*Hire class.* The Hire class is the same entity class that featured in the analysis model. As with Payment and Customer objects, each :Hire has a unique hireId which is generated by the class variable hireCount. The attributes customer and bike are used to hold references to Customer and Bike objects. Although it has four get methods, the only method shown on the diagram is a constructor.

## The code

Don't be dismayed if you cannot follow all of the code. This book is not about programming. The purpose of this section is to demonstrate how the class diagram maps onto the code. For the moment, just ignore the parts you do not understand and accept that there is some reason why they have to be there. In this section, for three of the classes only, we compare the class diagram model of the class with the section of code that implements it.

### *StartUp class*

The class diagram for the StartUp class is shown in Figure 11.3, the code for the StartUp class is shown in Figure 11.4.

- Line 01 of the code declares that StartUp is in a package called bikeshop (as are all of the classes in the code).
- The StartUp class declaration is in line 05. Everything in the class must be contained within two curly brackets {}, these brackets tell the compiler where the class begins and ends. The opening bracket is on line 05, the closing one on line 27.
- The method, main(), is declared in line 07. All of the program instructions for this method are inside a second pair of curly brackets, the opening bracket is on line 07, the closing bracket is on line 26.
- The first instruction in main() creates an IssueBikeUI object, ui (line 13), the next four instructions are calls to the IssueBikeUI object. These four instructions implement the main steps in the 'Issue bike' use case scenario.
- Lines beginning with a double forward slash, //, are comments and are ignored by the compiler.[5] Comments can also be contained within /* */; the compiler ignores everything after an opening /* until it finds the closing */.

5. *A compiler is a program that translates source code (e.g. a Java program) into machine code, i.e. code which can be executed by a computer.*