Wrapper classes are classes that wrap around primitive data types. Wrapper classes allow primitive data types to be more versatile, for example a wrapper class has built in methods that can be used to convert data types. The Integer wrapper class has a method parseInt that allows for integer values to be parsed from strings.

Wrapper classes are useful for ArrayLists because ArrayLists can only be a collection of objects, and not primitive data types. This means that to have an ArrayList of doubles instead of using the primitive data type you need to use the wrapper class to create the ArrayList. ArrayLists are very powerful tools because they have additional functionality compared to arrays. ArrayLists have a method called add() that allows you to add elements to the ArrayList. This means that ArrayLists are resizable, which arrays are not.

Autoboxing is the process of automatically converting a primitive data type to its wrapper class, this is done by default in Java. This means automatically converting int to Integer. Unboxing is the reverse process where the Java compiler converts the wrapper class into the primitive datatype. Please see the screenshot below for an example of each.

A screen shot of a computer program

Description automatically generated

As you can see from the code the compiler automatically converts the primitive data type to the wrapper class and the other way. This is example has both implicit and explicit autoboxing and implicit unboxing. You see you can explicitly tell the compiler to autobox or unbox by using the methods contained in the wrapper class, or you can let the compiler do it implicitly by assigning a primitive a value of the wrapper object type, or by assigning an object the type of the primitive.

References

Eck, D. J. (2019). *Introduction to programming using Java*, version 8.1. Hobart and William Smith Colleges. <http://math.hws.edu/javanotes/>.