생성했을때 number1에서 num1과 num2를 각각 1씩 증가시키고 number2에서 num1와 num2를 각각 출력시켰을때는 num1은 1, num2는 0이 출력되었습니다. 왜 이런 현상이 나타났느냐면 인스턴스 변수는 인스턴스가 생성될 때마다 생성되므로 인스턴스마다 각기 다른 값을 가지지만 정적 변수는 모든 인스턴스가 하나의 저장공간을 공유하기에 항상 같은 값을 가지기에 나타난 현상입니다.

**정적(Static) 메서드 사용 예시**

Copy

class Name{

static void print() { *//클래스 메소드*

System.out.println("내 이름은 홍길동입니다.");

}

void print2() { *//인스턴스 메소드*

System.out.println("내 이름은 이순신입니다.");

}

}

public class Static\_ex {

public static void main(String[] args) {

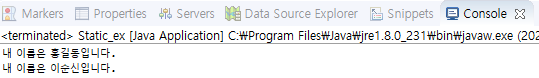
Name.print(); *//인스턴스를 생성하지 않아도 호출이 가능*

Name name = new Name(); *//인스턴스 생성*

name.print2(); *//인스턴스를 생성하여야만 호출이 가능*

}

}



정적 메소드는 클래스가 메모리에 올라갈 때 정적 메소드가 자동적으로 생성됩니다. 그렇기에 정적 메소드는 인스턴스를 생성하지 않아도 호출을 할 수 있습니다. 정적 메소드는 유틸리티 함수를 만드는데 유용하게 사용됩니다.