CYBAGE
Delivering Value. Scientifically.

Java Fundamentals

Author & Presenter –Asfiya Khan
(Training Specialist- Technical)

www.cybage.com

Agenda

- Variables in Java
- ‘static’ keyword
- Memory layout
- Concept of Garbage Collector
- Parameter Passing
- Arrays
- Enhanced for loop
- Packages

Variables

A variable is a name given to memory location. That memory is associated to a data type and can be assigned a value.

```
int id ;
float rateOfInterest;
char status;
double salary;
```

Variables

- Assigning a value to a variable.
- Initialization of a variable with primary value.

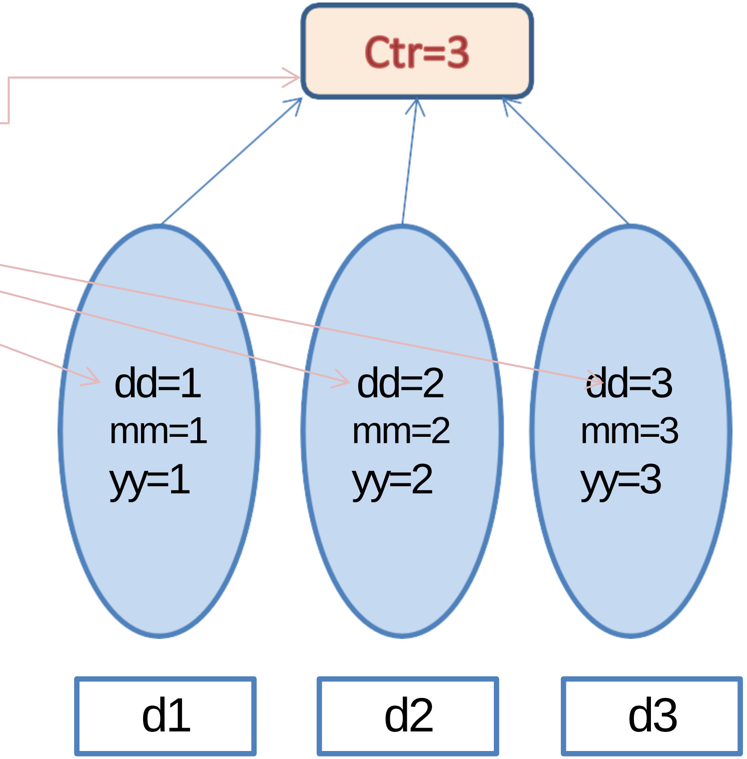
1. int accountNumber;
2. accountNumber =2123457788 ; // assignment
3. char choice = ‘A’; // initialization
4. double salary = 56777.24d; // initialization
5. float rateOfInterest = 10.45F;

Types of Variables

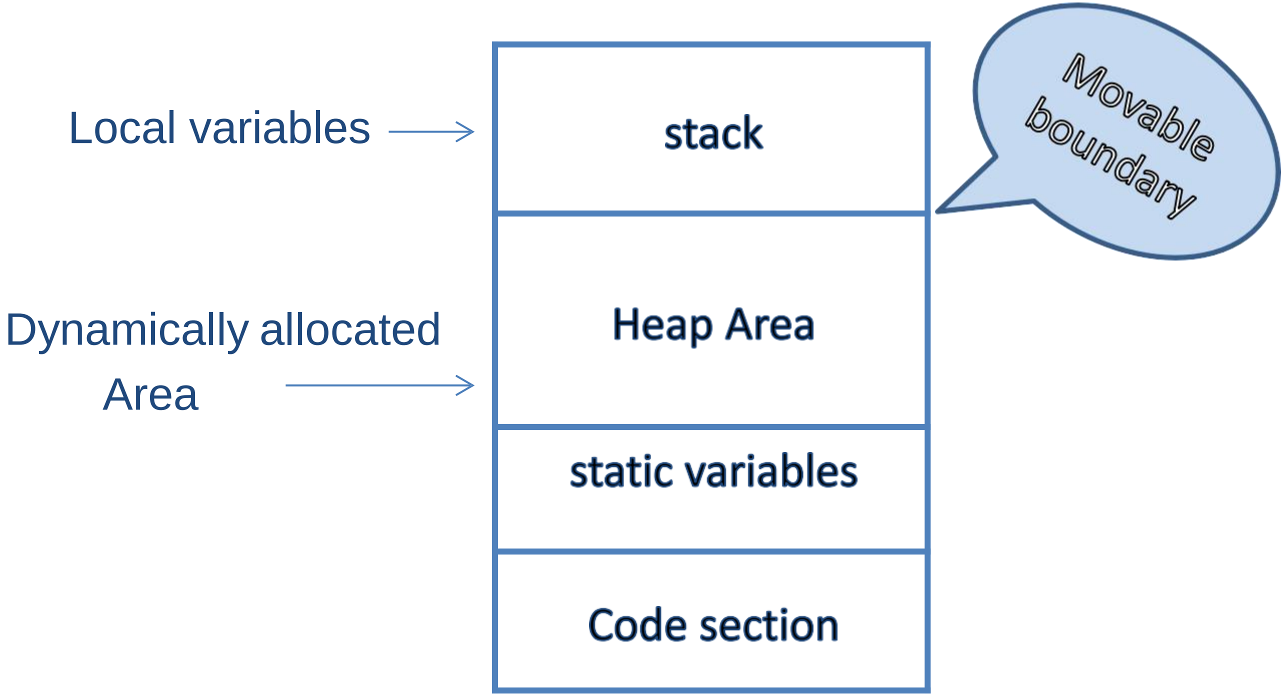
- Instance Variables : Copy exists per instance
- Static Variables : Class level variable i.e. copy exists per class
- Local Variables : Variables declared within methods or blocks. They are local to the block where they are declared

static Keyword

```
class Date
{
    static int ctr;
   int dd;
   int mm;
   int yy;
}
```



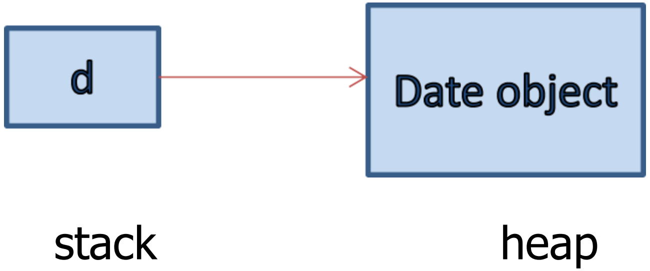
Memory Layout



Memory

Local variables are created on stack where as objects are created on heap.

```
class Demo {
  public static void main (String args[]) {
            Date d = new Date ();
     }
   }
```

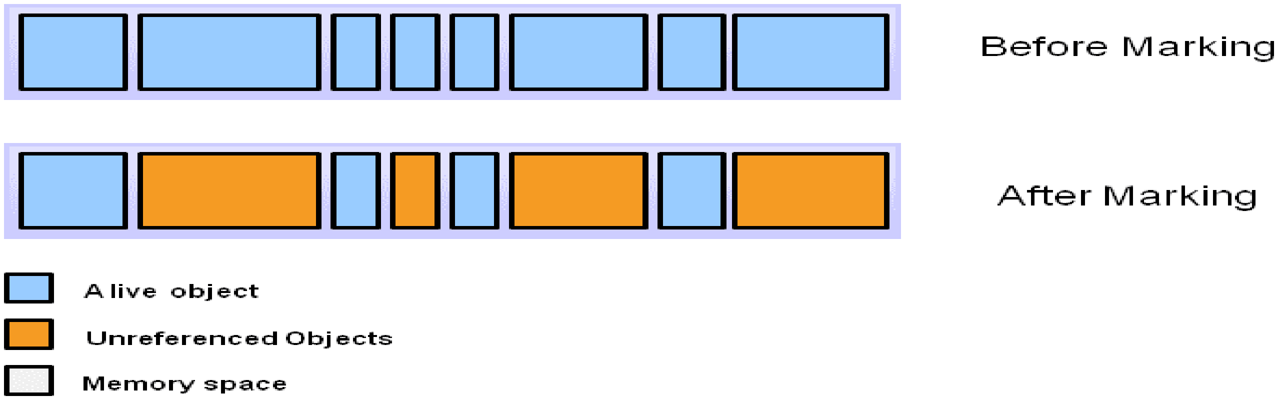


Concept of Garbage Collector

- Garbage collector generally reclaims orphaned object spaces
- Finalize() gets executed just before an object is garbage collected.
- Override Object class finalize method to write clean up code.

Garbage Collection Phases

Marking



Before Marking

After Marking

🟦 A live object

🟧 Unreferenced Objects

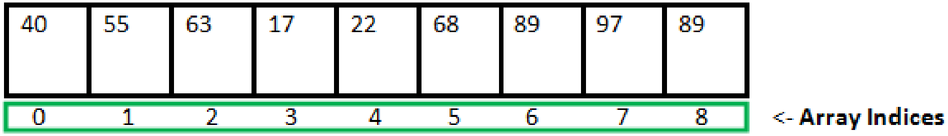
⬜ Memory space

Parameter Passing In Java

- In Java we can pass parameter to any method by value only.
- Whenever we pass any object reference to a method then reference value gets copied inside the formal parameter.
- Any modifications to the formal parameter variable inside the called function or method affect only the separate storage location and will not be reflected in the actual parameter in the calling environment

Arrays

- An array is a group of like-typed variables that are referred to by a common name.
- In Java all arrays are dynamically allocated i.e. allocated on heap.
- Since arrays are objects in Java, we can find their length using member length.
- Array can contains primitives data types as well as objects of a class depending on the definition of array.
- Array elements accessed by its numerical index.



40	55	63	17	22	68	89	97	89	
0	1	2	3	4	5	6	7	8	<- Array Indices

Array Length = 9
First Index = 0
Last Index = 8

Array Declaration

```
type var-name[];
OR
type[] var-name;
```

```
int intArray[];
OR
int[] intArray;
```

- An array declaration has two components: the type and the name.
- *type* declares the element type of the array.
- the element type for the array determines what type of data the array will hold
- name is unique identifier.

Instantiation of Array

- When an array is declared, only a reference of array is created.
- To actually create or give memory to array, you create an array like this:

The general form of *new* as it applies to one-dimensional arrays appears as follows:

```
int intArray[]; //declaring array

intArray = new int[20]; // allocating memory to array

            OR

int[] intArray = new int[20];   // combining both statements in one
```

Array Literal

- In a situation, where the size of the array and variables of array are already known, array literals can be used.

```
int[] intArray = new int[] { 1,2,3,4,5,6,7,8,9,10 };  // Declaring array literal
```

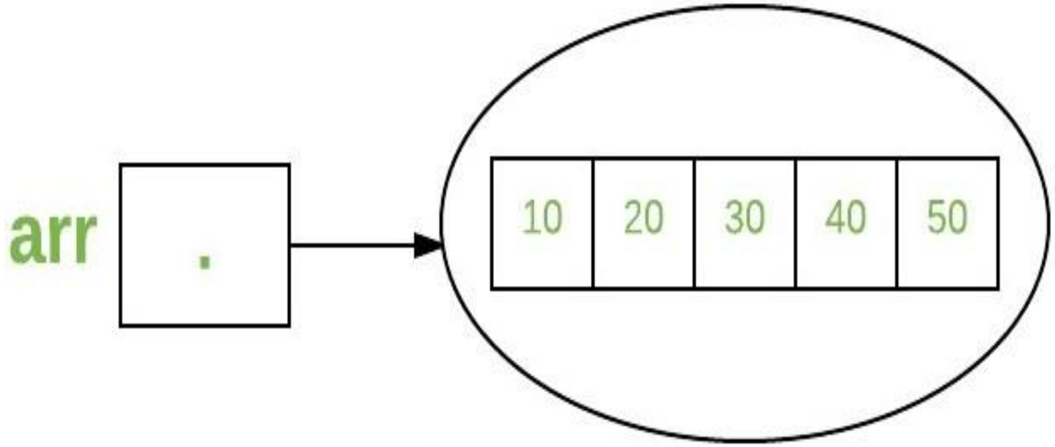
OR

```
int[] intArray = { 1,2,3,4,5,6,7,8,9,10 };
```

- The length of this array determines the length of the created array.
- There is no need to write the new int[] part in the latest versions of Java

One Dimensional Arrays

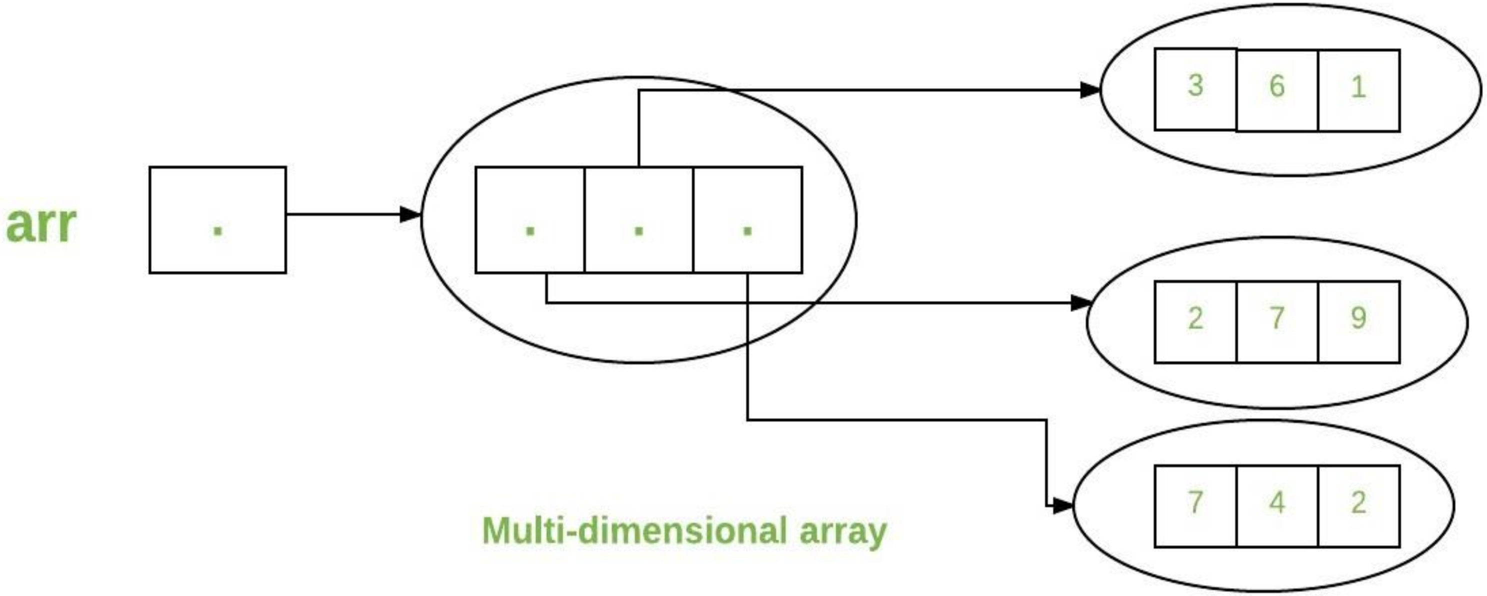
```
int[] arr;

arr = new int[5];

arr[0] = 10;
arr[1] = 20;
arr[2] = 30;
arr[3] = 40;
arr[4] = 50;
```



One-Dimensional Array

Multi-Dimensional Arrays

```
int arr[][] = {  {2,7,9},{3,6,1},{7,4,2} };
int arr[][] = new int[3][3];
```

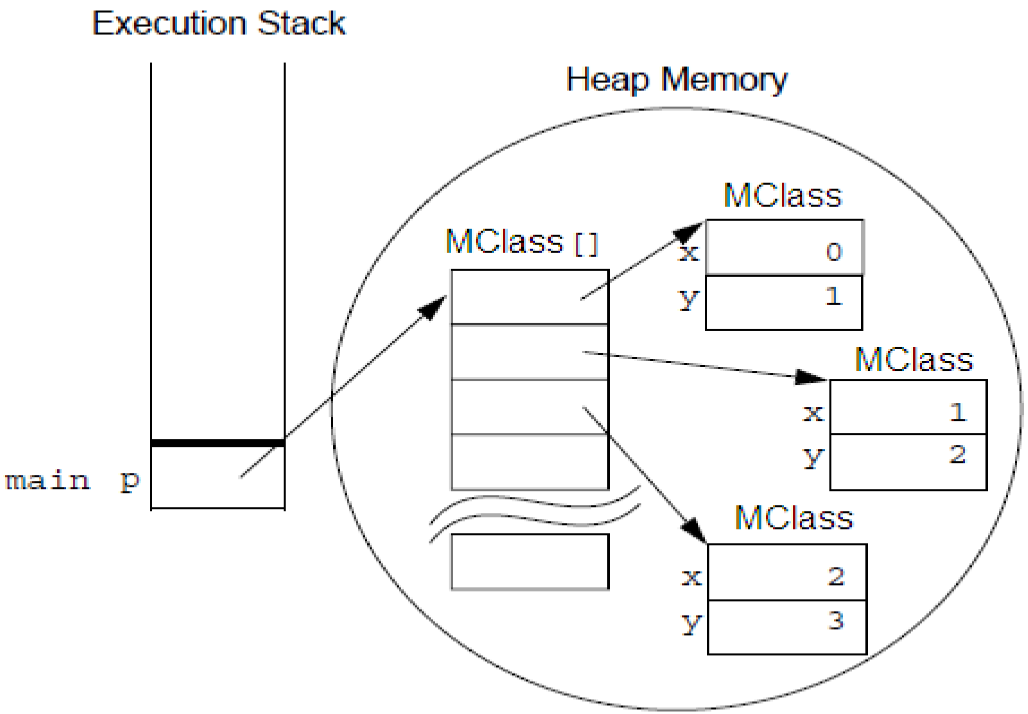


Creating Reference Arrays

Example:

```
public class MClass{

    public MClass(int x , int y){
    System.out.println("constr" + x + y );
    }

    public static void main(String [] a){
        MClass[] p;
        p = new MClass[10];
        for ( int i=0; i<10; i++ ) {
            p[i] = new MClass(i, i+1);
        }
    }
}
```

Creating Reference Arrays (Contid)



Array Bounds

All array index begin at 0:

```
public void disp() {

          Int [] x = new int[5];

          for (int i = 0; i < x.length; i++) {
                    System.out.println(x[i]);
          }
}
```

Packages

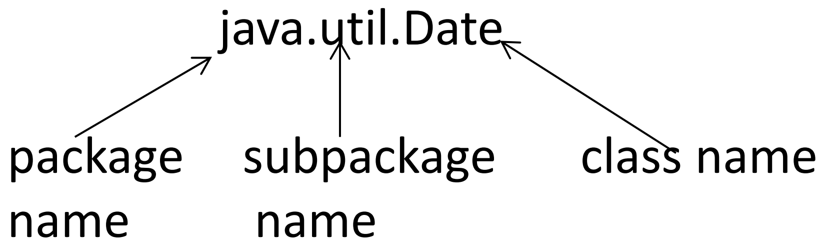
- A package is a grouping of related classes & interfaces providing access protection and name space management
- Programmers bundle groups of related types into packages to make it easier to find & use related classes & interfaces
- Packages help to avoid naming conflicts
- There can be only one package declaration per source file
- In case if no package is declared, then the class is placed into the default package i.e., the current folder

Packages Contd.

- If you want to create a package ,the package statement should be the first statement.

- The fully qualified name of a class is : packageName.className

- Fully qualified name of class Date in java is



java.util.Date

package name

subpackage name

class name

Packages Contd.

```
package p1;

import java.util.Date;
import java.sql.*;

class MyClass
{



}
```

Basic Structure Of Java File

- Package statement……optional
- Import statements …..optional
- Standard class writing

ANY
QUESTIONS
?

CYBAGE
Delivering Value. Scientifically.

Thank You!

www.cybage.com