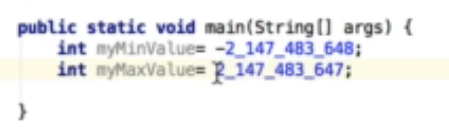
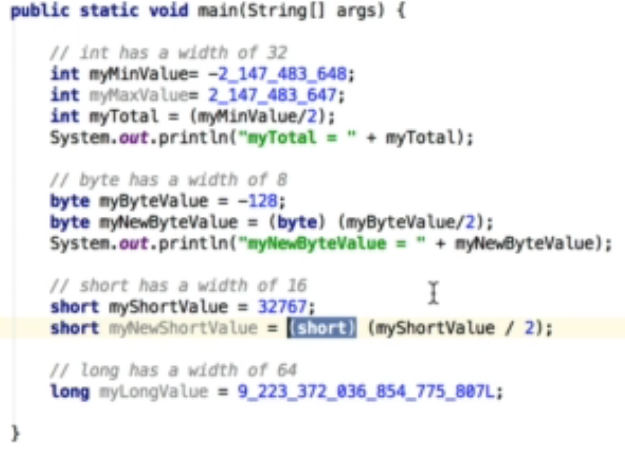
**Java – Refreshing memory**

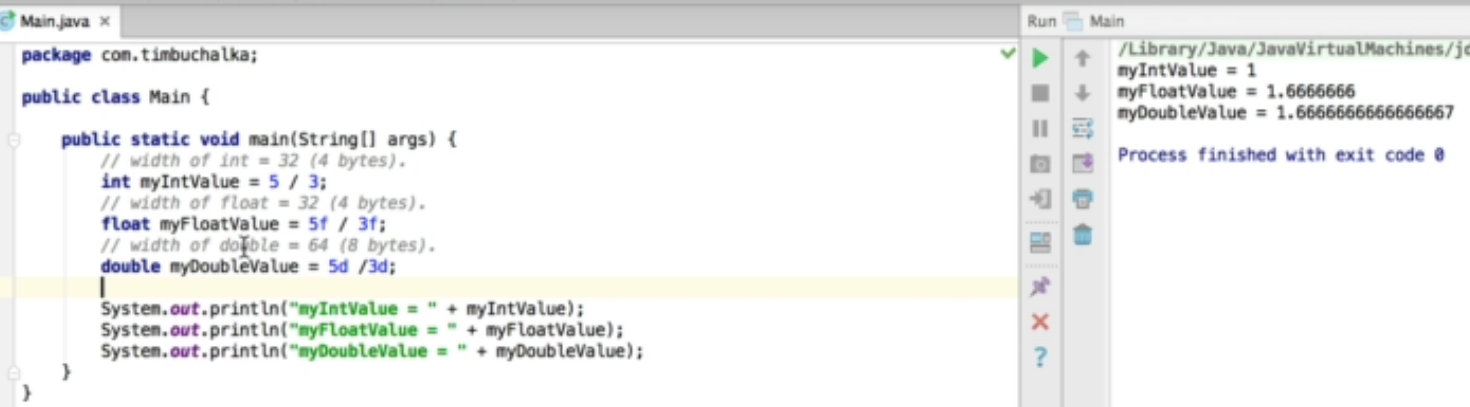
**Literal numbers can be underscored**



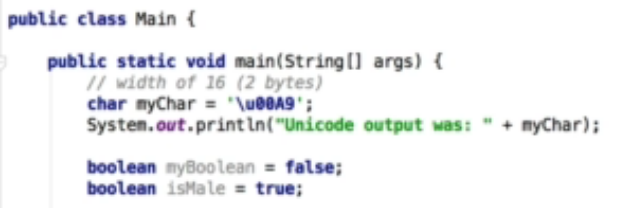
**Basic Types**



Double is fast in pretty fast in modern computers



Char takes 2 bytes

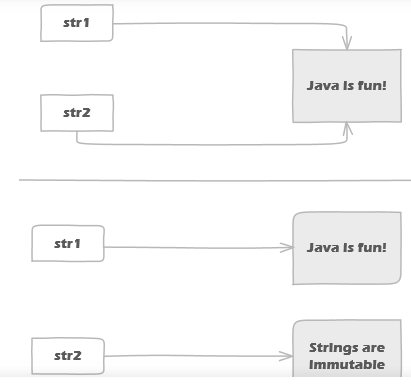


* **Strings are immutable in Java**

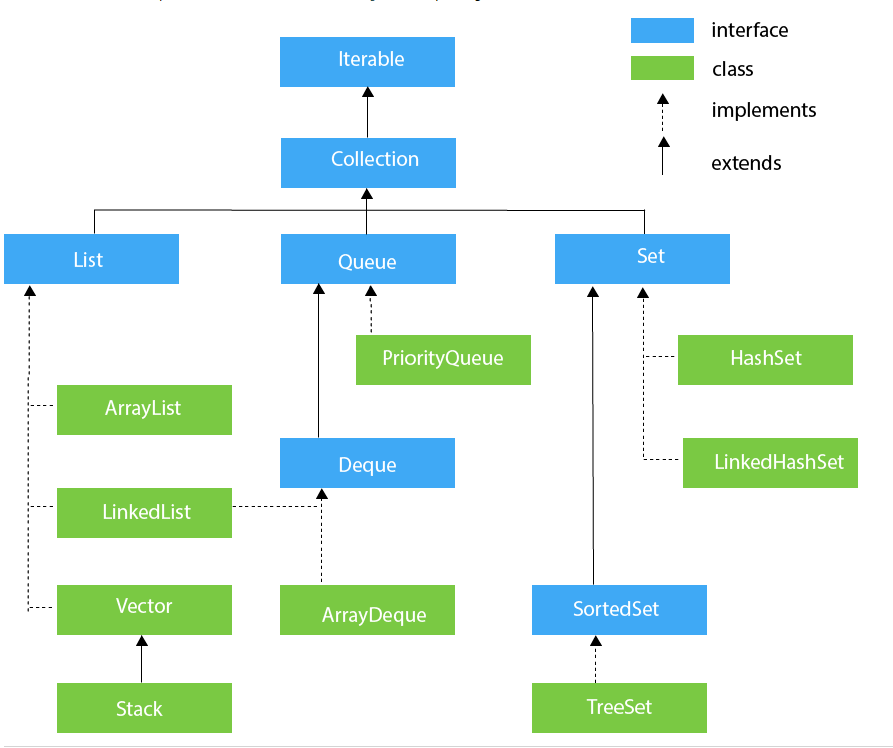
String str1 = "Java is fun!";  
String str2 = str1;

str2 = "Strings are immutable"; // str2 is no longer points to "Java is fun!"

System.out.println(str1); // Java is fun!  
System.out.println(str2); // Strings are immutable



* **List and Array List**



**import** java.util.ArrayList;  
**import** java.util.Iterator;  
  
**public class** Test {  
  
 **public static void** main(String args[]) {  
  
 ArrayList<String> identities = **new** ArrayList<String>();  
 identities.add(**"sun"**);  
 identities.add(**"sun1"**);  
  
 **for** ( String x :identities ) {  
 System.***out***.println(x);  
 }  
  
 Iterator<String> iterator = identities.iterator();  
  
 **while** (iterator.hasNext()) {  
 System.***out***.println(iterator.next());  
 }  
 }  
}

* **Autoboxing and unboxing**

Using Integer instead of int – wherever we need to use as reference

**public class** Test {  
  
 **public static void** main(String args[]) {  
  
 *// AutoBoxing* Integer x = 20; *// or -- both same* Integer y = Integer.*valueOf*(30);  
  
 *// Unboxing* **int** i = x.intValue();  
 System.***out***.println(x+y+i);  
 }  
 }

* **LinkedList**

**Array list** – any elements added in middle . The whole thing has to be shifted down because it is not added in consecutive address

**Linked List** – Subsequent address is used to add the elements. Any addition it just relinks. This need to shift anything down. Due to this reason it is pretty fast. In case an element is removed – Java automatically performs garbage collection

Java has implemented doubly linked list. Hence it is easy to traverse back

**Interfaces**

* Can implement multiple interfaces but can inherit only one

**Inner Classes**

* **Non Static Class**

**Generics**