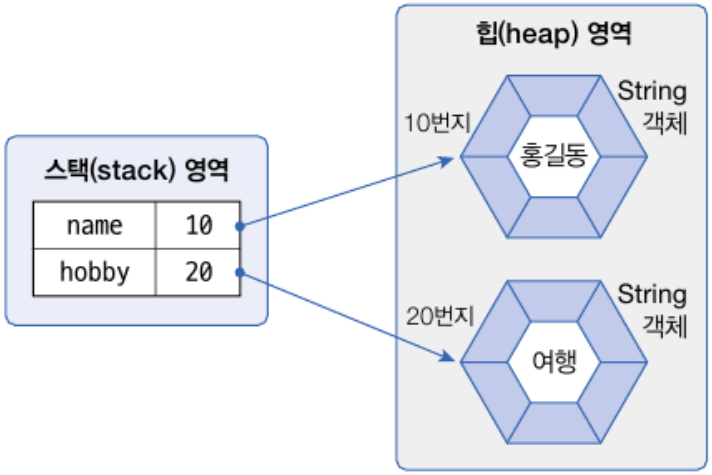
문자열(String) 타입

▣ 문자열(String) 타입

- 자바의 문자열은 String 객체로 생성됩니다.

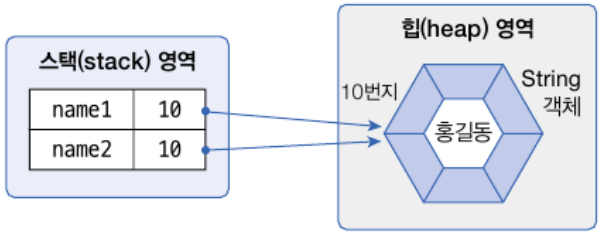
```
String name;              //String 타입 변수 name 선언
name = "홍길동";          //name 변수에 문자열 대입
String hobby = "여행";    //String 타입 변수 hobby를 선언하고 문자열 대입
```



● 문자열 비교

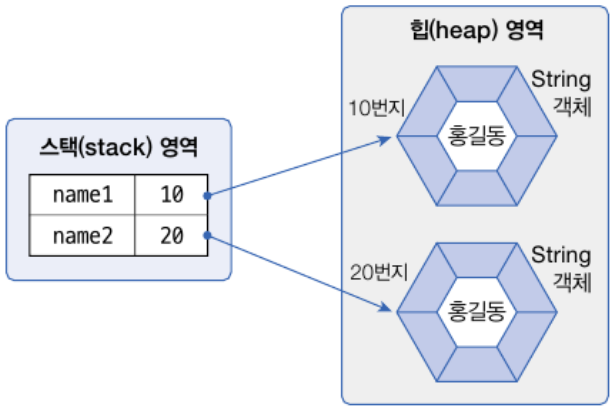
- 자바는 문자열 리터럴이 동일하다면 String 객체를 공유하도록 설계되어 있다.

```
String name1 = "홍길동";
String name2 = "홍길동";
```



- String 변수에 문자열 리터럴을 대입하는 것이 일반적이지만, new 연산자로 직접 String 객체를 생성하고 대입할 수도 있다. new 연산자는 새로운 객체를 만드는 연산자로 객체 생성 연산자라고 한다.

```
String name1 = new String("홍길동");
String name2 = new String("홍길동");
```



- String 객체를 문자열 리터럴로 생성하느냐, new 연산자로 생성하느냐에 따라 비교 연산자의 결과가 달라질 수 있다.

```
String name1 = "홍길동";
String name2 = "홍길동";
String name3 = new String("홍길동");
```

```
name1 == name2   //결과: true
name1 == name3   //결과: false
```

- 동일한 String 객체든 다른 String 객체든 상관없이 내부 문자열만을 비교할 경우에는 String 객체의 equals() 메소드를 사용한다.

```
boolean result = str1.equals(str2);     //문자열이 같은지 검사(대소문자 구분)
                 원본 문자열     비교 문자열

boolean result = !str1.equals(str2);    //문자열이 다른지 검사
```

- String 변수에 빈 문자열(“”)을 대입할 수도 있다. 빈 문자열도 String 객체로 생성되기 때문에 변수가 빈 문자열을 참조하는지 조사하려면 equals() 메소드를 사용해야 한다.

```
public class EmptyStringExample {
  public static void main(String[] args) {
    String hobby = "";
    if(hobby.equals("")) {
       System.out.println("hobby 변수가 참조는 String 객체는 빈 문자열");
    }
  }
}
```

실행 결과

```
hobby 변수가 참조는 String 객체는 빈 문자열
```

● 문자열 추출

- 문자열에서 특정 위치의 문자를 얻고 싶다면 charAt( ) 메소드를 이용할 수 있다. chatAt( ) 메소드는 매개값으로 주어진 인덱스의 문자를 리턴한다.

```
String subject = "자바 프로그래밍";
char charValue = subject.charAt(3);
```

자	바		프	로	그	래	밍
0	1	2	3	4	5	6	7

● 문자열 길이

- 문자열에서 문자의 개수를 얻고 싶다면 length( ) 메소드를 사용한다.

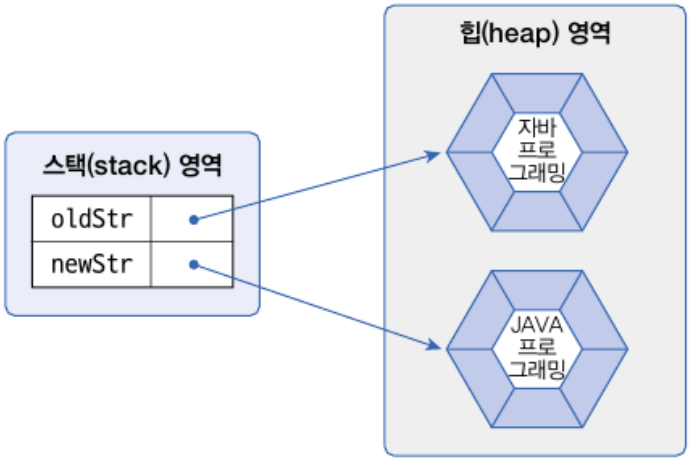
```
String subject = "자바 프로그래밍";
int length = subject.length();
```

= 총 14 자리

● 문자열 대체

- 문자열에서 특정 문자열을 다른 문자열로 대체하고 싶다면 replace() 메소드를 이용한다.
  replace() 메소드는 기존 문자열은 그대로 두고, 대체한 새로운 문자열을 리턴한다. (문자열은 불변)

```
String oldStr = "자바 프로그래밍";
String newStr = oldStr.replace("자바", "JAVA");
```

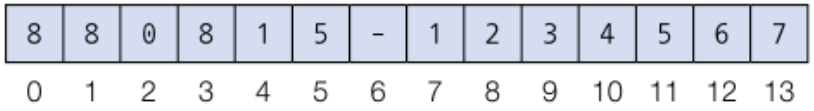


● 문자열 잘라내기

- 문자열에서 특정 위치의 문자열을 잘라내어 가져오고 싶다면 substring() 메소드를 사용한다.

메소드	설명
substring(int beginIndex)	beginIndex에서 끝까지 잘라내기
substring(int beginIndex, int endIndex)	beginIndex에서 endIndex 앞까지 잘라내기

```
String ssn = "880815-1234567";
String firstNum = ssn.substring(0, 6);
String secondNum = ssn.substring(7);
```



● 문자열 찾기

- 문자열에서 특정 위치의 문자열의 위치를 찾고자 할 때에는 indexOf() 메소드를 사용한다.
  indexOf() 메소드는 주어진 문자열이 시작되는 인덱스를 리턴한다. “

```
String subject = "자바 프로그래밍";
int index = subject.indexOf("프로그래밍");
```

- 만약 주어진 문자열이 포함되어 있지 않으면 indexOf() 메소드는 -1을 리턴한다.

```
int index = subject.indexOf("프로그래밍");
if(index == -1) {
  //포함되어 있지 않은 경우
} else {
  //포함되어 있는 경우
}
```

- 만약 주어진 문자열이 단순히 포함되어 있는지만 조사하고 싶다면 contains() 메소드를 사용하면 편리하다.
  원하는 문자열이 포함되어 있으면 true, 없으면 false를 리턴한다.

```
boolean result = subject.contains("프로그래밍");
```

● 문자열 분리

- 문자열이 구분자를 사용하여 여러 개의 문자열로 구성되어 있을 경우, 이를 따로 분리해서 얻고 싶다면 split() 메소드를 사용한다.

```
String board = "번호,제목,내용,글쓴이";
String[] arr = board.split(",");
```