A Brief Introduction

This lesson welcomes you to the world of Object Oriented Programming.

WE'LL COVER THE FOLLOWING

Procedural Programming

Presumably, you are familiar with the basics of programming and you should have used methods in your programs.

Procedural programming is a programming paradigm. In procedural programming, a program is divided into smaller parts called methods. These **methods** are the **basic entities** used in this technique. The focal point of procedural programming technique is to use methods for code reusability. However, the implementation of a complex real-world scenario becomes a difficult task using this paradigm.

What Object-Oriented Programming is?

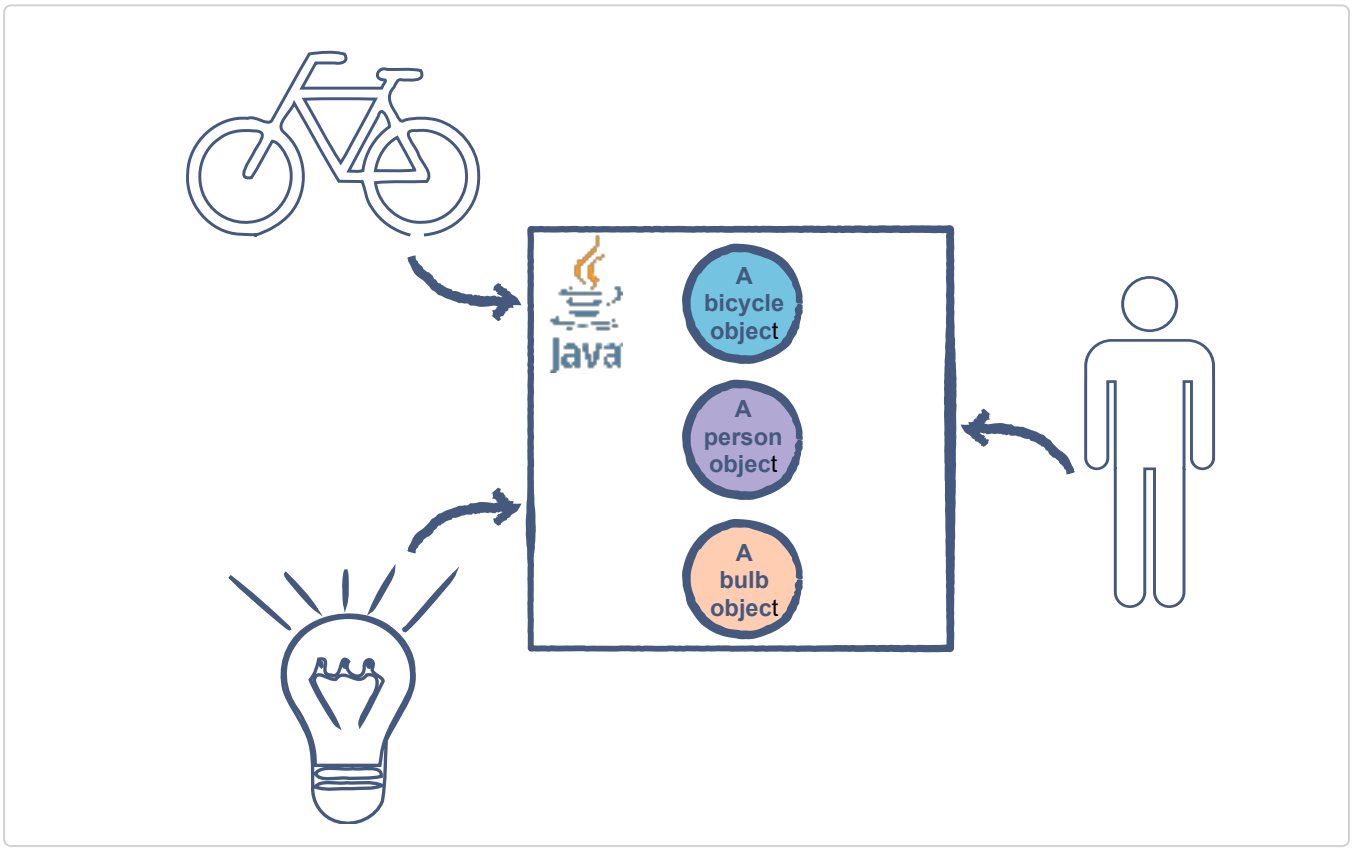
> *Object-oriented programming*, also referred to as **OOP** is a programming paradigm that includes or relies on the concept of classes and objects.

The basic entities in object-oriented programming are **classes and objects.**

Programming won't make much sense if one can't model real-world scenarios using programming languages right? This is where Object-Oriented Programming comes into play!

> The basic idea of OOP is to divide a complex program into a bunch of **objects** talking to each other.

Objects in a program frequently represent real-world objects.



Many other objects serve application logic and have no direct real-world parallel objects that manage authentication, templating, request handling, or any of the other myriad features needed for a working application.

Anatomy of Objects and Classes

Objects may contain data in the form of *fields* (variables) and methods to operate on that data. For a while, think about the real-world objects around you."*What are the characteristics of these objects*? Take the example of a *light bulb*. It has a **state** i.e. either it is *on* or *off*. It also has a **behavior** i.e. when you turn it on it lights up and when turned off, it stops spreading light. To conclude this one can say:

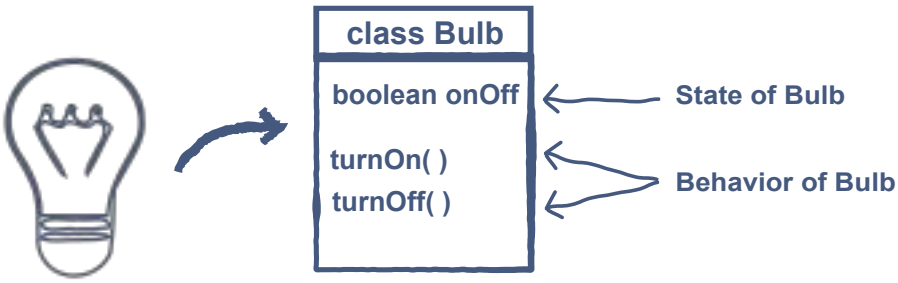
> Objects have **state** and **behavior**.

Interesting! isn't it? But the question is *"where do these objects come from?"*

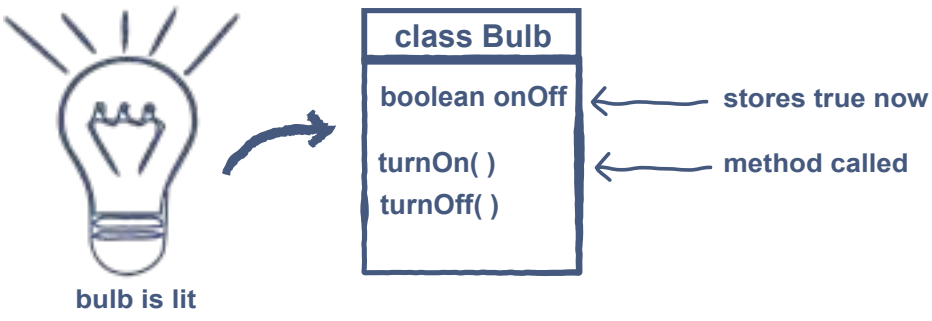
Well, the answer to the above question is **classes**.

> A **Class** can be thought of as a *blueprint* for creating objects.

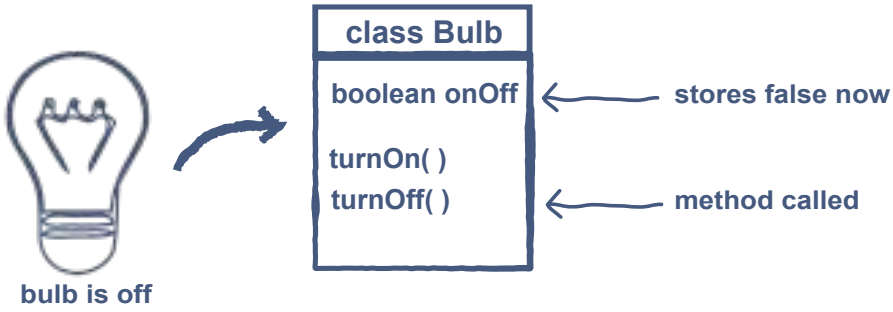
The below illustration shows what a `LightBulb` class should look like.



1 of 3

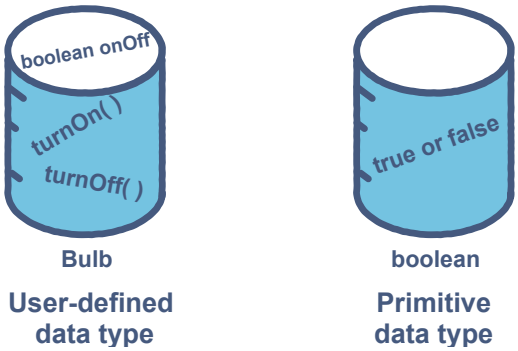


2 of 3



3 of 3

From the above illustration, it can be concluded that the state of the objects is generally modeled using *variables* in a class and the behavior is modeled by implementing the *methods*.



It can be inferred from the discussion above that classes are user-defined data types implemented using the primitive data types e.g. `boolean`, `int`, `char` etc.

We will leave this discussion for now and come back to it in a later section.

In the next lesson, you will get to know about the modern Object-Oriented languages and how Java is one of these.