Project 1: Singleton Objects for Java

CSC 7351, Fall 2012

Due: 2 October 2012

For this project, you will add a singleton object construct to Java and implement it using the Polyglot compiler.

Sometimes, we only want a single object of something. For example, a compiler typically has only one parser, or a real operating system has only one TCP/IP stack (unless it's Windows). A standard way to implement this in an object-oriented language is to define a class that maintains its single instance and ensures that no other instances can be generated. This is called the Singleton pattern.

Add an object construct to Java that looks syntactically similar to a class but defines a single object, e.g.,

```
object O implements I {
    int foo() { return 42; }
}

I p = O;
int i = p.foo();
```

To implement this you need to:

- add a command line option to enable the language extension,
- add the keyword `object` to the lexical analyzer,
- make the parser recognize the object construct,
- add appropriate error checks (e.g., '`new O()`' would be illegal),
- and generate code by transforming the abstract syntax tree.

Since each byte code (`.class`) file corresponds to one Java class, you need to generate code that looks like it came from a regular Java class. The easiest way to generate such code is to transform the parse tree for an object construct into a parse tree for a class that implements the Singleton pattern.

The object construct above is not fully specified. E.g., what does it mean for an object to be defined by inheritance from another object (or class) or to have protected or static methods? Think through these language design issues, justify your design choice, and implement the semantic analysis checks and tree transformation correspondingly.

Submission

Download Polyglot 2.5 from

```
http://www.cs.cornell.edu/projects/polyglot/
```

For building it you also need to download and install Ant and JFlex. On Windows, you will also need Cygwin. For source code version control, I recommed you use Git or CVS.

To submit your projejct, upload the code to your `classes.csc.lsu.edu` account in a directory `prog1` and submit it using

```
~cs7351_bau/bin/p_copy 1
```

Please provide a README file (as text, PDF, or Word) as well as test cases with your submission. In the README file document and justify your language design choices, explain the architecture of your code, and explain what's working and what's not yet implemented. Write test cases to highlight the details of your language design.