Engineering Design for Embedded Systems: Assignment 4

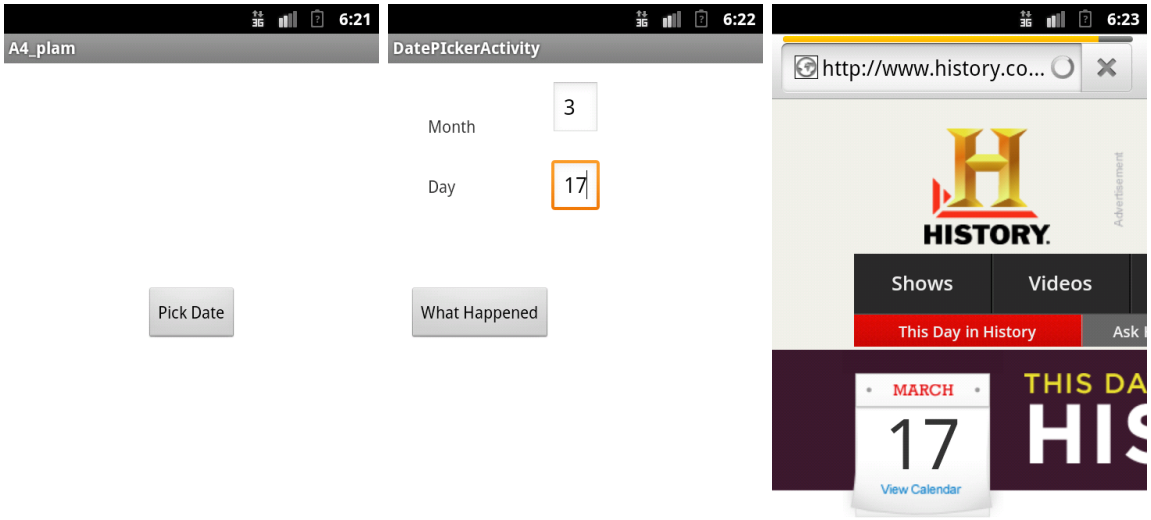
Due Date: February 4, 2013

This assignment gives you an opportunity to use Android Intents and to practice debugging. Lecture 10 PDF notes contain most of the information you need for the first part of the assignment.

If I find any cases of plagiarism on this assignment, I will apply the standard penalties and report the case to the Associate Dean, as per Policy 71.

1 Android Programming: Intents (5 points)

The key notion here is that of an `Intent`, which allows you to request that Android brings up a new `Activity`. In this part of this assignment, you are going to use an `Intent` to launch a second `Activity` and then accept the result of that `Intent` to launch a web page. My solution to this part of the assignment had fewer than 50 lines of code. Here's screenshots of what you have to do.



Main Activity

The screenshot on the left illustrates the `MainActivity` for this assignment. It contains one button.

Task. Create a button in your main activity. When the user clicks the button, your app will use an explicit `Intent` to launch your `DatePickerActivity`.

By the way, the lecture notes contain this code fragment:

```
Intent intent = new Intent(this, OtherActivity.class);
startActivity(intent);
```

If you use an anonymous inner class for the click listener, you must write `MainActivity.this` instead of just `this`. Plain old `this` will refer to the inner class, which is not what you want.

Date Picker Activity

The second activity allows your user to select a day and month and return that information to the calling `Activity`.

Task. Create two `EditText` boxes for the month and the day. (It's OK if you choose to use the Android date-picking widget as well, but I'm not providing any help on that.) You should also create labels for these boxes. Also, create a button which reports the result to the calling activity, which is your `MainActivity`.

When the user clicks the button, your activity needs to read the values from the text boxes, perform error checking, and return the values to the calling activity. It should do that by creating a new `Intent`, using `putExtra` on that intent to store the month and day, and calling `setResult()` to set `RESULT_OK`.

finish() In lecture, I forgot to mention that if you actually want to return control to the first activity, you need to call `finish()`. Both `setResult()` and `finish()` are defined on the `Activity` class, so that your `DatePickerActivity` can just call these methods.

Reading off the returned results

The final task is to handle the month and day that the date picker returned.

Task. Implement a `onActivityResult()` method. If it's called with the appropriate `resultCode` and `requestCode`, it should start an implicit intent which opens the following URI:

```
http://www.history.com/this-day-in-history/[month]/[day]
```

Note: User Interface Design. By the way, I'm proposing a terrible UI design (on so many levels). You may (for no credit) include a discussion of why the UI design is terrible and suggest a better UI design. But, this contrived design does let you practice using `Intent`s.

2 Debugging of past ECE 150 Programming Assignment 4, Car Dealership Inventory (5 points)

A past ECE 150 Programming Assignment 4. This assignment asked for the design and implementation of a Car Dealership Inventory program. The program had to use a singly linked list of `Car`s. This was a fairly large assignment, amounting to several hundred lines of code.

The a6-CarInvProgram.zip file in this folder contains one implementation of PA4. For your reference, the folder also includes a sample .csv file with car data and the F'10 description of PA4. This implementation was returned by a car dealership with the complaint that it "behaved strangely".

Questions. What debugging tactics should be used to localize and identify the underlying bug(s)? What debug test cases should be run, and what diagnostic information should be obtained? What are the actual bug(s)?

Task. As before. After having written down answers to the questions above, apply the tactics that you have suggested. Keep a record of the changes that you apply to the implementation and the test cases that you investigate, including the one that activates the bug. Identify and correct the bug, explaining how the information that you collected helped.