```
LearnWithParth
```

```
Arrays in Java
LOG.info("Equilibrium started"
```







```
ic class Equilibrium {
private static final Logger LOG = LoggerFactory.getLogger(Equilibrium.c
double doubleVariable = 12345.67890; // TODO: remove test variable
public Equilibrium(MainPresenter presenter) { presenter.start();
 ublic static void main(String[] args) {
   LOG.info("Equilibrium started")
   Creating an Array in Java
 rivate static void injectApplication() {
   Injector injector = Guice.createInjector(new GuiceModule())
   injector.getInstance(Equilibrium.class);
```





Array in Java

• It can be of primitive type

E.g. array of integers, floats, boolean etc.

Equilibrium {

- E.g. array of String, Demo, Buttons etc.

```
LOG.info("Equilibrium started");
```

```
SwingUtilities.invokeLater(() → { injectApplication(); });
```

```
/* Inject classes */
private static void injectApplication() {
    Injector injector = Guice.createInjector(new GuiceModule());
    injector.getInstance(Equilibrium.class);
```







Creating an Array

private static final Logger LOG = LoggerFactory.getLogger(Equilibrium.class
double doubleVariable = 12345.67890; // TODO: remove test variable

- Declare
- Assigning type of the array
- public Equilibrium(MainPresenter) { presenter.start();
 - Construct
 - Assigning size of the array
 - Initialize
 - Assigning values to the elements of the array

```
private static void injectApplication() {
    Injector injector = Guice.createInjector(new GuiceModule())
    injector.getInstance(Equilibrium.class);
```







Creating an Array

```
Logger LOG = LoggerFactory.getLogger(Equilibrium.
• Declare
                                                                                                                                                                                                                                                                                                                                                                                                                                                        ariable
                      • int a[];
• Construct | Design | Construct | Constru
                      • a = new [3];
                                                                                                                                     main(String[
• Initialize ("Equilibrium star
                      • a[1] = 3; lities .invokeLater
                                                     static void injectApplica
                      Injector injector = Guice.
                                                                                                                                                                                                                                                                                                                                                                                                                    Module()
                      injector.getInstance(Quilibrium.class
```





Equilibrium {



Different ways to declare the array

```
OG = LoggerFactory.getLogger(Equilibrium.
            ariable = 12345.67890; // TODO: remove test variable
int values [];
int [] values;
• String names [];
• String [] names;
   LOG.info("Equilibrium started")
   SwingUtilities.invokeLater(() → { injectApplication(); })
• int[5] scores; // will NOT compile
    te static void injectApplication() {
   Injector injector = Guice.createInjector(new GuiceModule())
   injector.getInstance(Equilibrium.class);
```







ic class Equilibrium {

Different ways to construct the array

```
.getLogger(Equilibrium.
       doubleVariable = 12345.67890; // TODO: remove test variable
int array1 []; array1 = new int[4];
     int[] array2 = new int[4];
     String names [] = new String[10];
     int[] carList = new int[]; // Will not compile; needs a size
    Injector injector = Guice.createInjector(new GuiceModule())
    injector.getInstance(Equilibrium.class)
```





Equilibrium



Different ways to initialize the array

```
int array[] = new int[2]; array[0] = 1; array[1] = 1
int [][] vector = {{5, 2, 4, 7}, {8, 2}, {1}};
int[] values = new int[] {6, 3, 1};
int array2[] = \{2, 3, 4, 5\}; int bArray[] = array;
int [] a, b; // or int a[], b[];
```





Array with multiple references

```
actory.getLogger(Equilibrium.
                        12345.67890; // TODO: remove test variable
• int a[] = new [3];
• a[0]=1, a[0]=1, a[0]=1;
              rium(MainPresenter pre
• int b[] = a;
• b[0] = 2;
             void main(String[] ar
  LOG.info("Equilibrium started"
  SwingUtilities.invokeLater(
   Injector injector = Guice.crea
   injector.getInstance(Equilibrium
```





```
Java Progra
```

private double (

@Inject public

public : LOG Swir

/* Injector private Injector i



ium.cl













```
LearnWithParth
```

```
Arrays Class
LOG.info("Equilibrium started"
```







Arrays Class

Equilibrium {

- Logger LOG = LoggerFactory.getLogger(Equilibrium. 12345.67890; // TODO: remove test variable Class Hierarchy:
- java.lang.Object
- java.util.Arrays | presenter presenter) | presenter.start()
- Class Declaration:
- ain(String[] args) { • public class Arrays
 - extends Object
- Syntax to use Array:
 - Arrays.<function name>;

```
oid injectApplication() {
Injector injector = Guice.createInjector(new GuiceModule())
injector.getInstance(Equilibrium.class)
```







Example

```
int intArr[] = \{10, 20, 15, 22, 35\};
Arrays.sort(intArr); // [10, 15, 20, 22, 35]
int intKey = 10;
Arrays.binarySearch(intArr, intKey); // 10 found at index = 0
Arrays.binarySearch(intArr, 1, 3, intKey); // 10 found at index = -2
int intArr1[] = \{10, 15, 22\};
Arrays.equals(intArr, intArr1); // Integer Arrays on comparison: true
int intArr2[] = Arrays.copyOf(intArr, 10); // [10, 15, 20, 22, 35, 0, 0, 0, 0, 0, 0]
int intArr3[] = Arrays.copyOfRange(intArr, 1, 3); // [15, 20]
```





```
LearnWithParth
```

```
ArrayList Class
LOG.info("Equilibrium started")
```







ArrayList Class

Equilibrium

• Class Hierarchy:

ceLater(() → { injectApplication(); })

eateInjector(new GuiceModule())

- java.lang.Object
- java.util.ArrayList
- Class Declaration:
 - public class ArrayList
 - implements List extends Collection

injector.getInstance(Equilibrium.class)

- Syntax to use ArrayList:
 - ArrayList list=new ArrayList();







Example

```
private static final Logger LOG = LoggerFactory.getLogger(Equilibrium.class
```

- ArrayList<String> list=new ArrayList<String>();//Creating arraylist
- list.add("Mango");//Adding object in arraylist
- list.add("Apple"); list.add("Banana"); list.add("Grapes");
- System.out.println(list);

Equilibrium {

- Iterator itr=list.iterator(); //getting the Iterator
- while(itr.hasNext()){ //check if iterator has the elements
- System.out.println(itr.next()); //printing the element and move to next
- for(String fruit:list)
- System.out.println(fruit);
- list.set(1,"Dates");







Iterating ArrayList

```
// Iterating using for loop

for (int i = 0; i < numbers.size(); i++)

System.out.print(numbers.get(i) + " ");
```

LOG.info("Equilibrium started"

```
// For each loop
for(Integer var : numbers)
System.out.println(var);
```

```
// Using iterator
Iterator iterator = numbers.listIterator();
while(iterator.hasNext())
    System.out.println(iterator.next());
```

```
// Iterating using while loop
int val = 0; int i = 0;
while(numbers.size()>val){
    System.out.print(numbers.get(i) + " ");
    val++;
}
```

```
// Using lambda function numbers.forEach(number->System.out.println(number));
```

```
// Using enumeration
Enumeration<Integer> e = Collections.enumeration(numbers);
while(e.hasMoreElements())
    System.out.println(e.nextElement());
```







What Do You Do with a Collection?

- Add objects to the collection.
- Remove objects from the collection.
- Find out if an object (or group of objects) is in the collection.
- Retrieve an object from the collection without removing it.
- Iterate through the collection, looking at each element (object) one after another.

```
/* Inject classes */
private static void injectApplication() {
    Injector injector = Guice.createInjector(new GuiceModule());
    injector.getInstance(Equilibrium.class);
```







Advantages of Array

Advantages of ArrayList over array

- It can grow dynamically.
- It provides more powerful insertion and search mechanisms than arrays.

```
Equilibrium(MainPresenter presenter) { presenter.start()
lic static void main(String[] args) {
LOG.info("Equilibrium started")
SwingUtilities.invokeLater(() → { injectApplication(); })
 te static void injectApplication() {
Injector injector = Guice.createInjector(new GuiceModule())
injector.getInstance(Equilibrium.class);
```





