Return to Module Overview Page (https://onlinelearning.berkeley.edu/courses/665040/pages/module-ten)

10.5: Methods

You can even call an arbitrary method on an object using the Reflection API. The class java.lang.reflect.Method is the key to this:

```
public class Foo
{
     public String s(){ return "A String";}
}
Object o = new Foo();

... (pass o to a fnx that doesn't know about Foo)

Class c = o.getClass();
Method[] m = c.getMethods();
int i = 0;
// See Note #1 below
while(!m[i].getName().equals("s"))
   i++;
System.out.println(m[i].getReturnType().getName());
System.out.println(m[i].getName());
System.out.println(m[i].invoke(o, null)); // Note #2
```

(prints)

```
java.lang.String
s
A String
```

Note #1: m[ ] is a fairly large array, because it contains all the methods either defined by or inherited by Foo. Foo inherits all the methods of java.lang.Object, so we have to search until we find s( ).

Note #2: The second argument to invoke is an array of Object; it should contain the arguments to pass to the function being invoked.