목차

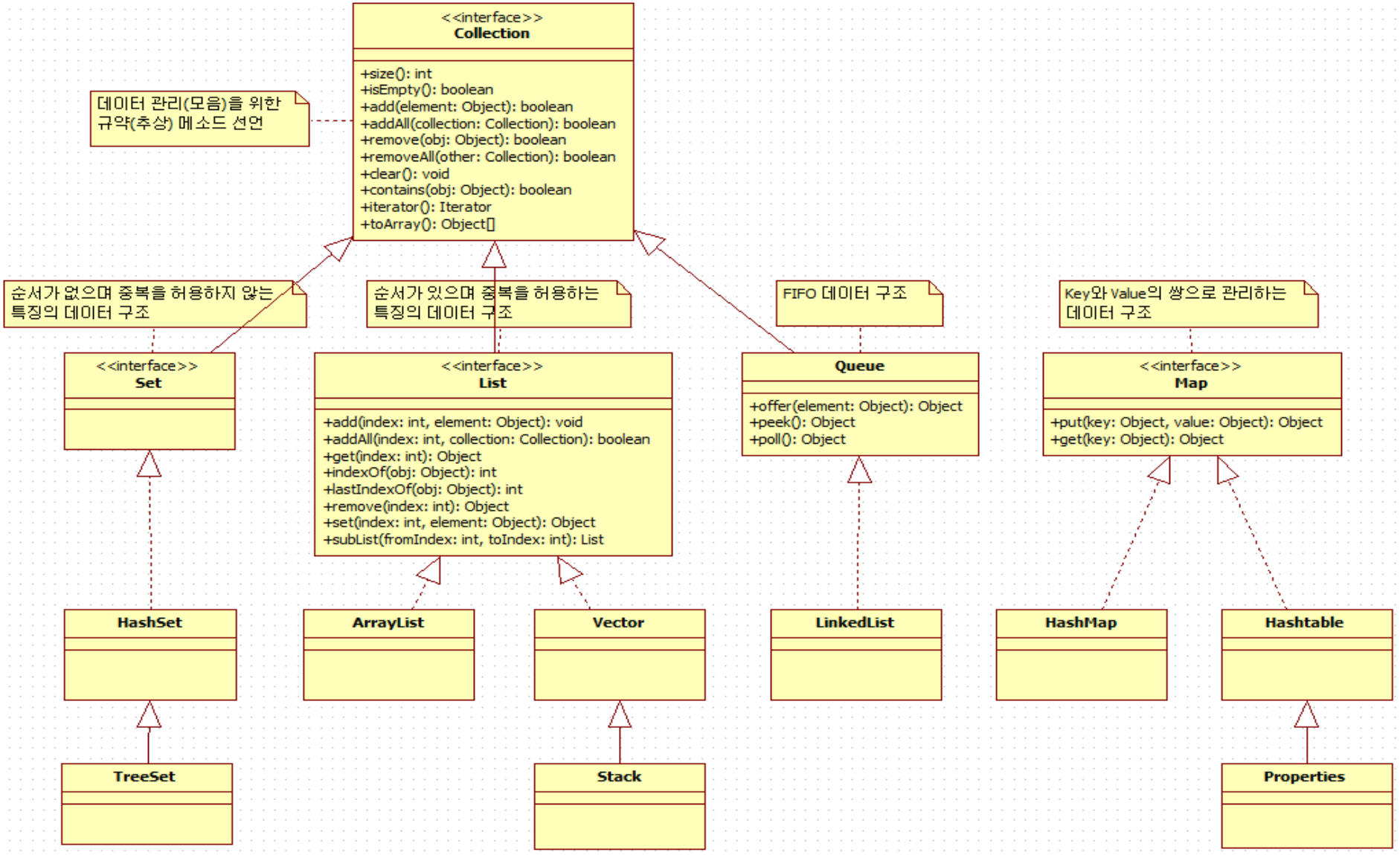
- 컬렉션 프레임워크
- Collection 인터페이스
- Set 인터페이스
- List 인터페이스
- 확장 for 문

Collection Framework

- 프레임워크란? 애플케이션의 전체적인 구조(뼈대)와 제어흐름을 담당하는 재사용 가능한 인터페이스와 클래스들의 집합

- 자바 표준 API에는 배열의 단점을 개선한 데이터 구조를 위한 Collection Framework를 제공한다

Collection Framework 구조



Collection 인터페이스의 주요 메소드

메소드	설 명
int size()	요소가 몇 개 들었는지를 반환
boolean isEmpty()	컬렉션이 비었는지를 반환
boolean add(Object element)	요소 추가 성공시 true 반환
boolean remove(Object obj)	요소 삭제 성공시 true 반환
boolean removeAll(Collection other)	요소 전체 삭제
boolean contains(Object obj)	해당 객체가 컬렉션 클래스에 포함되어 있으면 true, 그렇지 않으면 false
Iterator iterator()	Iterator 인터페이스를 얻어냄
Object[] toArray()	컬렉션에 들어 있는 요소를 객체 배열로 바꿈

Set 인터페이스

```
HashSet<String> set;
set=new HashSet<String>( );

set.add("사과");
set.add("바나나");
set.add("귤");
set.add("오렌지");
set.add("바나나");
```

Iterator 인터페이스

메소드	설 명
boolean hasNext()	요소가 있으면 true 반환 없으면 false 반환
E next()	요소를 얻어낸다.

```
Iterator elements=set.iterator();
while(elements.hasNext())
    System.out.println(elements.next());
```

List 인터페이스

- List란 데이터를 일렬로 늘어놓은 자료 구조를 말한다.
- ArrayList, Vector, Stack,LinkedList가 모두 List 인터페이스를 구현한 클래스이다.
- List는 Set과는 달리 요소가 순차적으로 처리된다.

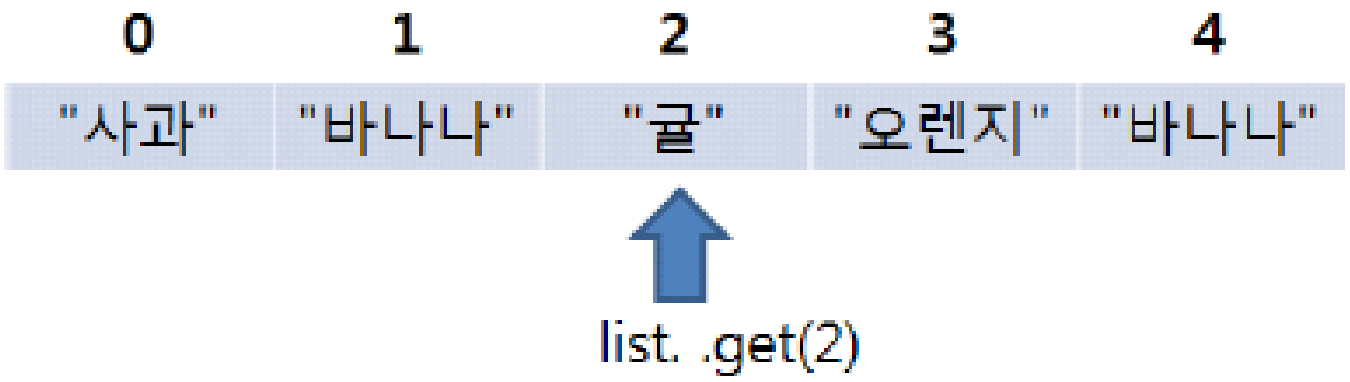
ArrayList 클래스

```
ArrayList<String> list;
list =new ArrayList<String>( );
list.add("사과");
list.add("바나나");
list.add("귤");
list.add("오렌지");
list.add("바나나");
```

0	1	2	3	4
"사과"	"바나나"	"귤"	"오렌지"	"바나나"

get 메소드

String item=list.get(2);



ArrayList에서 원소 추가, 변경, 삭제하기

메소드	설 명
void add(int index, E element)	index 위치에 element로 주어진 객체를 저장한다. 해당 위치의 객체는 뒤로 밀려난다.
E set(int index, E element)	index 위치의 요소를 element로 주어진 객체를 대체한다.
E remove(int index)	index 위치의 객체를 지운다.

ArrayList에서 원소 추가, 변경, 삭제하기

ArrayList에서 데이터 검색

메소드	설 명
int indexOf(Object o)	전달 인자로 준 객체를 앞에서부터 찾아 해당 위치를 반환. 못 찾으면 –1 반환
int lastIndexOf(Object o)	객체를 마지막 위치부터 찾음

Vector 클래스

메소드	설 명
public Vector()	디폴트 생성자로 빈 벡터 객체를 생성한다.
public Vector(int initialCapacity)	initialCapacity로 지정한 크기의 벡터 객체를 생성한다.
public Vector(int initialCapacity, int capacityIncrement)	initialCapacity로 지정한 크기의 벡터 객체를 생성하되, 새로운 요소가 추가되어 원소가 늘어나야 하면 capacityIncrement만큼 늘어난다.

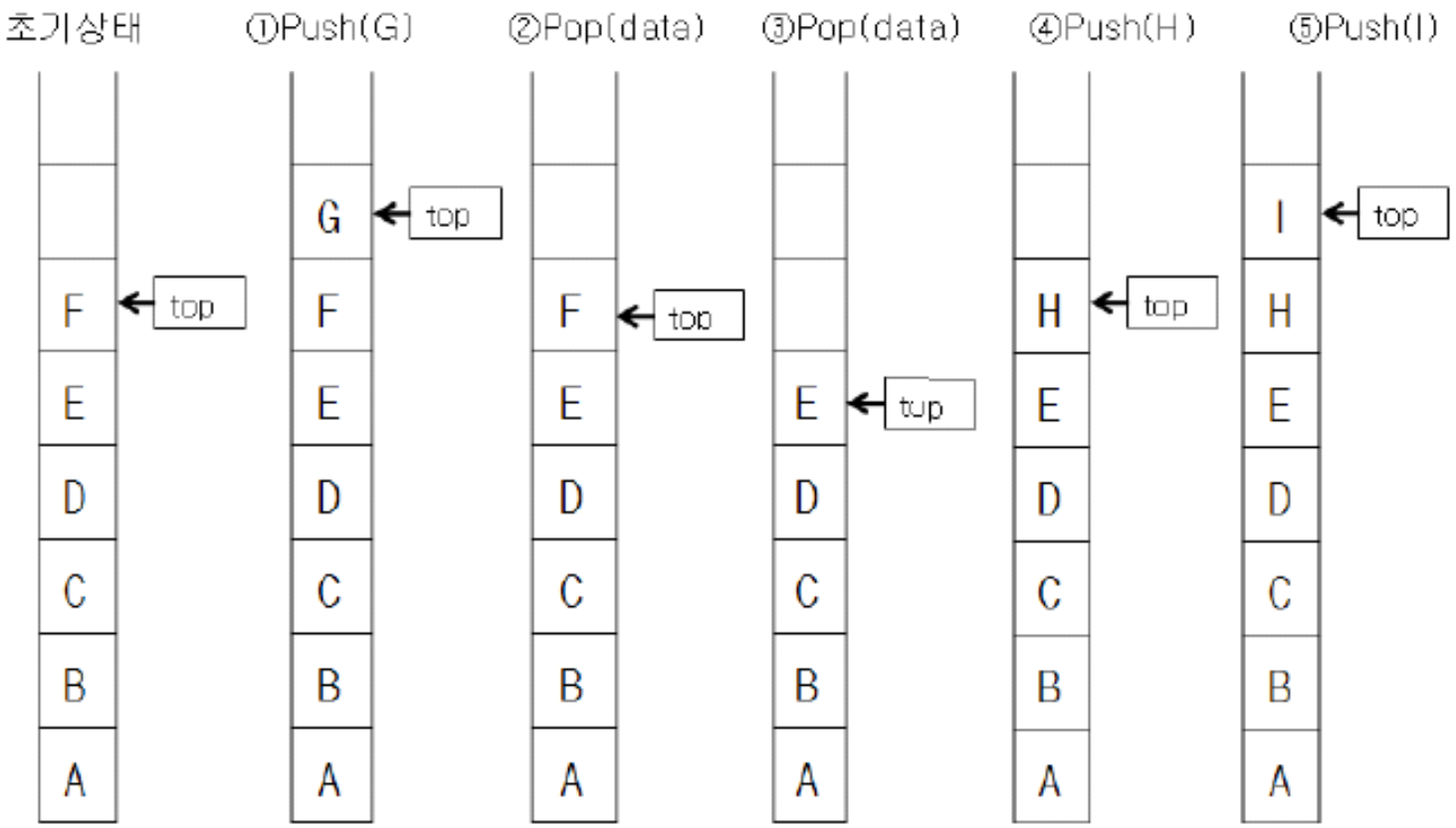
Vector 클래스

메소드	설 명
addElement()	객체를 저장함. add()도 같은 기능 제공
setElement()	index로 지정한 위치의 객체를 설정한다. set도 같은 기능 제공
get(int index)	index로 지정된 요소를 반환한다.
firstElement()	Vector의 첫 번째 요소에 반환한다.
lastElement()	Vector의 마지막 요소를 반환한다.
void trimToSize()	Vector의 용량을 현재 크기로 줄여 준다.
int capacity()	벡터의 용량을 반환
Enumeration elements()	벡터 요소들에 대한 Enumeration 객체를 반환

Enumeration 인터페이스

메소드	설 명
boolean hasMoreElements( )	요소가 있으면 true를 반환하고 없으면 false를 반환한다.
E nextElement( )	요소를 얻어낸다.

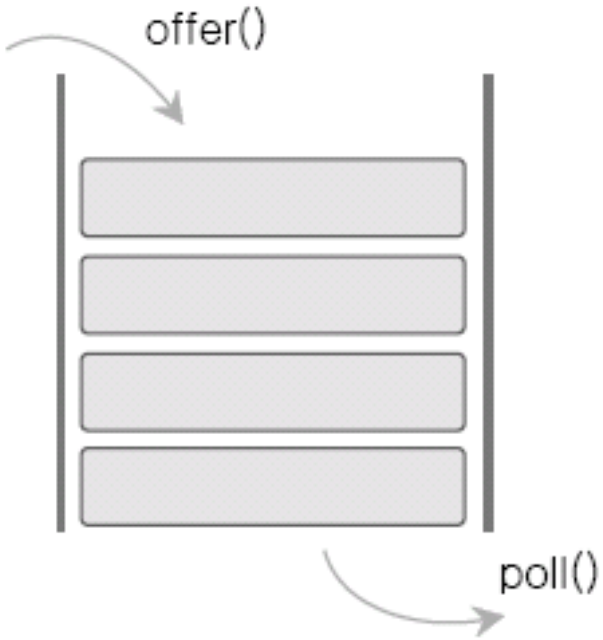
Stack 클래스


초기상태
①Push(G)
②Pop(data)
③Pop(data)
④Push(H)
⑤Push(I)
top
F
E
D
C
B
A
G
H
I

Stack 클래스의 주요 메소드

메소드	설 명
pop()	스택의 맨 위에서 객체를 제거하고 그 객체를 반환한다.
push(E item)	스택의 맨 위에 객체를 추가한다.
peek()	스택의 맨 위에 있는 객체를 반환한다. 이때 객체를 스택에서 제거하지는 않는다.
Boolean empty()	현재 스택이 비어 있는지를 확인한다. isEmpty도 동일한 기능을 한다.

Queue 인터페이스 LinkedList 클래스



Queue 인터페이스 LinkedList 클래스

메소드	설 명
boolean offer(E o)	큐에 객체를 넣는다.
E poll()	큐에서 데이터를 꺼내 온다. 만일 큐가 비어 있다면 null을 반환한다.
E peek()	큐의 맨 위에 있는 객체를 반환한다. 이때 객체를 큐에서 제거하지는 않는다. 그리고 큐가 비어 있다면 null을 반환한다.

Map 인터페이스 Hashtable 클래스

키(이름)	값(전화번호)
"홍길동"	"010-111-1111"
"성춘향"	"010-222-2222"
"한석봉"	"010-333-3333"

키에 대한 데이터 타입 ↓ 키에 대한 데이터 타입 ↓

```
Hashtable<String, String> ht=new Hashtable<String, String>;
```

↑ 값에 대한 데이터 타입 ↑ 값에 대한 데이터 타입

Map 인터페이스 Hashtable 클래스

형식

put(E key, E value)

```
ht.put("홍길동", "010-111-1111");
ht.put("성춘향", "010-222-2222");
ht.put("한석봉", "010-333-3333");
```

형식

E get(E key)

```
String phone=ht.get("성춘향");
```