ECOBATTERY

https://github.com/comp195/senior-project-fall-2022-batterysaverapp.git

Syeda Aliha Naeem (s_naeem@u.pacific.edu)

Mah Noor (m_noor@u.pacific.edu)

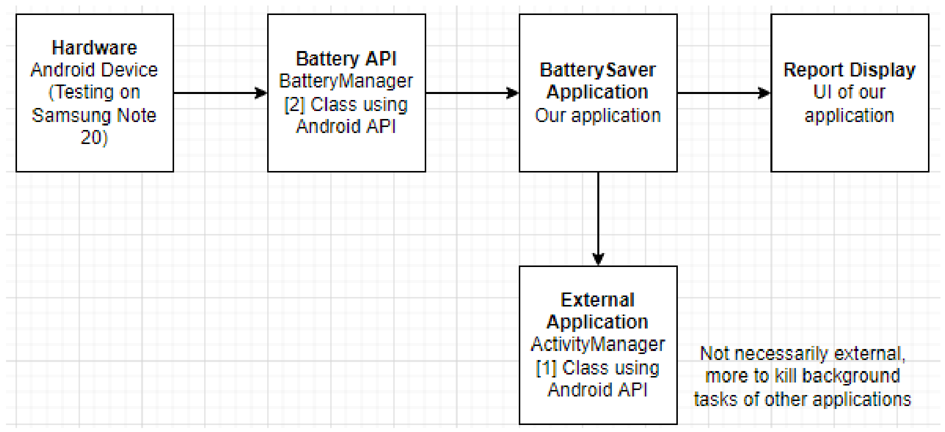
Andrew Kim (a_kim86@u.pacific.edu)

09/18/2022

Project Introduction and Scope

EcoBattery will be an android application used to help users maximize conservation of their batteries. The application will display essential battery data and provide the functionality to keep other applications enabled or disabled while in use. While Android devices have built-in features to preserve battery, our application will help to maintain battery more efficiently, whether it be an old or slow Android device, while the user is traveling or sleeping, etc.

System Architecture


Hardware
Android Device (Testing on Samsung Note 20)
Battery API
BatteryManager [2] Class using Android API
BatterySaver Application
Our application
Report Display
UI of our application
External Application
ActivityManager [1] Class using Android API
Not necessarily external, more to kill background tasks of other applications

Hardware, Software and System Requirements

Hardware Requirements:

[3] (Specs defined by oldest Android 4.1 device, specifically the Samsung Google Nexus S)

- CPU Speed: 1GHz

- RAM: 512MB

- Disk Space: 1GB

- GPU: PowerVR SGX540

Software Requirements:

- Android API 16

System Requirements:

- Android 4.1 Jelly Bean

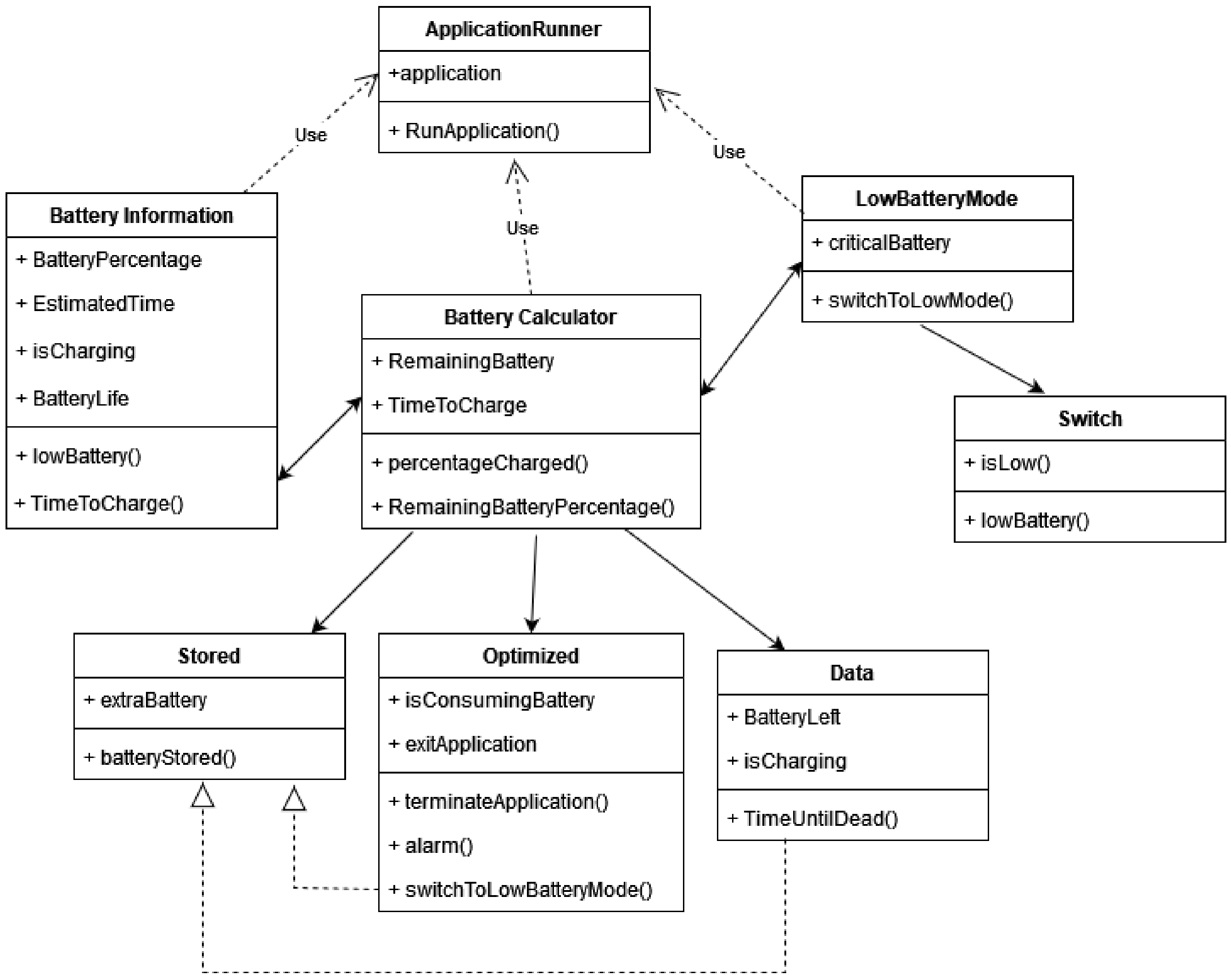
External Interfaces

Project does not include external interfaces.

Software Design

UML Class Diagram

(The implementation is the same but some of the classes have been integrated into already existing ones)



Class Specifications

Description of main classes and main relationships

There are four main classes in this project. The applicationRunner class is the main driving class for this application. The battery information class holds all the information about the battery. The battery calculator class is where all the calculations are going to be done. The battery calculator class uses functions from the classes stored, optimized and data to perform several calculations. The low Battery Mode class is the going to implement the battery saver aspect of this application. The switch class is used to switch to the low battery mode.

Implementation of each Class

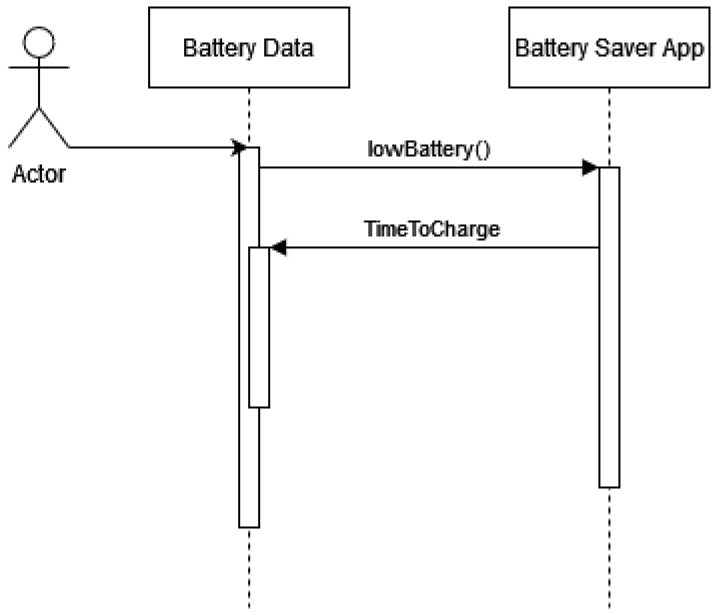
1. **ApplicationRunner** : The main runner class.
    a. +runApplication() : void
       This method is used to run the entire application.
2. **BatteryInformation**: The class with all the information about the battery
    a. +lowBattery() : void
       This method is used to evaluate if the current battery is considered to be low
    b. +timeToCharge() : static Time
       This method is going to be used to give the user an estimate on how much time is left until the user needs to start charging again
3. **BatteryCalculator** : This class is used to perform all the calculations related to the battery saver application.
    a. +percentageCharged() : int
       This method is used to calculate the percentage of battery that is charged.
    b. +remainingBatteryPercentage() : int
       This method shows the remaining battery left
4. **LowBatteryMode**: This class will deal with all the functionalities of the application in low power mode.
    a. +switchToLow() : void
       This method is used to switch to low battery mode of the device
5. **Stored** : This class is used detail information about the stored battery
    a. +batteryStored(): void
       This method is used to keep logs on the battery storage
6. **Optimized** : This class is used to do all the battery optimizations.
    a. +terminateApplication() : void
       This method is used to terminate all the applications that are in the background.

   b. +alarm() : void
      This method is used to set off an alarm to alert the user of low battery levels
   c. +switchToLowBatteryMode() : void
      This method is used to switch to low battery mode after the battery is dropped below a certain level

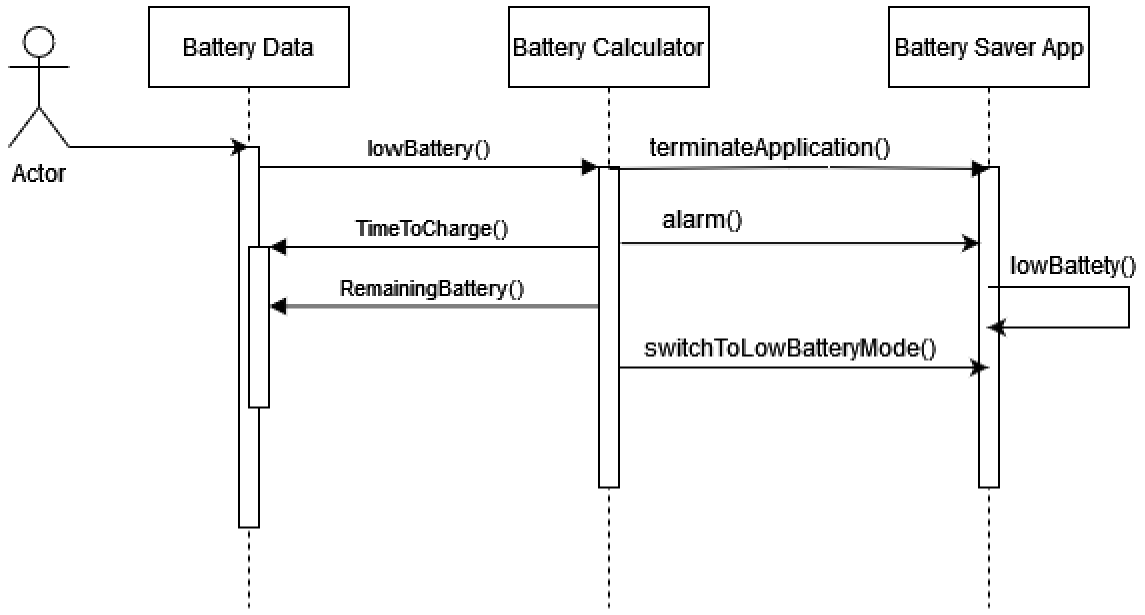
7. **Data**: This class stores all the data about battery
   a. +timeUntilDead(): void
      This method is used to determine the time that is left until the device completely shuts off.
8. **Switch**: This class is used to implement the actual switch to low battery mode
   a. +lowBattery(): void
      This function is used to trigger the switch to low battery mode.

Sequence Diagram:

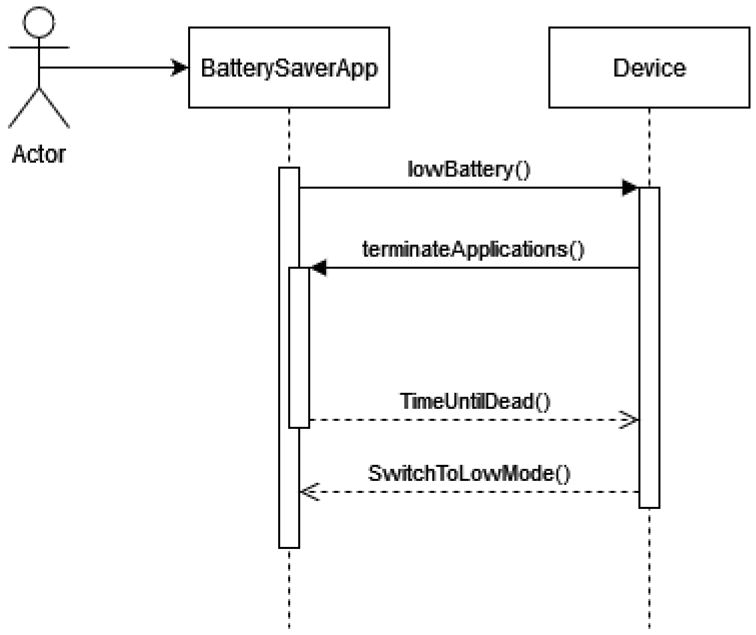
Use Case 1: Battery Infomatics



Use Case 2: Battery Calculations



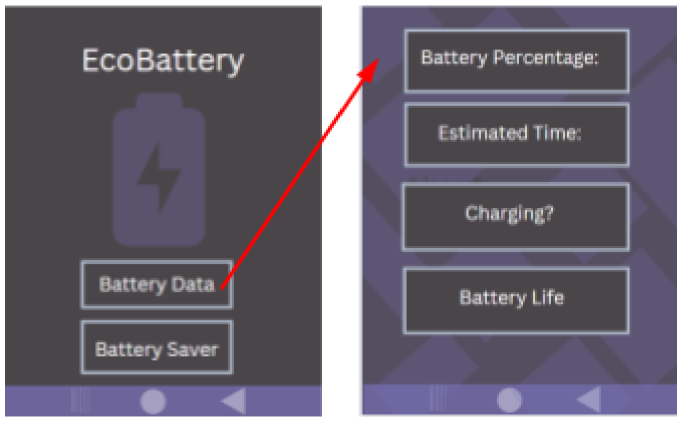
Use Case 3: Battery Saver



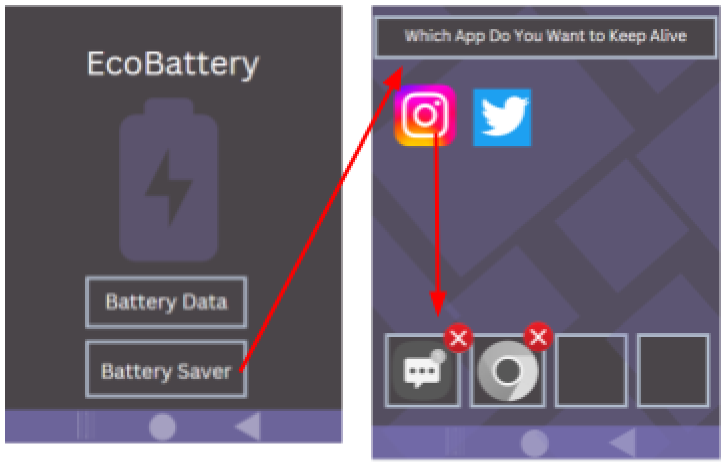
Design Considerations:

In order to assign responsibilities to the objects, we employed design principles such as Expert Doer, High Cohesion, or Low Coupling. The responsibilities of the objects are to store information, send back messages and retrieve information. By the Expert Doer principle, the Low Battery Mode should make the call. However, this choice would lower the cohesion and increase coupling, which would need to pass the retrieved list to the rendering object. The High Cohesion design principle favors assigning R2 to the Low Battery Mode. Therefore, we have a conflict among the design principles.

User Interface Design



Use Case 1



Use Case 2

Glossary of Terms

Term	Definition
Battery	Physical hardware component holding electric power for the device
BatteryPercentage	Remaining battery capacity percentage of total capacity
BatteryLife	BatteryPercentage including Extra Health
Estimated Time	Time the phone's battery life will last while it is either charging or not connected.
Charging	Phone is connected to external charging system that allows the app to tell its user whether the device is being charged properly.
TerminateApplication	Kill the background task of other applications to preserve battery
BatterySaver	Option to the user within the graphical interface to select which apps they would like to keep awake while trying to conserve battery.
BatteryData	Option to the user within the graphical interface to list the information about the user's device, including the battery percentage, battery life, estimated time before device runs out of battery and the external battery's life.

References

[1] ActivityManager : android developers. Android Developers. (n.d.). Retrieved

September 18, 2022, from

https://developer.android.com/reference/android/app/ActivityManager

[2] Batterymanager : android developers. Android Developers. (n.d.). Retrieved

September 18, 2022, from

https://developer.android.com/reference/android/os/BatteryManager

[3] Android 4.1.1 Jelly Bean smartphones. PhoneMore. (n.d.). Retrieved September 18,

2022, from https://www.phonemore.com/systems/android/4-1-1-jelly-bean/