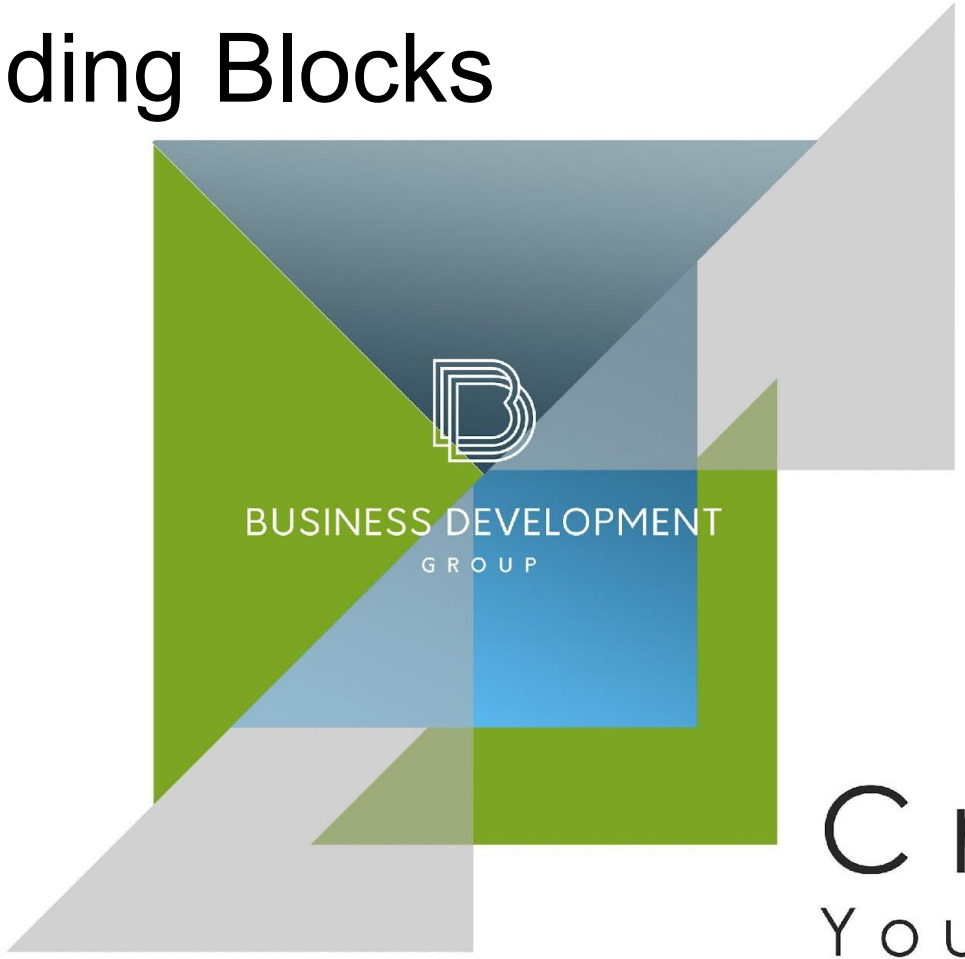
Java Building Blocks



Create
Your Future

BUSINESS DEVELOPMENT
GROUP

Consider the following points

- Java Class Structure
- Memory management
- Garbage Collection

BUSINESS DEVELOPMENT
GROUP

Java Class Structure

An object is a runtime instance of a class in memory.

Class members

- **Fields** hold the state of the program
- **Methods** operate on that state

```
1   public class Person {
2       String name;
3
4       public String getName() {
5           return name;
6       }
7
8       public void setName(String name) {
9           this.name = name;
10      }
11  }
12
```

BUSINESS DEVELOPMENT
GROUP

Order of Initialization

1. Static variables
static initializers
2. Instance variables
instance initializers
3. Constructor

```
1
2  public class Chick {
3
4      private String name = "Fluffy";
5
6      {
7          System.out.println("Instance initializers");
8      }
9
10     static {
11         System.out.println("Static initializers");
12     }
13
14     Chick() {
15         name = "Tiny";
16         System.out.println("Constructor");
17     }
18
19     public static void main(String[] args) {
20         Chick chick = new Chick();
21         System.out.println(chick.name);
22     }
23 }
```

BUSINESS DEVELOPMENT
GROUP

Constructor

1. The name of the constructor matches the name of the class
2. There's no return type

```
1   public class Hamster {
2
3       private String color;
4
5       private int weight;
6
7       public Hamster(int weight) {
8           this(weight,  color: "brown"); // Must be first line
9       }
10
11      public Hamster(int weight, String color) {
12          this.weight = weight;
13          this.color = color;
14      }
15  }
```

BUSINESS DEVELOPMENT
GROUP

Constructor Rules

1. The first statement of every constructor is a call to another constructor within the class using this(), or a call to constructor in the direct parent class using parent class using super().
2. The super() call may not be used after the first statement of the constructor.
3. If no super() call is declared in a constructor, Java will insert a no-argument super() as the first statement of the constructor.
4. If the parent doesn't have a no-argument constructor and the child doesn't define any constructors, the compiler will throw an error and try to insert a default no-argument constructor into the child class.
5. If the parent doesn't have a no-argument constructor, the compiler requires an explicit call to a parent constructor in each child constructor.

BUSINESS DEVELOPMENT
GROUP

Package Declarations and Imports

- Java.lang package is automatically imported
- Use wildcards (*) to import all classes in a package

```
1   import java.lang.System;
2   import java.lang.*;
3   import java.util.Random;
4   import java.util.*;
5
6   public class ImportExample {
7
8       public static void main(String[] args) {
9           Random r = new Random();
10
11      }
12  }
```

Naming Conflicts

```
1   import java.util.*;
2   import java.sql.*;
3
4   public class Conflicts {
5
6       public static void main(String[] args) {
7           Date date = new Date();
8       }
9   }
```

BUSINESS DEVELOPMENT
GROUP

Rules for JavaBeans naming conventions

1. Properties are private.
2. Getter methods begin with is if the property is a boolean, otherwise get.
3. Setter methods begin with set.
4. The method name must have a prefix of set/get/is, followed by the first letter of the property in uppercase, followed by the rest of the property name.

BUSINESS DEVELOPMENT
GROUP

Immutable classes

```
1   public class ImmutableSwan {
2
3       private int numberEggs;
4
5       public ImmutableSwan(int numberEggs) {
6           this.numberEggs = numberEggs;
7       }
8
9       public int getNumberEggs() {
10          return numberEggs;
11      }
12  }
13
```

BUSINESS DEVELOPMENT
GROUP

Ordering Elements in a Class

Elements	Example	Required?
Package declaration	package model;	No
Import statements	import java.util.*	No
Class declaration	public class C	Yes
Field declarations	int value;	No
Method declarations	void method()	No

BUSINESS DEVELOPMENT
GROUP

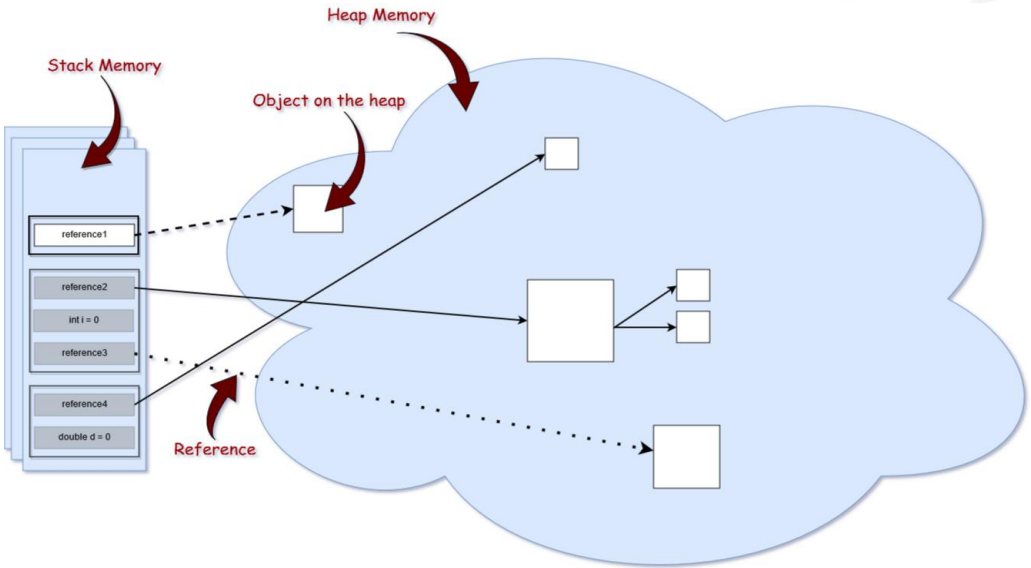
Objects vs References

1. The type of the object determines which properties exist within the object in memory.
2. The type of the reference to the object determines which methods and variables are accessible to the Java program.

BUSINESS DEVELOPMENT
GROUP

Memory Management

1. A reference may or may not be created on the heap. All references are the same size, no matter what their data type is, and are accessed by their variable name.
2. Objects are always on the heap. They have no name and can only be accessed via a reference. Objects are different in size depending on their class definition.

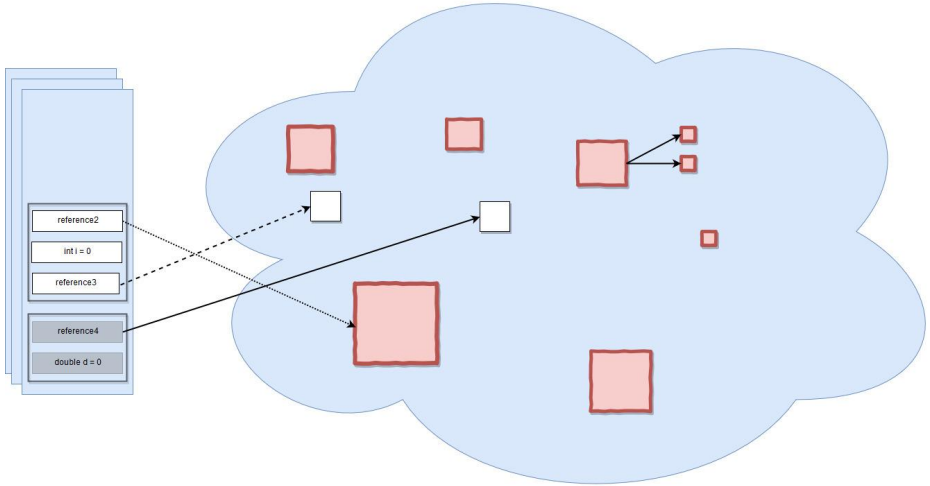


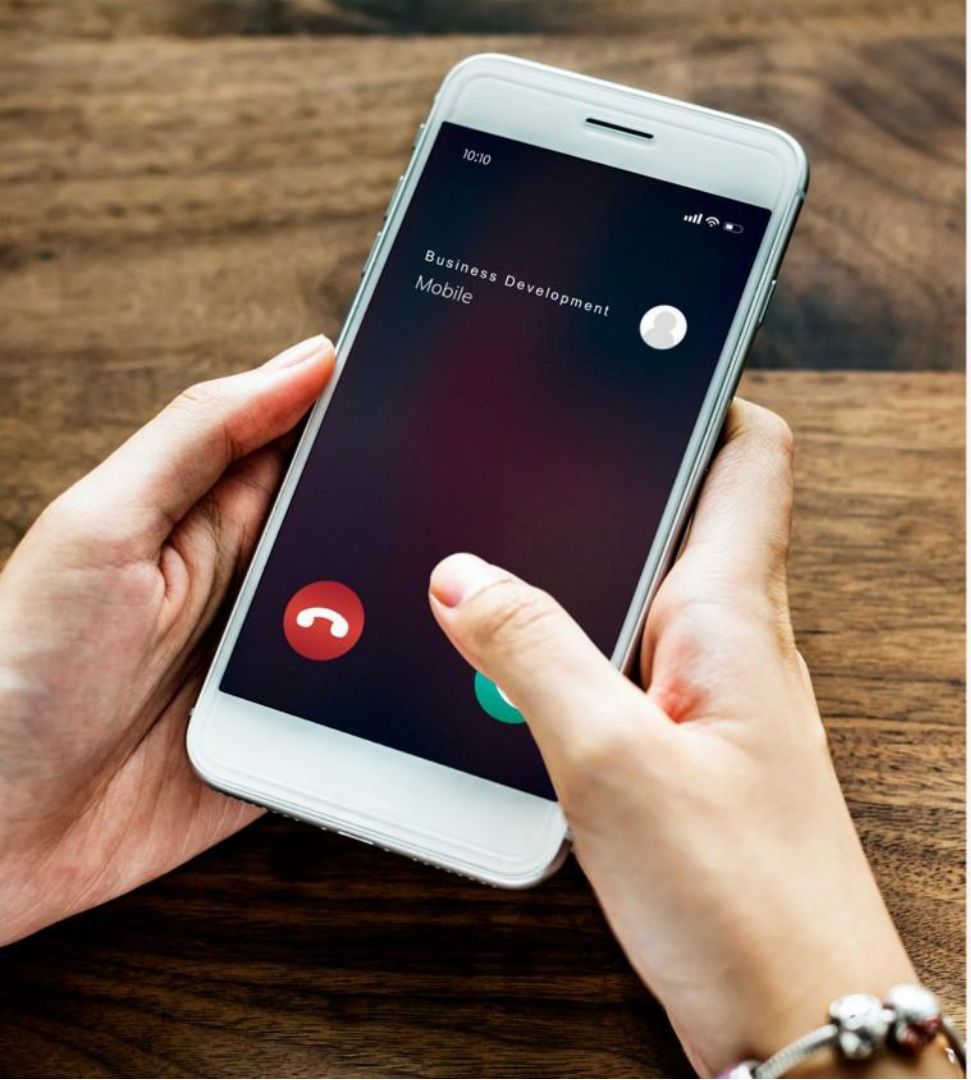
BUSINESS DEVELOPMENT
GROUP

Garbage Collection

1. Java provides a method called System.gc().
2. finalize() is only run when the object is eligible for garbage collection.

```
1  public class Scope {
2
3      public static void main(String[] args) {
4          String one, two;
5          one = new String( original: "a");
6          two = new String( original: "b");
7          one =two;
8          String three = one;
9          one = null;
10     }
11 }
```





Contact US

Phone: +374 55 201 209

E-mail: info@bdg.am

Address: Hr. Kochar 4

Web: www.bdg.am

https://www.facebook.com/bdg.trainings/

bdg.trainings

https://www.linkedin.com/in/bdg-trainings/