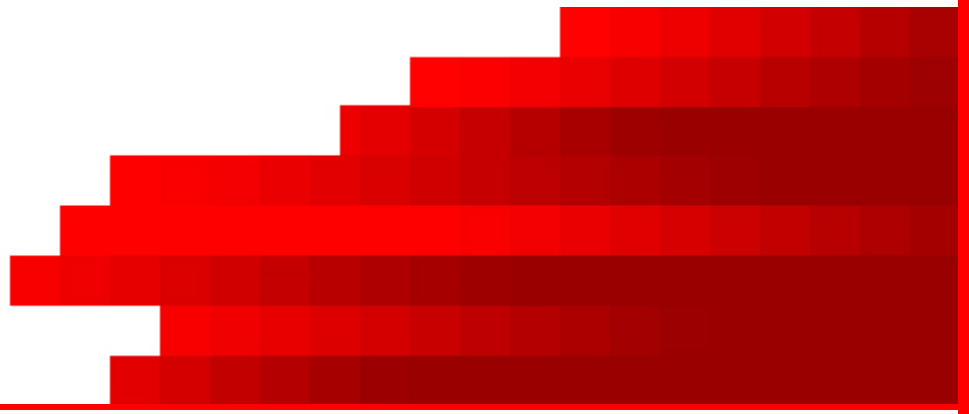
Core Java



HSBC Technology and Services

HSBC
The world's local bank

Introducing Classes

- Class fundamental
- The general form of a class

```
class classname
{
    type instance variable;
    type methodname(parameter list)
    {
        body of the method
    }
}
```

Declaring Objects

```
Box mybox = new Box();

Box  mybox;
mybox = new Box();
```

Assigning Object Reference Variables

```
Box b1 = new Box ();
Box b2 = b1;
```

Introducing Methods

```
type name (parameter-list)
  {
        //body of the method
  }
```

Adding a Method to the Box Class

- BoxDemo3.java

Returning a value

The type of data returned by a method must be compatible with the return type specified by the method.
The variable receiving the value returned by a method must also be compatible with the return type specified for the method

BoxDemo4.java

Adding a method that takes parameter

- BoxDemo5.java

Constructors

- A constructor initializes an object immediately upon creation.It has the same name as the class in which it resides and is syntactically similar to a method. Once defined a constructor is automatically called.They don’t have any return type, not even void.

- BoxDemo6.java

Parameterized Constructors

- BoxDemo7.java

The this Keyword

- **this can be used inside any method to refer to the current object. this is always a reference to the object on which the method was invoked.**

```
}
} // A redundant use of this.
} Box(double w, double h, double d)
} {
}   this.width = w;
}   this.height = h;
}   this.depth = d;
} }
```

```
}
} /* Use this to resolve name-space collisions.*/
} Box(double width, double height, double depth)
} {
}   this.width = width;
}   this.height = height;
}   this.depth = depth;
} }
```

Overloading Methods

- In Java it is possible to define two or more methods within the same class that share the same name, as long as their parameter declarations are different. When this is the case, the methods are said to be overloaded, and the process is referred to as method overloading. It is one of the ways that Java implements Polymorphism.

- Overload.java

Automatic type conversion for overloading

- Overload1.java

Benefits of Overloading

Overloading Constructors

- OverloadCons.java

}

- The absolute path is system dependent and may include relative indicators
- For example, the following code creates a file 'test2.txt' in the directory directly above the current directory: File f1 = new File("..", "test2.txt") f1.createNewFile(); System.out.println(f1.getAbsolutePath()); Output (on Win98): D:\Java\jeg\io\..\test2.txt
- the canonical path is the same as the absolute path BUT all relative indicators are resolved
- For example, System.out.println(f1.getCanonicalPath()); Output (on Win98): // '..' in absolute path is resolved D:\Java\jeg\test2.txt

About HSBC Technology and Services

HSBC Technology and Services (HTS) is a pivotal part of the Group and seamlessly integrates technology platforms and operations with an aim to re-define customer experience and drive down unit cost of production. Its solutions connect people, devices and networks across the globe and combine domain expertise, process skills and technology to deliver unparalleled business value, thereby enabling HSBC to stay ahead of competition by addressing market changes quickly and developing profitable customer relationships.

Presenter's Contact Details:
Name: Deepak Ratnani
Role: Team Leader
Direct: + 91-20 -66423608
Email: DeepakRatnani@hsbc.co.in