

목 차

- 1 Vector 클래스.....
- 2 Stack 클래스.....
- 3 Dictionary 클래스.....
- 4 Hashtable 클래스.....
- 5 Properties 클래스.....

Generic

Generic이란?

- 클래스내부에서 사용할 데이터 타입을 외부에서 지정하는 방법

```
class Demo<T> {
    public T data;
}
```

//메인

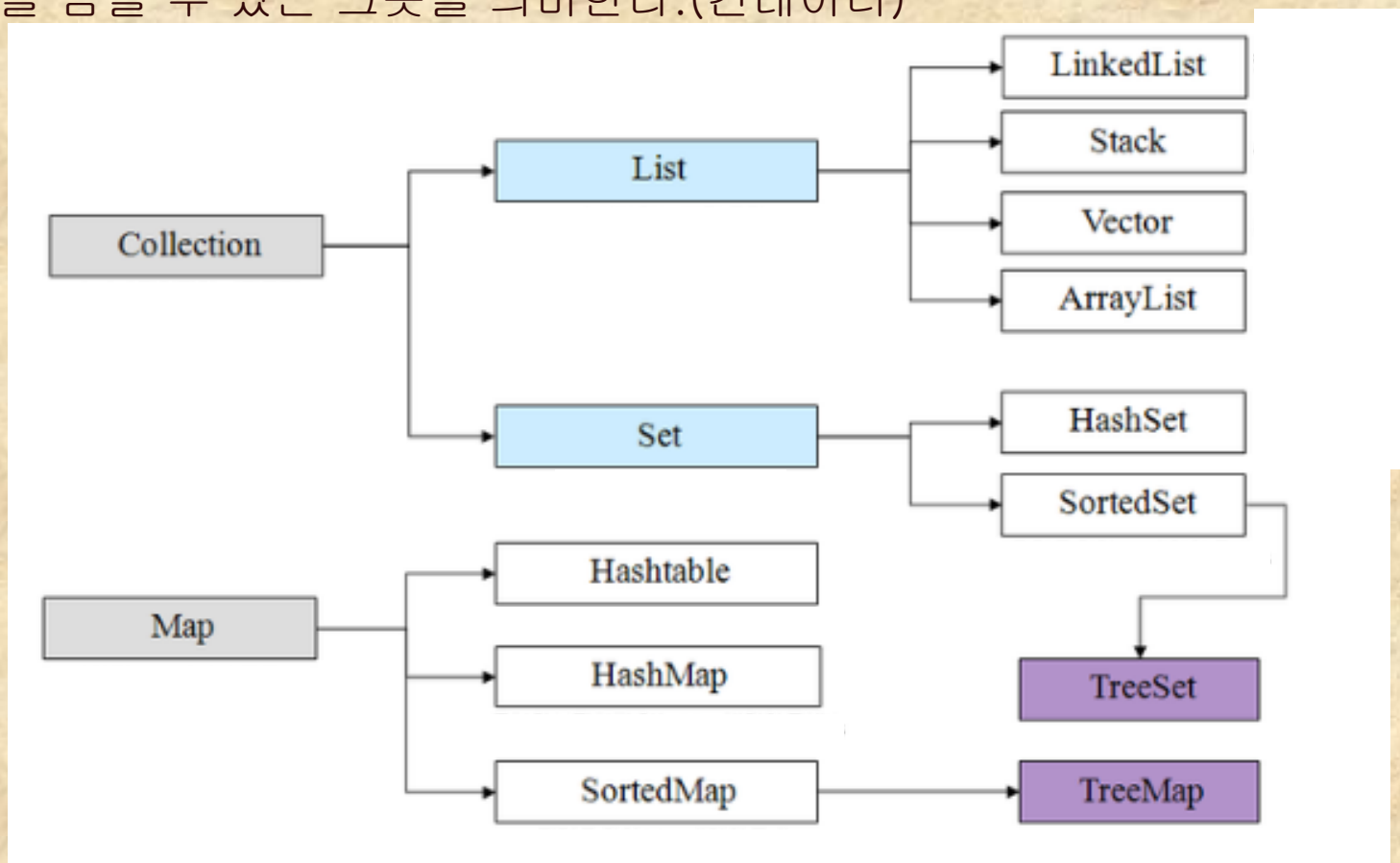
```
Demo<String> d1 = new Demo<String>();
```

```
Demo<Integer> d2 = new Demo<Integer>();
```

Collections Framework

Collections Framework 이란?

- 값을 담을 수 있는 그릇을 의미한다.(컨테이너)



1. Vector 클래스


- ❖ 대부분의 프로그래밍 언어에서는 배열구조를 제공한다. 하지만 배열이라는 구조는 정적인 크기만을 가지고 있다 동적으로 변하지 못한다.
- ❖ 그러나 자바에서는 **Vector클래스**를 이용해 동적인 크기를 갖는 **동적배열**을 제공한다.
- ❖ Vector는 Object클래스의 객체 레퍼런스로 저장된다. Object클래스는 모든 클래스의 부모클래스이므로 모든 클래스형의 객체가 Vector에 저장될 수 있다.
- ❖ Vector 클래스는 기본 데이터 타입(byte,int,float...)을 저장하지 못한다. 이 경우에는 Wrapper클래스(Byte,Int,Float...)를 사용해야한다.

- ❖ Vector 클래스는 java.util 패키지에 속해있으므로 Vector 클래스를 사용하기 위해서는 java.util 패키지를 import해야한다.
- ❖ JDK1.5 버전이후부터 Vector 클래스의 객체를 생성시 Vector에 저장되는 객체의 타입을 지정하도록 되어 있으며, 그렇지 않을 경우는 경고가 발생한다.
- ❖ 객체의 타입을 지정해두면 나중에 형변환을 지정하지 않아도 된다.


Vector 클래스형 객체를 생성하는 문장

```
Vector<저장할 객체타입> 객체명 = new Vector<저장할 객체타입>( );
```

JDK1.5버전 이후 추가요구



JDK1.5버전 이후 추가요구



vector

```
1 package 내장클래스;
2 import java.util.Vector;
3 public class VectorTest2 {
4     public static void main(String[] args) {
5         Object obj;
6
7         Vector<Object> vec =new Vector<Object>(2);
8         obj=10;
9         vec.add(obj);//엘리먼트 추가
10        System.out.println("용량은#1 : "+vec.capacity());
11
12        System.out.println("크기는#1 : "+vec.size());
13        obj="hi";
14        vec.addElement(obj);//엘리먼트 추가
15        System.out.println("용량은#2 : "+vec.capacity());
16        System.out.println("크기는#2 : "+vec.size());
17        obj="Nice Day";//엘리먼트 추가
18        vec.addElement(obj);
19        System.out.println("용량은#3 : "+vec.capacity());
20        System.out.println("크기는#3 : "+vec.size());
21    }
22 }
```

2. Stack 클래스

- ❖ Stack 구조는 접시를 닦아서 쌓듯이 나중에 들어온것이 먼저 서비스를 받는 구조이다.
- ❖ 자바의 Stack 클래스는 이러한 구조를 제공한다.
- ❖ Stack 클래스는 Vector 클래스의 서브 클래스로 vector를 stack으로 취급할 수 있도록 push, pop, peek, empty, search 등의 5개 연산을 메소드로 제공한다.
- ❖ Stack 클래스는 객체만을 저장할 수 있으며, java.util 패키지에 속해있으므로 이를 사용하기 위해서는 java.util 패키지를 import해야한다.

Stack 클래스형 객체를 생성하는 문장

```
Stack<저장할 객체타입> 객체명 = new Stack<저장할 객체타입>( );
```

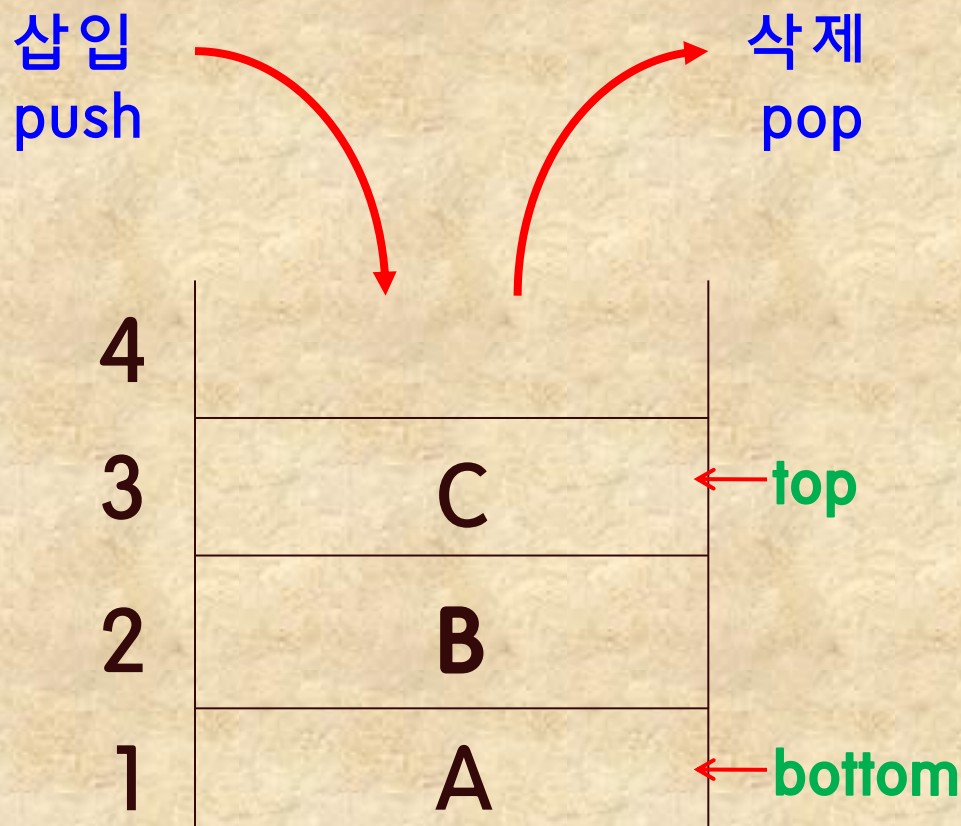
JDK1.5버전 이후 추가요구



JDK1.5버전 이후 추가요구



stack의 구조



삽입연산(Push)

```

Top = Top + 1
if Top > M Then
    Overflow
Else
    X(Top) ← Item
    
```

삭제연산(Pop Up)

```

if Top = 0 Then
    Underflow
Else
    Item ← X(Top)
    Top = Top - 1
    
```

M : 스택의 크기
Top : 스택포인터
X : 스택의 이름

Eclipse 창에서 다음과 같이 입력한다.

```
1 package 내장클래스;
2 import java.util.Stack;
3 public class StackTest {
4     public static void main(String[] args) {
5         Object obj;
6         Stack<Object> st = new Stack<Object>();
7
8         if(st.empty()){//스택이 비어 있으면
9             for(int i=1; i<=3; i++){//스택에 데이터를
10                 st.push(new String("Hi!" + i));
11                 System.out.println(st.peek());
12             }
13         }
14         st.pop();
15         System.out.println(st.peek());
16         st.pop();
17         System.out.println(st.peek());
18         st.push(new String("everybody!"));
19         System.out.println(st.peek());
20         st.push(new String("Nice Day!"));
21         System.out.println(st.peek());
22     }
23 }
```

```
VectorTest.java StackTest.java
1 package pk24;
2 import java.util.Stack;
3 public class StackTest {
4     public static void main(String[] args) {
5         // TODO Auto-generated method stub
6         Object obj;
7         //empty 스택생성
8         Stack<Object> st = new Stack<Object>();
9
10        if(st.empty()){//스택이 비어 있으면
11            for(int i=1; i<=3; i++){//스택에 데이터를 3개 추가
12                st.push(new String("Hi!" + i));
13                System.out.println(st.peek());
14            }
15        }//of if
16        st.pop();
17        System.out.println(st.peek());
18        st.pop();
19        System.out.println(st.peek());
20        st.push(new String("everybody!"));
21        System.out.println(st.peek());
22        st.push(new String("Nice Day!"));
23        System.out.println(st.peek());
24    }
25 }
26
```

실행창에 결과값이 출력된 것을
확인할 수 있다.

Problems @ Javadoc Declaration Console
terminated StackTest [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe

Hi!1
Hi!2
Hi!3
Hi!2
Hi!1
everybody!
Nice Day!

3. Dictionary 클래스

- ❖ Dictionary 클래스는 추상클래스로서 Hashtable 클래스의 부모 클래스이며, key와 value의 쌍으로 구성되어 있다.
- ❖ 한 Dictionary 객체에서 모든 키는 최소한 하나의 값과 연결되어 있어, 키를 가지고 연결된 값을 찾아 낼 수 있다.
- ❖ Dictionary 클래스는 java.util 패키지에 속해있으므로 이를 사용하기 위해서는 java.util 패키지를 import해야한다.

4. Hashtable 클래스

- ❖ Hashtable 클래스는 해시테이블을 구현한 클래스로서 key와 value의 쌍으로 구성되어 있다. null이 아닌 객체는 키 또는 값으로 사용될 수 있다.
- ❖ 해시테이블에서 안정적으로 객체를 저장하고 회수하기 위해서는 key로 사용되는 객체는 반드시 hashCode()와 equals()메소드를 사용하는 것이 효과적이다.
- ❖ Hashtable 클래스는 java.util 패키지에 속해있으므로 이를 사용하기 위해서는 java.util 패키지를 import해야한다.

5. Properties 클래스

- ❖ Properties 클래스는 Hashtable 클래스의 자식 클래스로서 파일로 저장된 key, value를 읽어들이거나 key, value 쌍을 파일로 저장할 수 있는 객체이다.
- ❖ Properties 클래스는 입,출력 스트림을 통해서 읽거나 저장될 수 있으며, 각 key와 이에 연결된 value는 문자열로 구성된 Properties리스트로 되어있다.
- ❖ java.util 패키지에 속해있으므로 이를 사용하기 위해서는 java.util 패키지를 import해야한다.