OOC Documentation Documentation

Release 0

Paul Schmitt and Morgan Vigil

June 03, 2013

CONTENTS

Contents:

CHAPTER

ONE

INTRODUCTION

OpportunisticOwnCloud (OOC) is an Android application designed to interface with WebDAV servers in contexts with variable connectivity. For this particular project, we assume that the application will be interfacing with an instance of an ownCloud server, but this does not necessarily need to be the case (although we have not tested with alternative WebDAV servers).

ownCloud is an open source technology that provides cloud ownership and management tools. The technology was designed to be used on a small, residential scale for file storage and sharing. However, the http://moment.cs.ucsb.edu/?q=content/villageshare-localized-network-architecture project has developed modules for ownCloud that allow for its sharing properties to extend over multiple ownCloud instances over wide area networks. While ownCloud currently has an Android application, it is designed around the paradigm of ownCloud operating as a single instance for users with easy access to network connectivity. We designed OOC in hopes that it might act as a complementary technology to the VillageShare use of ownCloud instances–that is, facilitating the sharing and storage of locally generated content in communities with limited resources.

1.1 Future of OOC

This application began as a project for the Modern Programming Languages and Implementations class at UCSB. We intend to incorporate this opportunistic model for the deployed VillageShare version of the ownCloud Android application. Presently, we are not certain if this will entail modifying the original ownCloud Android application or incorporating more of the functionality into OpportunisticOwnCloud.

1.2 Intstalling OOC vs. Contributing to OOC

While OOC should work with any WebDAV server, we have only tested it using ownCloud servers. To install and configure your own lightweight instance of ownCloud, see https://mvigil-cs263-technology-project.readthedocs.org/en/latest/ and http://owncloud.org/install/.

For those readers interested in installing OOC, please refer to the Phone page before going to the Installation section.

For reasers interested in contributing to OOC, please refer to the Android section before proceeding to Installation section.

CHAPTER

TWO

REQUIREMENTS FOR OPPORTUNISTICOWNCLOUD

2.1 Phone

We have developed OpportunisticOwnCloud to target Android SDK 17 (Jelly Bean). The minimum requirement is Android SDK 8 (Froyo). We have run OOC successfully on systems running Froyo and Gingerbread.

2.2 WebDAV Server

While OOC should be able to interface with any WebDAV server using the Sardine library, we recommend installing and running an ownCloud server instance to use with OOC. See <*http://owncloud.org/install/*>.

2.3 Android

DEVELOPERS ONLY

In order to contribute to OOC, you must have Android ADT. We recommend downloading it from http://developer.android.com/sdk/index.html as this provides the entire Android SDK, including Eclipse with ADT plugins.

CHAPTER

THREE

SETUP

The setup assumes you have the Android ADT and Android SDK installed.

3.1 From the code

In the terminal, enter:

    git clone git://github.com/mvigil90/OpportunisticOwnCloud.git

Open Eclipse with ADT support.

1. Go to 'File'>'Import'.
2. Select 'Android'>'Existing Code into Workspace'.
3. Browse to /path/to/OpportunisticOwnCloud263/actionbarsherlock.
4. Again, select 'Android'>'Existing Code into Workspace'.
5. Browse to /path/to/OpportunisticOwnCloud263
6. Now, right-click the OpportunisticOwnCloud project in the Eclipse Package Explorer.
7. Select 'Properties'
8. In the Android tab, ensure that actionbarsherlock has been selected as a Library
9. Ensure that simple-xml-2.6.2.jar is in the libs/ folder of the OpportunisticOwnCloud project.

At this point, you should be able to run the application on the emulator or on a device. If you are using an emulator, make sure you have created one and explicitly set the RAM size (512 MB works fine). Otherwise, the sdcard is mounted as Read-only and the application will not work.

CHAPTER

FOUR

INSTALLATION

In order to install, you must have Android ADT installed.

From Eclipse: #. Right click the OpportunisticOwnCloud project #. Select ‘Run’>’Run as Android Application’ #. If an emulator has not yet been created, create an emulated device with a target SDK of 17 and specify 512 MB of RAM. If you are using a device, your device should automatically appear as a device option in Eclipse. you can verify its identify using the shell command:

```
adb devices
```

OpportunisticOwnCloud should now be running on your emulator or device.

CHAPTER

FIVE

CLASSES USED IN THE OPPORTUNISTICOWNCLOUD APPLICATION

Here we describe the classes used in the OpportunisticOwnCloud application.

5.1 ConnectivityChangeReceiver

The ConnectivtyChangeReceive class is responsinle for listeining for network events and creating Intent classes to handle the network events. This class has a lifetime that endures throughout the application lifetime.

Extends: BroadcastReceiver Implements: NONE

Public functions:

void onReceive(Context, Intent)

Sub-Classes: NONE

5.2 SyncService

The SyncService is created from the ConnectivityChangeReceive class when a network event is detected.

Extends: IntentService Implements: NONE

Public functions: NONE

Sub-Classes:

mThread

- Extends: Thread
- Implements: NONE
- Public functions:

  void run()

DownLoadThread

- Extends: NONE

- Implements: Runnable
- Public functions:

    void run()

UploadThread

- Extends: NONE
- Implements: Runnable
- Public functions:

    void run()

5.3 MainActivity

The MainActivity class is primarily responsible for managing the GUI aspect of OpportunisticOwnCloud. It creates the LocalFileFragmentTab and the RemoteFileFragmetnTab instances. It also starts the SettingsActivity if the user has never set OpportunisticOwnCloud settings on the phone prior to running the applications. MainActivity is launched by the application when it is first opened.

Extends: SherlockFragmentActivity Implements: NONE

Public functions:

void onCreate(Bundle) void onPause() void onResume() boolean onCreateOptionsMenu(Menu) boolean onOptionsItemSelected(MenuItem)

Sub-Classes:

mThread

- Extends: Thread
- Implements: NONE
- Public functions:

    void run()

5.4 LocalFileFragmentTab

The LocalFileFragmentTab class is responsible for displaying the files located on the user's ownCloud directory on /sdcard. It provides navigation functionality for file browsing as well as on-demand upload functionality.

Extends: SherlockFragment Implements: ActionBar.TabListener

Public functions:

void onCreate(Bundle) void onTabSelected(Tab, FragmentTransaction) void onTabUnselected(Tab, FragmentTransaction) void onTabReselected(Tab, FragmentTransaction)

Sub-Classes:

mThread

- Extends: Thread
- Implements: NONE
- Public functions:

void run()

UploadThread

- Extends: NONE
- Implements: Runnable
- Public functions:

  void run()

5.5 RemoteFileFragmentTab

The RemoteFileFragmentTab class is responsible for displaying the user's files located on the ownCloud server. It provides navigation functionality for file browsing as well as on-demand download functionality.

Extends: SherlockFragment Implements: ActionBar.TabListener

Public functions:

void onCreate(Bundle) void onTabSelected(Tab, FragmentTransaction) void onTabUnselected(Tab, FragmentTransaction) void onTabReselected(Tab, FragmentTransaction)

Sub-Classes:

mThread

- Extends: Thread
- Implements: NONE
- Public functions:

  void run()

DownloadThread

- Extends: NONE
- Implements: Runnable
- Public functions:

  void run()

5.6 SettingsActivity

The SettingsActivity provides a simple interface for the user to set global settings, including username, password, and URL information for connecting with the ownCloud server as well as setting sync frequency and toggling opportunistic synchronization on and off. The data object created is used globally to provide necessary settings information to the other classes.

Extends: PreferenceActivity Implements: NONE

Public functions: NONE

Sub-Classes: NONE

CHAPTER

SIX

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*