What is the purpose of Object Oriented Code and what are the benefits?

OOP allows for real world entities to be mapped into discreet bundles of properties and actions (known as methods). Everything in the real world can be described as an object, for example individual trees are objects, individual people are objects and individual pencils are objects.

Let’s take a look at a person as a specific example:

A person has properties, these are things that describe what a person is and can be:

* Name
* Age
* Current address
* Eye color
* Gender
* etc….

All persons have these properties, however each individual person has a specific set of properties that makes them unique (In a way. Ok, two people can have the same set of properties as each other but for the purposes of this example they make them unique).

A person also has methods (actions that they can undertake), for example:

* Talk
* Walk
* Eat Food
* Sleep
* etc…

Again all persons can do these actions (or, at least a subset of actions as some persons can do some things other persons cannot for example) but with different properties. So for example,

PersonA

* .name(‘James’)
* .talk(languageSpoken) 🡪 .talk(‘English’)
* .sleep(sleepLength) 🡪.sleep(7.5)
* .walk(speed) 🡪 .walk(3.25)

PersonB

* .name(‘Suzanne’)
* .talk(languageSpoken) 🡪 .talk(languages[])
* .sleep(sleepLength) 🡪.sleep(5)
* .walk(speed) 🡪 .walk(4.1)

Above we have created 2 persons, one called James and another called Suzanne. These people objects have been derived or ‘instanced’ from an object class that we call ‘Person’. James and Suzanne are individual instances of this class.

OOP allows us to create these classes that then allow us to create the individual instances of that class as objects.

We can then work with those objects such as adding or updating properties as is necessary, executing methods against the objects, which in turn may cause other objects to update properties or execute their methods – for example, while person ‘a’ walks dog ‘a’, the person object is causing dog object, through their interaction, to execute its methods and update its properties as necessary. Finally, when we are done using the objects we can dispose of them and perhaps create new objects with completely different sets of properties. We are able to do this from the classes Person and Dog we have created.

Once we have written the Class code, it will allow us to ‘reuse the code’ over and over, not just in a single project but in any project we choose that has a need for creating its type of object. We put these classes together in what we call, a Class Library.