**Also see official documentation:** [**https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html**](https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html)

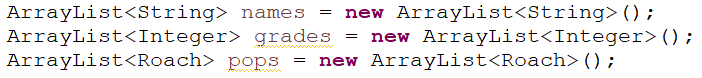
**An ArrayList -** is a special object type in the Java utility library that makes working with arrays much easier. ArrayLists grow and shrink dynamically as needed and come with many useful helper methods.

**Creating an ArrayList**

First, you must import the ArrayList class at the very top of your program, like you would a Scanner.

import java.util.ArrayList;

Next, create an arraylist object assigning it a name and type. There is no need to set a size.



etc

The data type of what you want to put in the ArrayList goes in the <>. Note that you have to provide the proper class name (Integer instead of int).

**ArrayList Methods**

Here are some of the methods you use to interact with your new ArrayList.

names.size(); //returns the number of elements, equivalent to .length

//add adds a name to the end of the list

names.add("Kyle");

names.add("Wendy");

//names is now --> {Kyle, Wendy}

names.add(1, "Stan"); //adds an element to the specified index, shifting others right

//names is now --> {Kyle, Stan, Wendy}

names.get(0); //returns the element at the index specified

names.remove(1); //removes the element at the specified index, compresses the list

names.remove("Wendy"); //removes the first occurrence of the specified element

names.set(0, "Craig"); //replace the value at the specified index

names.contains("Scott"); //returns true or false of the specified element is in the list

names.indexOf("Scott"); //returns the index of the first occurance of the specified element

names.clear(); //erases the entire list, size is now 0

**Printing an ArrayList**

You can print an ArrayList with a for loop just like you would an array. Using get(i) instead of [i]. You can also just print it directly like so:

System.out.println(names);

This will output the Arraylist in the format: [Element1, Element2, Element3, ...etc]

**ArrayLists vs Arrays**

So why use Arrays now that you now about ArrayLists? Well, ArrayLists use a lot of system overhead which can lead to performance issues. This can cause slowdowns or heavy memory usage when working with very large ArrayLists.

**ArrayLists use arrays internally.** They are not magic. They just hide all of the "ugly" array code from you. ArrayLists are fine in the thousands, but if you ever need to work with hundreds of thousands or millions of elements use an array.