“Mini-Project”: Inheritance and Dispatching in C# Versus Java 5

The C# code is for on-line submission, by 5pm Friday, Sept. 28, filename DispatchingNNN.cs, where NNN stands for your initials in uppercase (if you have 2 or 4 initials, that’s fine). The command to submit files foo1, foo2 from the machine fork (which you should use in preference to yeager) is:

~regan/submit_cse305 foo1 foo2

The report is now requested in hardcopy only, as part of Assignment 5. Handwritten answers are now fine. The report questions are below.

Directions—coding part: Translate the file Dispatching.java given in the directory ~regan/cse305/LANGUAGES/C#/LECREC07/ into C#. A checklist of “minor syntactic differences” and library differences has been given. The meaningful differences concern the presence or absence of the modifiers virtual, override, and/or new specifically on the methods named Foo in the C# code’s classes Base and Derived, and also whether the return type of the first Foo method in the class Derived is C or is E<Derived>.

The theme of this “mini-project” is different decisions made by the designers of Java and C# on handling these features. Once you have working C# code that complies with the other differences, here are the combinations to try and the observations to make:

1. First compile and run with the base class Foo having signature public C Foo(C arg), and the first Foo method in class Derived having the same. You should edit the text string inside the method to reflect the correct return type. You may note and also ignore a warning to use the keyword new on the latter, though of course you may put in the keyword. Make a table with six rows labeled b.Foo(c),...,d.Foo(ed) and two other calls labeled c1,c2, and columns saying what class object was constructed (following the words I am a/an), what the signature of the called method Foo was, and what class the argument belonged to (again following I am a/an).

2. Then compile and run with the base class Foo marked virtual, and the first method in Derived marked override. Build a similar table, and note the differences. This is the version that you should submit.

3. Then change the return type of the first Foo from C to E<Derived>. Note the compiler error message.

4. Finally, remove override from the code. Make a similar table. Also be sure to make the text string in the first Foo method say the return type is again E<Derived>, not C. Do you see any (other) differences?

Finally, explain in prose the results you obtained, What differences between C# and Java are revealed, in terms of when overriding is legal? (45 pts. for the code, and 45 pts. total for the tables and prose explanations, for 90 pts. in all. This incorporates (1) of “Assignment 5,” which has two other problems; it does not matter if your report is stapled to them or not.)

---

1You may also ignore the warning about the object a never being used, which curiously Java never gives even though my javac5 shell script uses the -Xlint option. (The original “lint” was a program to generate helpful warnings on C code.) Class A is there for future reference, and also in case you comment-out lines or “play with” the code in other ways that cause similar warnings about other objects, you’ll know what they mean.