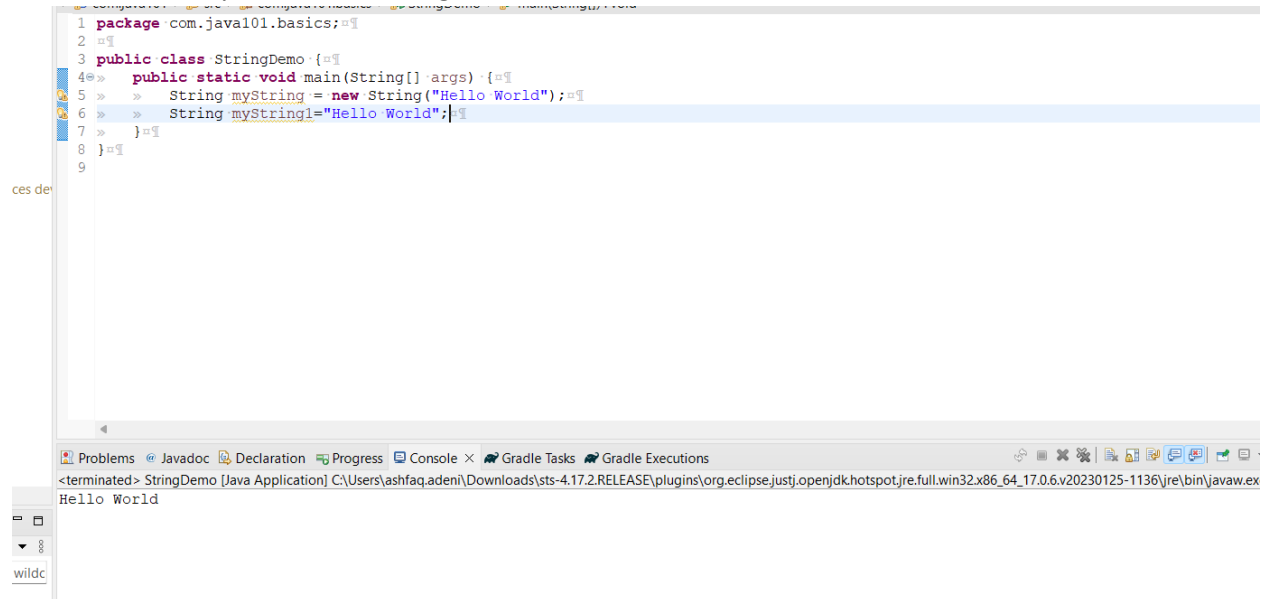


# Strings

There are two ways to create a string



```
1 package com.java101.basics;
2
3 public class StringDemo {
4     public static void main(String[] args) {
5         String myString = new String("Hello World");
6         String myString1 = "Hello World";
7     }
8 }
9
```

ces de

Problems Javadoc Declaration Progress Console x Gradle Tasks Gradle Executions

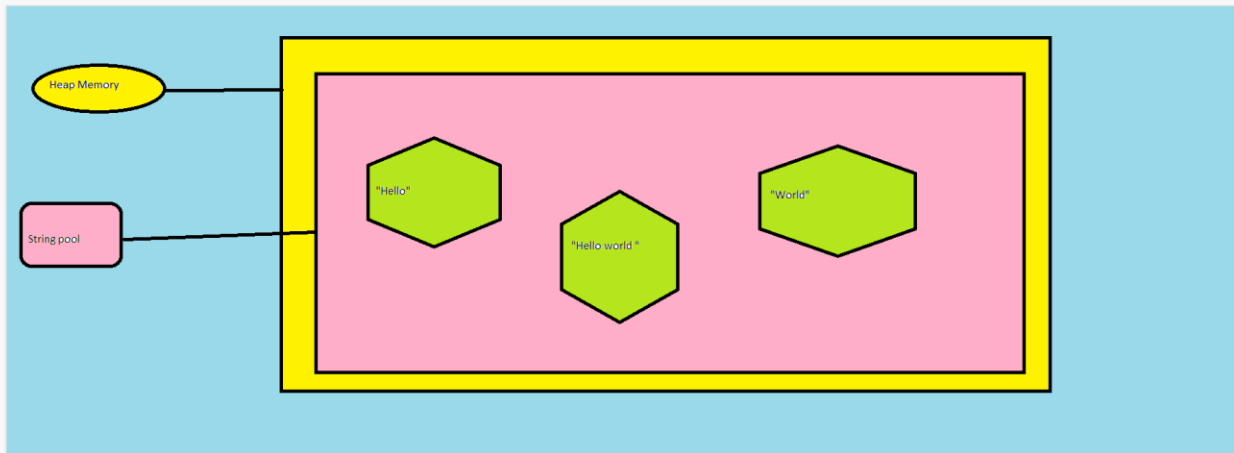
<terminated> StringDemo [Java Application] C:\Users\ashfaq.adeni\Downloads\sts-4.17.2.RELEASE\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.6.v20230125-1136\jre\bin\javaw.exe  
Hello World

wildc

One is using a new Keyword → a new string object is explicitly created using the **new** keyword and its constructor, which creates a new object on the heap memory. This means that a new object is created even if there is an existing object with the same value in memory.

Another one is using String Literal:

When the string literal is encountered by the compiler, it creates a new string object in a special memory area called the "string pool". If a string with the same value already exists in the pool, a reference to the existing object is returned instead of creating a new object. This can be more efficient in terms of memory usage, especially if the same string literal is used multiple times in the code.



When java compiler encounters a new string with different name and value = **“Hello World”**, then it won't create a new String which occupies space in heap memory it just refers to an existing string with value **“Hello World”** in String pool