

Java String class

API documentation is available at:

java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html

String

- String is the name of a predefined class.
 - actual name is **java.lang.String**
- The String class is a *little* special
 - support for literals, like "Hello Class"
 - support for operators:
 - + concatenation
 - += assignment and concatenation, as in
s += " Append to end of s";

Ask not what you can do with a `String`,
ask what a `String` can do for you.

- A `String` object can:
 - tell you it's length
 - tell you if it contains a substring (and where).
 - tell you if it matches a regular expression.
 - tell you what is at any position
 - create a copy that is all lowercase or uppercase
 - compare itself to another `String`

This page intentionally left blank

- If you are reading this outside of class, please proceed to the Java API docs for the class

`java.lang.String`

- memorize all the methods and constructors, there may be a test on this...

Some Background

- a String object is *immutable*
 - *you can't change it at all. Really!*
 - This sounds silly, but there are good reasons for this and it's not really a hassle.
- Using `==` to compare String objects is rarely what you want
 - only true if they are the same object
 - we usually want to know if two strings hold the same sequence of characters. Use **`String.equals()`**

Literals

- You can initialize a String with a literal:

```
String s = "Hello There";
```

- This is basically the same as:

```
String s = new String("Hello There");
```

- You can call methods on a literal!

```
int len = "Hello".length();
```

```
if ("Fred".equals(s))
```

```
    System.out.printf("Fred it is\n");
```

Converting Strings to primitive types

```
int x = Integer.parseInt("12");
```

```
double d = Double.parseDouble("3.141593");
```

```
boolean b = Boolean.parseBoolean("false");
```

These are all static methods!

Immutable – so what!

```
String s = new String();  
for (int i=0;i<10;i++) {  
    s = s + i;  
}  
System.out.println(s);
```



0123456789

A sample class/program

- We want a class named Square that has a size.
- We want to be able to print a Square object
 - it should report it's size, that's all.
 - we need to write a **toString()** method.
- We want to be able to tell the Square to draw itself.
 - we will just have it print characters out using `System.out.print()`.

Class Square (part 1).

```
/**
 * This is a javadoc comment for the class
 * Square. Square can draw itself using
 * nothing more than printable characters.
 */
class Square {
    int size;
```

Class Square (part deux).

```
/**
 * This is a javadoc comment for the
 * constructor.
 * Square needs a size, please give it
 * an integer size.
 */
Square(int x) {
    size = x;
}
```

Class Square (part tres).

```
/**
 * This is a javadoc comment for the
 * method toString().
 * as a String, a Square just reports
 * it's size.
 */
public String toString() {
    return("Square of size " + size);
}
```

Class Square (part quattro).

```
/**
 * Draw will generate a character based
 * representation of the Square by printing
 * characters to System.out
 */
public void Draw() {
    for (int i=0;i<size;i++) {
        for (int j=0;j<size;j++) {
            System.out.print("*");
        }
        System.out.println();
    }
}
```

Class Square (part ۵ = پنج).

```
/**
 * How about some test code? Sounds good.
 */
public static void main(String[] args) {
    Square s = new Square(Integer.parseInt(args[0]));
    System.out.println(s);
    s.Draw();
}
```

Class Square (part exi).

```
> javac Square.java
```

```
> java Square
```

```
Exception in thread "main"
```

```
java.lang.ArrayIndexOutOfBoundsException: 0 at
```

```
Square.main(Square.java:22)
```

```
> java Square 5
```

```
Square of size 5
```

```
*****
```

```
*****
```

```
*****
```

```
*****
```

```
*****
```