열거 타입

열거 타입

❖ 열거 타입(Enumeration Type)

- 한정된 값만을 갖는 데이터 타입

- 한정된 값은 열거 상수(Enumeration Constant)로 정의

열거 타입

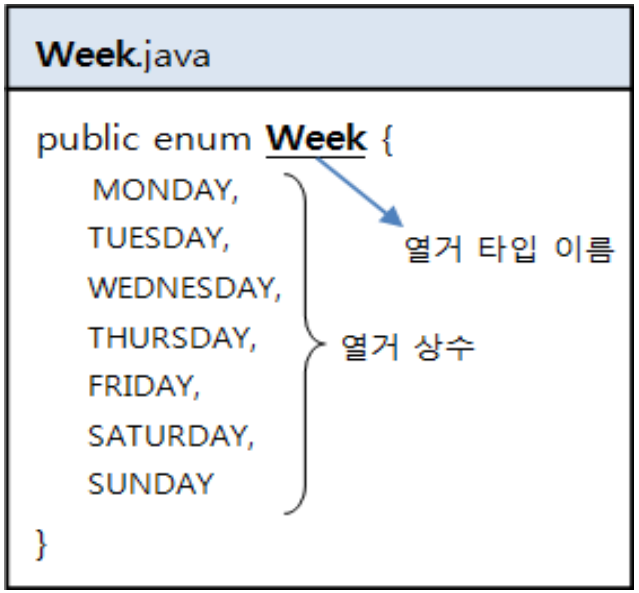
❖ 열거 타입 선언

- 파일 이름과 동일한 이름으로 다음과 같이 선언 (첫 글자 대문자)

```
public enum 열거타입이름 { ... }
```

- 한정된 값인 열거 상수 정의
  - 열거 상수 이름은 관례적으로 모두 대문자로 작성
  - 다른 단어가 결합된 이름일 경우 관례적으로 밑줄(_)로 연결

```
public enum Week { MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, ... }
public enum LoginResult { LOGIN_SUCCESS, LOGIN_FAILED }
```



열거 타입

❖ 열거타입과 열거상수 : Week.java

```
public enum Week {
      MONDAY,
      TUESDAY,
      WEDNESDAY,
      THURSDAY,
      FRIDAY,
      SATURDAY,
      SUNDAY
}
```

❖ 열거 타입 변수

- 열거 타입 변수 선언

```
열거타입 변수;
```

```
Week today;
Week reservationDay;
```

- 열거 상수 값 저장 - 열거 타입 변수값은 열거 상수 중 하나

```
열거타입 변수 = 열거타입.열거상수;
Week today = Week.SUNDAY;
```

- 열거 타입 변수는 참조 타입
  - 열거 타입 변수는 참조 타입이므로 null 값 저장 가능

```
Week birthday = null;
```

열거 타입

❖ 열거타입과 열거상수 : AdvancedForExample.java

```
import java.util.Calendar;

public class EnumWeekExample {
   public static void main(String[] args) {
      Week today = null;
      Calendar cal = Calendar.getInstance();
      int week = cal.get(Calendar.DAY_OF_WEEK);

      switch (week) {
      case 1:
         today = Week.SUNDAY; break;
      case 2:
         today = Week.MONDAY; break;
      case 3:
         today = Week.TUESDAY;  break;
      case 4:
         today = Week.WEDNESDAY; break;
      case 5:
         today = Week.THURSDAY; break;
```

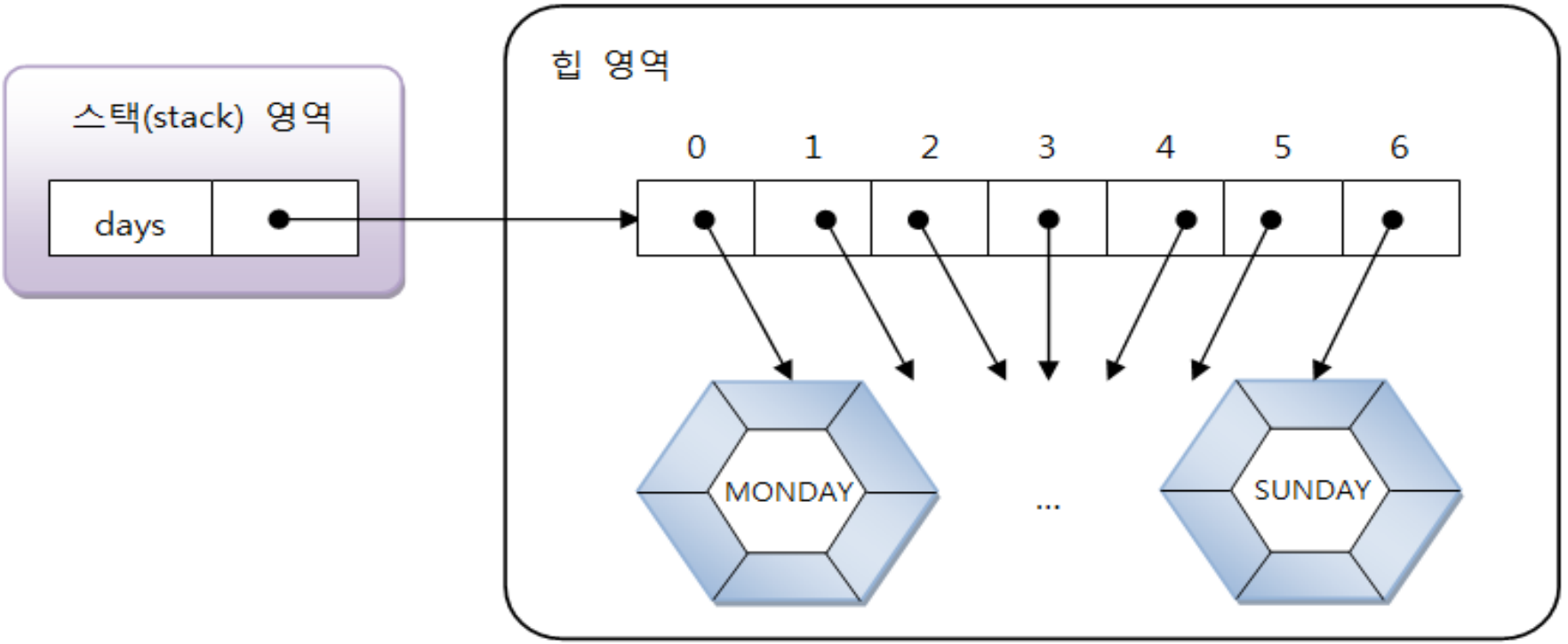
❖ 열거타입과 열거상수 : AdvancedForExample.java

```
        case 6:
            today = Week.FRIDAY;break;
        case 7:
            today = Week.SATURDAY;    break;
        }

        System.out.println("오늘 요일: " + today);

        if (today == Week.SUNDAY) {
            System.out.println("일요일에는 축구를 합니다.");
        } else {
            System.out.println("열심히 자바 공부합니다.");
        }
    }
}
```

열거 타입

❖ 열거 객체의 메소드

- 열거 객체는 열거 상수의 문자열을 내부 데이터로 가지고 있음



열거 타입

❖ 열거 객체의 메소드

- 열거 타입은 컴파일 시 `java.lang.Enum` 클래스를 자동 상속
  - 열거 객체는 `java.lang.Enum` 클래스의 메소드 사용 가능

리턴타입	메소드(매개변수)	설명
String	name()	열거 객체의 문자열을 리턴
int	ordinal()	열거 객체의 순번(0 부터 시작)를 리턴
int	compareTo()	열거 객체를 비교해서 순번 차이를 리턴
열거타입	valueOf(String name)	주어진 문자열의 열거 객체를 리턴
열거배열	values()	모든 열거 객체들을 배열로 리턴

열거 타입

❖ 열거 객체의 메서드: EnumMethodExample.java

```
public class EnumMethodExample {
   public static void main(String[] args) {
      // name() 메소드
      Week today = Week.SUNDAY;
      String name = today.name();
      System.out.println(name);

      // ordinal() 메소드
      int ordinal = today.ordinal();
      System.out.println(ordinal);

      // compareTo() 메소드
      Week day1 = Week.MONDAY;
      Week day2 = Week.WEDNESDAY;
      int result1 = day1.compareTo(day2);
      int result2 = day2.compareTo(day1);
      System.out.println(result1);
      System.out.println(result2);
```

❖ 열거 객체의 메서드: EnumMethodExample.java

```
//valueOf() 메소드
Week weekDay = Week.valueOf("SUNDAY");
if(weekDay == Week.SATURDAY || weekDay == Week.SUNDAY) {
   System.out.println("주말 이군요");
} else {
   System.out.println("평일 이군요");
}

if (args.length == 1) {
   String strDay = args[0];
   Week weekDay = Week.valueOf(strDay);
   if (weekDay == Week.SATURDAY || weekDay == Week.SUNDAY) {
      System.out.println("주말 이군요");
   } else {
      System.out.println("평일 이군요");
   }
}
```

열거 타입

❖ 열거 객체의 메서드: EnumMethodExample.java

```
        // values() 메소드
        Week[] days = Week.values();
        for (Week day : days) {
           System.out.println(day);
        }
    }
}
```

열거 타입

❖ 열거 상수를 다른 값과 연관짓기

- 다음의 열거 상수들을 "봄", "여름", "가을", "겨울"이라는 이름과 연관시킬 수 있음

```
enum Season {
    SPRING, SUMMER, FALL, WINTER
}
```

열거 타입

❖ 열거 상수를 다른 값과 연관짓기

- 1단계 : 각각의 열거 상수 뒤에 연관 값을 기술

```
enum Season {
    SPRING("봄"), SUMMER("여름"), FALL("가을"), WINTER("겨울") ;

}
```

필드, 생성자, 메서드를 선언할 수 있는 위치

열거 상수와 다른 구성요소를 구분하는 세미콜론

열거 타입

❖ 열거 상수를 다른 값과 연관짓기

- 2단계 : 연관 값을 저장할 필드 선언

```
enum Season {
    SPRING("봄"), SUMMER("여름"), FALL("가을"), WINTER("겨울") ;
    final private String name;
}
```

반드시 써야하는 키워드

열거 타입

❖ 열거 상수를 다른 값과 연관짓기

- 3단계 : 생성자의 선언

```
enum Season {
    SPRING("봄"), SUMMER("여름"), FALL("가을"), WINTER("겨울") ;
    final private String name;
    Season(String name) {
        this.name = name;
    }
}
```

이런 값이 생성자 파라미터로 넘어옵니다.

그 파라미터 값으로
필드를 초기화 해야 합니다.

열거 타입

❖ 열거 상수를 다른 값과 연관짓기

- 4단계 : 메소드의 선언

```
enum Season {
    SPRING("봄"), SUMMER("여름"), FALL("가을"), WINTER("겨울") ;
    final private String name;
    Season(String name) {
        this.name = name;
    }
    String value() {
        return name;
    }
}
```

열거 상수에 연관된 값을 리턴하는 메소드

열거 타입

❖ 열거 상수를 다른 값과 연관짓기 : Season.java

```
enum Season {
    SPRING("봄"), SUMMER("여름"), FALL("가을"), WINTER("겨울") ;
    final private String name;

    Season(String name) {
        this.name = name;
    }

    String value() {
        return name;
    }
}
```

열거 타입

❖ 열거 상수를 다른 값과 연관짓기 : SeasonExample.java

```
class SeasonExample {
    public static void main(String args[]) {
        printSeason(Season.SPRING);
        printSeason(Season.SUMMER);
        printSeason(Season.FALL);
        printSeason(Season.WINTER);
    }

    static void printSeason(Season season) {
        System.out.println(season.value());
    }
}
```