Disk Space Usage Monitor In Android

Document Showing Software Design And Verification

version 3.0

Prepared By:
Anoop K A -ETAJECS005
Asha M B-ETAJECS010
Nisha M C-ETAJECS037
Under The Guidance Of: Savyan P V

April 22, 2012

Contents

Chapter 1

Introduction

1.1 Purpose

The aim of the project is to build an application software That shows the usage of memory for android mobiles.With this application the user can see the used space of memory in the disk drives of the device,which is displayed with user friendly graphical interface . The project follows the incremental model,so that later it can be upgraded as a file explorer for android smart phones.

Fist we are implementing the software application in the eclipse Indigo which is a tool for android development.And the application is tested using the AVD[Android Virtual Device] available in the eclipse. The Eclipse includes java jdk. That is the java developer tool kit. So that it is easy to use the eclipse for developing android applications.

The purpose of making a simple application like the disk space usage monitor is to understand the new tools and techniques in android application development.

1.2 Scope

The scope of the application is that the user will get more interactive window for the required information about the drive. It is better than the inbuilt application for the disk space usage information. The application can be made with different views for displaying the information. The available products are paid software and they are highly expensive. So that the product will be helpful as it is user friendly.

1.3 References

- **Wikipedia**, *http://www.wikipedia.org/*
- **youtube**, *http://www.youtube.com*
- **android developer.com**,*http://developer.android.com/index.html*

Chapter 2

Definitions,Acronyms & Abbreviations

2.1 Definitions

- **JDK**:-The Java Development Kit (JDK) is an Oracle Corporation product aimed at Java developers. Since the introduction of Java, it has been by far the most widely used Java SDK. On 17 November 2006. Sun contributed the source code to the OpenJDK.

- **Android SDK**:-The Android software development kit (SDK) includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator , documentation, sample code, and tutorials. Currently supported development platforms include computers running Linux (any modern desktop Linux distribution), Mac OS X 10.4.9 or later, Windows XP or later. The officially supported integrated development environment (IDE) is Eclipse using the Android Development Tools (ADT) Plugin, though developers may use any text editor to edit Java and XML files then use command line tools (Java Development Kit and Apache Ant are required) to create, build and debug Android applications as well as control attached Android devices (e.g., triggering a reboot, installing software package(s) remotely).

- **AVD**:- AVD is the android virtual device , which is helpful in running the application in a computer. AVD is a virtually creates a mobile device,which helps in running the application in a computer.The program running in avd seems to be like running the application in android mobile phone.

- **Ambiguity between SDK and JDK**:- The JDK forms an extended subset of a software development kit (SDK). In the descriptions which accompany its recent releases for Java SE, EE, and ME, Sun acknowledges that under its terminology, the JDK forms the subset of the SDK which has the responsibility for the writing and running of Java programs. The remainder of the SDK comprises extra software, such as application servers, debuggers, and documentation.

- **java ADT**:- An abstract data type (ADT) is a mathematical model for a certain class of data structures that have similar behavior; or for certain data types of one or more programming languages that have similar semantics. An abstract data type is defined indirectly, only by the operations that may be performed on it and by mathematical constraints on the effects (and possibly cost) of those operations.
  Here for the working of eclipse jdk ,the java ADT is compulsory. The order of installing softwares to the windows based 64bit system is ..

  1. installing the JAVA JDK , which is downloaded from the site http://www.oracle.com/us/technologies/java/ente edition/overview/index.html and installed.
  2. Installing the android SDK. from developer.android.com
  3. downloading the eclipse latest version, now it is eclipse indigo. From thr eclipse.org.
  4. installing the android sdk inside the unzipped folder of eclipse indigo.
  5. updating and installing the sdk and avd manager from inside the eclipse.

2.2 Abbreviations

- **JDK** :-Java Development Kit
- **SDK** :-Software Development Kit
- **NDK** :-Native development kit
- **AVD** :-Android Virtual Device
- **GUI** :-Graphical User interface

Chapter 3

System Overview

The component of android application development and deployment includes a device and a computer that has installed eclipse in it. The eclipse has an AVD manager which include different types of avd s. The Avd is a device emulator which works as an android phone. The prepared application can run in this AVD device. The code will be created in .java format with eclipse .And the package is moulded to .apk file for installing it in to any android device. The interface designing can be done in eclipse with the help of XMl commands. The interface designing has separate section that contains several lay out interfaces buttons images etc.

The main components used in eclipse are the platforms or api levels, which is downloaded from the eclipse home page. It can be downloaded from inside the eclipse. The selected Api for our project is Eclipse platform 1.6 which includes 4 API s.The all other high level API will accept the low level API s. it is upward compatible. Sample device is given below. It is a sony emulator device with 8 gb memory card inserted to it.

Figure 3.1: Sony android device

Chapter 4

Architectural Design

The application development is by the help of the tool ECLIPSE Indigo . And the application can only run in android os. In eclipse there will be a AVD[Android Virtual Device] for testing the application. The AVD includes a virtual sd card according to the given size by the user. The main java file can be searched and retrieved with the project explorer in eclipse. After completion of the code it can be run in the Virtual Android Device[avd]. The in put window is created using eclipse XML lay out .There will be one main window that have three options,which accept the user in put through a tap in the buttons.The application starts when the main icon is tapped.By selecting the first button , And The second window opens with the details of the used, free and total phone memory with some a pictures like a rectangular filled box,and a standard hard disk image. The rectangular box is filled according to the used memory. In eclipse the creation of buttons are standardised and the input is given through tapping the icon of the disk space monitor,Which is created as a Main icon by the help of manifest file in project.

4.1 Data Flow Diagram

The first start of data flow is from the actor , which in this case is the user of the mobile device. The actor starts the data flow by tapping the button of the application . Which leads to the next stage and at last reaches the last stage. When the actor gives in put to the application the scanning module starts by creating an oncreate activity by the java classes and the memory details retrieves by the codes used. The the next stage is to process the details to the picture format. It has to components which are run parallel. The system memory details and the memory card details. Both the details are processed separately and made to picture by the convert module. And at last displayed through a window to the user with the display module onclicklistener.

In the src folder there will be one auto created java file ,when eclipse loads a project to the memory. The java file is modified with the codes to call the ui diagrams or user interface created with the XML codes in eclipse layouts. The main method will be same as the project name.

The first method used is the oncreate method for calling the main xml layout to the front of process.When the public oncreate method calls the R,main. Then the main xml lay out will be displayed. Selecting any button from the main XML lay out will leads to the corresponding activity window, if the selected one is the internal memory , the the internal memory is scanned though the second activity java file which is called from the first main java file. An there will be a back button in the second and third activity windows to come back to the main window.

The third window is for scanning and displaying details of the SD card memory used in the mobile device. The activity flow starts from the main window itself. By selecting the second button from the main screen or main activity window the sd card memory will be displayed by the 3rd window. And the back button in the window will be helpful to come back to the main screen. there will be an exit button only in the main window or main activity, which will be the third option button in the main screen. By pressing the exit button, all the activities will be distroyed and memory will be released by the application.

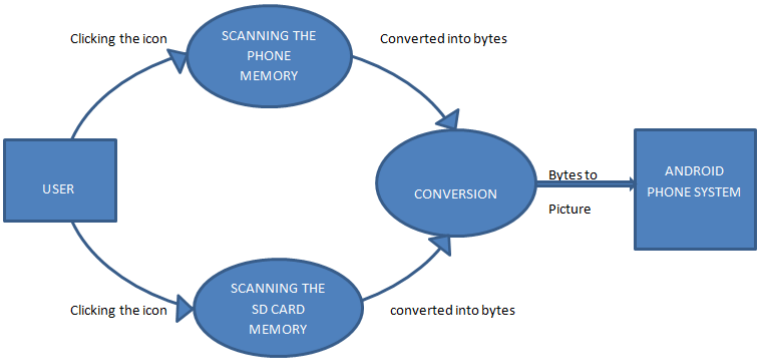
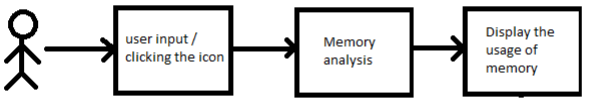
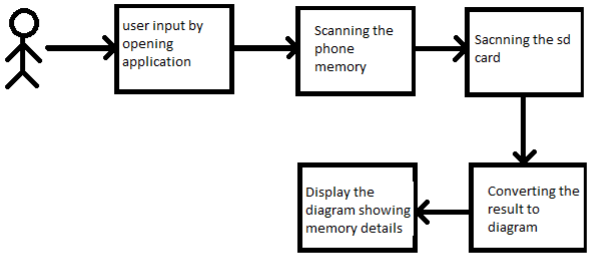


Figure 4.1: Data flow diagram



level 1: data flow diagram



level 2: data flow diagram

Figure 4.2: Data flow details

4.2 Activity Diagram

The activity diagram represents the order of activities in exact sequence.

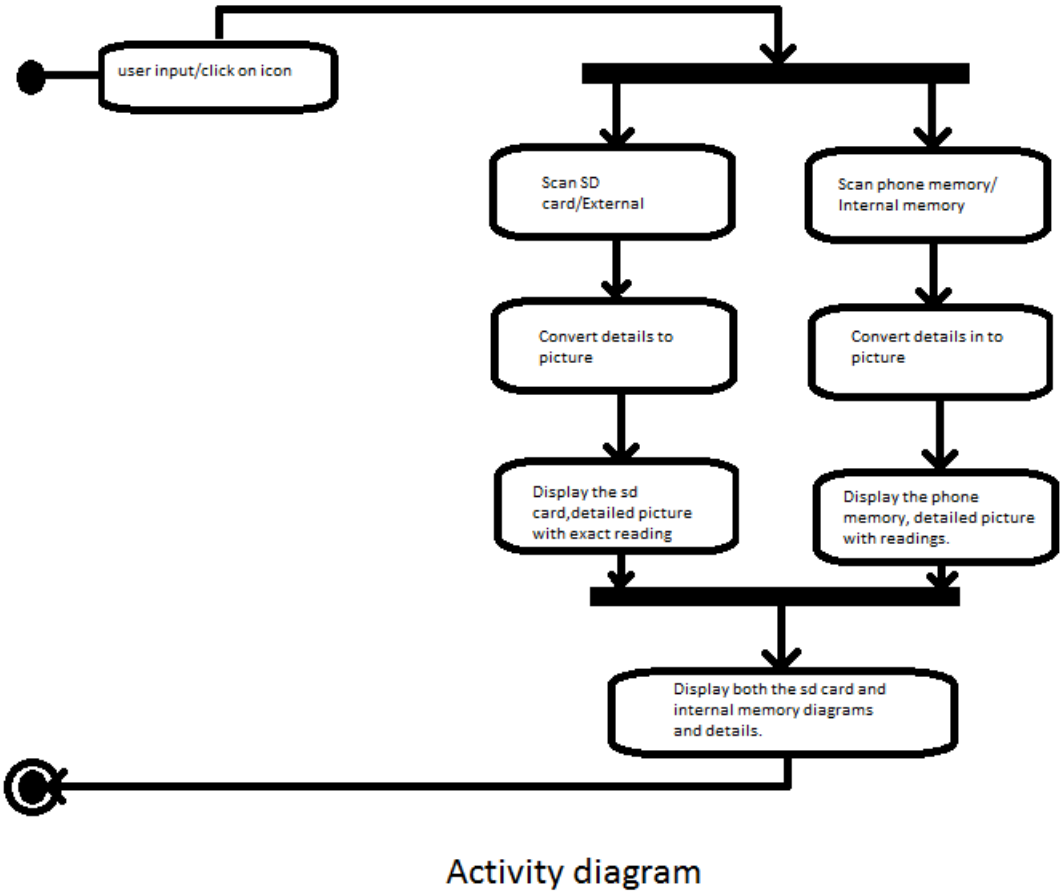


Figure 4.3: Activity diagram

The starting of activity is by the actor which here is the user of the application. When the user select the application the scanner module will work next and it will scan the phone memory and the external memory. the details will be converted to the equivalent diagram by the converter module and the display module will display the diagram later.

The android Activity flow can be expressed through a diagram below.The life cycle of on activity is explained detailed in this figure.The android activities are brought to front when they are called, the activity may not be closed if the process is not destroyed properly.

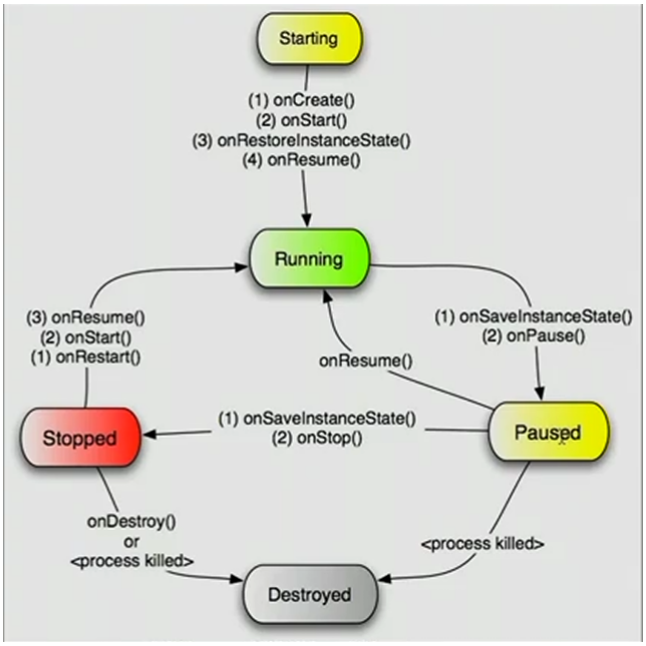


Figure 4.4: Life cycle of an activity

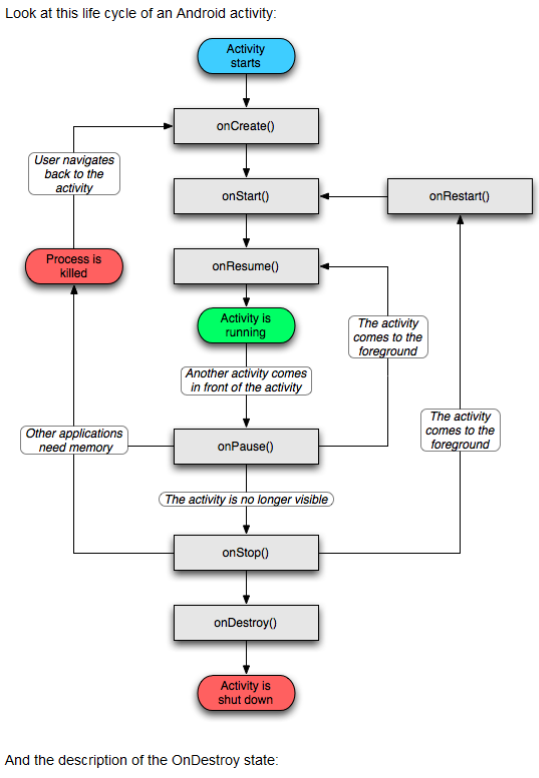


Figure 4.5: Flow chart of activity life cycle

4.3 GUI Diagrams [User Interface Design]

The GUI is the Graphical user interface of the application. Which Shows the user interfaces with the device. In this application there will be one main window that displays the details of memory as a picture. The other user interface is the icon of the application by simply clicking the application icon it starts the memory scanning and converting the result to a displayable diagram for the window. Some sample icons are listed below. Icon can be of any thing related to the memory, or name of the application.

The graphical interface for displaying the details need more images and its included to the processing and conversion modules with the java image process functions.The first image is the icon which is clicked by the user. It activates the main activity by on create method. It creates the window for main activity and displays or brought it to he front.The main icon that is selected for the application is given below

Figure 4.6: main icon

Figure 4.7: Sample icons

The Display interfaces may be different with different diagrams. It is possible to have several views with same details. Its is an optional requirement.Examples of some of the out put graphical user interfaces for disk usage monitor systems are listed below. The Android Virtual Device view will be different with the real device view.

In our application the main activity window will be like the diagram below.There will be two more windows that shows the phone memory and sd card memory separately.The main window will lead to the second window and third window which will be parallel to he second activity window. There will be option for coming back to the main window from all other windows.

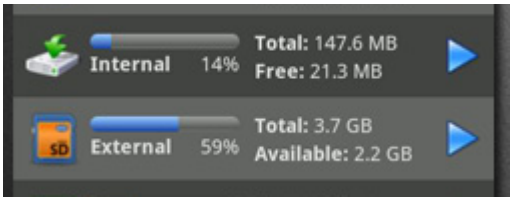


Figure 4.8: Sample GUI for memory analysis

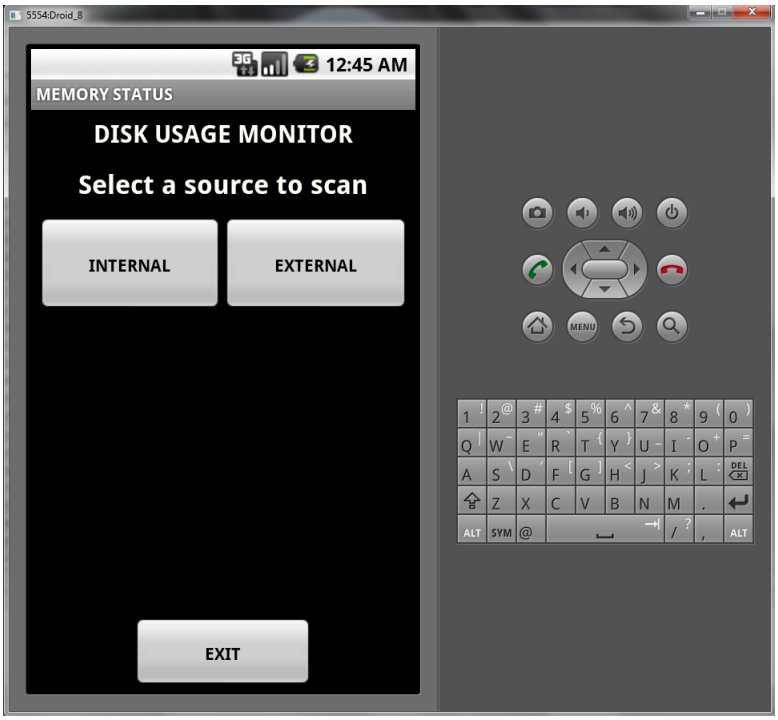


Figure 4.9: The main activity window

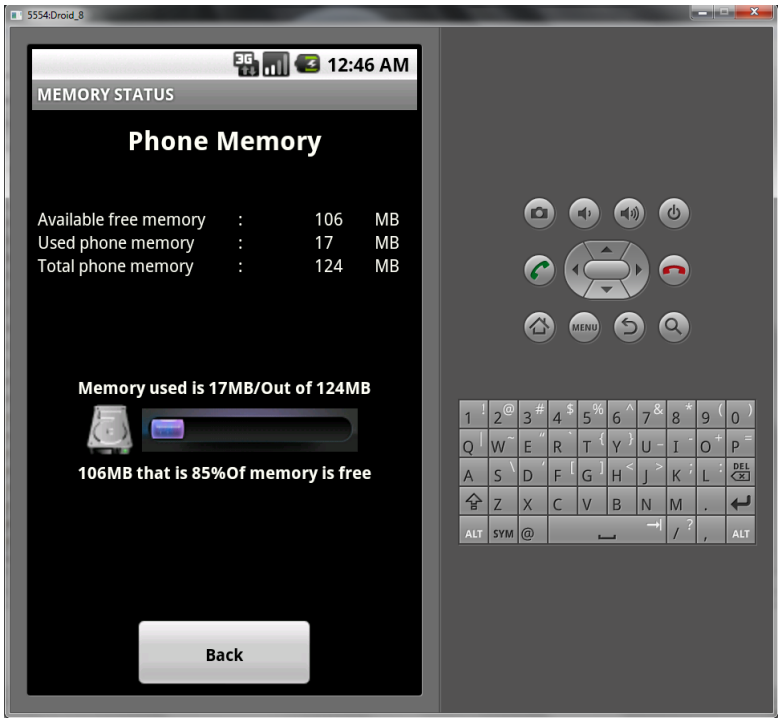


Figure 4.10: Secon activity - phone memory

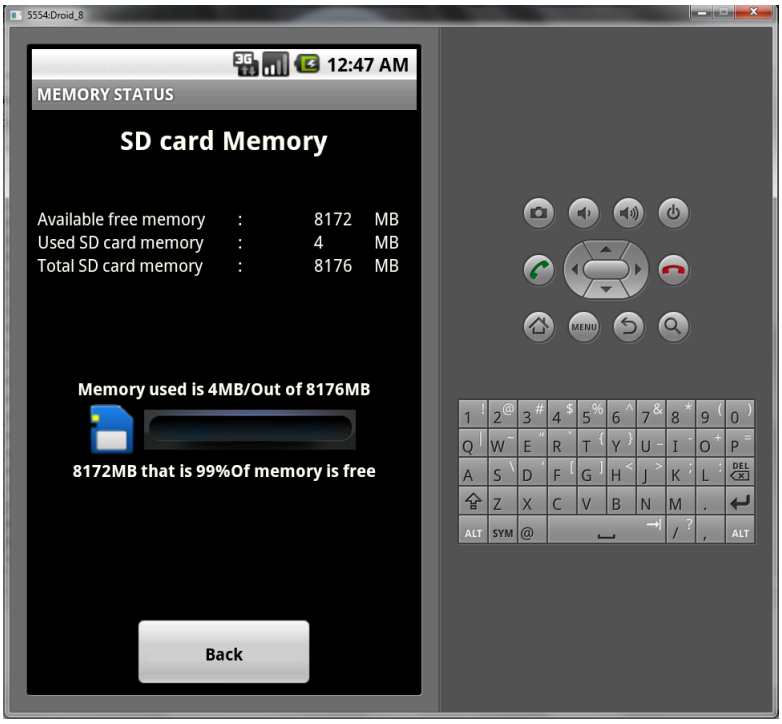


Figure 4.11: Activity window 3-sd card

Chapter 5

Component Design

Component design is a description of various components of the system. It includes a front end and back end.

5.1 Front End

The front end is the software used to create the user interface. Which is in our case is the XML codes in eclipse indigo. Through the XML graphical or program interface it is much easy to create a standard user interface, which may includes the buttons and texts necessary for the application. In our application there will be three windows or activities.They are created with the XML code so that each activity can be bring front to the user. The user has to select one activity from the main activity window which is created when the icon is taped on application is started by the user.The options are memory card and phone memory. When the user select one of the to windows the xml code for that activity works and user will be provided with the corresponding window.

5.2 Back End

The back end of eclipse is java program codes. Which includes the classes and functions to scan memory and conversion of data in to image. The java files are accessed with the help of the user interface created with the front end application. The java codes here includes functions to call the use disk space in the device memory and the memory card.When the second window is invoked by the user the functions for the second window that is the memory card space usage will be displayed to the user. Its is done by linking the back end with the front end of the application. The linking is done through a manifest file in eclipse. The icon linking also done with the manifest file.

Chapter 6

Modules & Functions

The java class uses modules and functions for the creating the application. The android stack is as follows..

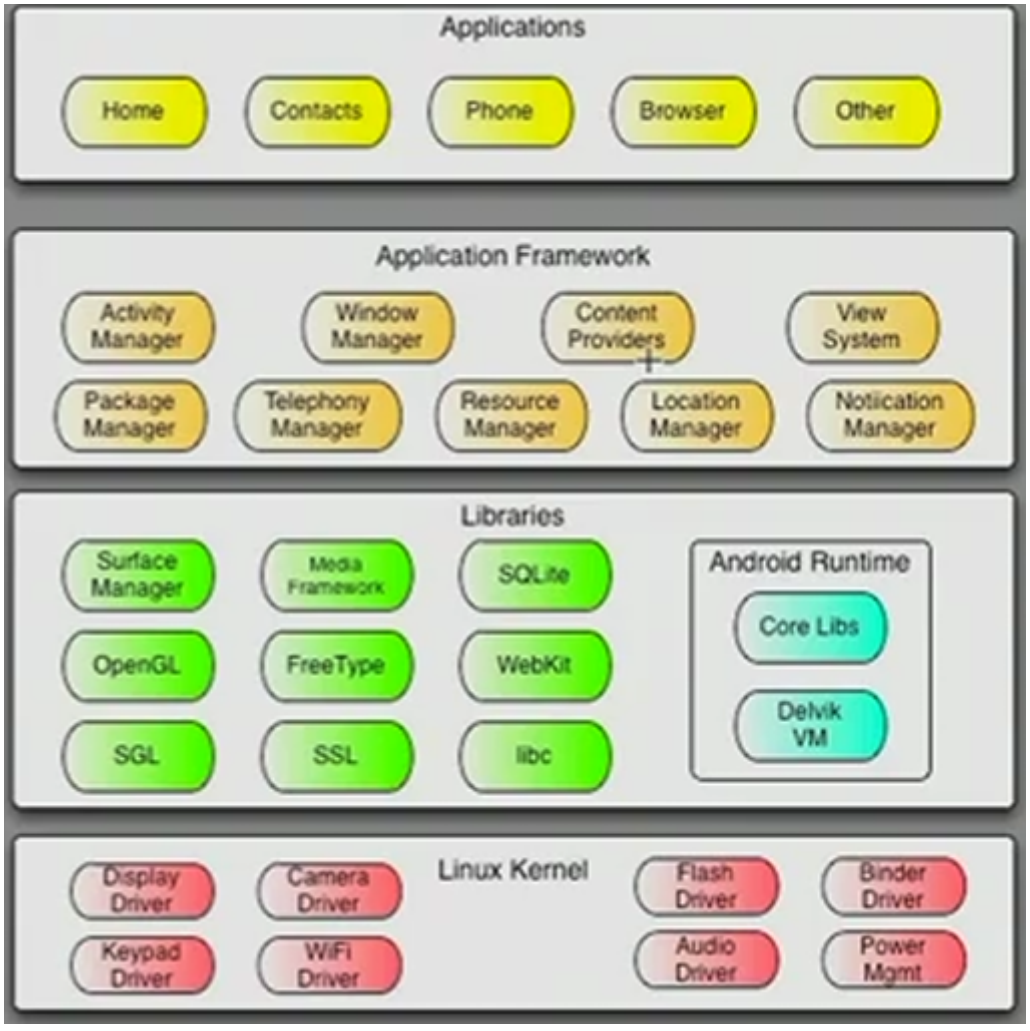


Figure 6.1: Android stack

6.1 Module- Scan memory

The first step of the process is to scan the memory of the device to retrieve the used and free space of memory in either sd card or system memory .The functions that can be used to get the details of memory are described below. The scanning module is the process to create the exact details of memory, it is further processed to make a pictorial out put that is displayed to the user.

6.1.1 get total space

getTotalSpace:
public long getTotalSpace()
Returns the size of the partition named by this abstract pathname.
Returns:
The size, in bytes, of the partition or 0L if this abstract pathname does not name a partition
Throws:
SecurityException - If a security manager has been installed and it denies RuntimePermission("getFileSystemAttributes") or its SecurityManager.checkRead(String) method denies read access to the file named by this abstract pathname.

The above is a common java code , But in android phones , the eclipse uses separate functions.
for SD card:

```
public static long sd_card_total(){

    File path = Environment.getExternalStorageDirectory();
    StatFs stat = new StatFs(path.getPath());
    long free_memory = (stat.getBlockCount()/(1024)) * (stat.getBlockSize()/(1024)); //return value

    return free_memory;
}
```

For phone memory:

```
public static long phone_storage_total(){
         File path = Environment.getDataDirectory();
         StatFs stat = new StatFs(path.getPath());
         long free_memory = stat.getBlockCount() * stat.getBlockSize(); //return value is in bytes

         return free_memory/(1024*1024);
     }
```

6.1.2 get free space

getFreeSpace:
public long getFreeSpace()
Returns the number of unallocated bytes in the partition named by this abstract path name.
The returned number of unallocated bytes is a hint, but not a guarantee, that it is possible to use most or any of these bytes. The number of unallocated bytes is most likely to be accurate immediately after this call. It is likely to be made inaccurate by any external I/O operations including those made on the system outside of this virtual machine. This method makes no guarantee that write operations to this file system will succeed.
Returns:
The number of unallocated bytes on the partition 0L if the abstract pathname does not name a partition. This value will be less than or equal to the total file system size returned by getTotalSpace().
Throws:
Security Exception - If a security manager has been installed and it denies Run timePermission("*getFileSystemAttributes*" or its SecurityManager.checkRead(String) method denies read access to the file named by this abstract pathname.

In the eclipse the code for function to get sd card free space will be as follows.

```
public static long sd_card_free(){

    File path = Environment.getExternalStorageDirectory();
    StatFs stat = new StatFs(path.getPath());

    long free_memory = (stat.getAvailableBlocks()/1024) * (stat.getBlockSize()/1024); //return value
```

```
    return free_memory;
}
```

Function for memory card free space:

```
 public static long phone_storage_free(){

         File path = Environment.getDataDirectory();
         StatFs stat = new StatFs(path.getPath());
         long free_memory = stat.getAvailableBlocks() * stat.getBlockSize(); //return value is in by

         return free_memory/(1024*1024);

     }
```

6.1.3 get usable space

getUsableSpace
public long getUsableSpace()
Returns the number of bytes available to this virtual machine on the partition named by this abstract pathname. When possible, this method checks for write permissions and other operating system restrictions and will therefore usually provide a more accurate estimate of how much new data can actually be written than getFreeSpace().
The returned number of available bytes is a hint, but not a guarantee, that it is possible to use most or any of these bytes. The number of unallocated bytes is most likely to be accurate immediately after this call. It is likely to be made inaccurate by any external I/O operations including those made on the system outside of this virtual machine. This method makes no guarantee that write operations to this file system will succeed.
Returns:
The number of available bytes on the partition or 0L if the abstract pathname does not name a partition. On systems where this information is not available, this method will be equivalent to a call to getFreeSpace().
Throws:
Security Exception - If a security manager has been installed and it denies Run time Permission("*getFileSystemAttributes*
or its Security Manager.checkRead(String) method denies read access to the file named by this abstract pathname.
Eclipse code for both sd card and phone memory is as follows:
Sdcard:

```
public static long sd_card_used(){

    File path = Environment.getExternalStorageDirectory();
    StatFs stat = new StatFs(path.getPath());
    long free_memory = (((stat.getBlockCount()/(1024)) - (stat.getAvailableBlocks()/(1024)))) * (sta

    return free_memory;
```

phone memory:

```
public static long phone_storage_used(){
         File path = Environment.getDataDirectory();
         StatFs stat = new StatFs(path.getPath());
         long free_memory = (stat.getBlockCount() - stat.getAvailableBlocks()) * stat.getBlockSize()

         return free_memory/(1024*1024);
     }
```

6.2 Module-Convert data

It is the second step of process to convert the details that are given by the scanning module to a pictorial representation.It include the functions to include pictures and icons. The display will be according to the

used memory status. The rectangular bar fills according to the memory that is been used.the Graphic functions will be called through this module. The percentage of used memory is calculated and according to that information the picture graph is changed. the percentage of memory used is found by

```
float r=(phone_storage_used()*100)/(phone_storage_total());
```

the image is declared first to certain value,with the code below.

```
ImageButton image = (ImageButton) findViewById(R.id.imageButton1);
```

then the image is changed with the percentage value retrieved by r. example, if the value is less than 5%then the following code is used

```
 if(r<=5.00&&r>1.00){
      image.setImageResource(R.drawable.mem_v3_05);
       }
```

Thus both the memory is converted to graph by chnging picture according to the value retrieved.

6.3 Module-Showdata

It is the last stage of process that displays the datais as a pictorial representation. The gui diagram shows the detailed out put interface of the application.
The total window will be displayed with the command given below.

```
public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.disk_internal_monitor);
```

the r.internal changed to r. external in the external activity to show the gui for external scanning result of memory.

Chapter 7

Conclusion

This project aim to implement an application for android .it requires more knowledge about the tools that can be used to implement such application. Building such an application helps to learn about the new technologies adapted to implement and design different software products. The Disk Space Usage monitor,monitors the usage of the memory in any android mobile.it can be tested in the android avd manager. And can be deployed as an .apk file. Which is application that can install in an android phone.