String Class

String Class

- 문자열을 표현하기 위해서 Java에서 표준으로 정의한 Class
- 문자열을 사용할 경우에 자동으로 String이라는 class의 instance화 되어 처리된다.

- "문자열"을 표현하면 Java는 문자열을 저장하는 하나의 String형 Instance를 생성하고 Instance의 주소값을 반환해준다.

- 문자열의 표현은 전통적으로 컴퓨터공학에서 골칫거리다.

```
String str = "ABC";
```

String	str	=	"ABC";
참조변수 Str은 String형 객체의 주소값을 받아야 한다.	참조변수		문자열을 선언하면 String Class의 Instance화 되서 메모리에 저장되고, Instance의 메모리 주소값을 반환 해준다.

주의 : Stirng형 Instance에 저장된 Data(문자열)은 바꿀 수가 없다 (상수화된 인스턴스)!

- 문자열은 프로그램에서 매우 빈번하게 등장한다. 문자열이 생성될 때마다 Instance를 만드는 것은 비효율적이다.
- 동일한 문자열로 String형 Instance가 생성될 경우에는 새로운 Instance를 생성하지 않고, 이전에 생성된 Instance의 메모리 주소값만 반환

```
String str1 = "ABC";
String str2 = "ABC";

if(str1 == str2) // 참조값 비교!
     System.out.println("동일 객체 참조");
```

str1이 가리키는, "ABC"를 담고 있는 instance의 주소값 참조

결국, str1, str2는 동일한 instance를 참조하게 된다. 이 때문에 저장된 문자열의 변경을 허용하지 않는다!

Code

```
String str1 = "String Instance";
String str2 = "My String";

System.out.println("Hello Java");
```

System.out.println("My Coffee");

i. 문자열이 instance화 되고 주소값이 반환됨

ii. Println Method에 문자열이 전달되는 것이 아니고 문자열로 인해 생성된 instance 주소값이 전달

(println Method의 parameter는 String형 참조변수이다.)

(println Method는 숫자도 받을 수 있으므로 Overloadig된 Method이다!)

```
class StringInstance {
     public static void main(String[] args) {
          java.lang.String str = "My name is Android";
          int strLen1 = str.length();

          System.out.println("길이 1 : "+strLen1);

          int strLen2 = "한글의 길이는 어떻게?".length();
```

(.) 접근연산자의 왼편에는 instance 참조값이 위치해야 하는데, 문자열이 표현되자마자 String형 instance가 생성이 되고 주소값을 반환해 주기 때문에 String Class에 정의되있던 Method 사용 가능!

```
          System.out.println("길이 2 : "+strLen2);
     }
}
```
