

컴퓨터 프로그래밍

String class

# String class

- Java의 문자열은 String class로 표현

```
String str1 = new String();
```

```
String str2 = new String("abc");
```

```
String str3 = "String Instance";
```

```
String str4 = "My String";
```

```
class StringInstance {
```

```
    public static void main(String[] args) {
```

```
        String str = "Hello";
```

```
        int strLen1 = str.length();
```

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

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

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

```
    }
```

```
}
```

# String class Methods

- <http://docs.oracle.com/javase/7/docs/api/>

The screenshot shows a web browser window displaying the Java Platform SE 7 API documentation for the `String` class. The browser's address bar shows the URL `http://docs.oracle.com/javase/7/docs/api/`. The left sidebar contains a navigation pane with a tree view of Java packages and classes. The main content area is titled "Method Summary" and lists various methods of the `String` class.

Modifier and Type	Method and Description
char	<b>charAt</b> (int index) Returns the char value at the specified index.
int	<b>codePointAt</b> (int index) Returns the character (Unicode code point) at the specified index.
int	<b>codePointBefore</b> (int index) Returns the character (Unicode code point) before the specified index.
int	<b>codePointCount</b> (int beginIndex, int endIndex) Returns the number of Unicode code points in the specified text range of this <code>String</code> .
int	<b>compareTo</b> ( <code>String</code> anotherString) Compares two strings lexicographically.
int	<b>compareToIgnoreCase</b> ( <code>String</code> str) Compares two strings lexicographically, ignoring case differences.
<b>String</b>	<b>concat</b> ( <code>String</code> str) Concatenates the specified string to the end of this string.
boolean	<b>contains</b> ( <code>CharSequence</code> s) Returns true if and only if this string contains the specified sequence of char values.
boolean	<b>contentEquals</b> ( <code>CharSequence</code> cs) Compares this string to the specified <code>CharSequence</code> .
boolean	<b>contentEquals</b> ( <code>StringBuffer</code> sb) Compares this string to the specified <code>StringBuffer</code> .
static <b>String</b>	<b>copyValueOf</b> (char[] data) Returns a <code>String</code> that represents the character sequence in the array specified.
static <b>String</b>	<b>copyValueOf</b> (char[] data, int offset, int count) Returns a <code>String</code> that represents the character sequence in the array specified.
boolean	<b>endsWith</b> ( <code>String</code> suffix) Tests if this string ends with the specified suffix.
boolean	<b>equals</b> ( <code>Object</code> anObject) Compares this string to the specified object.

# Useful Methods

- `public int length()`
  - `public boolean equals(Object anObject)`
  - `public int compareTo(String anotherString)`
  - `public char charAt(int index)`
  - `public String substring(int beginIndex, endIndex)`
  - `public boolean contains(CharSequence s)`
  - `public int indexOf(int ch)`
  - ...
- 
- API document에서 각 메소드의 기능을 확인

# Useful Methods

```
class StringMethod {  
    public static void main(String[] args) {  
        String str1 = "Smart";  
        String str2 = " and ";  
        String str3 = "Simple";  
        String str4 = str1.concat(str2).concat(str3);  
  
        System.out.println(str4);  
        System.out.println("문자열 길이 : "+str4.length());  
  
        if (str1.compareTo(str3) < 0)  
            System.out.println("str1이 앞선다");  
        else  
            System.out.println("str3이 앞선다");  
    }  
}
```

Smart and Simple 문자열 길이 : 16 str3이 앞선다
--

# String +

- Java compiler는 + 연산을 적절한 형태의 method 호출로 변환

```
public static void main(String[] args) {
```

```
    String str1 = "Lemon" + "ade";
```

```
    String str2 = "Lemon" + 'A';
```

```
    String str3 = "Lemon" + 3;
```

```
    String str4 = 1 + "Lemon" + 2;
```

```
    str4 += '!';
```

```
    System.out.println(str1);
```

```
    System.out.println(str2);
```

```
    System.out.println(str3);
```

```
    System.out.println(str4);
```

```
}
```

→

```
"Lemon".concat("ade");
```

```
"Lemon".concat(String.valueOf('A'));
```

```
"Lemon".concat(String.valueOf(3));
```

valueOf 메소드의 오버로딩

- public static String valueOf(boolean b)
- public static String valueOf(char c)
- public static String valueOf(int i)
- public static String valueOf(long l)
- public static String valueOf(float f)
- public static String valueOf(double d)

## 시스 템

- 임의의 문자열을 입력받아 shift cipher로 암호화하여 출력
  - Shift cipher: 암호키 값(정수  $k$ )을 정한 후 각 문자를  $k$  문자 뒤의 문자로 변경하여 암호화하는 방법
  - 문자 알파벳은 순환 배열되어 있다고 가정. 즉 영문자  $z$  다음 문자는  $a$
  - 예) 암호키  $k=3$ 일 때 문자열 "zebra"를 입력하면 "cheud"로 출력
- 임의의 문자열을 입력받아 문자열에 포함된 모든 소문자  $a$ 를 대문자  $A$ 로 교체하여 출력