Debugging Domain-Specific Modelling to Android

Hakan Ozler (20124419)

Overview

- ❖ Introduction
- ❖ Debugging Tool
- ❖ Debugging DSM
- ❖ Meta Models
- ❖ Android
- ❖ Aspect Oriented Programming
- ❖ Socket Programming
- ❖ Simple Debugging Example
- ❖ Future Work

Universiteit Antwerpen

Why do we need to debug?

- We want to know where the bug occurs
- Debugging provides different techniques
- Easily check with debugging tool
  - E.g : breakpoint

Universiteit Antwerpen

Breakpoint

- A *breakpoint* is a signal that tells the debugger to temporarily suspend execution of your program at a certain point
- Can be set as disable or enable without having change your program's source code
- Stop / Resume mode
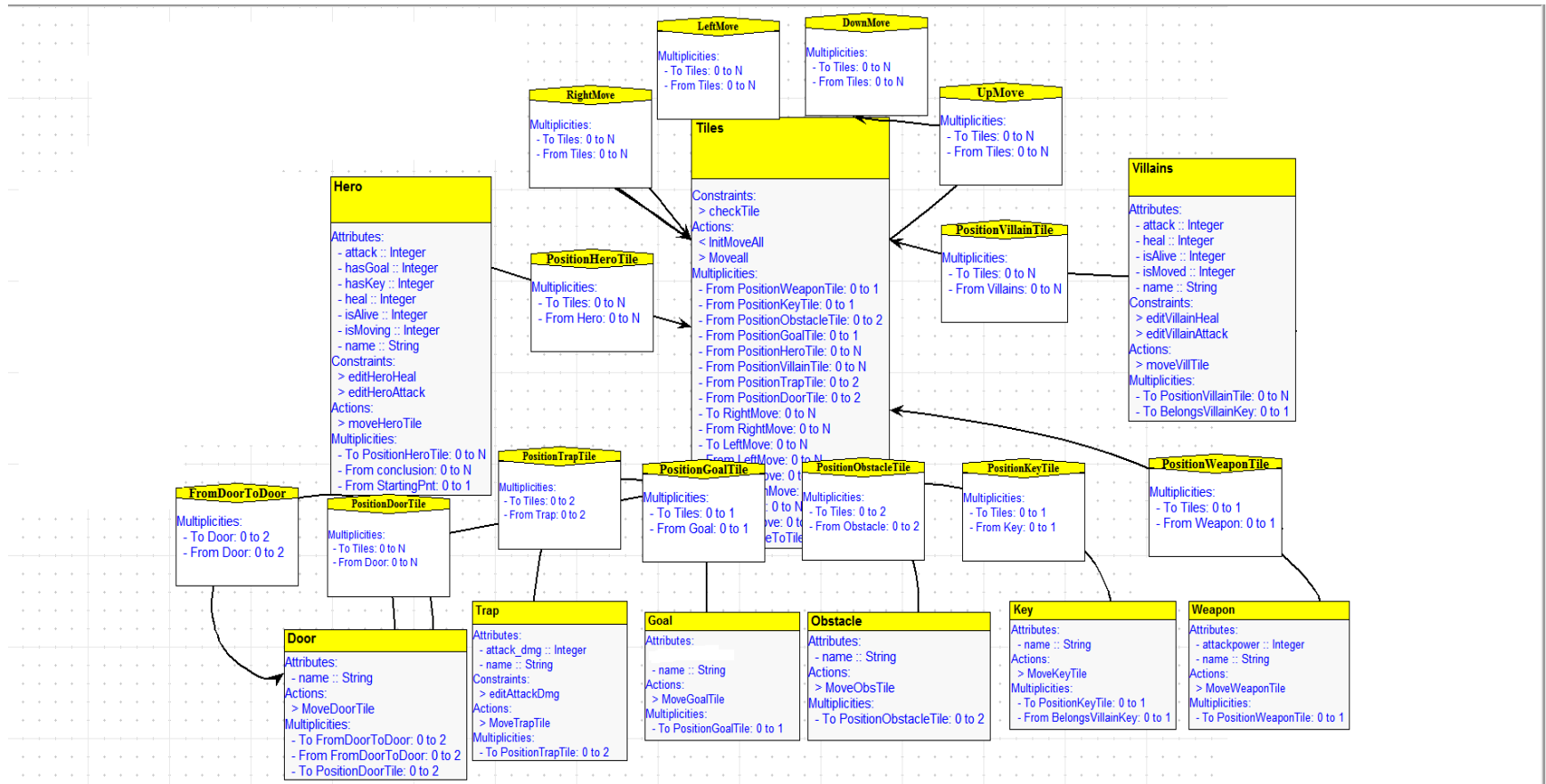- Provide a powerful tool where you need

Universiteit Antwerpen

Debugging DSM

- DSM permits a special role for each debugging facility
- It is essential to make a transform from graphical entities to the code
- DSM is providing to set breakpoint into the domain-specific code,
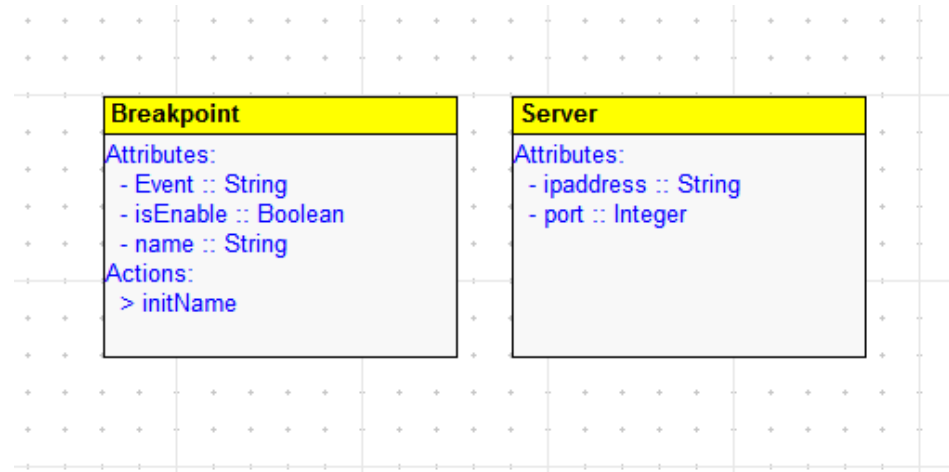   for instance → pause/resume execution

Meta Model RPG Game

Meta Model Breakpoint

Android Application

- Android Manifest → provides user permissions, activity, service, provider elements and presents important information

- E.g: <uses-permission android:name=*"android.permission.INTERNET" />*
 <uses-permission android:name=*"android.permission.ACCESS_WIFI_STATE" />*
  <uses-permission android:name=*"android.permission.ACCESS_NETWORK_STATE" />*

- Layout is typically a xml file → presents the visual structure for user interface

Universiteit Antwerpen

Aspect Oriented Programming

- AOP languages typically extend object-oriented languages.
- AOP provides to separate each concern
- AspectJ is an extension of Java.
- AspectJ is an implementation of aspect-oriented programming (AOP) for Java.
- It is a new way to improve the separation of concerns of programs.
- There are several new constructs which are pointcuts, advice and aspects

- The aspect oriented programming tries to solve two main problems basically

- code tangling
- code scattering

MainActivity
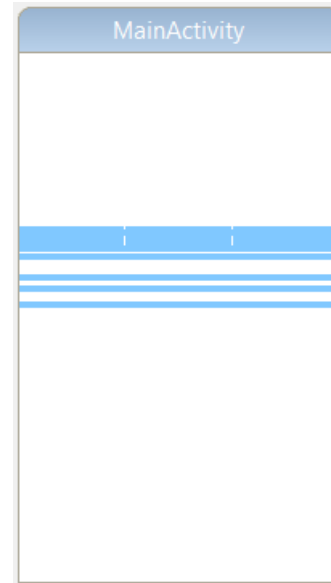
Aspect Oriented Programming(Cont.)

Main Aspect constructs;

- Join point: some point in the execution of an application
- Pointcut: a set of logically related join points
- Advice: some behavior that should become active whenever a join point is encountered

Aspect Oriented Programming(Cont.)

```
205    pointcut ObserverGetsObject(RPGame rpg, object obj) :
206            (call(* RPGame.get*(..)) || call(* RPGame.dmgTrap(..))) && withincode(* RPGame.run()) && this(rpg) && args(obj);
207
208    after(RPGame rpg, object obj) returning: ObserverGetsObject(rpg,obj){
209        for (BreakPoint ob : rpg.breakpoints) {
210            if (ob.getEnable() && rpg.Xin.existBreakPoint(ob)
211                    && obj.existBreakPoint(ob)) {
212                Message msg = MainActivity.myHandler.obtainMessage(1,"Breakpoint is triggered! -> " + ob.getEvent());
213                MainActivity.myHandler.sendMessage(msg);
214                run(ob.getEvent(), rpg);
215                Message msg2 = MainActivity.myHandler.obtainMessage(1, "Pause");
216                MainActivity.myHandler.sendMessage(msg2);
217                rpg.onPause();
218            }
219        }
220    }
```

    {…} RPGame: method-call(void mde.project.RPGame.getKey(mde.project.object))
    {…} RPGame: method-call(void mde.project.RPGame.getWeapon(mde.project.object))
    {…} RPGame: method-call(void mde.project.RPGame.getGoal(mde.project.object))
    {…} RPGame: method-call(void mde.project.RPGame.dmgTrap(mde.project.Trap))

Universiteit Antwerpen

Socket Programming in Python

- A socket is an endpoint of a two-way communication link between two programs running on the network

- two basic sockets constructs
  - *Socket*, provides standard low-level networking interface
  - *SocketServer*, provides classes that simplify the development of network servers

```
sock = socket.socket ( socket.AF INET , socket.SOCK STREAM )
sock.bind( ( ' ', 2525) )
sock.listen( 5 )
newsocket, (host , port) = sock.accept()
```

Socket Programming in Java

- two key classes from the java.net
  - Socket
  - ServerSocket

```
Socket s = new Socket ( MainActivity.ipaddress , MainActivity.portn );
PrintWriter outp = new PrintWriter ( s.getOutputStream() , true);
outp.println (event);
s.close ()
```

Simple Debugging Example

Simple Debugging Example(Cont.)

Universiteit Antwerpen

Simple Debugging Example(Cont.)

Universiteit Antwerpen

Simple Debugging Example(Cont.)

Universiteit Antwerpen

Simple Debugging Example(Cont.)

Simple Debugging Example(Cont.)

Universiteit Antwerpen
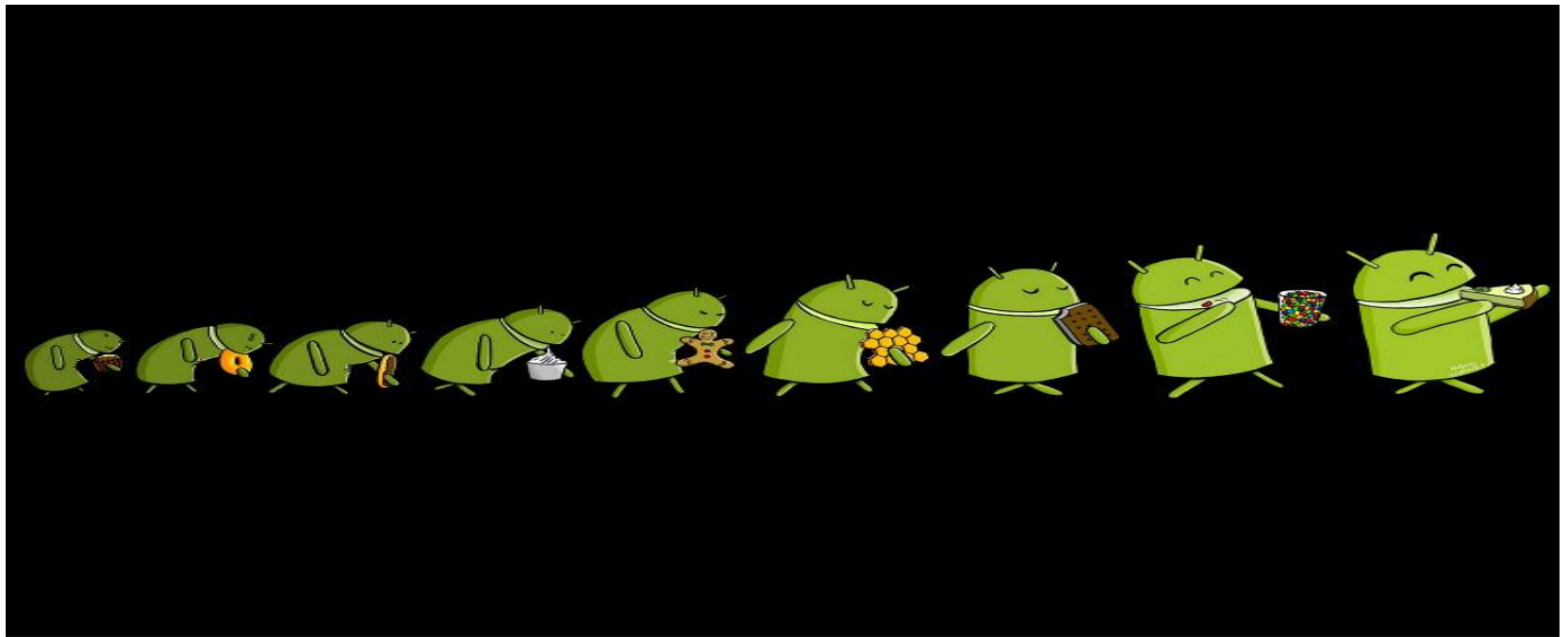
Simple Debugging Example(Cont.)

Future Work

- It can be made more complex breakpoint models
- It can be made when incoming message comes then Atom^3 send an ack message to android devices
- It can be created to send a message step-by-step for each execution

Thank you for your attention!

References

1. AToM3 Programming WebSite

    i.   http://atom3.cs.mcgill.ca/people/jlara/AToM3Programming/index.dtml

2. Aspect Oriented Programing

    i.   http://www.eclipse.org/aspectj/doc/next/progguide/index.html

3. Aspect Oriented Android Development

    i.   http://deansserver.co.uk/ dean/2011/07/18/aspect-oriented-android-
          development-tool-integration/

4. Debugging in Domain-Specific Modelling, Raphael Mannadiar, Hans
   Vangheluwe

5. AspectJ Development Tools

    i.   http://www.eclipse.org/ajdt/

6. Sockets programming in Python M. Tim Jones,

    i.   Sockets programming in Python,04 Oct 2005

Universiteit Antwerpen

Demo & Questions?

Universiteit Antwerpen