

# Instances

---



In order to use **objects** and **classes**, it is necessary to create **instances**, which are covered in this section.

## Instances

A class is simply a group of properties and behaviours (behaviours will be covered later) grouped together and given a name.

Think of a class as a detailed description of an object. A class is not an object, but it contains all the information needed to create an object. It is like the relationship between a blueprint and a house. An **instance** is a single entity created from the class. It will contain all the properties and behaviours of the class with its own values. It is a house built from the blueprints but, for example, its colour or roof type may vary from another house instance made from the same blueprint.

Lets us continue with the blueprint/house analogy. We may have the blueprint, the class, but it is of no immediate use to us if we want to live somewhere, that is in a house. We have to use this blueprint, this class, to 'build' our house. So in the house world we would 'build' our house using the blueprint. Similarly in the object-oriented modelling world we too would have to 'build' our object. The process of 'building the object from the class description is called **instantiation** (making an instance of).

**Instantiation** is the process of constructing an object for a given class.

The process of instantiation is sometimes compared to a **factory** which manufactures related products. The factory produces products (classes), where each item (object) can be customised (instantiation). For example, a car manufacturer produces vehicles (class *Car*) where automobiles (objects *car1*, *car2*, *car3*, etc) will be built to a certain specification (**class description**), that is, each car will have four wheels, an engine, 4 seats, and so on. However, each individual car (instance) has certain properties, for example, some cars will have alloy wheels, engines are either diesel or petrol and different sizes, some cars will have leather seats, ....

**An object is an instance of a class!**

Each object has all the properties and other characteristics of the class, which can then be customised. When modelling an individual object, each property has to be set to valid value, for instance, Dima's hair is blond, his eyes are green, his jumper is blue, his trousers are red and so on.



**Figure 1:** The object Dima

Because Dima belongs to the *Sloopy* class, all properties/attributes are already provided and only values have to be attached. The advantage of instantiation (making instances) is that as many Sloopies (objects or instances) can be created as are necessary, with very little effort. Furthermore, if an additional property has to be added to Sloopy objects, it only has to be done at the class level; all instances will automatically contain the new properties/attributes.

As the name suggest, UML is a language that focuses on the modelling of (software) systems and thus does not support implementations of these models. This is the reason the initial UML notation did not cater for objects and also why there was no representation for instances. The latest version of UML has rectified this situation and introduced **Object Models**, which are beyond the scope of this introductory course.

## Summary

A class is simply a group of properties and behaviours rather like a blueprint for a house. The blueprint is used to build a house and similarly the class and its description are used to build objects in the class, which are called instances. The process of creating objects from classes is called **instantiation**.



We have been looking at instances at object level. In the next Section we will look at instances at class level.