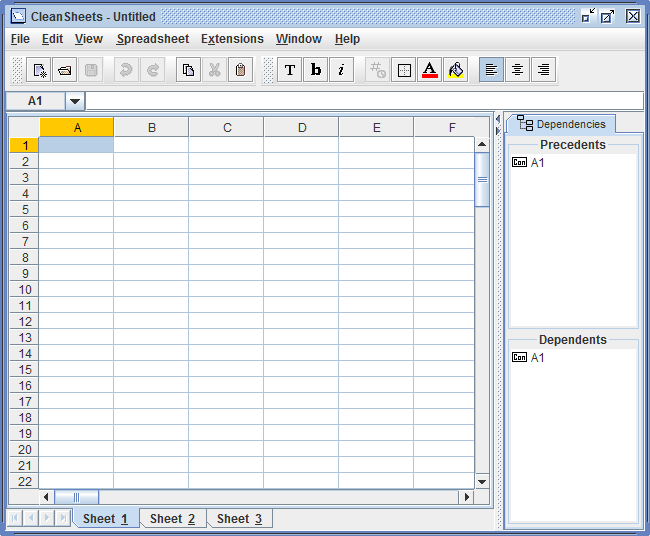
# This is an old CSSE 375 exam – it’s just for practice. It’s not your current 375 exam.

# CSSE 375: Exam 2

This is a take-home exam. Please do not discuss this exam with anyone else. Please be careful with your solution code to ensure that it isn’t left somewhere another student could access it. If you have any questions about Exam 2, contact Professor Hewner directly.

## Scenario

In this exam you will modify an existing software package – the Cleansheets Java spreadsheet application. Download/install the eclipse project from Moodle (it is nearly identical to the one available from the official website, but does have a few modifications for this exam). Run it and you should see a window like this:



The stated goal of this scenario is that we want to internationalize this application. This exam will concentrate on one aspect of that: internationalizing the “Spreadsheets” menu.

## Part 1: Playing around

Find you way around the system and determine how the UI works and the menu system is populated, using whatever techniques you wish.

**[10%]** Modify the code so that the title of the main window has you name (e.g. rather that say “Cleansheets” my title now says “Cleansheets Mike Hewner Edition”).

## Part 2: Java Internationalization

Obviously, we don’t want to make this change without first having some unit tests. Look in the package csheets.testing – you can see I’ve gotten things started there for you. One thing you’ll notice is the existing test testJavaInternationalizationExample. This is designed to show you how java’s internationalization system works:

**public** **void** testJavaInternationalizationExample() {

ResourceBundle messages;

messages = ResourceBundle.*getBundle*("CommandTextBundle");

*assertEquals*("Hello", messages.getString("greetings"));

Locale france = **new** Locale("fr", "FR");

Locale.*setDefault*(france);

messages = ResourceBundle.*getBundle*("CommandTextBundle");

*assertEquals*("Bonjour", messages.getString("greetings"));

}

You can see that strings are loaded from ResourceBundles. Depending on the global default locale, either the default (English) strings are loaded or the French strings are loaded. Take a look at the CommandTextBundle.properties file and the CommandTextBundle\_fr\_FR.peroperties files (they’re in the root of src). You’ll notice that all the strings you need to internationalize are already in there and setup for you to use.

If you’d like some more details about internationalization in Java, check here:

<http://docs.oracle.com/javase/tutorial/i18n/intro/index.html>

You may be a bit concerned that we’re using the default locale as a global variable, rather than passing the locale as a parameter to UI functions. Yeah, if you had to explicitly pass locale around it’ll be a bigger job than is worthwhile for this exam. So don’t worry about it.

You should be able to run the tests, and this test should succeed.

## Part 3: Testing challenges.

Our goal is to write tests that test the MenuBar class.

You should also see in the test file two other tests that do this. These tests don’t run properly, however, because CleanSheets and UIController (two parameters to MenuBar’s constructor) are being passed as null. Constructing them normally, however, also causes problems (try it and see).

**[32.5%]** Using Feather’s techniques, get the testMenuBar and testMenuBarFrench tests working (though of course testMenuBarFrench will fail until you complete the internationalization features). Make your changes in a way that *minimizes modification to the existing non-test source code*. You can do stuff like Extract Interface/Extract Method through the Eclipse UI, but functional modification to the code of CleanSheets or UIController will cost you points.

**[7.5%]** Using the terminology of the Feather’s book, explain what you did to get the code under test in a comment in the testMenuBar function.

## Part 4: Internationalize the Spreadsheets menu

**[22.5%]** Modify the code of the CleanSheets project to internationalize the Spreadsheets menu of the CleanSheets UI. When you are finished both tests should run successfully and the Cleansheets application should work as it does now. If you modify the code to set the default locale to French in the CleanSheets main function, you should be able to see that the spreadsheet’s menu is French.

**[22.5%]** This change should be done in a way that is clean from an OO design perspective, and fits in well with the existing code architecture. The design should be straightforward to extend to all the other menus in the system, and it should be possible to do this without a lot of duplicated code.

**[5%]** When you are finished, set the default locale in main and take a screenshot (it should look like the one below – I used the Print Screen button to catch the menu while it was open). Include the screenshot with your submission.

