CSc 317
Local Storage

Benjamin Dicken

Announcements

- Quiz 4
- API level for M1
- PA 3

A New App

- Create a new Application called **Tasks**
- Will use to experiment with local storage today, Wednesday and maybe Friday too

Local Storage

- Android uses a file system, similar to other operating systems that use disk-based or SSD-based file systems
- Can store information private to application, or made available to other apps and the user

Types of Local storage

- **App-Specific Storage**
  - Files only accessible to one application
  - https://developer.android.com/training/data-storage/app-specific#internal-access-files
- **Shared Storage**
  - Files accessible by multiple apps, and user
- **Preferences**
  - Persistent key/value mappings
- **Databases**
  - Store data in a structured way

App-Specific File Write

```java
String fileContents = "Some\ncontent\n";
FileOutputStream outputStream =
        openFileOutput("file_name.txt", this.MODE_PRIVATE);
outputStream.write(fileContents.getBytes());
outputStream.close();
```

App-Specific File Read

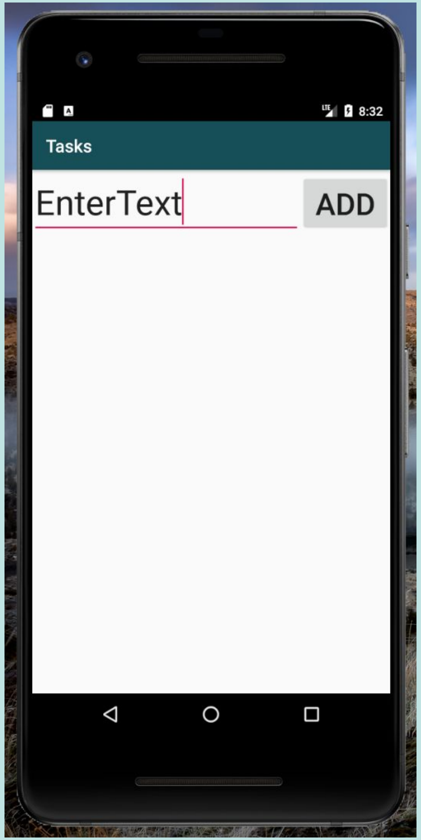
```
FileInputStream inputStream = this.openFileInput("file_name.txt");
InputStreamReader streamReader = new InputStreamReader(fileInputStream);
BufferedReader bufferedReader = new BufferedReader(inputStreamReader);

String line = bufferedReader.readLine(); // If returns NULL, can stop
```

ICA

Tasks App Interface

- Match the interface Shown
- Use a LinearLayout or ConstraintLayout, whichever you prefer

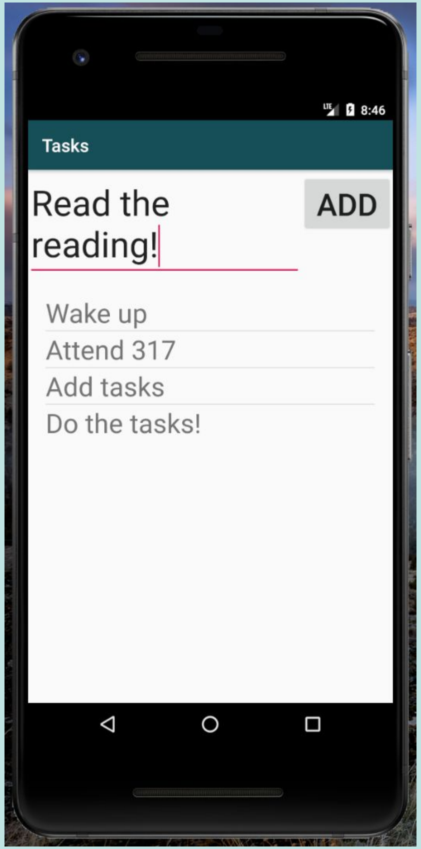


ICA

Tasks App Interface

- Add a ListView, SimpleAdapter, and some dummy tasks

```
private ListView tasksListView = null;
private ArrayAdapter<String> taskArrayAdapter = null;
private ArrayList<String> tasks = new ArrayList<String>(
                    Arrays.asList("homework"...));
...
tasksListView =
      (ListView)findViewById(R.id.tasks_list_view);
taskArrayAdapter = new
      ArrayAdapter<String>(this, R.layout.task_row,
      R.id.task_item, tasks);
tasksListView.setAdapter(taskArrayAdapter);
```



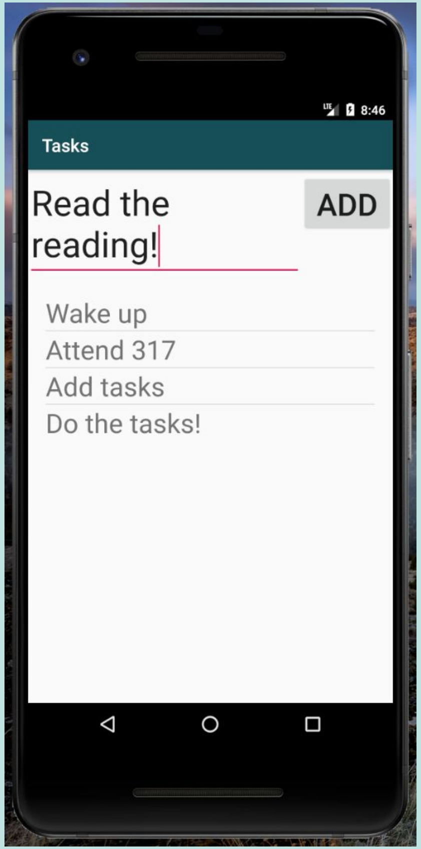
ICA

Tasks App Interface

Next, Create file to Store tasks, if it doesn't already exist. Try this:

```
File file = this.getFileStreamPath(tasksFileName);

if (!file.exists()) {
    String fileContents = "A\nB\nC\nD\n";
    FileOutputStream outputStream;
    outputStream = openFileOutput(
            tasksFileName, this.MODE_PRIVATE);
    outputStream.write(fileContents.getBytes());
    outputStream.close();
}
```



Read the reading!

Wake up
Attend 317
Add tasks
Do the tasks!

ICA

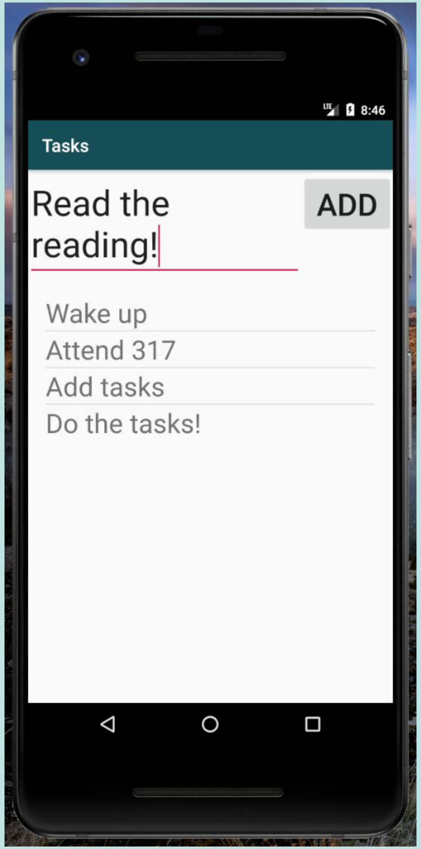
Tasks App Interface

- Next, In another function, load tasks from the tasks file, assuming it is already there

```
Context context = getApplicationContext();

FileInputStream fileInputStream =
      context.openFileInput(tasksFileName);

InputStreamReader inputStreamReader = new
      InputStreamReader(fileInputStream);
BufferedReader bufferedReader = new
      BufferedReader(inputStreamReader);
String taskLine = bufferedReader.readLine();
```



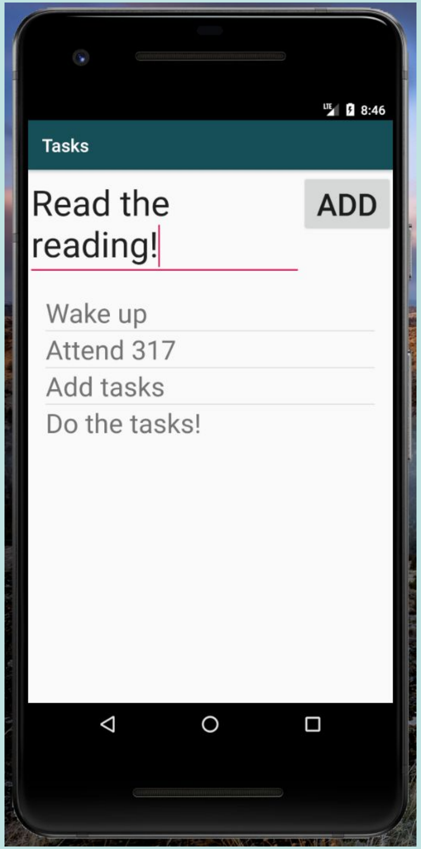
Wake up
Attend 317
Add tasks
Do the tasks!

Tasks App Interface

- Next, load each line into the task list

```
. . .

String taskLine = bufferedReader.readLine();
while (taskLine != null) {
    tasks.add(taskLine);
    taskLine = bufferedReader.readLine();
}
```



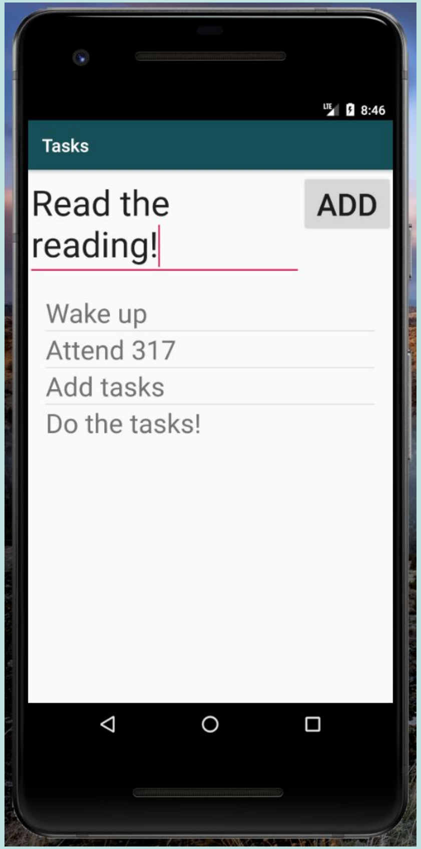
Wake up
Attend 317
Add tasks
Do the tasks!

Save / Load

Ensure that:

1. When the App / Activity is started, the tasks from the file get loaded in and displayed

2. When the App / Activity is closed, everything in the task list gets saved for future reference!



Wake up
Attend 317
Add tasks
Do the tasks!

Shared Preferences

- Key/Value pairs
- Can be accessible to only app, or more broadly available
- Can store Strings, ints

Save to Shared Preferences

```java
SharedPreferences sharedPrefs =
        this.getPreferences(Context.MODE_PRIVATE);
SharedPreferences.Editor editor = sharedPrefs.edit();

editor.putString("Key_1", "a string value");
editor.putInt("Key_2", 123);

editor.commit();
```

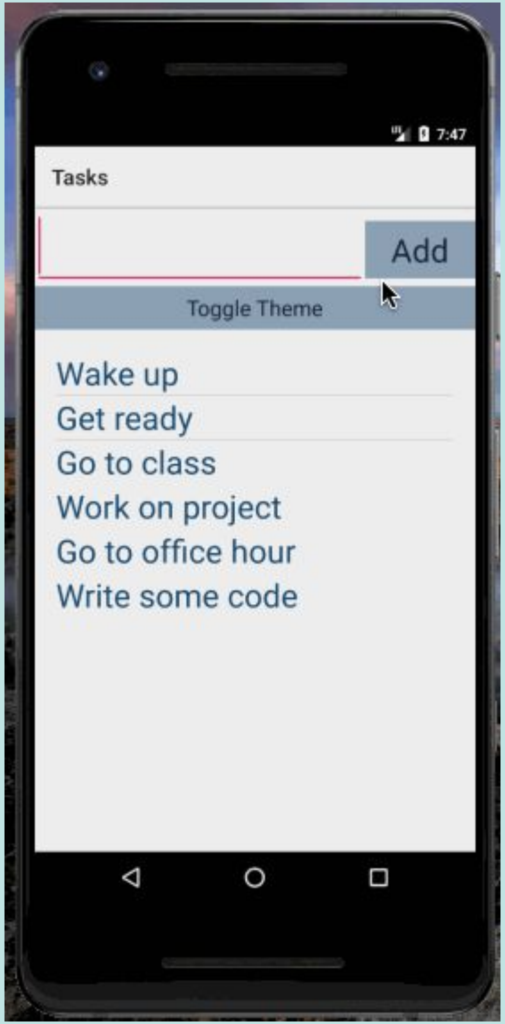
Read from Shared Preferences

```
SharedPreferences sharedPrefs =
        this.getPreferences(Context.MODE_PRIVATE);

String value1 = sharedPrefs.getString("Key_1", "default");
int value1 = sharedPrefs.getInt("Key_2", 123);
```

Saving Theme

- Change the app to save user theme preference
  - Add Button to select theme
  - Create two different themes
  - **`onClick`**, theme should be changed
    - Also, selection persisted



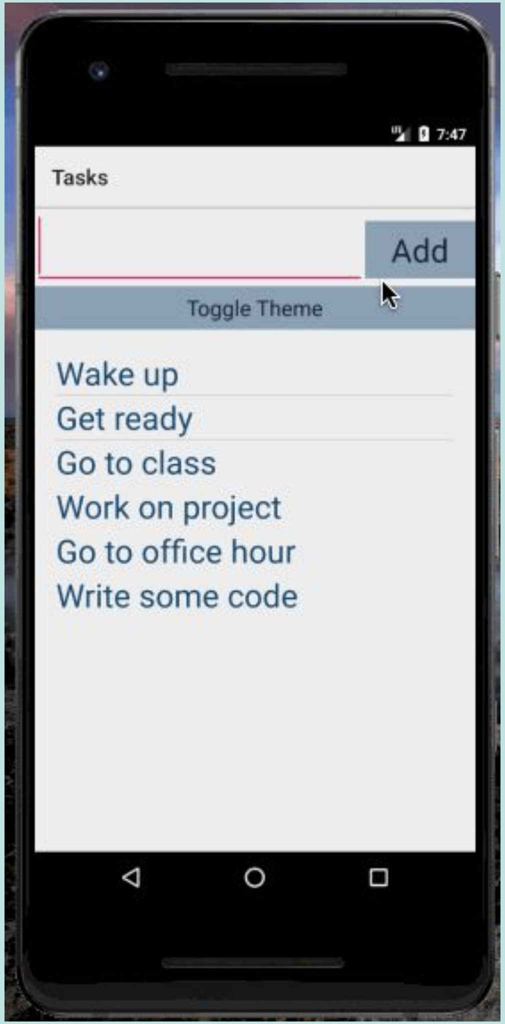
Wake up
Get ready
Go to class
Work on project
Go to office hour
Write some code

ICA

Saving Theme

```
<style name="AppThemeLight"
        parent="Theme.AppCompat.Light">
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="android:textColor">#154970</item>
    <item name="android:background">#EEEEEE</item>
</style>

<style name="AppThemeDark"
        parent="Theme.AppCompat.Light.DarkActionBar">
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="android:textColor">#98C9EE</item>
    <item name="android:background">#242424</item>
</style>
```



Wake up
Get ready
Go to class
Work on project
Go to office hour
Write some code

ICA

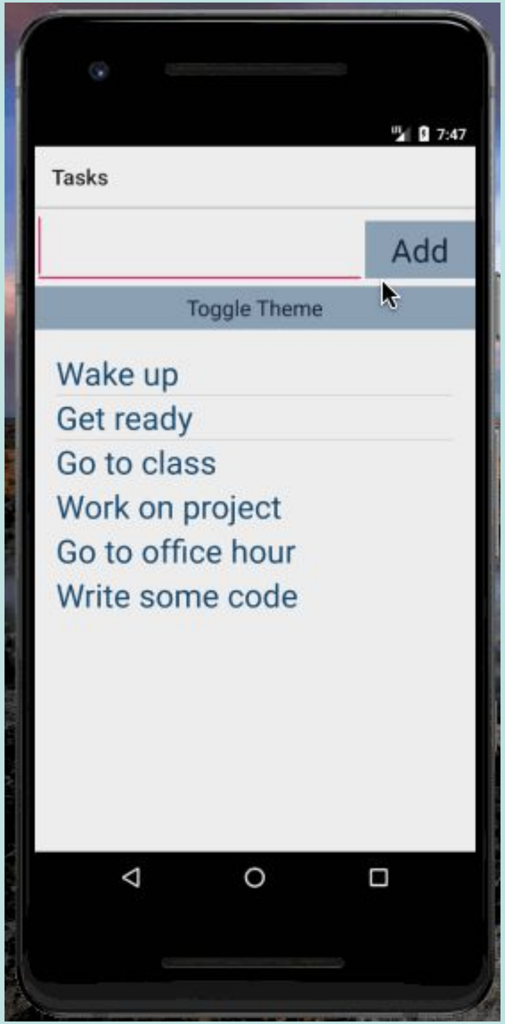
Saving Theme

```
// Before calling setContentView
SharedPreferences sharedPrefs = this.getPreferences(Context.MODE_PRIVATE);
String theme  = sharedPrefs.getString(TASKS_THEME, THEME_LIGHT);

if (theme.equals(THEME_LIGHT)) {
   setTheme(R.style.AppThemeLight);
} else {
   setTheme(R.style.AppThemeDark);
}

// ...
// When theme button is pressed
SharedPreferences sharedPrefs = this.getPreferences(Context.MODE_PRIVATE);
String theme  = sharedPrefs.getString(TASKS_THEME, THEME_LIGHT);
SharedPreferences.Editor editor = sharedPrefs.edit();

if (theme.equals(THEME_LIGHT)) {
    // Update Theme, recreate the activity with recreate()
} else {
    // Update Theme, recreate the activity with recreate()
}
```



Wake up
Get ready
Go to class
Work on project
Go to office hour
Write some code