List 컬렉션

▣ List 컬렉션

- List 컬렉션은 객체를 인덱스로 관리하기 때문에
  객체를 저장하면 인덱스가 부여되고 인덱스로 객체를 검색, 삭제할 수 있는 기능을 제공한다.

- List 컬렉션에서 공통적으로 사용가능한 List 인터페이스 메소드는 아래와 같다.

기능	메소드	설명
객체 추가	boolean add(E e)	주어진 객체를 맨 끝에 추가
	void add(int index, E element)	주어진 인덱스에 객체를 추가
	set(int index. E element)	주어진 인덱스의 객체를 새로운 객체로 바꿈
객체 검색	boolean contains(Object o)	주어진 객체가 저장되어 있는지 여부
	E get(int index)	주어진 인덱스에 저장된 객체를 리턴
	isEmpty( )	컬렉션이 비어 있는지 조사
	int size( )	저장되어 있는 전체 객체 수를 리턴
객체 삭제	void clear( )	저장된 모든 객체를 삭제
	E remove(int index)	주어진 인덱스에 저장된 객체를 삭제
	boolean remove(Object o)	주어진 객체를 삭제

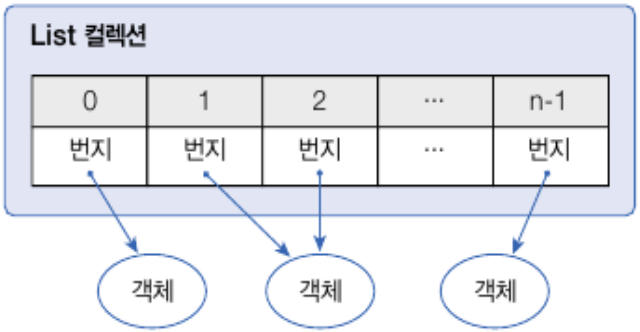
⦿ ArrayList

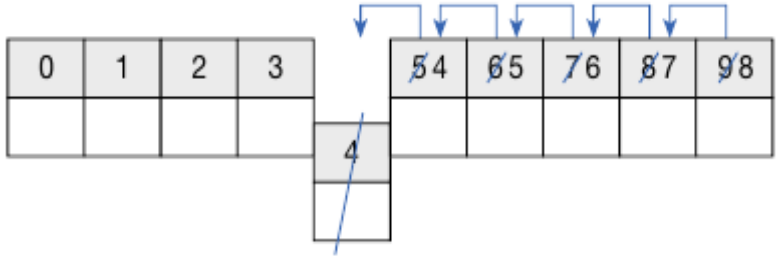
- ArrayList에 객체를 추가하면 내부 배열에 객체가 저장된다. 일반 배열과 차이점은 ArrayList는 제한 없이 객체를 추가할 수 있다는 것이다.

- List 컬렉션은 객체 자체를 저장하는 것이 아니라 객체의 번지를 저장한다.
  또한 동일한 객체를 중복 저장할 수 있는데, 이 경우에는 동일한 번지가 저장된다. null 또한 저장이 가능하다.

- 객체를 추가하면 인덱스 0번부터 차례대로 저장된다. 특정 인덱스의 객체를 제거하면 모두 앞으로 1씩 당겨진다.
  특정 인덱스에 객체를 삽입하면 마지막 인덱스까지 모두 1식 밀려난다.

- 따라서 빈번한 객체 삭제와 삽입이 일어나는 곳에서는 ArrayList를 사용하는 것은 바람직하지 않다.
  대신 이런 경우라면 LinkedList를 사용하는 것이 좋다.






```
List<E> list = new ArrayList<E>();    //E에 지정된 타입의 객체만 저장
List<E> list = new ArrayList<>();     //E에 지정된 타입의 객체만 저장
List list = new ArrayList();          //모든 타입의 객체를 저장
```

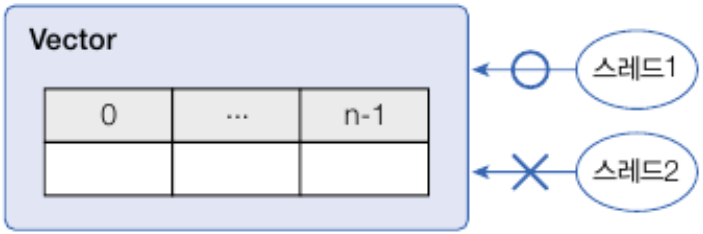
※ 컬렉션은 제네릭(generics) 기법으로 구현됨

⦿ Vector

- Vector는 ArrayList와 동일한 내부 구조를 가지고 있다.

- 차이점은 Vector는 동기화된(synchronized) 메소드로 구성되어 있기 때문에
  멀티 스레드가 동시에 Vector() 메소드를 실행할 수 없다는 것이다.

- 그렇기 때문에 멀티 스레드 환경에서는 안전하게 객체를 추가 또는 삭제할 수 있다.



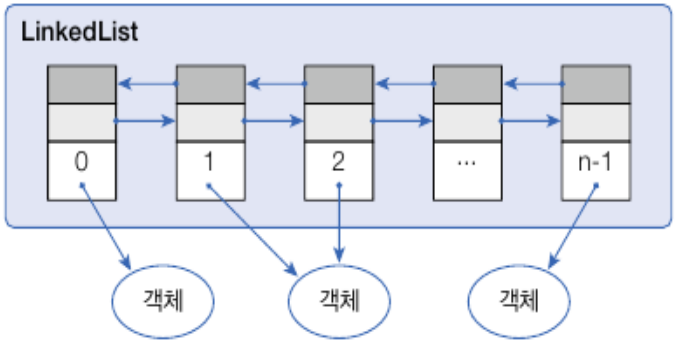
- **동기화 방식**은 한 자원에 대해 동시에 접근하는 것을 제한하는 방식이다.
- 다시말해서, **순서를 지키겠다**는 말이다.
- 비동기화 방식은 한 자원에 대해 동시에 접근이 가능하다

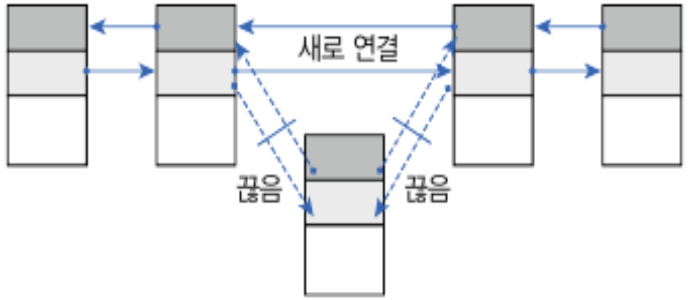
```
List<E> list = new Vector<E>();    //E에 지정된 타입의 객체만 저장
List<E> list = new Vector<>();     //E에 지정된 타입의 객체만 저장
List list = new Vector();          //모든 타입의 객체를 저장
```

⦿ LinkedList

- LinkedList는 ArrayList와 사용 방법은 동일하지만 내부 구조는 완전히 다르다.
  ArrayList는 내부 배열에 객체를 저장하지만, LinkedList는 인접 객체를 체인처럼 연결해서 관리한다.

- LinkedList는 특정 위치에서 객체를 삽입하거나 삭제하면 바로 앞뒤 링크만 변경하면 되므로
  빈번한 객체 삭제와 삽입이 일어나는 곳에서는 ArrayList보다 좋은 성능을 발휘한다.






```
List<E> list = new LinkedList<E>();    //E에 지정된 타입의 객체만 저장
List<E> list = new LinkedList<>();     //E에 지정된 타입의 객체만 저장
List list = new LinkedList();          //모든 타입의 객체를 저장
```