CSc 133 Lecture Notes

2 – Introduction to Mobile App Development and CN1

Computer Science Department
California State University, Sacramento

Overview

- Why to Use a Mobile Programming Environment?
- Why to Choose Codename One (CN1)?
- CN1 Features
- CN1 vs Java
- CN1 Installation
- CN1 Hello World App
- CN1 and Assignments
- Assignment#0
- CN1 Online resources

Why to Use a Mobile Programming Environment?

- Mobile computing is **ubiquitous** and allows:
  - Instant retrieval of information
  - Constant communication
  - Easy access to games, company products etc.
- Hence, there is an **ever growing need** for mobile app developers.
- Also, knowing how to program in this contemporary environment is **fun** and **cool**!

Why to Use a Mobile Programming Environment? (cont.)

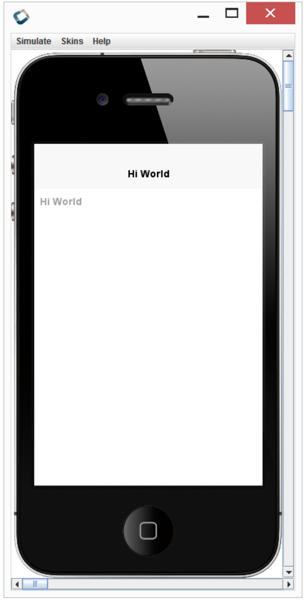
- CSC 133 topics are widely applicable to a mobile programming environment.
- Hence, using this environment in the **lectures** and **assignments**, will help to:
  - Enhance learning by relating CSC 133 topics to their **contemporary use cases**
  - Provide a base for **further exploration** of this environment (apply it to other CSC topics or create your own brilliant app!)
  - Build a stronger **resume**

Why to Choose Codename One (CN1)?

- There are various popular mobile programming environments:
  - Platform specific:
    - e.g., Android, iOS SDK
  - Cross-platform (write one program and run it on various platforms - iOS, Android, Windows, etc.):
    - e.g., Codename One (CN1), Flutter, Xamarin
- We choose CN1, because it is:
  - Java-based
  - Cross-platform

CODENAME ONE

CN1 Features



- Features we will use:
  - Free and open source
  - Have **simulator** environment (does not require you to have a mobile device)
- Features that we will **not** use:
  - Build and cloud **servers** (converts the CN1 code to a native app, e.g., Android, iOS, Windows app)
  - **GUI builder** (provides drag and drop tools to automatically create GUI components)

CN1 vs Java

- CN1 API was initially limited to subset of Java 1.3 and then added support for subset of Java 5 and now supports some Java 8 language features.
- Does not support Java features that are not suitable for mobile devices e.g.:
  - Reflections
  - Desktop APIs such as java.net, java.io.File etc. (provides its own alternatives)
  - Swing library (provides Swing redesigned for mobile environment in its UI API/package)

CN1 Installation

- Install “Java SE JDK version 11”:

https://www.oracle.com/java/technologies/downloads/#java11

- CN1 can be installed to one of the following IDEs: Eclipse, NetBeans, or IntelliJ IDEA which run on various operating systems.
- “Eclipse IDE for Java Developers” is **required.** CSC 133 software is tested with Eclipse IDE 2022-03 R (version 4.23.0):

https://www.eclipse.org/downloads/packages/release/2022-03/r/eclipse-ide-java-developers

- Windows is recommended.

CN1 Installation (cont.)

- Install CN1 plugin to Eclipse using instructions at:

https://www.codenameone.com/how_di_i/how-do-i-create-a-basic-hello-world-application-send-it-to-my-device-using-eclipse.html

In Eclipse select:

“Help” → “Eclipse Marketplace” → search for “Codename One”

and follow the installation process…

- Eclipse and CN1 are installed at school machines. See “Home” section of Canvas for the list of labs that you can use for CSC 133…

CN1 Hello World App

- Steps for Eclipse:
  - File → New → Project → Codename One Project
  - Give a project name “HiWorldPrj” and **uncheck “Java 8 project”**. Hit “Next”.
  - Give a main class name “HiWorld”, package name “com.mycompany.hi”, and select a “native” theme, and **“Hello World(Bare Bones)” template (for manual GUI building)**. Hit “Finish”.
- It generates and builds the project. You can view your main class under the package explorer:

HiWorldPrj → src → com.myCompany.hi → HiWorld.java

CN1 Hello World App (cont.)

- Run the app on the simulator in Eclipse by right clicking the last entry of the project under the package explorer:

  HiWorldPrj → Simulator_HiWorldPrj.launch

  - Select "Run As" to run and "Debug As" to debug your app.
- You can also run it directly from the command-line. Get into the HiWorldPrj directory and (in Windows) type:

```
java -cp dist\HiWorldPrj.jar;JavaSE.jar
com.codename1.impl.javase.Simulator
com.mycompany.hi.HiWorld
```

(all in one line, but put spaces between sub-lines)

CN1 Hello World App (cont.)

- Unix-like operating systems (such as Linux and Mac OS X) use "forward-slash" and "colon" (instead of "back-slash" and "semicolon":

```
java -cp dist/HiWorldPrj.jar:JavaSE.jar
com.codename1.impl.javase.Simulator
com.mycompany.hi.HiWorld
```

(all in one line, but put spaces between sub-lines)

- You can switch through different skins in the simulator. For assignments we will use ipad3_os7.skin (download it via "Skins" → "More" -> "iPad III")

Troubleshooting Problems

- If dist\HiWorldPrj.jar is not generated: Signup for a CN1 account at https://www.codenameone.com, right click on project and hit “Codename One → Send Android Build”, hit OK if a warning is given, then login with your CN1 account, if still does not work:

set **JAVA_HOME** environment variable to JDK directory

In Windows: goto “Control Panel → System → Advance System Settings → Environment Variables” and add JAVA_HOME as

*C:\Program Files\Java\jdk-**11.0.16*** to “System Variables” ***(gray numbers indicate an earlier release of Java 11, the latest release would have more or larger numbers after the second dot)***

- If the command line complains that: ‘java’ is not recognized … : add

*C:\Program Files\Java\jdk-**11.0.16**\bin* to **PATH**

JavaSE.jar cannot be found … : (first make sure you are in the project directory that has JavaSE.jar) add current directory (indicated by a single period “.”) to **CLASSPATH**

(see Appendices.pdf for tips)

Troubleshooting Problems (cont…)

- You may have different JDK versions installed on your system. Make sure you are using JDK 11 to run:
  - Eclipse IDE itself:
    - set **JAVA_HOME** to *C:\Program Files\Java\jdk-**11.0.16***
    - If this does not work, edit the “eclipse.ini” file to force Eclipse to use your JDK 11 install. See https://wiki.eclipse.org/Eclipse.ini for more info.
  - applications under Eclipse:
    - Goto “Window → Preferences → Java → Installed JREs” and add *C:\Program Files\Java\jdk-**11.0.16*** to the list and make it the default JRE.
  - applications from command-line:
    - add *C:\Program Files\Java\jdk-**11.0.16**\bin* to **PATH**

CN1 and Assignments

- For each assignment create a different CN1 project.

- You **must** create all assignments in the same way as HiWorldPrj example:
  - uncheck “Java 8 project”, select “native” theme, and **“Hello World (Bare Bones)”** template.
  - change the project, main class, and package names…

CN1 and Assignments (cont.)

- For instance for Assignment#1:
  - Project Name: A1Prj
  - Main Class Name: Starter (keep the same for all assignments)
  - Package: com.mycompany.a1
- Main class has the following structure:

```
public class Starter {
   …
   public void init(…) {…}
   public void start() {…}
   public void stop() {…}
   public void destroy() {}
}
```

CN1 and Assignments (cont.)

- Solve the assignment by modifying **start()** in Starter.java (**do NOT delete other methods**) and adding more java files (right click on the package then hit “New” → “Class”).
- **Make sure** dist\A1Prj.jar is up to date (if not, in Eclipse, right click on dist directory and hit “Refresh” or right click on project and hit “Codename One → Send to Android Build”, then hit “Cancel”)
- **Make sure** your submission works from command-line. Go into the A1Prj directory and type:

```
    java -cp dist\A1Prj.jar;JavaSE.jar
com.codename1.impl.javase.Simulator
com.mycompany.a1.Starter
```

(all in one line, but put spaces between sub-lines)

CN1 and Assignments (cont.)

- Again, for Unix-like operating systems (such as Linux and Mac OS X) use “forward-slash” and “colon”:

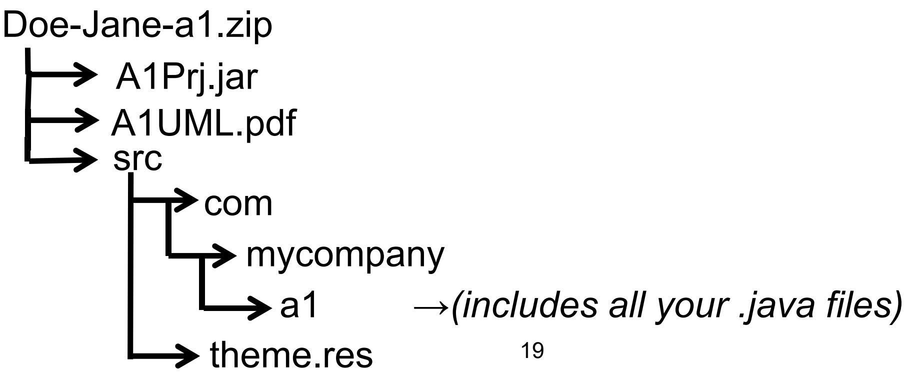
```
    java -cp dist/A1Prj.jar:JavaSE.jar
    com.codename1.impl.javase.Simulator
    com.mycompany.a1.Starter
```

(all in one line, but put spaces between sub-lines)

- Preparing the deliverable:
  For **ALL** assignments, you must **separately submit TWO files**:
  1) Doe-Jane-a1.zip (**ZIP** file)
  2) readme.txt (**TEXT** file)

CN1 and Assignments (cont.)

- **About the Zip file:**
  - Create a ZIP file containing the *.jar* file generated under *dist* dir and entire *src* dir (do **NOT** include other dirs such as *bin* etc.).
  - In A#1 and A#2, you will be asked to also include pdf of the UML diagram of your assignment in the ZIP file.
  - For A#1 the ZIP file would have the following structure:



Doe-Jane-a1.zip

A1Prj.jar

A1UML.pdf

src

com

mycompany

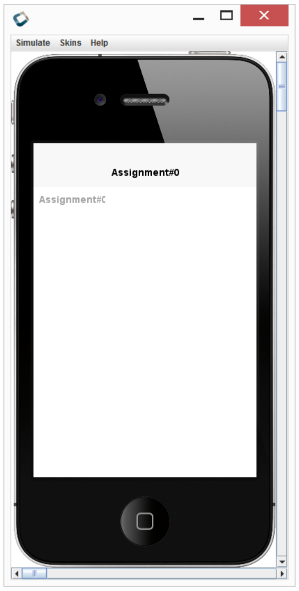
a1 →*(includes all your .java files)*

theme.res

CN1 and Assignments (cont.)

- **About the Text file:**
  - Create a **TEXT (i.e., not a pdf, doc etc.)** file called "readme.txt" that indicates the **remotely accessible** lab machine you have used to generate the *.jar* file (under *dist* dir) and test your assignment. Specify the lab number and the name of the specific machine you have used in that lab. **Make sure that you include the *src* folder and *jar* file generated/tested on the lab machine in the ZIP file.**
  - You may also include additional information you want to share with the grader in this text file…
  - You will receive the grader comments on your text file when grades are posted.
  - Do **NOT** place this TEXT file inside the ZIP file, instead submit it **separately.**

Assignment#0



- Find a lab computer that has CN1 or install CN1 to your computer.
- Following the instruction in the previous slides, generate an empty project called A0Prj.
- Modify Starter.java by replacing the texts “Hi World” with “Assignment#0”. Run the simulator.
- Experiments with debugging options of your IDE.
- Verify that your submission also works from the command line.

Assignment#0 (cont.)

- In addition, verify that your program works on a lab machine (i.e., a machine in a **remotely accessible** lab).
- To build and test your assignment in the lab, you can copy your assignment directory located in the Eclipse workspace directory of your machine to the Eclipse workspace directory of the lab machine and use “File → Import → General → Existing Project into Workspace” option of Eclipse to import your project to the lab workspace.

Do not submit A0 via Canvas (its purpose is to make sure you have access to CN1 and ready to solve real assignments)

CN1 Online resources

- Developers guide:

CN1 Developer Guide - Revision 3.6 (pdf is available at Canvas)

- Video tutorials can be found at:

https://www.codenameone.com/how-do-i.html

(note: mostly give examples that use the GUI builder which we will not utilize)

- You can view JavaDocs of APIs:

https://www.codenameone.com/javadoc/index.html