

ArrayList objects store a collection of reference type objects.

### Primitive

int

double

boolean

→

→

→

### Reference

Integer

Double

Boolean

String

Any Objects

### Arrays

VS

### ArrayList

- more rigidly  
structured data

- Less space in  
RAM

- Much  
more

flexible

### Creating an ArrayList

ArrayList <String> listName = new ArrayList<String>();

Annotations:

- ArrayList <String>: ArrayList type (under ArrayList), Reference type of List Contents (under <String>)
- listName: List / var name
- =: Assign op.
- new: "new"
- ArrayList<String>(): ArrayList Constructor (under ArrayList), Reference type of List Contents (under <String>), Rtn. (under parentheses)

★ Lists start out emp

## ArrayList Methods

`list.get(i);`

↳ Returns the list item at a given index.

`{"Hi", "Hello", "Bye"}`

`list.get(0);`

`>>> "Hi"`

`list.add(v);`

↳ Adds a certain item to the end of the list.

`{"Hi", "Hello"}`

`list.add("Bye");`

`{"Hi", "Hello", "Bye"}`

`list.add(i, v);`

↳ Adds a certain item to the list at the location specified (i)

`{"Hi", "Hello"}`

`list.add(0, "Bye");`

`{"Bye", "Hi", "Hello"}`

`list.set(i, v);`

↳ Sets the value of an element at a certain location (overrides)

`{"Hi", "Hello"}`

`list.set(0, "Bye");`

`{"Bye", "Hello"}`

`list.remove(i);`

`{"Hi" "Hello"}`

↳ Removes item from  
list at an index

`list.remove(0);`  
`{"Hello"}`

`list.size();`

`{"Hi", "Hello", "Bye"}`

↳ Returns the number  
of elements in  
a list.

`list.size();`  
`>>> 3`

### Traversing ArrayLists

`for (int i = 0; i < list.size(); i++) {`

`System.out.println(list.get(i));`  
`}`