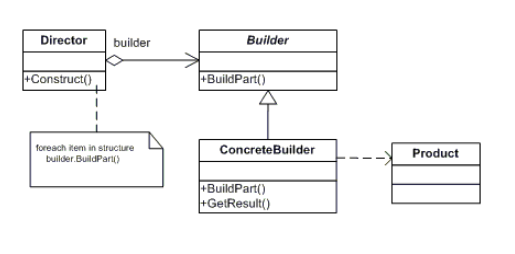
# Builder Pattern

**Motivation**

**Intent**

The Builder design pattern separates the construction of a complex object from its representation so that the same construction process can create different representations.

**UML Class Diagram**



The classes and objects participating in this pattern include:

* Builder  (VehicleBuilder)
  + specifies an abstract interface for creating parts of a Product object
* ConcreteBuilder  (MotorCycleBuilder, CarBuilder, ScooterBuilder)
  + constructs and assembles parts of the product by implementing the Builder interface
  + defines and keeps track of the representation it creates
  + provides an interface for retrieving the product
* Director  (Shop)
  + constructs an object using the Builder interface
* Product  (Vehicle)
  + represents the complex object under construction. ConcreteBuilder builds the product's internal representation and defines the process by which it's assembled
  + includes classes that define the constituent parts, including interfaces for assembling the parts into the final result