

Advanced Programming

“DebuggerCLI”

Group 23:

Nuno Cameira	69769
Iryna Shvydyuk	70352
Simone Griesmayr	82703

2nd Semester
2014/2015

Project Base
“DebuggerCLI”

DebuggerCLI

MyTranslator

InspectionObject

DebuggerCLI

- It stores the *CallStack*, the *LastObject* and the *thrownException*
- It provides the functionality for the shell and a “*run*” method that is used for the method invocation in *MyTranslator*

MyTranslator

- It contains the implementation of our expression editor
- This is used to inject code to the methods
- The expression editor edits each method call by the following methods of the DebuggerCLI:
 - `setLastObject;`
 - `addToStack;`
 - `run.`

ExprEditor

```
for(CtMethod ctm : ctmethods){
    ctm.instrument(new ExprEditor() {
        public void edit(MethodCall m) throws ...{
            try {
                m.replace(" ... setLastObject();
                ... addToStack();
                ... Object o = run();
                ...if(o instanceof Exception){
                    throw o;
                } else {
                    $_ = ($r)o;
                }")
            } catch(...){....}
        }
    })
}
```

Run()

```
try {
    return invokeMethod();
} catch {
    Object o = startShell();
}
...
```

InspectionObject

- Handles the invocation of static and non-static methods
- Uses java Reflection API for getting and setting fields

Problems

- First implementation with *addCatch()*
- Filtering the packages
 - `$class` is used in the implementation of our Expression Editor

Questions?