Android: Setup

Hello, World: Android Edition

due by noon ET on Thu 2/24

Ingredients.

- Android Development Tools Plugin for Eclipse
- Android Software Development Kit
- Eclipse
- Java

Help.

Help is available throughout the week at `http://help.cs76.net/`! We'll do our best to respond within 24 hours. Be sure, though, to take advantage of lectures and sections as well as videos thereof!

Academic Honesty

All work that you do toward fulfillment of this course's expectations must be your own unless collaboration is explicitly allowed by some project. Viewing, requesting, or copying another individual's work or lifting material from a book, magazine, website, or other source—even in part—and presenting it as your own constitutes academic dishonesty, as does showing or giving your work, even in part, to another student.

Similarly is dual submission academic dishonesty: you may not submit the same or similar work to this course that you have submitted or will submit to another. Nor may you provide or make available your or other students' solutions to projects to individuals who take or may take this course (or CSCI S-76) in the future.

You are welcome to discuss the course's material with others in order to better understand it. You may even discuss problem sets with classmates, but you may not share code. You may also turn to the Web for instruction beyond the course's lectures and sections, for references, and for solutions to technical difficulties, but not for outright solutions to problems on projects. However, failure to cite (as with comments) the origin of any code or technique that you do discover outside of the course's lectures and sections (even while respecting these constraints) and then integrate into your own work may be considered academic dishonesty.

If in doubt as to the appropriateness of some discussion or action, contact the staff.

All forms of academic dishonesty are dealt with harshly.

Grades.

Your work on this project will be evaluated along one primary axis.

Correctness. To what extent is your code consistent with our specifications and free of bugs?

Installing and Preparing the Android SDK.

☐ The Android software development kit (SDK) is a collection of tools that allow a developer to write, test, and deploy native Android applications written in Java. Though, really, when you download the SDK, it's just a basic collection of tools; it does not even include the utilities required to build an application! The SDK by itself also does not have a built-in integrated development environment (IDE), which would provide a GUI-based application to help develop your application. For this reason, installing the SDK is a multi-step process in which you download an IDE, the SDK itself, a plugin that acts as an interface between the IDE and the SDK, platform tools that allow you to build native Android applications, and finally create some Android Virtual Devices to use the emulator.

One website that you will find quite useful while developing Android applications is the Android Developer site at `http://developer.android.com/`.

We have some things to install, so let's get started!

☐ First up: Eclipse, the one IDE that supports the Android SDK and its Android Development Tools (ADT) Plugin. You can use a pre-existing installation of Eclipse if you already have it installed, but do make sure you are using the latest version, 3.6.1. If you already have the latest version installed, skip ahead to the next step. If you don't have it installed, visit the download page: `http://www.eclipse.org/downloads/` and download "Eclipse Classic 3.6.1". You might be presented with two download options: 32-bit or 64-bit. If you are unsure which applies to your specific machine, use one of the following links that is appropriate for your platform:

- ☐ Windows: `http://support.microsoft.com/kb/827218`
- ☐ Mac OS X: `http://support.apple.com/kb/ht3696`
- ☐ Linux: open a terminal window and type `uname -p`. The result will be `i386` for 32-bit machines and `x86_64` for 64-bit machines.

☐ Once you've downloaded the appropriate Eclipse version, install it by unzipping the downloaded file and moving the `eclipse` folder to some appropriate location, like `/Applications/` on a Mac or `C:\Program Files\` on a Windows PC.

☐ Now we're ready for the Android SDK itself! To download the development kit, visit `http://developer.android.com/sdk/index.html` and download the SDK appropriate for your platform. If you downloaded a ZIP or TGZ, uncompress the downloaded file and place the entire SDK folder into the `eclipse` folder from above. If you use the Windows installer, we recommend installing the SDK in the same `eclipse` folder and leaving checked the box that says **Start SDK Manager** to install some required components. Incidentally, it's not required that you place or install the SDK components in the `eclipse` folder, but we find that it keeps things tidy this way. If you do decide to place the SDK elsewhere, simply remember its location so that you can properly configure the ADT Plugin in the upcoming steps.

☐ Next up, let's tell Eclipse how to use the Android SDK by installing the ADT Plugin. To do this, first open Eclipse and follow the directions from the "Downloading the ADT Plugin" from:
`http://developer.android.com/sdk/eclipse-adt.html#downloading`

☐ Follow the directions from "Configuring the ADT Plugin" to so that Eclipse can find the SDK:
`http://developer.android.com/sdk/eclipse-adt.html#configuring`
You might receive a message like "SDK Platform Tools component is missing! Please use the SDK Manager to install it." If so, it's fine to click **OK** and ignore since we're installing this next.

☐ The SDK does not actually contain all of the tools specific to each Android version to build applications. We need to install the SDK Components to achieve this functionality, though if you have already done so after using the SDK installer on Windows you may skip to the next step. Otherwise, open Eclipse, select the **Window** menu and open the **Android SDK and AVD Manager**. From there, install **all** available components while following the instructions at:
`http://developer.android.com/sdk/adding-components.html#InstallingComponents`
Installing older versions in addition to the newest ones will allow you to build Android applications for the oldest version of Android your code can support, which will be useful to create applications that are compatible with as many devices as possible.

☐ Almost there! Everything is installed, but you can't quite open up an Android emulator quite yet. For that, we need to create an Android Virtual Device, or AVD. To do this, open the **Android SDK and AVD Manager** (remember how?) if it isn't already open. Select **Virtual Devices** from the options on the left, and then the **New…** button. Give your virtual device a name; we recommend that you give it a name that reflects the version of Android and any features you've selected for the AVD. In the **Target** pull-down, select "Android 2.3.1 – API Level 9", which reflects the most recent version publicly available on devices. Note that installing the SDK components from the previous step provided all available targets in that pull-down. If you don't see the Android 2.3.1 in the target pull down, you should double-check that you've installed all components.

The remainder of the options in the Create new Android Virtual Device (AVD) window defines additional properties of the emulated device, such as an (emulated) SD card or any special hardware or features, but leaving the defaults is fine for now. When done, click the **Create AVD** button. You should see your new AVD in the Android SDK and AVD Manager window, where you can click on your new AVD and click **Start…** to load the emulator and the AVD. If the resolution on your screen is low and you find the emulated screen is too large to fit on your display, you might need to select the "Scale display to real size" checkbox and change the DPI to reflect that of your own display in the Launch Options window that appears when starting an AVD. If this is the case, click on the **?** button next to the **Monitor dpi** field to help you calculate your screen's DPI.

☐ Now create another AVD but use a target of "Android 1.5 – API Level 3", and be sure to give yourself some time to play with both AVDs to get a feel for how Android has matured over time.

☐ Whew, you've done it! You've installed all of the SDK components and prepared yourself for native Android application development. Let's just recap quickly to make sure we're on the same page. You should have done the following:

- ☐ Installed Eclipse Classic and the Android SDK.
- ☐ Downloaded and configured the ADT Plugin to make Eclipse aware of the Android SDK.
- ☐ Installed all of the available SDK components.
- ☐ Created an AVD for Android 1.5 and Android 2.3.1 and played with both.

☐ Take a break, grab a sandwich and make some tea. Or perhaps you'd prefer some gingerbread[1].

Looking Ahead: Student's Choice.

☐ In a few weeks we will be releasing the "Android: Student's Choice" project. This project is meant as an opportunity for you to develop a native application using Eclipse and the Android SDK that you would like to see on the Android platform, but you will need to first get an idea approved by your Teaching Fellow before starting on that project. As part of your submission for this Android Setup project you'll propose an idea (or two) that you'd like to implement for the Student Choice project. Since we haven't done much work yet with the SDK itself it might be tough for you to know what sort of features you could implement, but that is precisely the purpose of the proposal: your TF will provide feedback to you about how realistic it will be to complete your proposed project in the two week time frame.

It's fine if your idea is intended for a project you'd like to continue working on outside of the scope of class, and it's also fine if you intend to use the same idea for the "iOS: Student's Choice" project. Just be sure that this is a new project and not one that you have previously written code. This proposal is also not binding and you can change it if you'd like; just be sure to let your TF know and get the new idea approved before you start working.

Submit your proposal before this project's due time at:
`https://www.cs76.net/login/?form=android-setup`

You'll be asked to log in with PIN authentication like you do for `help.cs76.net`. It will pre-populate some fields; be sure **not** to change these fields in any way or we will not be able to match your proposal back to you.

[1] `http://en.wikipedia.org/wiki/Android_OS#Version_history`

Hello, World.

☐ Now that you've installed all of the necessary components to do native application development for the Android platform, it's time to make your first application! This app isn't intended to be difficult but rather to familiarize you with the tools and the development environment. For that purpose, your first Android application should be a Hello, World[2] application.

Implementation Details.

- ☐ This is meant to be the most basic of native Android applications. Your application should compile without errors and open in an Android device or the emulator and say, simply, "Hello, cs76!" in legible text. The remainder of the app's aesthetics is left up to you.
- ☐ The project should be targeted to work on Android version 1.5 and later.
- ☐ Be sure the project and application name are both `Hello########`, where `########` is your 8-digit Harvard ID (HUID), the same credential that you use to log into `help.cs76.net`.
- ☐ Your project's package name should be: `net.cs76.setup.Hello########`

☐ Psst, want a hint? You might find a helpful tutorial on the Android Developer site.

How to Submit.

☐ Before the project's due date, export your project in Eclipse for submission. Open Eclipse and click the **File** menu and then **Export**. In the window that appears, click on the triangle next to the **General** section so that you can see the options contained within. Select **Archive File** and click the **Next** button. On the next window, check the box directly to the left of your project you'd like to submit. Be sure no others are selected, or you will export those as well. Click on the **Browse** button to select a location you'd like to export the ZIP file and be sure to name it `########.zip`, where `########` is your 8-digit Harvard ID (HUID), the same credential that you use to log into `help.cs76.net`.

After selecting where the ZIP file will be placed, make sure the export options are correct. Notably that you are saving as a ZIP file (and not tar), that a directory structure is created for files, and that the contents are compressed. When ready, click **Finish** to export your app.

Then head to `https://www.cs76.net/submit`, click the **login** link at top-right, click the link to your TF's dropboxes at top-left, click this project's own folder, click **Upload File**, and upload your ZIP file as prompted; no need to give it a title. Be sure not to click the wrong project's folder. You may re-submit in this same manner as many times as you'd like. Just take care to delete any prior submissions.

Be sure not to submit or re-submit after this project's deadline.

[2] `http://en.wikipedia.org/wiki/Hello_world_program`