```java
// Example 8.3 joi/examples/ReflectionDemo.java
//
//
// Copyright 2003 Bill Campbell and Ethan Bolker

import java.lang.reflect.*;

// A short program to illustrate how Java uses
// class information dynamically.
//
// This file declares class Greeting as well as class
// ReflectionDemo. Java requires that a public class be
// declared in a file that matches its name, but Greeting
// is not marked public.
//
// %> java ReflectionDemo
// Greeting@93dee9 is an instance of class Greeting
// classOfG.toString(): class Greeting
// classOfG.getName():  Greeting
// fields in class Greeting (not inherited):
// name: message, type: class java.lang.String
// methods in class Greeting (not inherited):
// invoking hello
// hello, world!
// Creating an object when you know the name of its class:
// g = (Greeting)Class.forName("Greeting").newInstance();
// g.toString(): Greeting@6f0472
// Try to create an instance of nonexistent class Foo:
// java.lang.ClassNotFoundException: Foo

public class ReflectionDemo
{
    public static void main( String[] args )
    {
        Greeting g = new Greeting();
        Class classOfG = g.getClass();
        out(g.toString() + " is an instance of " +
                                classOfG.toString());
        out("classOfG.toString(): " + classOfG.toString());
        out("classOfG.getName():  " + classOfG.getName());

        out("fields in class Greeting (not inherited):");

        Field[] greetingFields = classOfG.getFields();
        for (int i=0; i < greetingFields.length; i++) {
            Field f = greetingFields[i];
            if (f.getDeclaringClass() == classOfG) {
                out("name: " + f.getName() + ", type: " + f.getType());
            }
        }

        out("methods in class Greeting (not inherited):");

        Method[] greetingMethods = classOfG.getMethods();
        for (int i=0; i < greetingMethods.length; i++) {
            Method m = greetingMethods[i];
```

```java
            if (m.getDeclaringClass() == classOfG) {
                out("invoking " + m.getName());
                try {
                    m.invoke(g, null);
                }
                catch( Exception e) {
                    out(e.toString());
                }
            }
        }

        out("Creating an object when you know the name of its class:");
        out("g = (Greeting)Class.forName(\"Greeting\").newInstance();");
        try {
            g = (Greeting)Class.forName("Greeting").newInstance();
            out( "g.toString(): " + g.toString());
        }
        catch (Exception e) { // couldn’t find class
            out(e.toString());
        }

        out("Try to create an instance of nonexistent class Foo:");
        Object o;
        try {
            o = Class.forName("Foo").newInstance();
        }
        catch (Exception e) { // couldn’t find class
            out(e.toString());
        }
    }

    // too lazy to type "System.out.println()
    public static void out( String s )
    {
        System.out.println(s);
    }
}

class Greeting
{
    public String message = "hello, world";

    public void hello()
    {
        System.out.println(message + "!");
    }
}
```