Strings

|  |  |  |
| --- | --- | --- |
| Data Type | Memory Allocated by Compiler | Size |
| boolean flag | |  | | --- | |  | | 1 byte |
| char symbol | |  |  | | --- | --- | |  |  | | 2 bytes |
| int count | |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | | 4 bytes |
| double price | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  | | 8 bytes |
| String title | ?? | ? |

A String value can have any length! And it can change as the program runs!

|  |  |  |
| --- | --- | --- |
| String value | Length | Bytes |
| “” | 0 | 0 |
| “FIU” | 3 | 6 |
| “Florida International University” | 32 | 64 |

So the Java compiler can’t know how much memory to allocate for a String variable.

Instead, the String variable does not store the String value. When the length of the String becomes known at run time, sufficient storage space is allocated somewhere in the memory, and the String variable in the program is updated to record the location of the allocated memory – a reference**.**

Since a reference is a fixed size memory address (it is always 8 bytes), the compiler knows how big to make the string variable in the program – but it doesn’t store the String value, just a reference to it. String variables actually have the same size as double variables – 8 bytes!

When a String variable has not been assigned, its value is null.

String title;

|  |
| --- |
| null |

After assignment, its value is a reference to the actual character string somewhere else in memory!

title = “Florida International University”;

|  |
| --- |
|  |

|  |
| --- |
| F l o r i d a I n t e r n a t i o n a l U n I v e r s i t y |