University at Buffalo The State University of New York | REACHING OTHERS

Creating a more energy aware integrated development environment (IDE) for smartphones

Edward Poon, Advisor: Dr. Steven Ko
The University at Buffalo Department of Computer Science and Engineering

Abstract

With the advent of increased computing on mobile devices such as phones and tablets, it has become crucial to pay attention to the energy consumption of mobile applications. The software engineering field is now faced with a whole new spectrum of energy-related challenges, ranging from power budgeting to test and debugging the energy consumption. To that end, we are creating a plugin for the Eclipse IDE that helps profile the energy consumption of each app on the Android Phone. This will help developers produce more energy efficient apps which in turn lead to a longer lasting battery.

Purpose of the Plugin

Create a plugin for an integrated development environment called Eclipse. The plugin will utilize the battery settings from the phone to determine how much of each application uses the phone's life.

This is the battery settings which will be used to utilize the application's power consumption.

Power Profiling Issues Of App Developers

- A plugin that checks the power consumption of the application during the testing stages is non existent.
- Other power profilers exist but they require disassembly of the phone.
- There are apps that check the power draw of the other apps but loading the app into the phone and testing it is very time consuming.

Motivation

- More people are using mobile devices i.e. tablets and smartphones.
- The battery life of said devices are a big concern because they don't last very long.
- Researching energy consumption is fairly new to the software engineering so the field has a lot of potential.
- Though energy consumption has been studied extensively in networking it has not been applied to mobile devices.

Objective

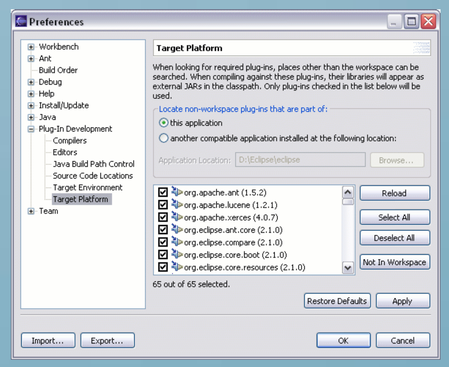
- Obtain the android source code.
- Analyze the code in the Ddmsplugin.java, LogCatMonitor.java, and LogCatMonitorDialog.java.
- Understand how the android phone communicates with the DDMS to create the plugin.
- Create the plugin and its icon.

Approach

- Find code that links up previous plugins to create a power profiler.
- Implement this into the Plugin Manifest File.

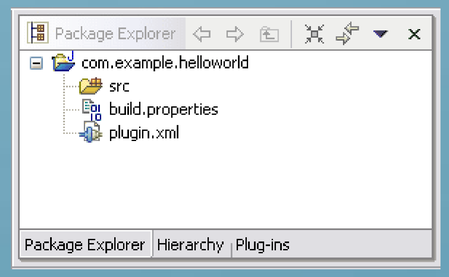
```
/*
 * Load the extension point implementations.
 * The first step is to load the IConfigurationElement representing the implementations.
 * The 2nd step is to use these objects to instantiate the implementation classes.
 *
 * Because the 2nd step will trigger loading the plug-ins providing the implementations,
 * and those plug-ins could access DDMS classes (like ADT), this 2nd step should be done
 * in a Job to ensure that DDMS is loaded, so that the other plug-ins can load.
 *
 * Both steps could be done in the 2nd step but some of DDMS UI rely on knowing if there
 * is an implementation or not (DeviceView), so we do the first steps in start() and, in
 * some case, record it.
 *
 */
```

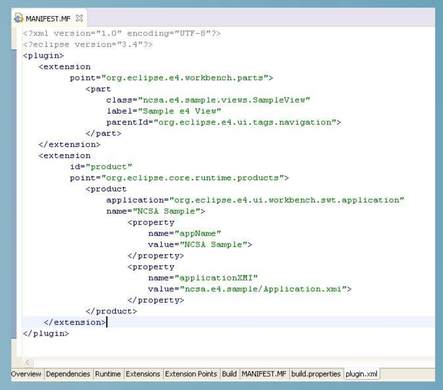
- In order to develop Plugins we have a Plugin Development Environment.
- Specify the Target Platform.
- Tells your plugin the location of other plugins that will be used.
- Almost all plugins require plugins that exist to function.



Approach (2)

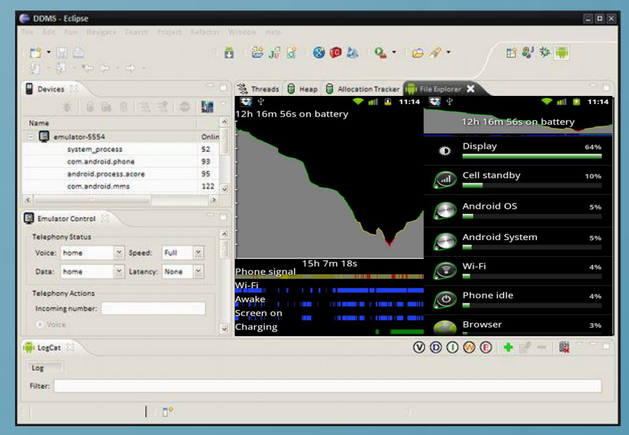
- Created the project
- Has: manifest file, plug-in content, build.properties





Discussion

- Although this project utilizes only one way to see the power consumption of the applications, there exist other power profilers that provide finer granularity.
- Some of the power profilers include: Eprof, Power Tutor etc.
- These power profilers might allow the developer to see where in the code their application takes up the most energy and allow the developer to optimize that section of code.
- I plan to make the plugin support multiple power profilers.



Acknowledgements

•Dr. Steven Ko
•Computer Science and Engineering Department.
•McNair Scholars Program

www.buffalo.edu