

SWAN for Android Apps

Introduction

Swan is a framework which makes it easy to build context aware applications for the Android platform. It is designed to run as a background service in the phone, being accessed by any other application through its API. This tutorial explains the installation and usage of Swan app on Android devices.

Prerequisites: A working Android development tool installed. Check Android developer tools guide for more information (<http://developer.android.com/sdk/index.html>)

Setting up the project in Eclipse

Current version of Swan is available at https://github.com/nadinasovaiala/swan_sense_integration. It contains two projects, both needed for a complete installation of swan.

- interdroid-swan-project, is the updated version of the Swan framework
- sense-android-library, is the slightly modified Sense library developed by Common Sense (check <http://developer.sense-os.nl/CommonSense/>)

Dependencies:

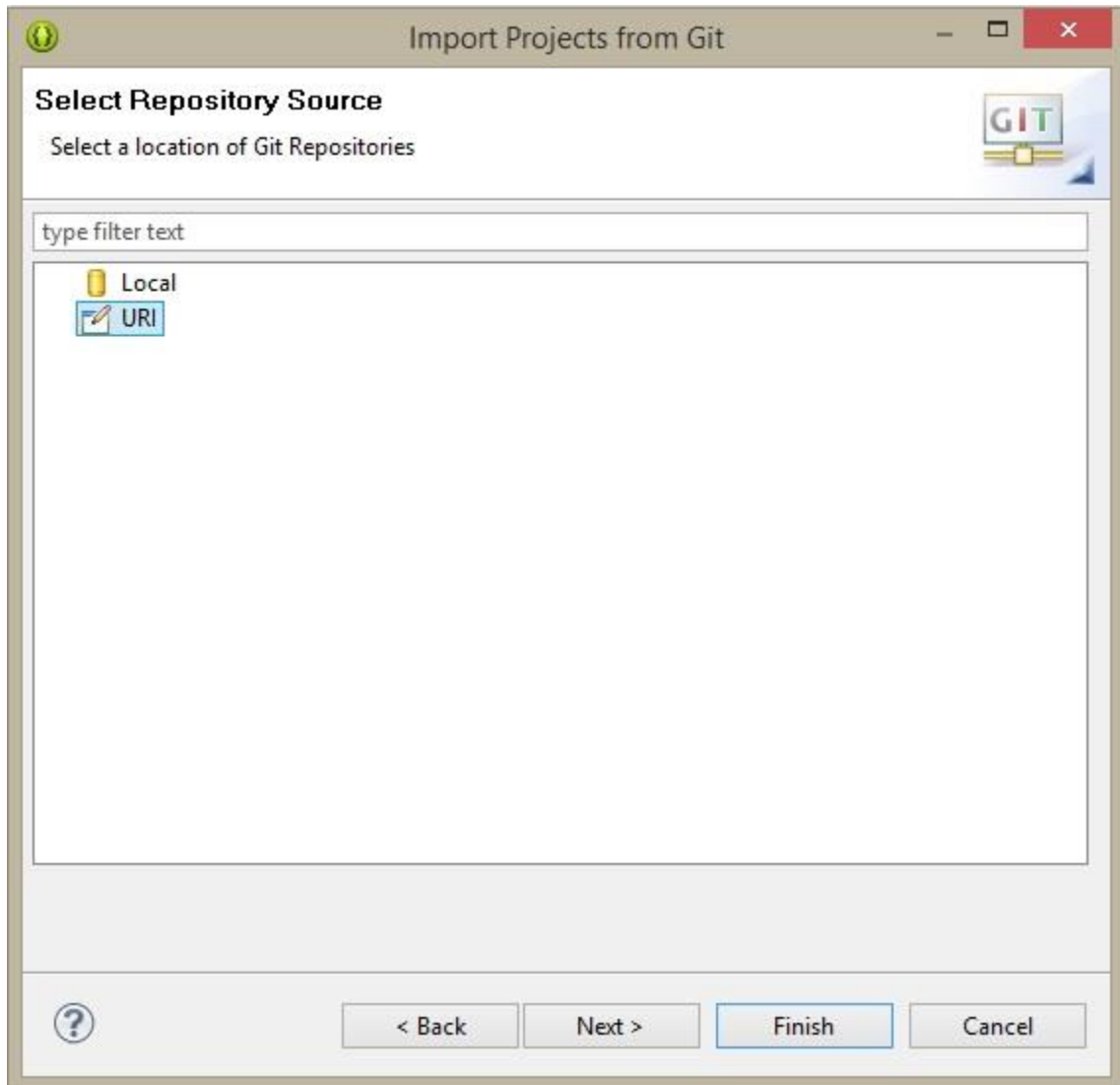
- google-play-services (check <https://developer.android.com/google/play-services/setup.html> to see how to add Google Play services to your project)
- sense-android-library

Import using EGit

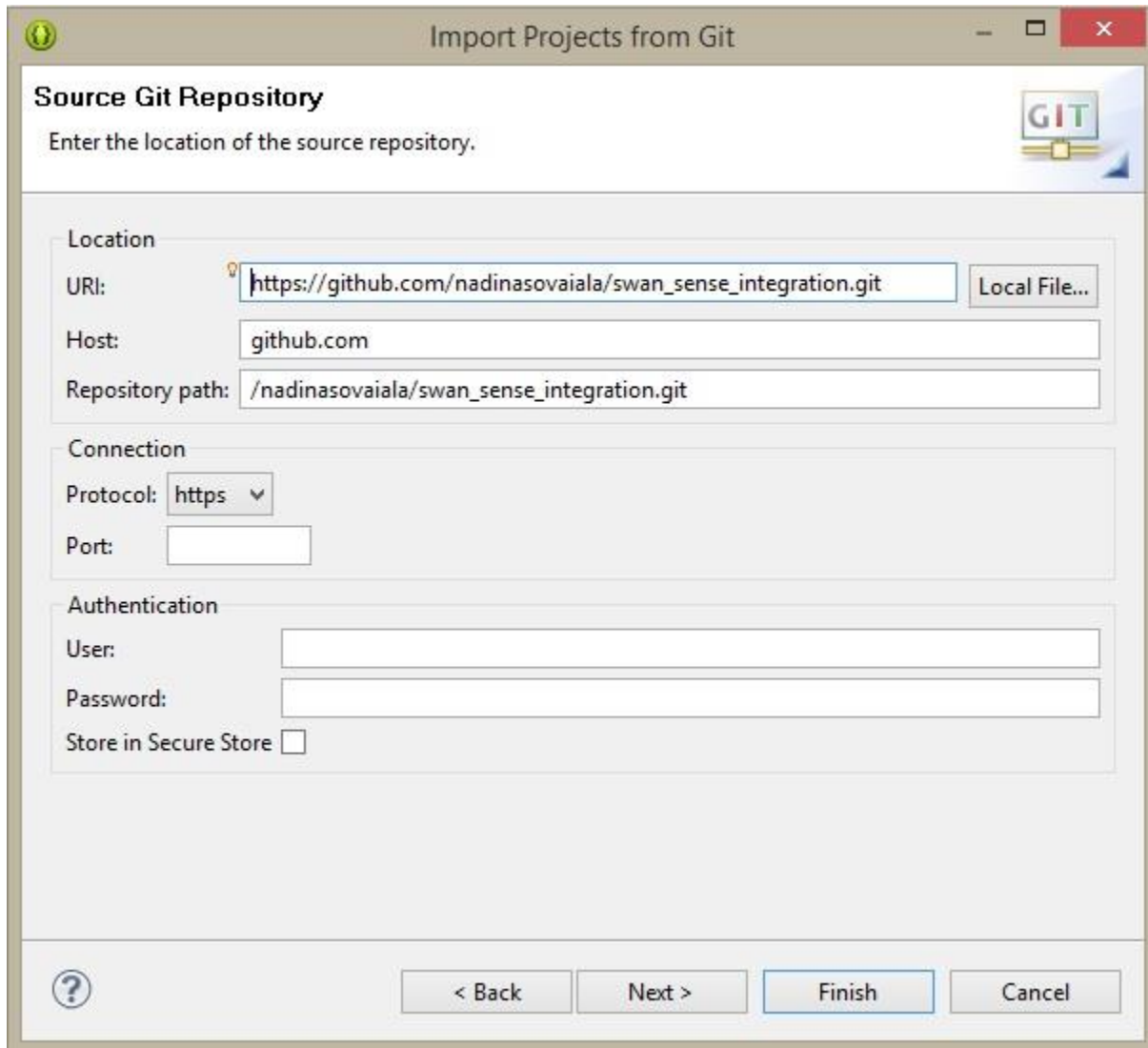
1. Open File -> Import. Select *Projects from Git*



2. Select URI as your repository resource



3. Select Git repository – add https://github.com/nadinasovaiala/swan_sense_integration.git as URI and the rest of the fields will be automatically filled



Import Projects from Git

Source Git Repository

Enter the location of the source repository.

Location

URI:

Host:

Repository path:

Connection

Protocol: ▼

Port:

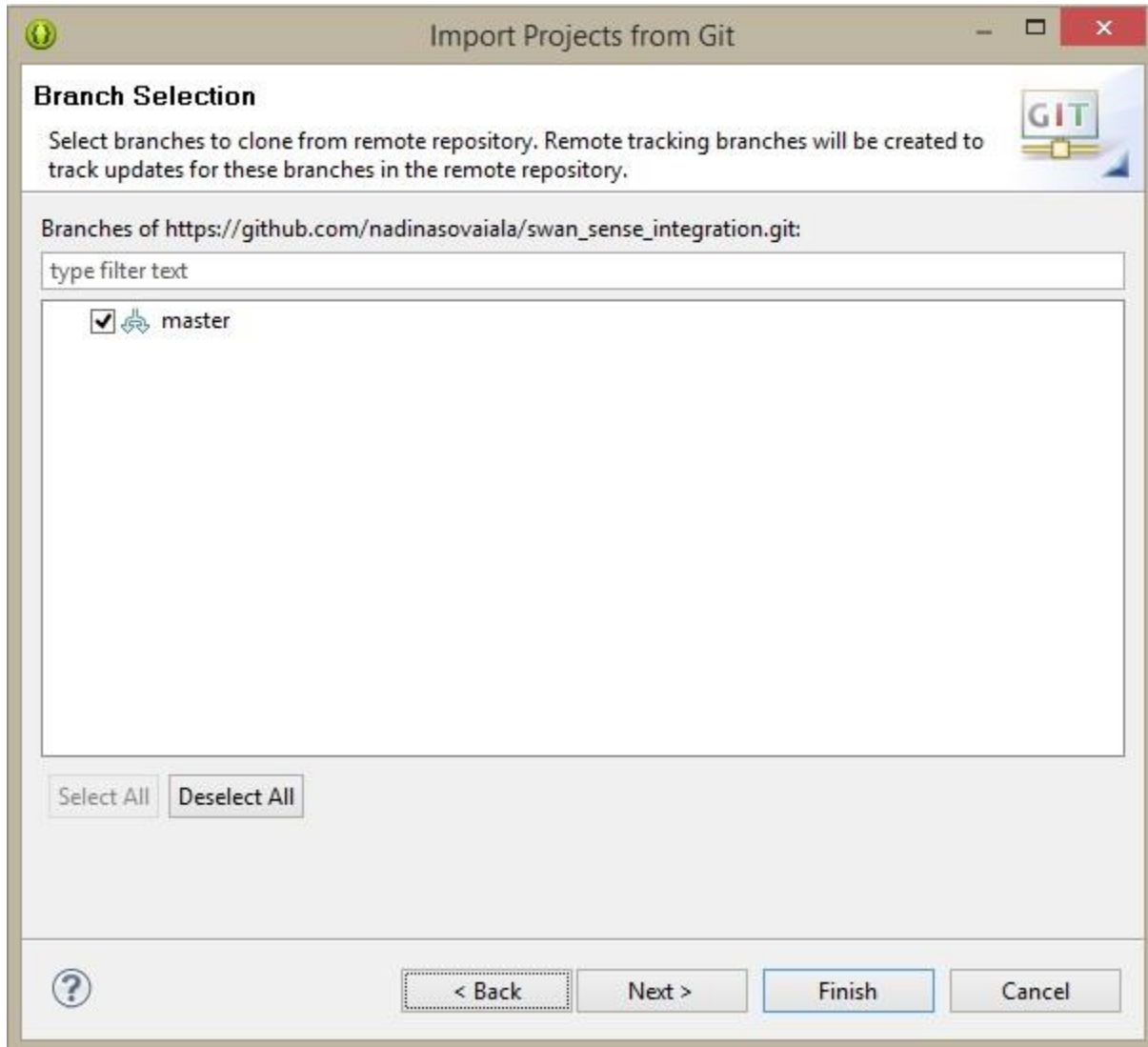
Authentication

User:

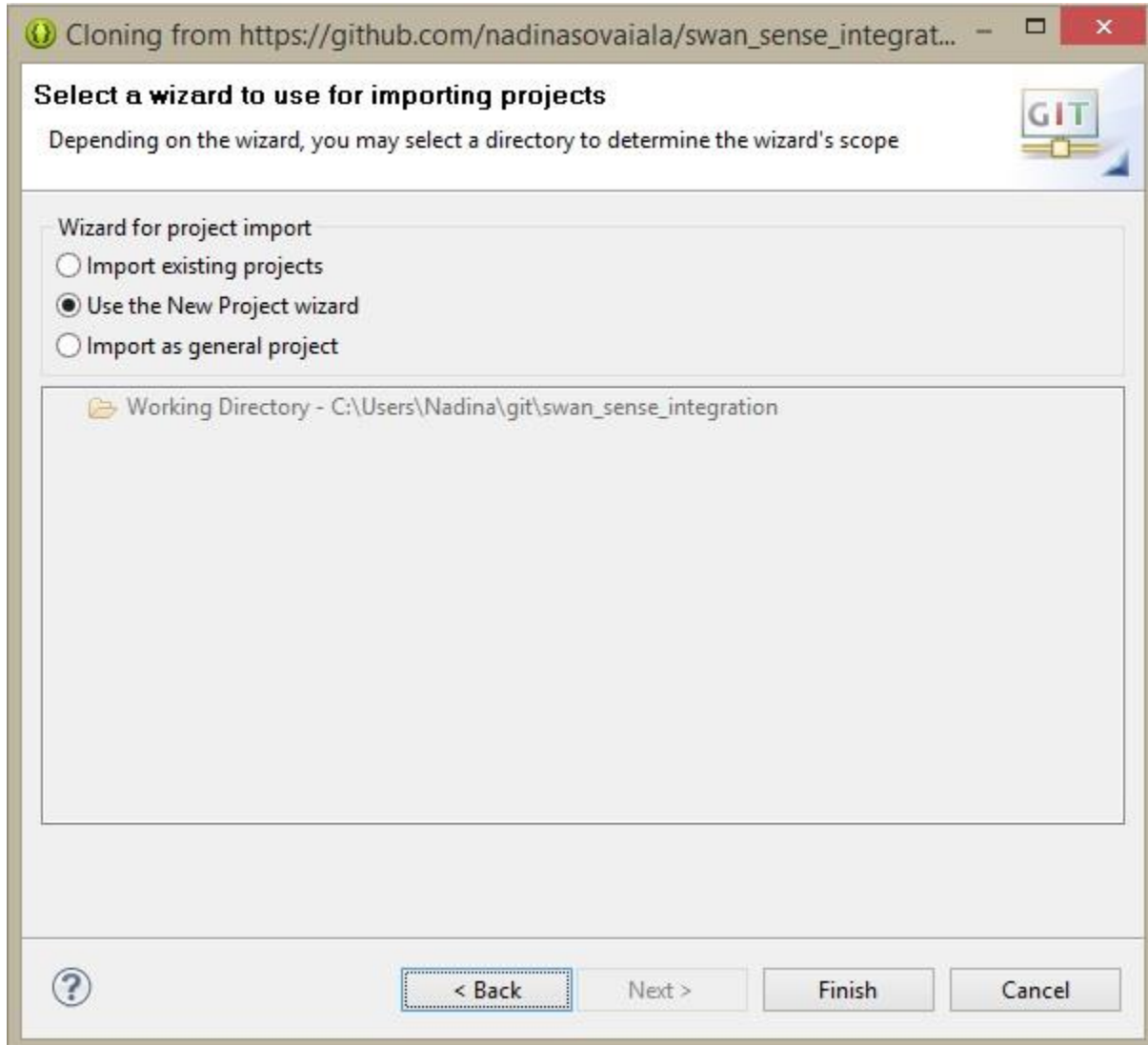
Password:

Store in Secure Store ☐

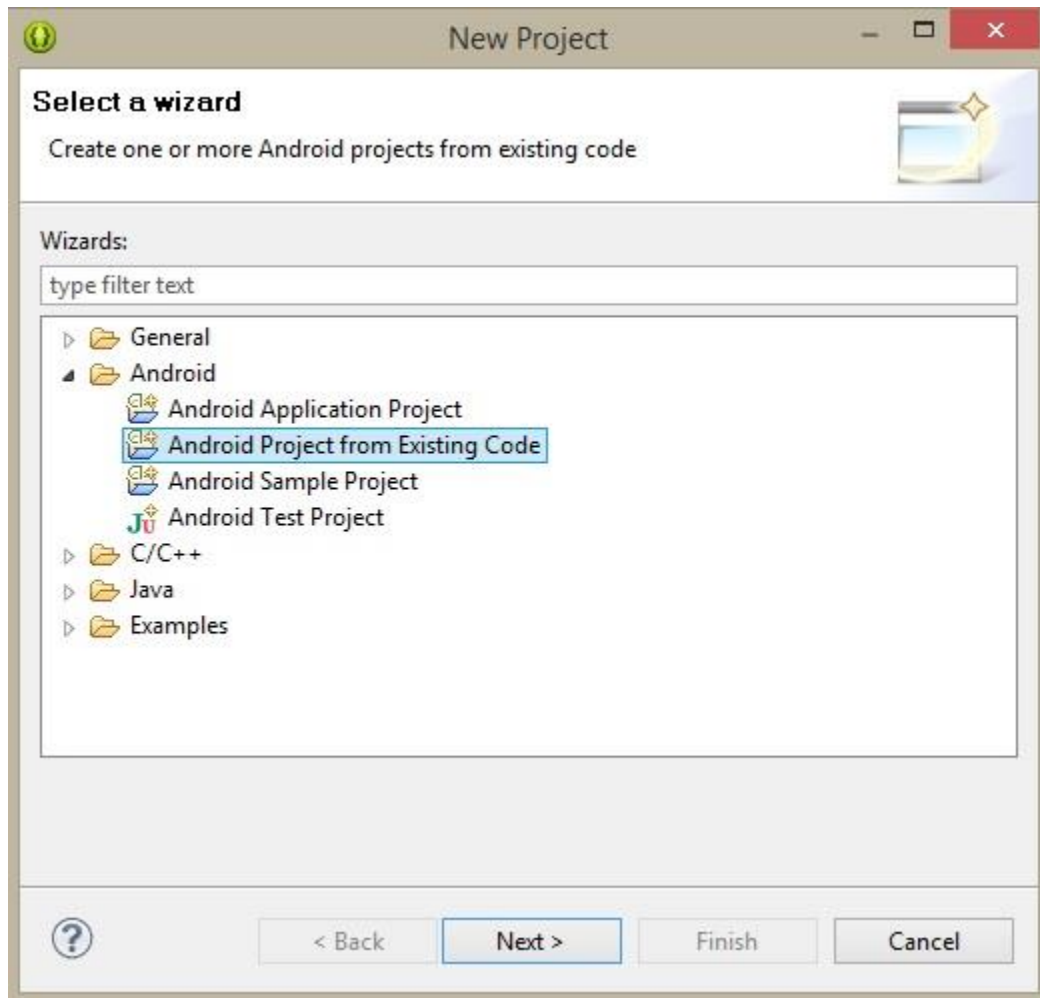
4. Select a branch (usually *masters*, as it contains the most stable and up to date version of the app)



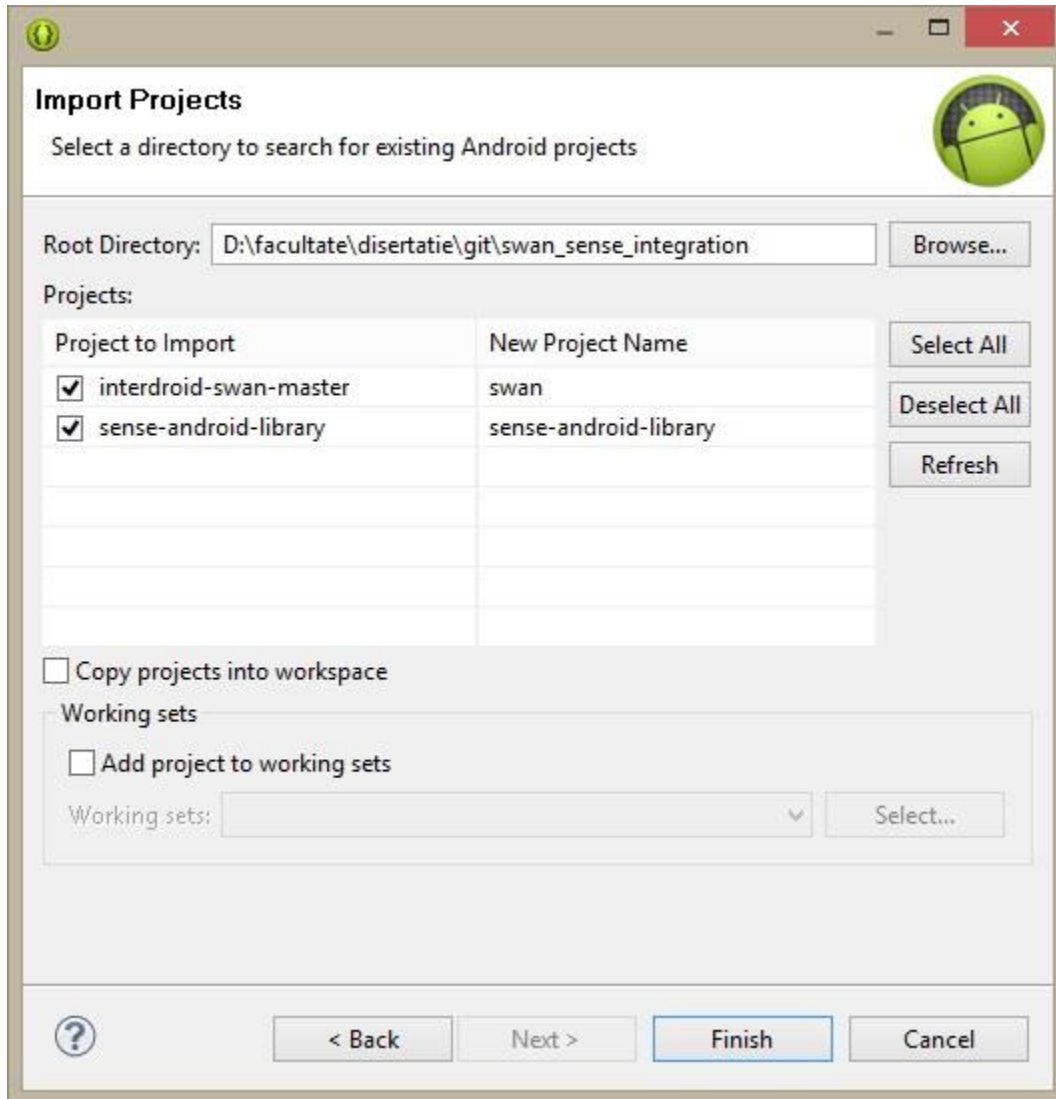
5. Enter the location where you want to clone the git repository and select *Use the New Project Wizard*



6. Select *Android project from existing code*



7. The wizard should find two projects: interdroid-swan-master and sense-android-library. Select both and press *Finish*. Swan project will be imported with the name "swan".



Import Projects

Select a directory to search for existing Android projects

Root Directory:

Projects:

Project to Import	New Project Name
<input checked="" type="checkbox"/> interdroid-swan-master	swan
<input checked="" type="checkbox"/> sense-android-library	sense-android-library

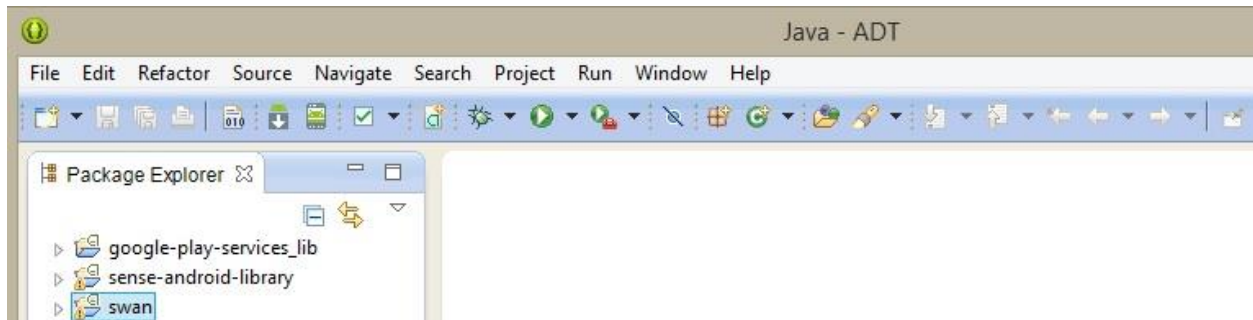
☐ Copy projects into workspace

Working sets

☐ Add project to working sets

Working sets:

8. Import google-play-service project and add it as a reference to swan. The package explorer should look like this:



Import using other Git clients

Clone the source code from GitHub repository. In Eclipse, start the new project wizard using 'File -> New -> Project'. Continue by following the EGit tutorial starting at step 6.

Adding sensors in Swan

New sensors can be easily plugged in using the Service Provider Interface. Internally, swan sensors are background services that gather data from a particular resource, such as hardware sensor, Bluetooth manager, or even external (online) sources like news podcasts or train schedules.

Sensors implementations are in the “interdroid.swan.sensors.impl” package. Any new sensor object should be added in this package and it must extend the `AbstractCloudSensor` base class. After overriding the methods from the base class, the new sensor class should also include a Configuration Activity class containing a XML Android Preference interface (which should be added in `res/xml` folder). This allows the users of swan-based app to properly configure the sensors from the phone.

Because a swan sensor is actually a service containing an activity, both the service and the activity need to be declared in the manifest file. The service should contain the “exported” attribute set to true (allows other apps to use the service) and the “process” attribute should be a unique name with a colon “:” at the beginning, setting the service to run in a private process, separate from the default application. The ConfigurationActivity must declare an intent-filter with the “interdroid.swan.sensor.DISCOVER” action. In this way it will be triggered every time the swan-based app sends a broadcast with this message. In addition, it should include any configuration properties for the user’s choice, like value paths and units. Don’t forget to add the required permissions.

How to register expressions

In order to get Swan running, the app must register a Swan-Song expression. *ExpressionManager* class offers a public API for registering and unregistering expressions, as well as querying for existing sensors. Once an expression is registered, SWAN will notify the app by sending a broadcast message every time the result of the expression evaluation changes.

There are two possibilities of retrieving the results: registering a broadcast receiver with action intent “EXTRA_NEW_VALUE” or implementing a listener and passing it as the 4th parameter in the register method, similar for both types of expressions (value or tristate):

```
ExpressionManager.registerValueExpression(this, id, (ValueExpression)
ExpressionFactory.parse(expression), new ValueExpressionListener() {

    @Override
    public void onNewValues(String id, TimestampedValue[] arg1) {
    }
});
```

If the second approach is used, the *ExpressionManager* will handle receiving the broadcasted message with the new values and forwarding it to the app by calling the associated listener.

Installing SWAN on your smartphone

After a successful build, Swan can be installed in your phone as any other app. This will trigger the Swan Lake Activity, where the user can choose to pair the device with other devices. Also, if it is the first time swan is opened, the app will ask to login to Common Sense platform. To create an account, follow these steps: <https://accounts.sense-os.nl/>

Testing

For testing Swan, use Swan Monitor app, available at <https://github.com/interdroid/swan-monitor>. It shows all the values of all available sensors.

Troubleshooting

A possible problem that may occur when building swan is:

[Dex Loader] Unable to execute dex: Multiple dex files define Linterdroid/swan/swansong/Parseable;

[SwanLakeActivity] Conversion to Dalvik format failed: Unable to execute dex: Multiple dex files define

Linterdroid/swan/swansong/Parseable;

The problem is that two ant files (buildgrammar and buildjar) are not being built. Two possible solutions (hacked) are cleaning the project or copying the generated files of swansong package in the swan project.

For troubleshooting with the sense library, check Common Sense website.

Contact

More information can be found on the Interdroid project website: <http://interdroid.net>

The latest source repository tree is accessible through Git at: `git@github.com:interdroid/interdroid-swan.git`

You can send bug reports, feature requests, cries for help, or descriptions of interesting way in which you have used this project to: `palmer at cs.vu.nl`

This software has been developed as part of the Interdroid project, a software project of the Computer Systems group of the Computer Science department of the Faculty of Sciences at the Vrije Universiteit, Amsterdam, The Netherlands. The main goal of the Interdroid project is to create distributed middleware for mobile systems.

This is free software. See the file "LICENSE.txt" for copying permissions.

This software contains components from several other open source projects.

Jars from these projects are stored in the "libs" directory along with licenses for each of these projects.