문자열 클래스

▣ 문자열 클래스

- 자바에서 문자열과 관련된 주요 클래스는 다음과 같다.

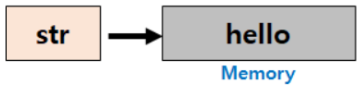
클래스	설명
String	문자열을 저장하고 조작할 때 사용
StringBuilder / StringBuffer	효율적인 문자열 조작 기능이 필요할 때 사용
StringTokenizer	구분자로 연결된 문자열을 분리할 때 사용

String 객체는 내부 데이터를 수정할 수 없으므로
"ABC" 에 "DEF"가 추가된 **"ABCDEF" 라는 새로운 String 객체가 생성**된다.

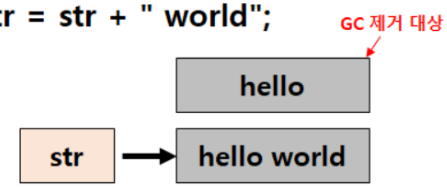
data 변수는 **새로 생성된 String 객체를 참조**하게 된다.

+ 연산자를 사용한 만큼 **String 객체 수가 늘어나기 때문에 성능을 느리게 할 수 있다.**
따라서 문자열 변경 작업이 많을 경우,
java.lang 패키지의 StringBuffer 또는 StringBuilder 클래스를 사용하는 것이 좋다.

1. String str = new String("hello");



2. str = str + " world";



이 두 클래스는 내부 버퍼(buffer : 데이터를 임시로 저장하는 메모리)에 문자열을 저장해 두고,
그 안에서 추가, 수정, 삭제 작업을 할 수 있도록 설계되어 있다.
즉, String 처럼 새로운 객체를 만들지 않고 문자열을 조작할 수 있다.

StringBuffer와 StringBuilder 의 사용 방법은 동일하다.
(동기화 O) (동기화 X)

◎ String 클래스

- String 클래스는 문자열을 저장하고 조작할 때 사용한다.
- 문자열 리터럴은 자동으로 String 객체로 생성되지만,
  String 클래스의 다양한 생성자를 이용해서 직접 객체를 생성할 수도 있다.

- 한글 1자를 UTF-8로 인코딩하면 3byte가 되고, EUC-KR로 인코딩하면 2byte가 된다.
  따라서 인코딩할 때 사용한 문자셋으로 디코딩을 해야만 한글이 올바르게 복원될 수 있다.

- 프로그램을 개발하다 보면 byte 배열을 문자열로 변환하는 경우가 종종 있다.
  이때는 String 생성자 중에서 두 가지를 이용해 String 객체로 생성할 수 있다.

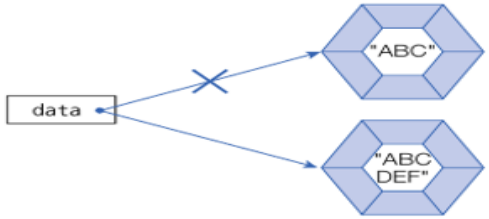
```
//기본 문자셋으로 byte 배열을 디코딩해서 String 객체로 생성
String str = new String(byte[] bytes);
```

```
//특정 문자셋으로 byte 배열을 디코딩해서 String 객체로 생성
String str = new String(byte[] bytes, String charsetName);
```

◎ StringBuilder 클래스

- String은 내부 문자열을 수정할 수 없다. 즉, 문자열의 + 연산은 새로운 String 객체가 생성되고 이전 객체는 계속 버려지는 것이기 때문에 효율성이 좋다고는 볼 수 없다.

```
String data = "ABC";
data += "DEF";
```



- 잦은 문자열 변경 작업을 해야 한다면 String보다는 StringBuilder를 사용하는 것이 좋다.
- StringBuilder는 내부 버퍼(데이터를 저장하는 메모리)에 문자열을 저장해두고 그 안에서 추가, 수정, 삭제 작업을 하도록 설계되어 있다. 따라서 String처럼 새로운 객체를 만들지 않고도 문자열을 조작할 수 있다.

- StringBuilder가 제공하는 메소드중 toString()을 제외한 다른 메소드는 StringBuilder를 다시 리턴하기 때문에 연이어서 다른 메소드를 호출할 수 있는 메소드 체이닝(chaining) 패턴을 사용할 수 있다.

리턴 타입	메소드(매개변수)	설명
StringBuilder	append(기본값 | 문자열)	문자열을 끝에 추가
StringBuilder	insert(위치, 기본값 | 문자열)	문자열을 지정 위치에 추가
StringBuilder	delete(시작 위치, 끝 위치)	문자열 일부를 삭제
StringBuilder	replace(시작 위치, 끝 위치, 문자열)	문자열 일부를 대체
String	toString( )	완성된 문자열을 리턴

◎ StringTokenizer 클래스

- 문자열이 구분자로 연결되어 있을 경우, 구분자를 기준으로 문자열을 분리하려면
  String 클래스의 split() 메소드를 이용하거나, java.util 패키지의 StringTokenizer 클래스를 이용할 수 있다.
- split() 메소드는 정규 표현식으로 구분하고, StringTokenizer는 문자로 구분한다는 차이점이 있다.

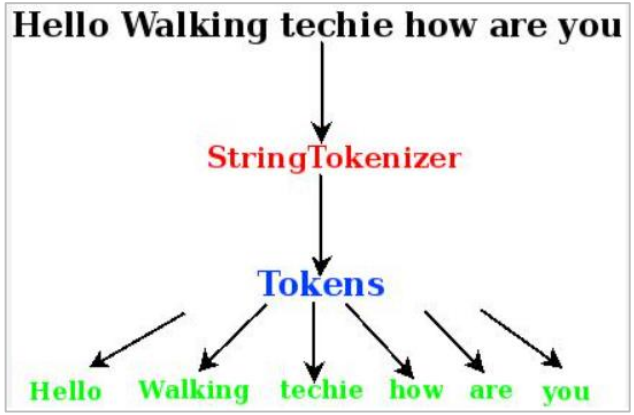
<유형1>
문자열에서 &, 쉼표(,), 하이픈(-)으로 구분된 사람 이름을 뽑아낼 경우에는 정규 표현식으로 분리하는 split() 메소드를 사용해야 한다.

```
String data = "홍길동&이수홍,박연수,김자바-최명호";
String[] names = data.split("&|,|-");
```

<유형2>
첫 번째 매개값으로 전체 문자열을 주고, 두 번째 매개값으로 구분자를 주면 된다.
만약 구분자를 생략하면 공백이 기본 구분자가 된다.

```
String data = "홍길동/이수홍/박연수";
StringTokenizer st = new StringTokenizer(data, "/");
```

리턴 타입	메소드(매개변수)	설명
int	countTokens( )	분리할 수 있는 문자열의 총 수
boolean	hasMoreTokens( )	남아 있는 문자열이 있는지 여부
String	nextToken( )	문자열을 하나씩 가져옴



※ nextToken() 메소드는 분리된 문자열을 하나씩 가져오고, 더 이상 가져올 문자열이 없다면 예외를 발생시킨다.
그래서 nextToken()을 사용하기 전에 hasMoreTokens() 메소드로 가져올 문자열이 있는지 먼저 조사하는 것이 좋다.