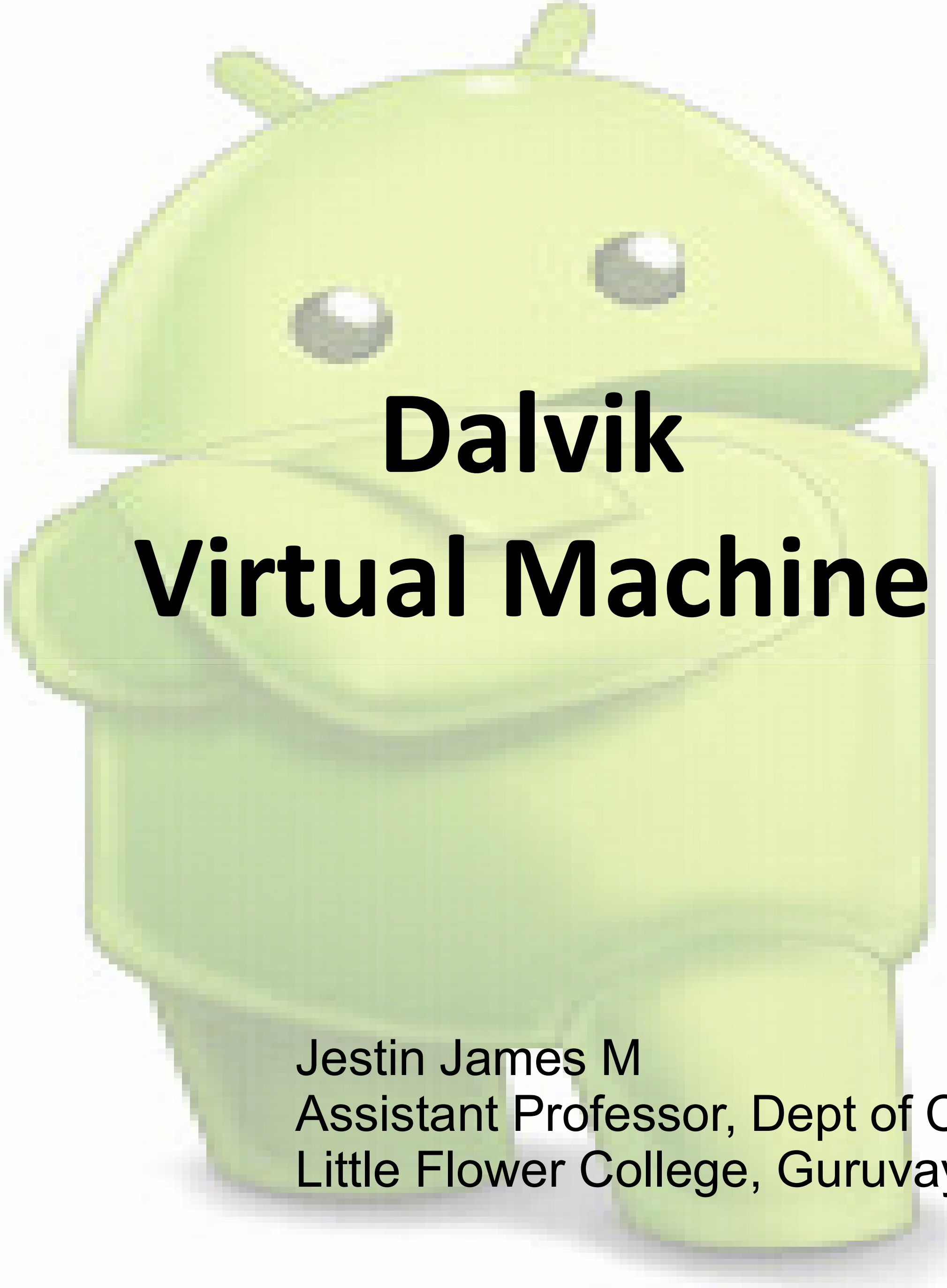
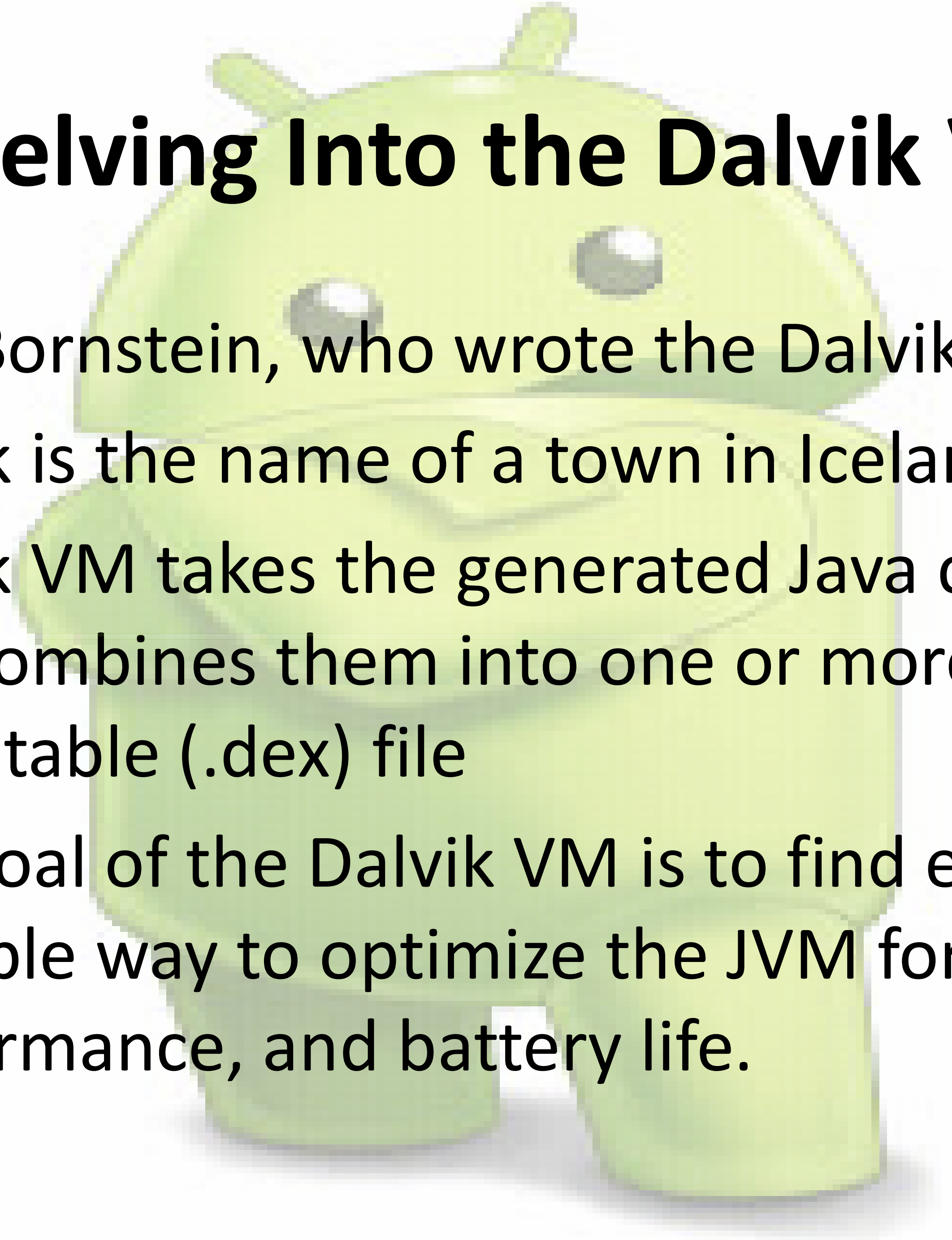


Dalvik
Virtual Machine

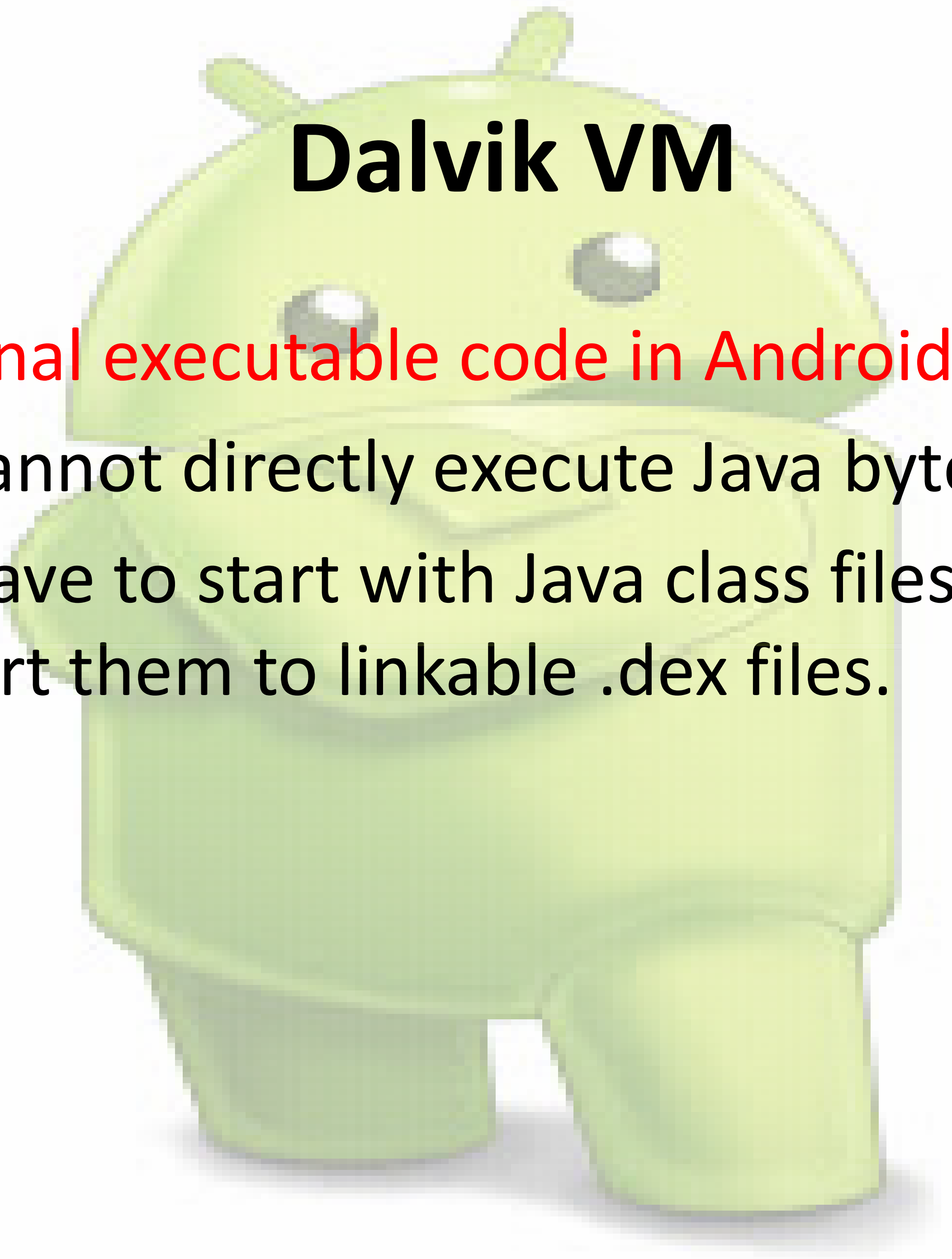
Jestin James M
Assistant Professor, Dept of Computer Science
Little Flower College, Guruvayoor

Delving Into the Dalvik VM



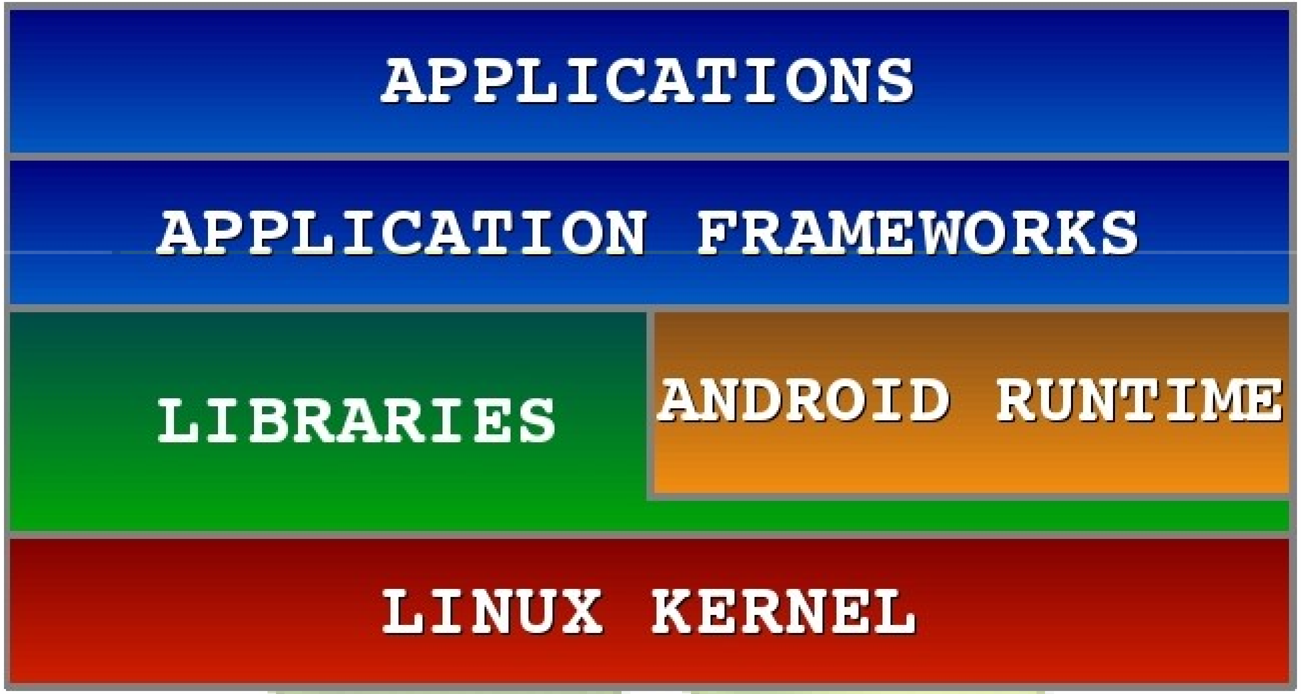
- Dan Bornstein, who wrote the Dalvik VM
- Dalvik is the name of a town in Iceland
- Dalvik VM takes the generated Java class files and combines them into one or more Dalvik Executable (.dex) file
- The goal of the Dalvik VM is to find every possible way to optimize the JVM for space, performance, and battery life.

Dalvik VM



- The final executable code in Android .dex files
- you cannot directly execute Java byte code
- you have to start with Java class files and then convert them to linkable .dex files.

Android Software Stack



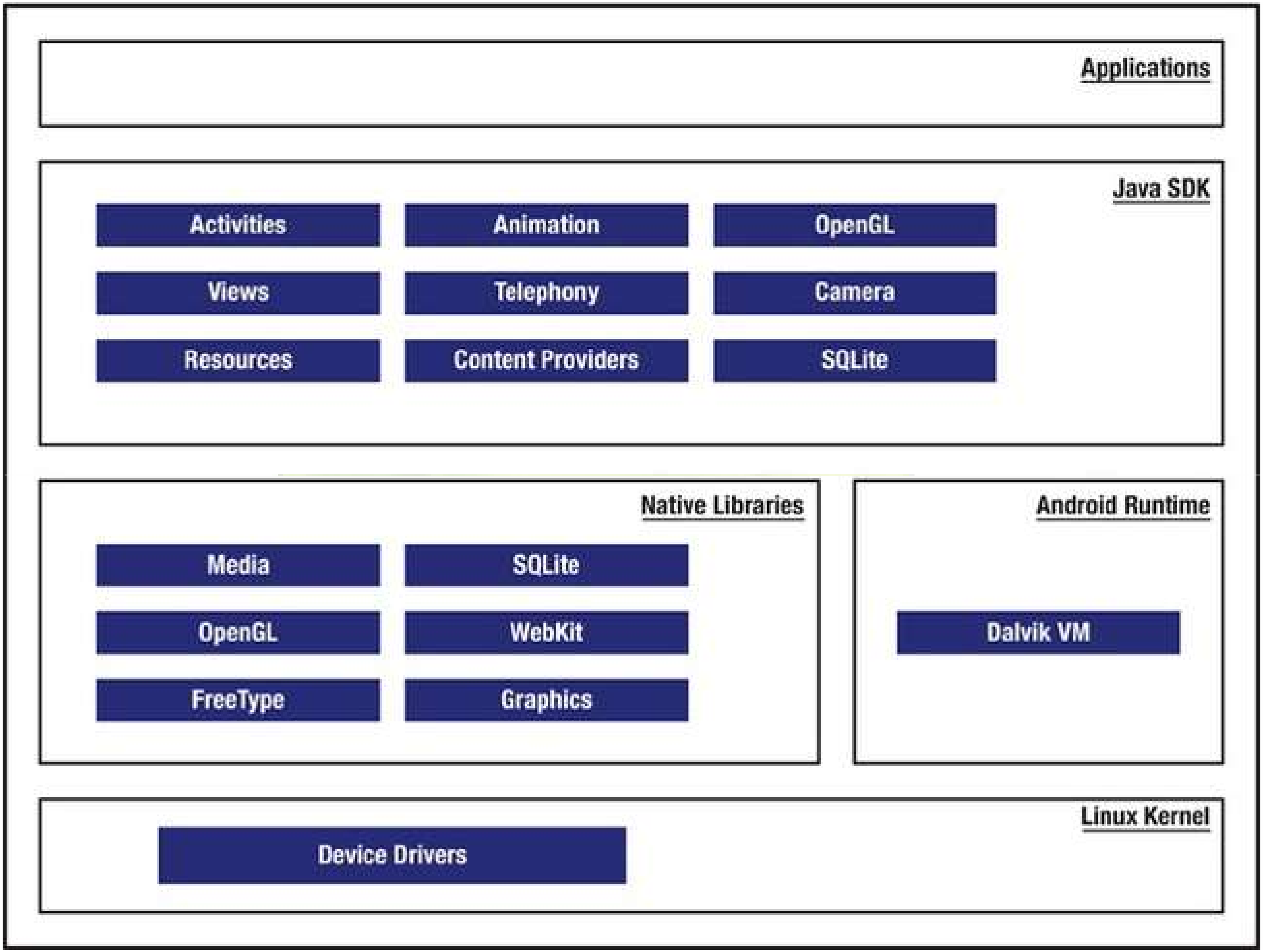
APPLICATIONS

APPLICATION FRAMEWORKS

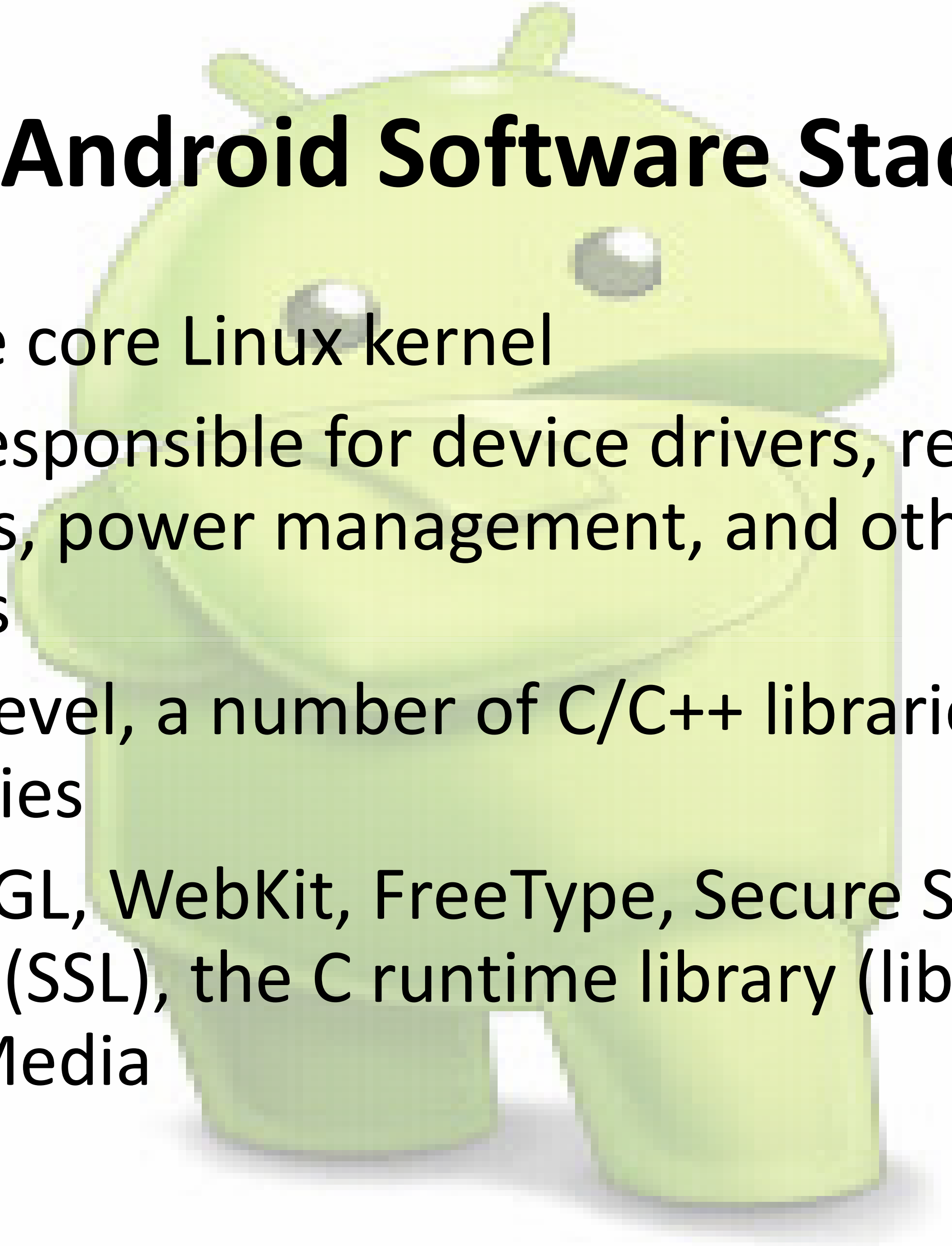
LIBRARIES

ANDROID RUNTIME

LINUX KERNEL

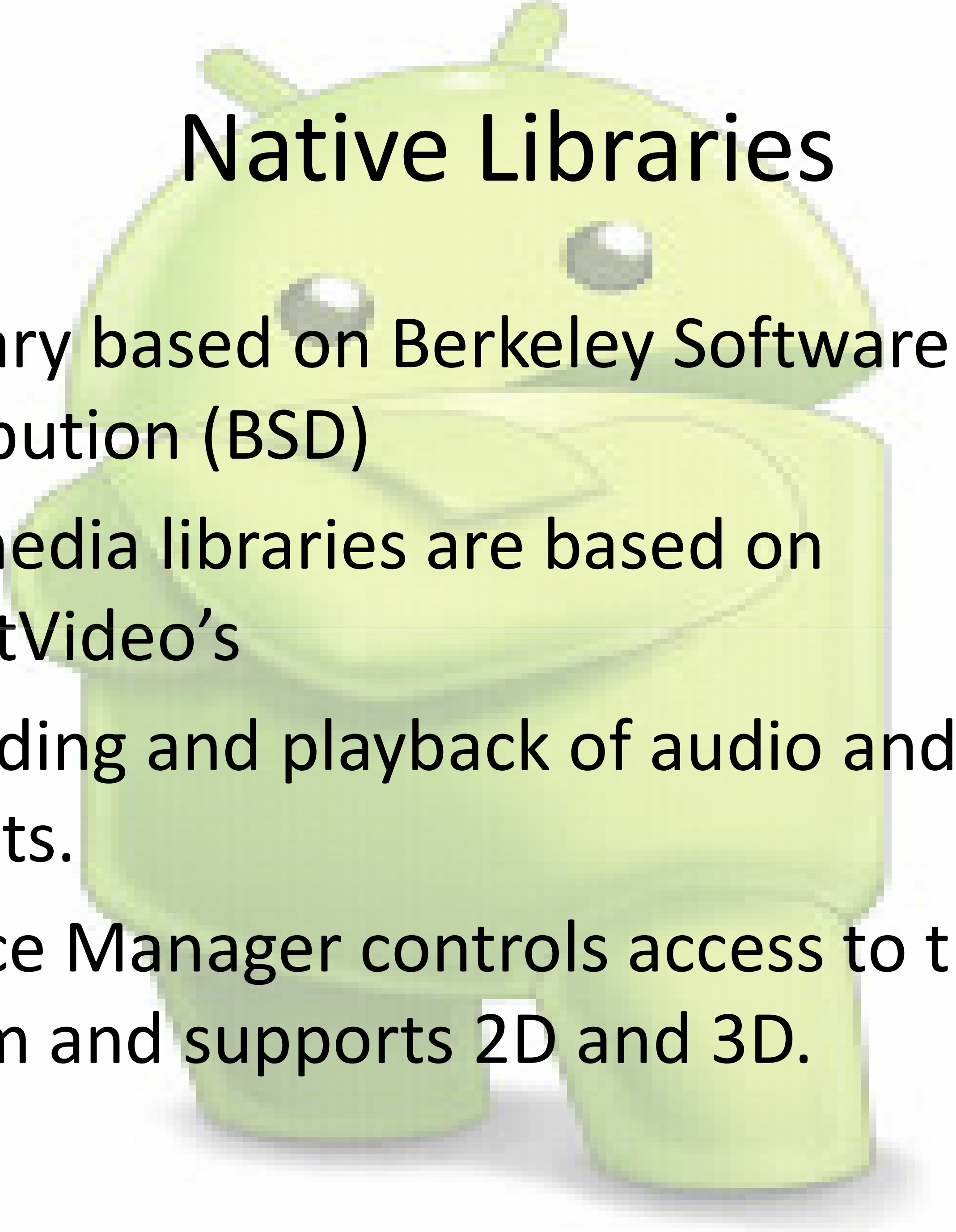

Applications
Java SDK
Activities
Animation
OpenGL
Views
Telephony
Camera
Resources
Content Providers
SQLite
Native Libraries
Media
SQLite
OpenGL
WebKit
FreeType
Graphics
Android Runtime
Dalvik VM
Linux Kernel
Device Drivers

Android Software Stack



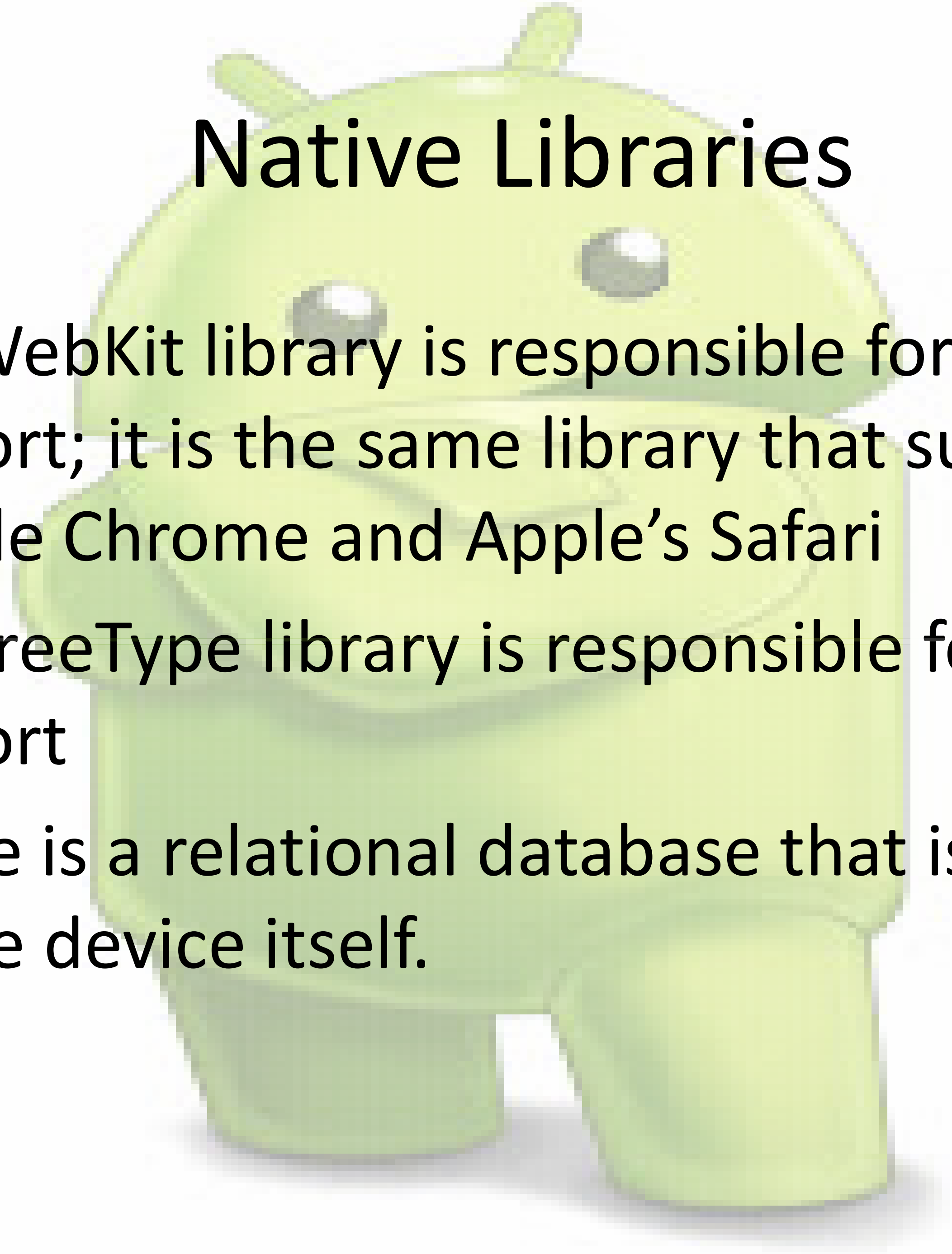
- At the core Linux kernel
- It is responsible for device drivers, resource access, power management, and other OS duties
- next level, a number of C/C++ libraries(native Libraries
- OpenGL, WebKit, FreeType, Secure Sockets Layer (SSL), the C runtime library (libc), SQLite, and Media

Native Libraries



- C library based on Berkeley Software Distribution (BSD)
- The media libraries are based on PacketVideo’s
- recording and playback of audio and video formats.
- Surface Manager controls access to the display system and supports 2D and 3D.

Native Libraries

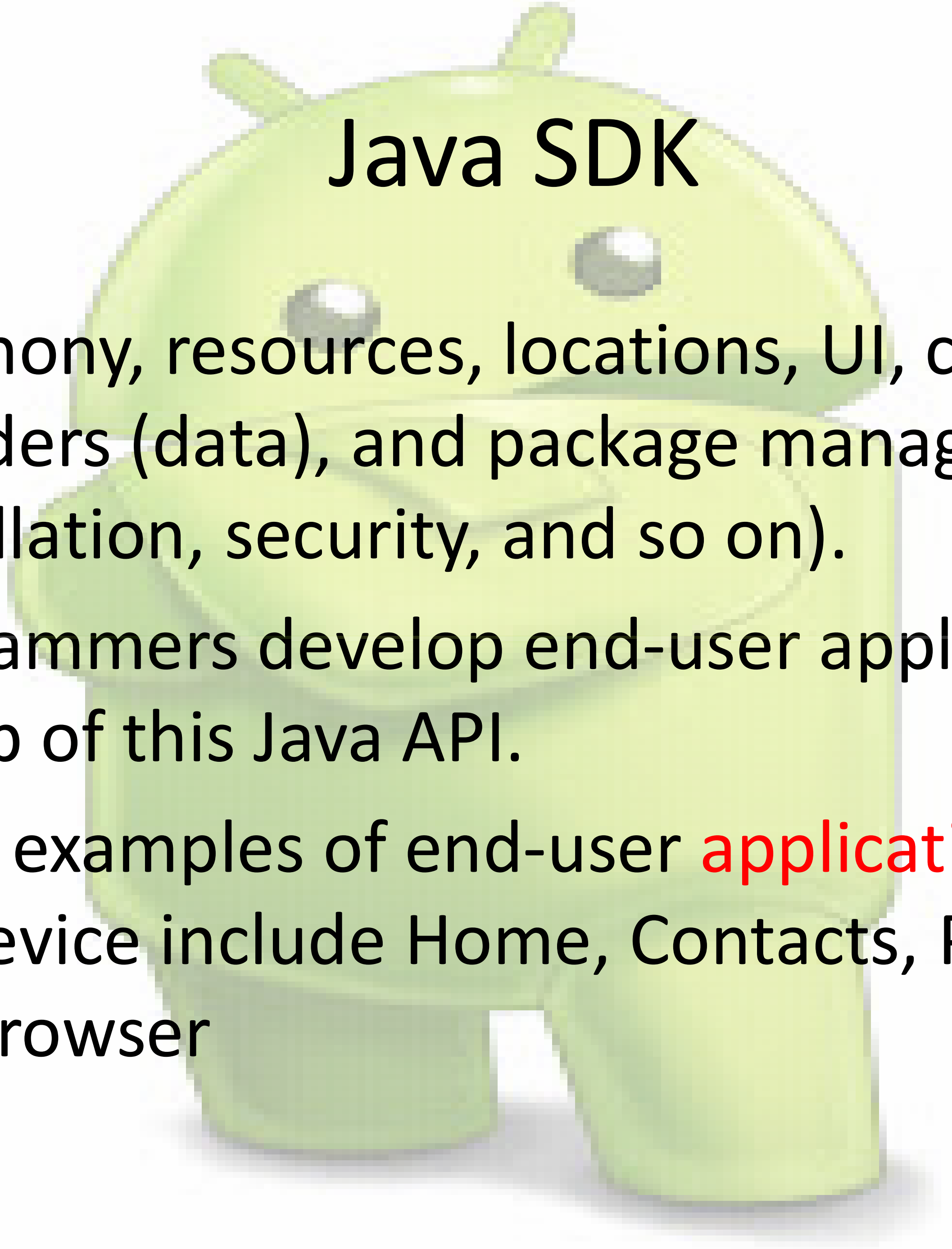


- The WebKit library is responsible for browser support; it is the same library that supports Google Chrome and Apple’s Safari
- The FreeType library is responsible for font support
- SQLite is a relational database that is available on the device itself.

Android runtime

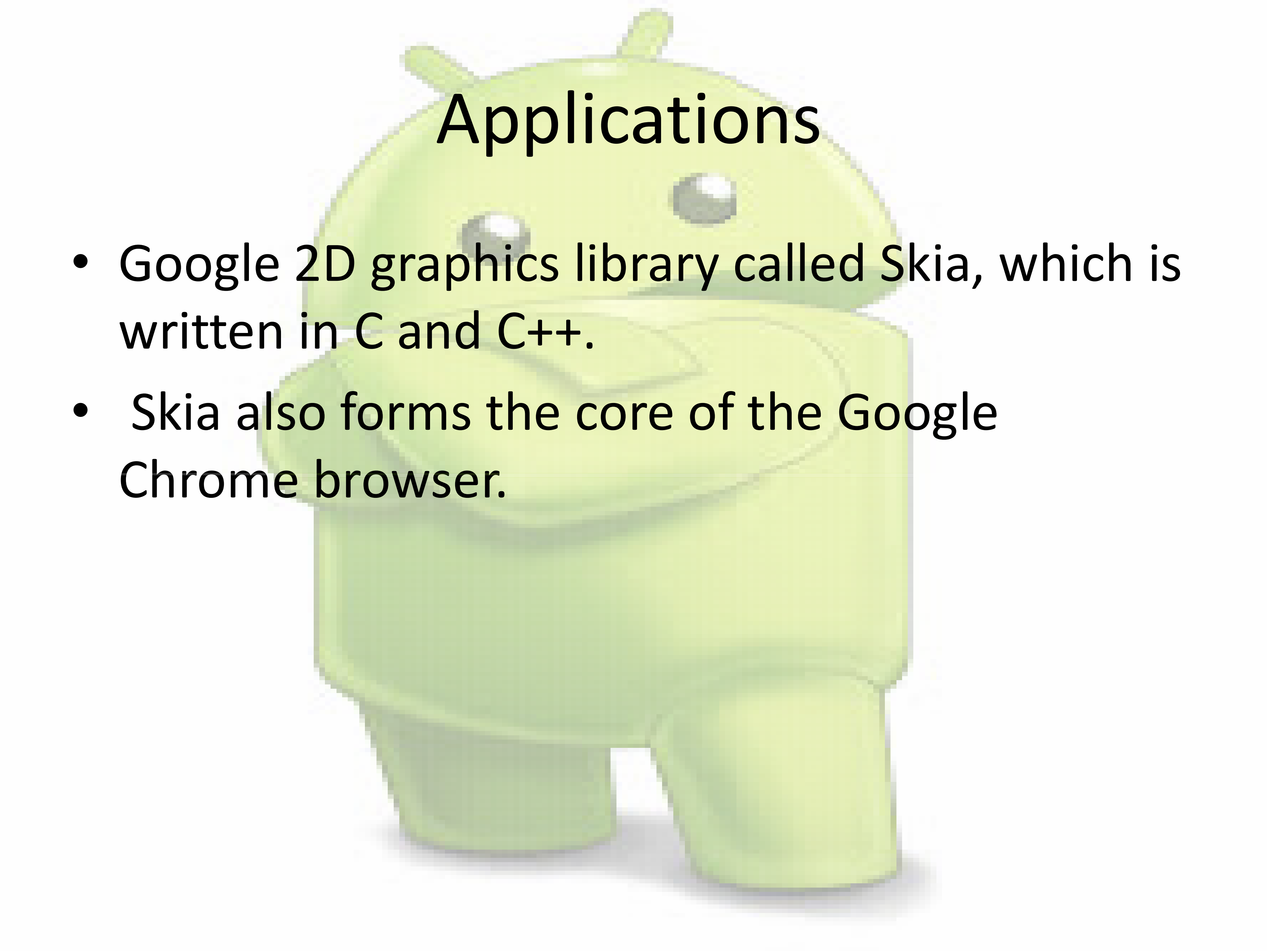
- Most of the application framework accesses these core libraries through the Dalvik VM, the gateway to the Android platform
- As Java applications access these core libraries, each application gets its own VM instance.

Java SDK



- telephony, resources, locations, UI, content providers (data), and package managers (installation, security, and so on).
- Programmers develop end-user applications on top of this Java API.
- Some examples of end-user applications on the device include Home, Contacts, Phone, and Browser

Applications



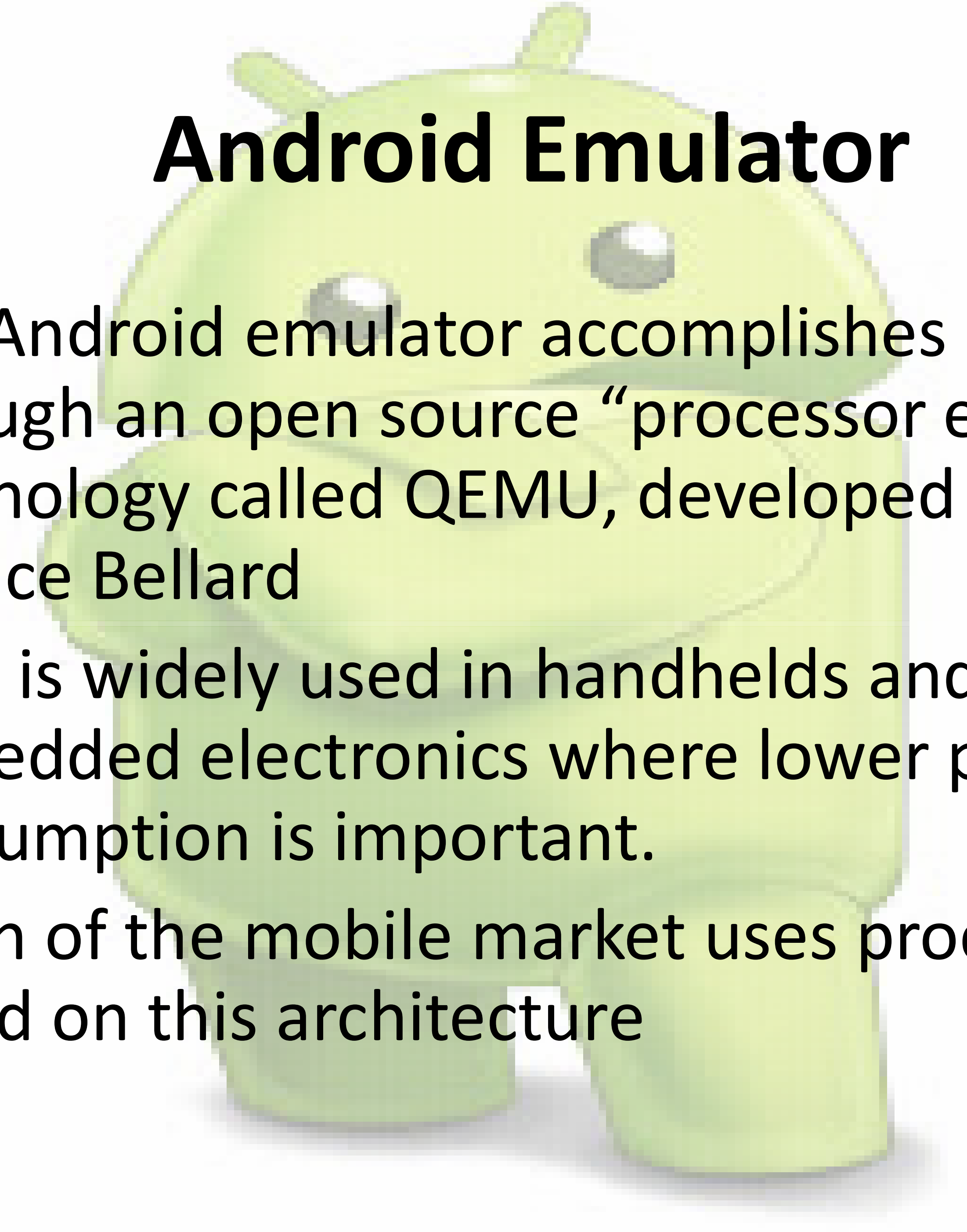
- Google 2D graphics library called Skia, which is written in C and C++.
- Skia also forms the core of the Google Chrome browser.

Developing an End-User Application with the Android SDK

Android Emulator

- The Android SDK ships with an Eclipse plug-in called Android Development Tools (ADT).
- Eclipse is a IDE tool for developing, debugging, and testing your Java applications
- The full-featured Android emulator mimics most of the device features.
- The emulator limitations include USB connections, camera and video capture, headphones, battery simulation, Bluetooth, Wi-Fi, NFC, and OpenGL ES 2.0.

Android Emulator

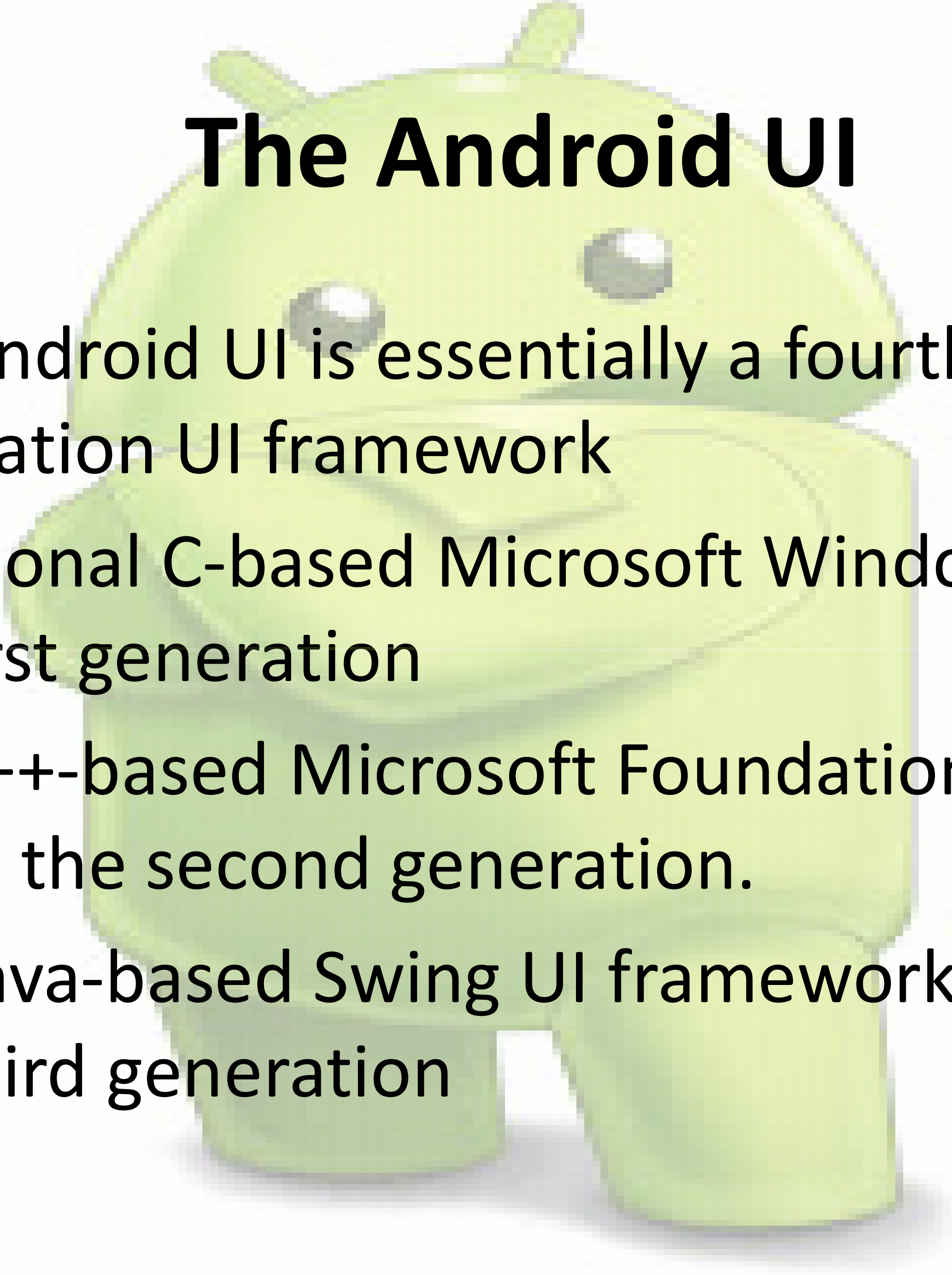


- The Android emulator accomplishes its work through an open source “processor emulator” technology called QEMU, developed by Fabrice Bellard
- ARM is widely used in handhelds and other embedded electronics where lower power consumption is important.
- Much of the mobile market uses processors based on this architecture

Android Emulator

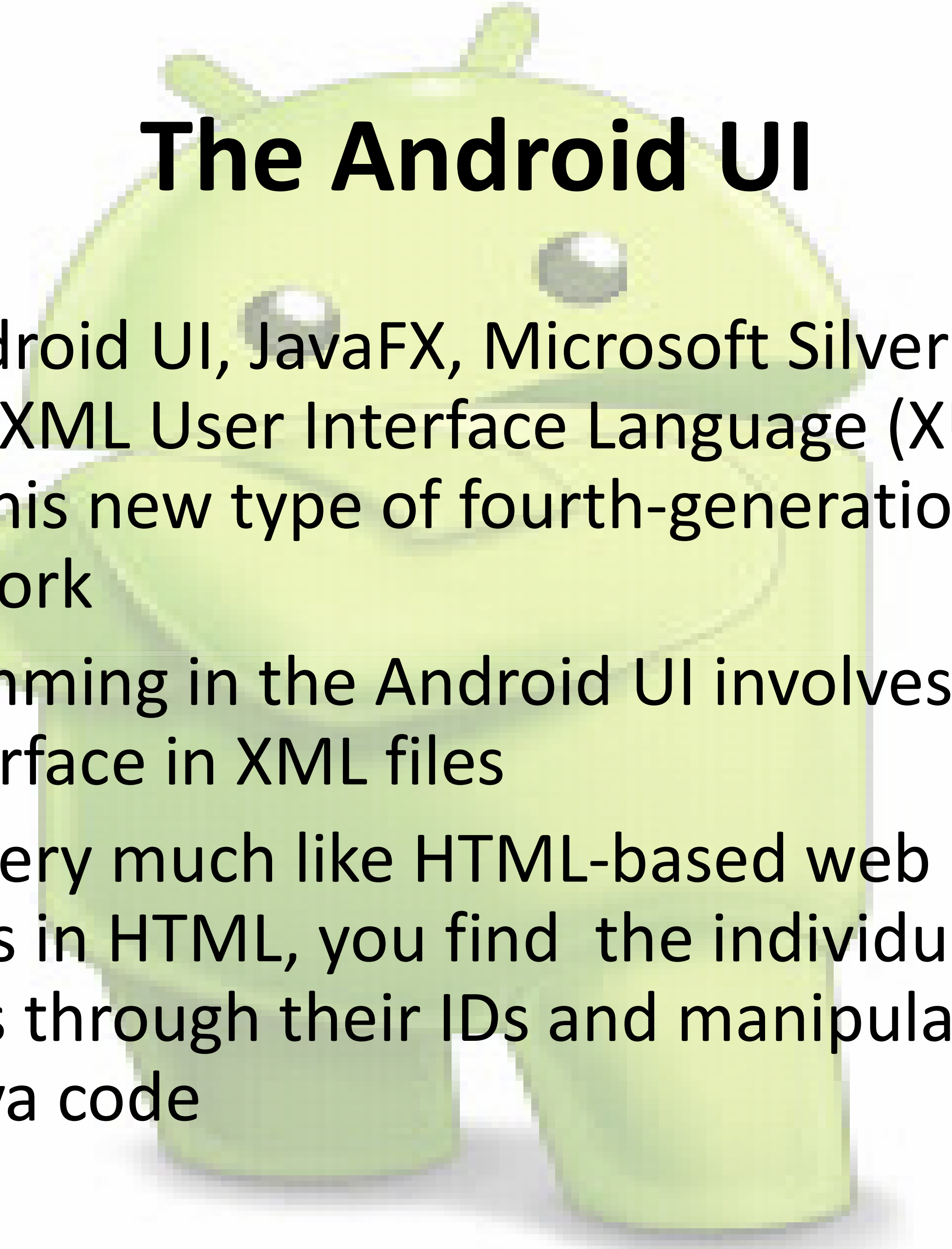
- ARM is a 32-bit microprocessor architecture based on Reduced Instruction Set Computing (RISC),

The Android UI



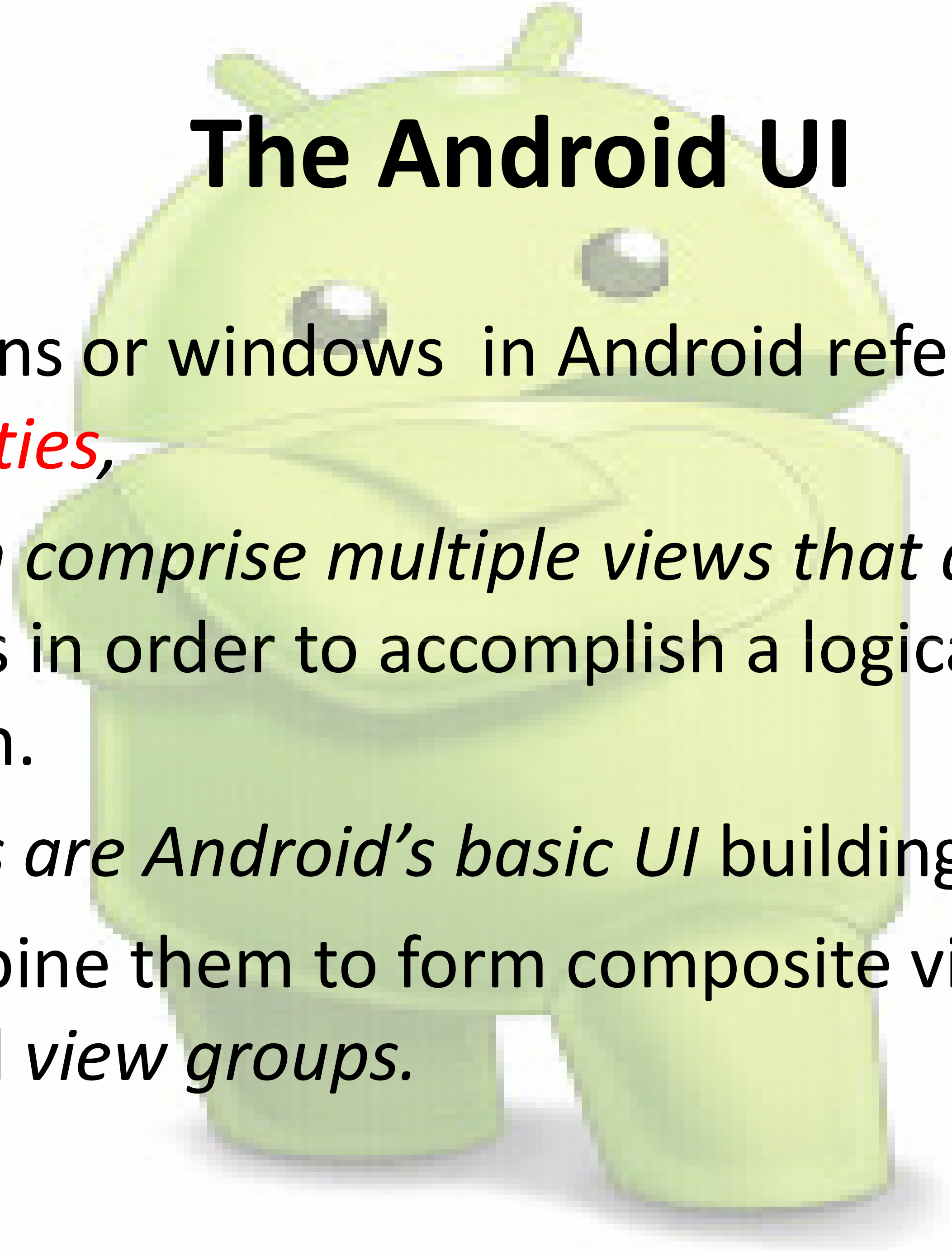
- The Android UI is essentially a fourth-generation UI framework
- traditional C-based Microsoft Windows API the first generation
- the C++-based Microsoft Foundation Classes (MFC) the second generation.
- The Java-based Swing UI framework would be the third generation

The Android UI



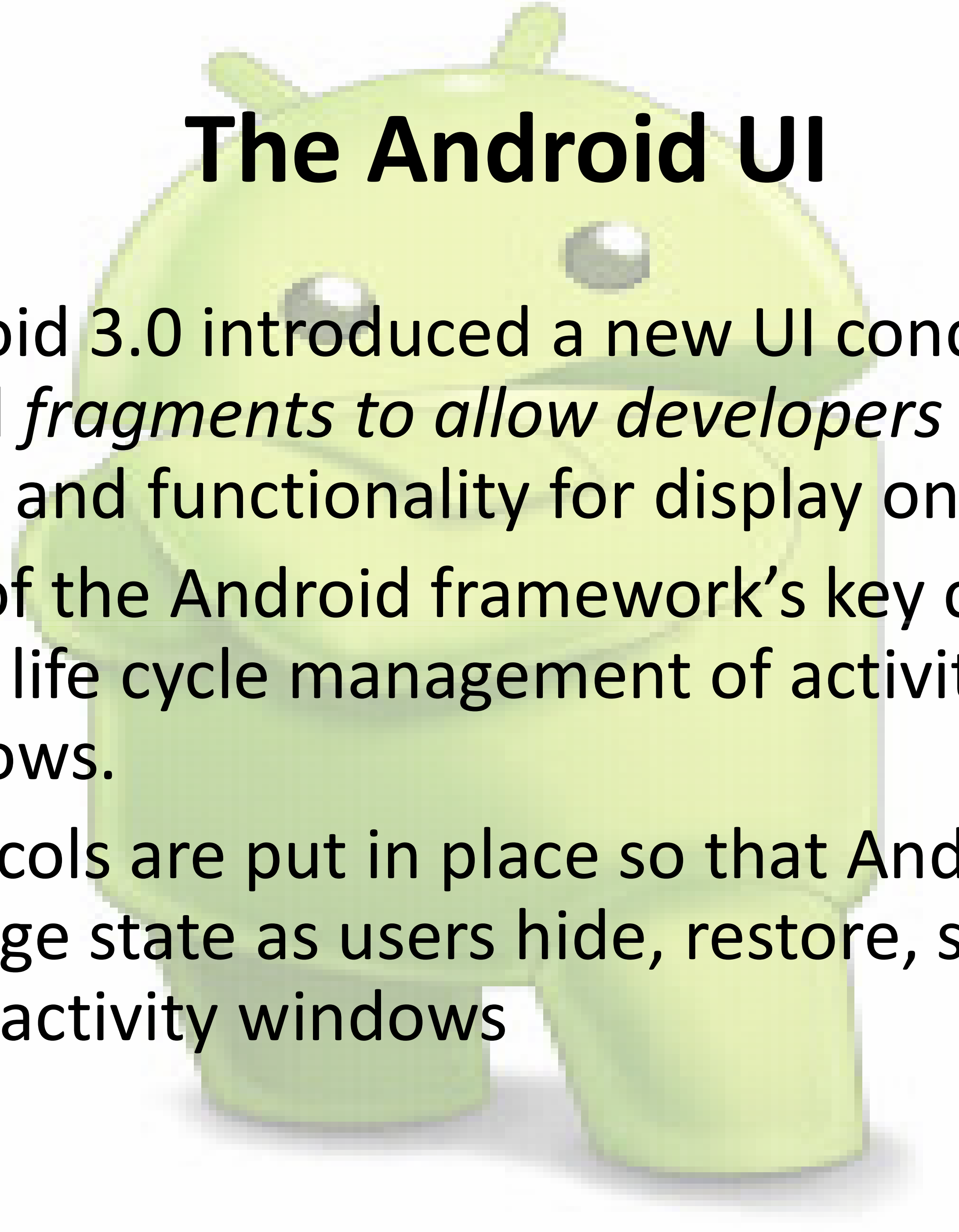
- The Android UI, JavaFX, Microsoft Silverlight, and Mozilla XML User Interface Language (XUL) fall under this new type of fourth-generation UI framework
- Programming in the Android UI involves declaring the interface in XML files
- This is very much like HTML-based web pages. Much as in HTML, you find the individual controls through their IDs and manipulate them with Java code

The Android UI



- Screens or windows in Android referred to as *activities,*
- *which comprise multiple views that a user* needs in order to accomplish a logical unit of action.
- *Views are Android's basic UI* building blocks
- combine them to form composite views called *view groups.*

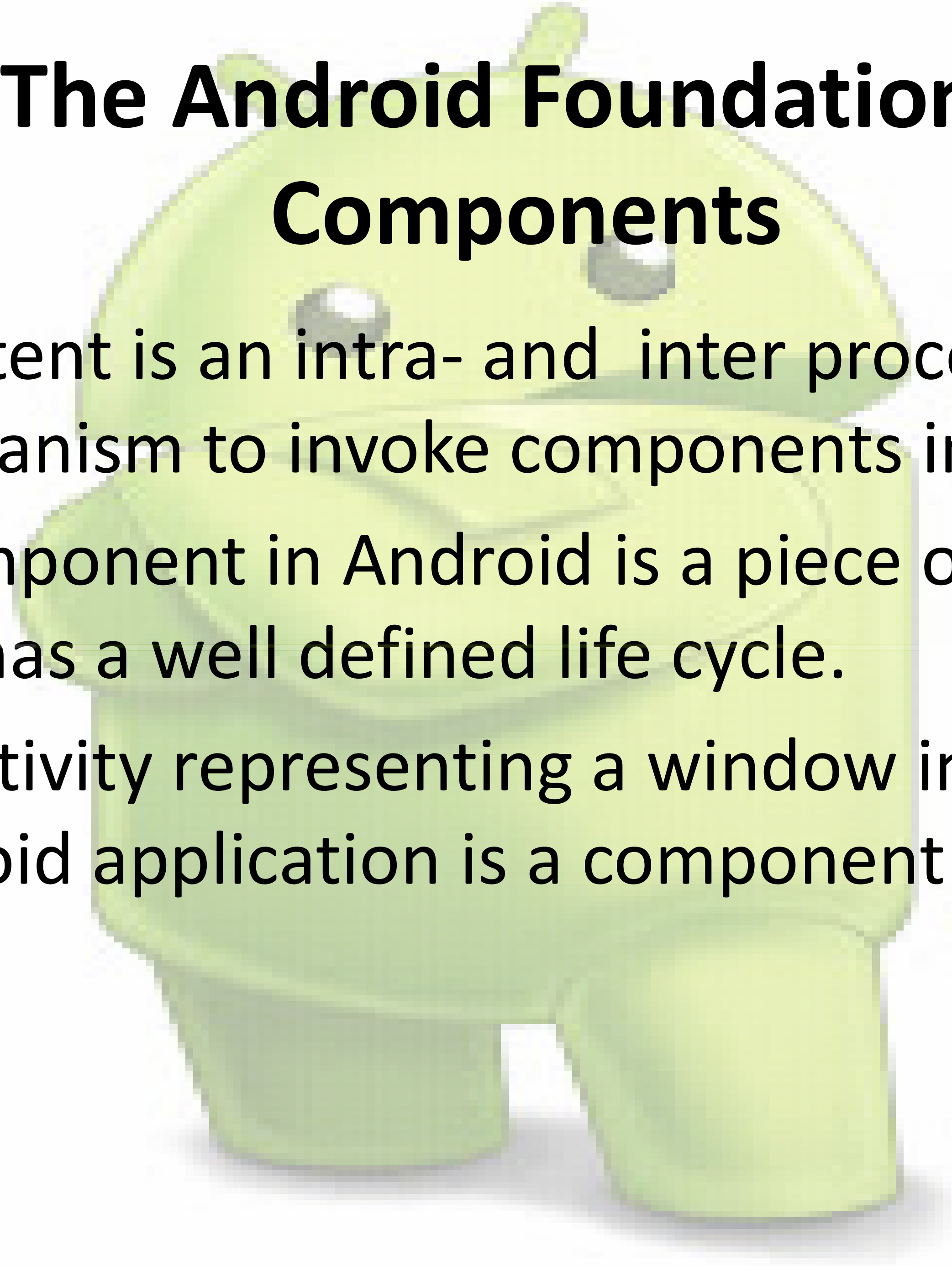
The Android UI



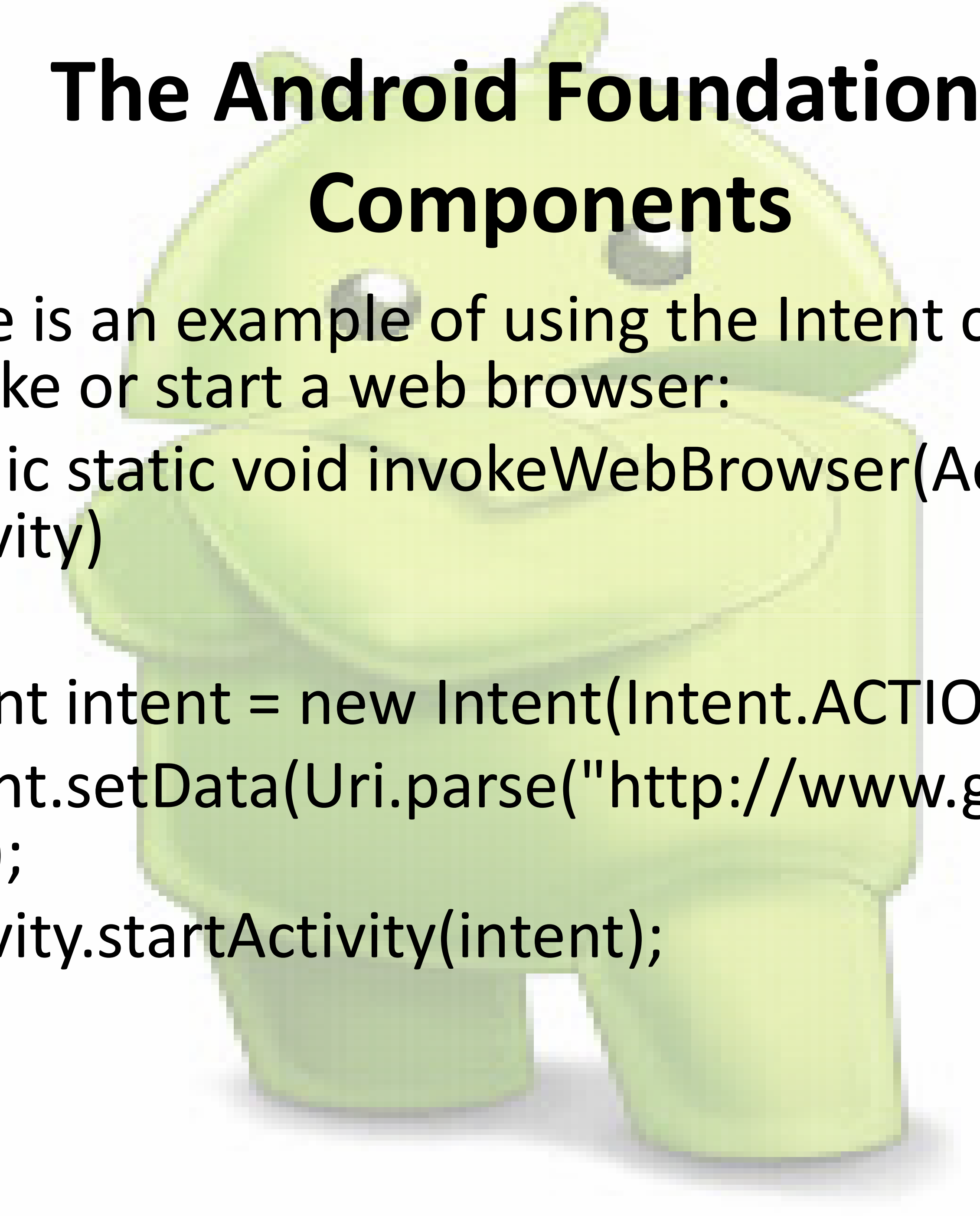
- Android 3.0 introduced a new UI concept called *fragments to allow developers to chunk* views and functionality for display on tablets
- One of the Android framework's key concepts is the life cycle management of activity windows.
- Protocols are put in place so that Android can manage state as users hide, restore, stop, and close activity windows

The Android Foundational Components

- An intent is an intra- and inter process mechanism to invoke components in Android.
- A component in Android is a piece of code that has a well defined life cycle.
- An activity representing a window in an Android application is a component



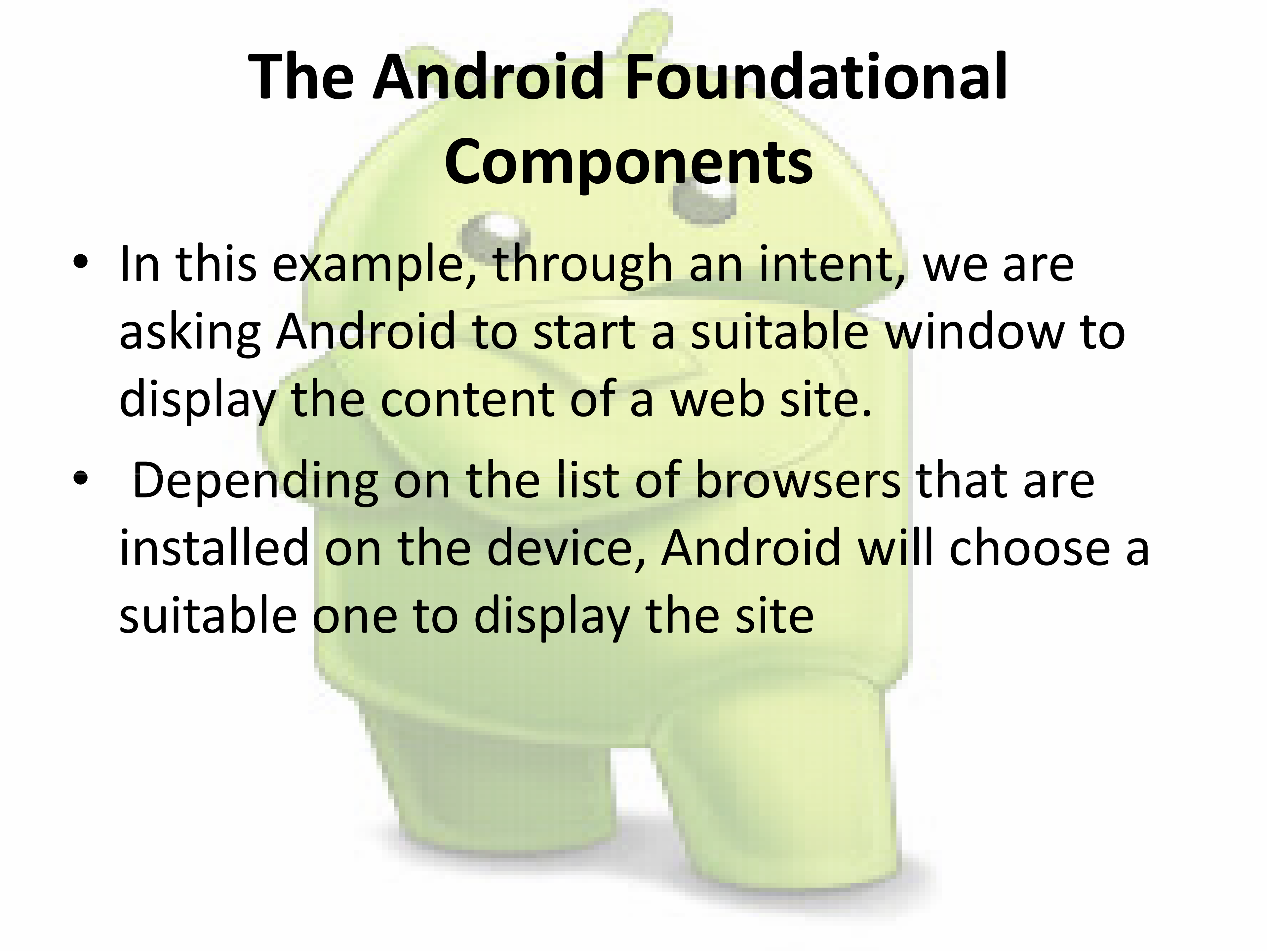
The Android Foundational Components



- Here is an example of using the Intent class to invoke or start a web browser:
- public static void invokeWebBrowser(Activity activity)

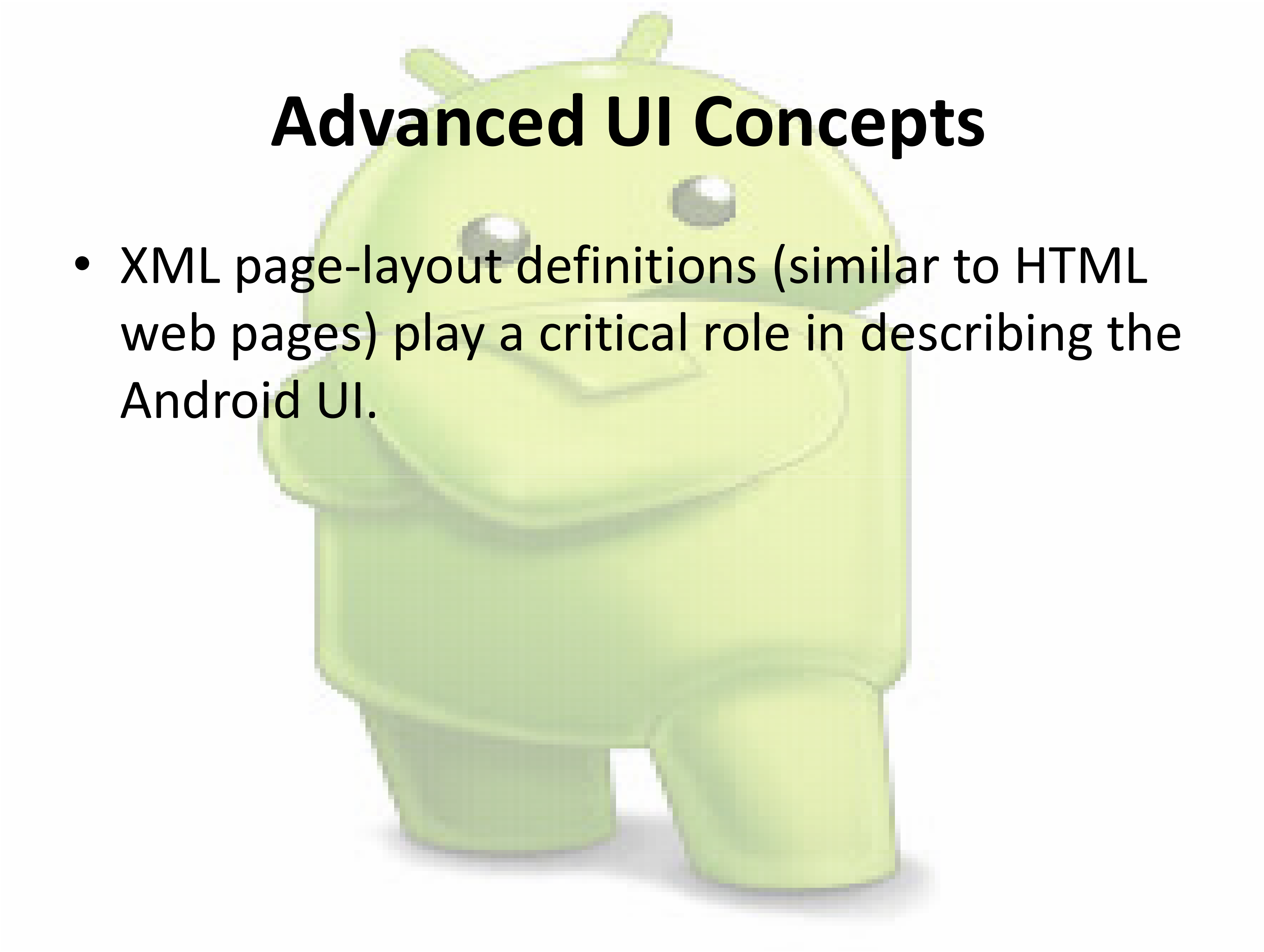
```
{
    Intent intent = new Intent(Intent.ACTION_VIEW);
    intent.setData(Uri.parse("http://www.google.com"));
    activity.startActivity(intent);
}
```

The Android Foundational Components



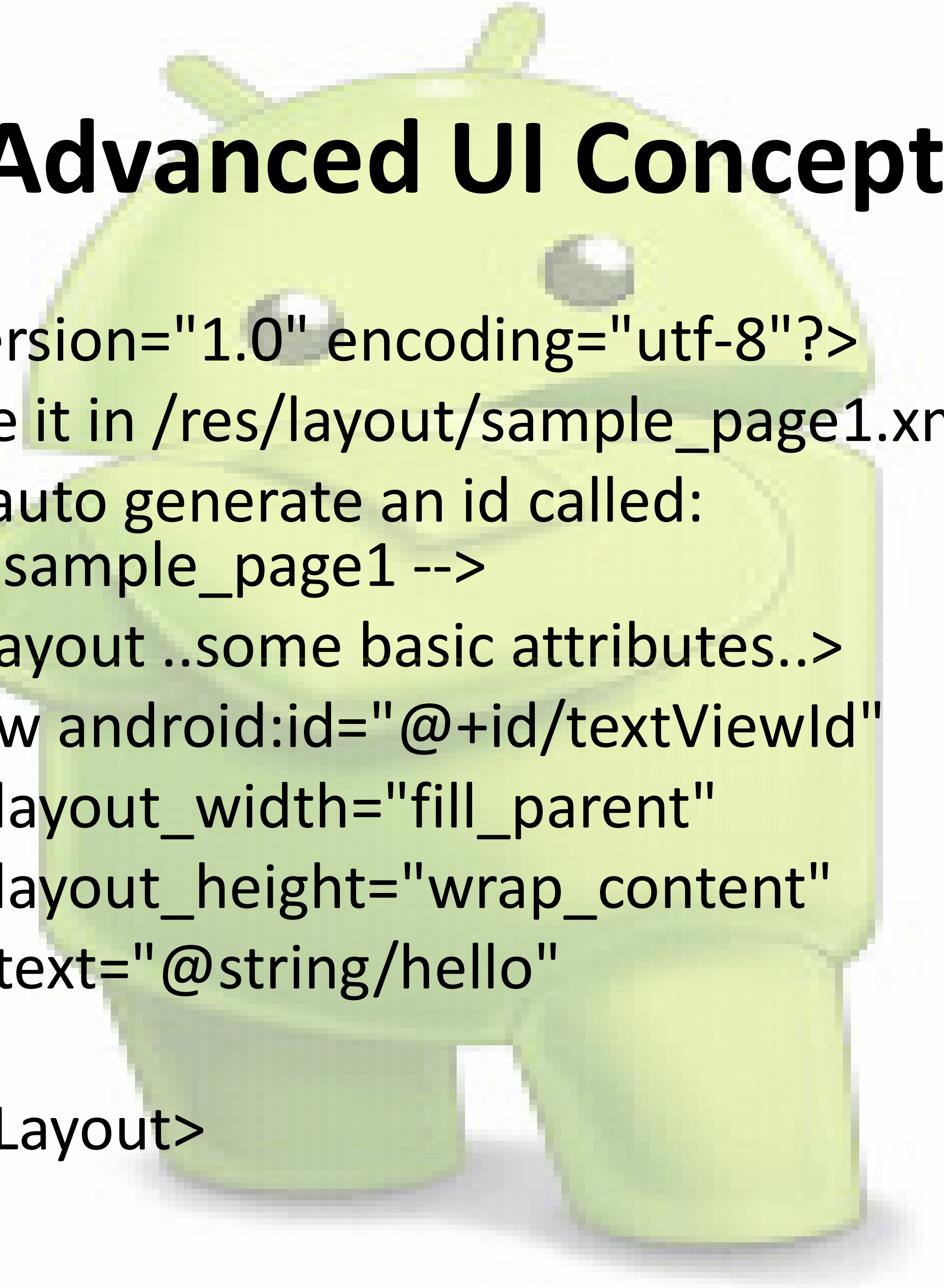
- In this example, through an intent, we are asking Android to start a suitable window to display the content of a web site.
- Depending on the list of browsers that are installed on the device, Android will choose a suitable one to display the site

Advanced UI Concepts



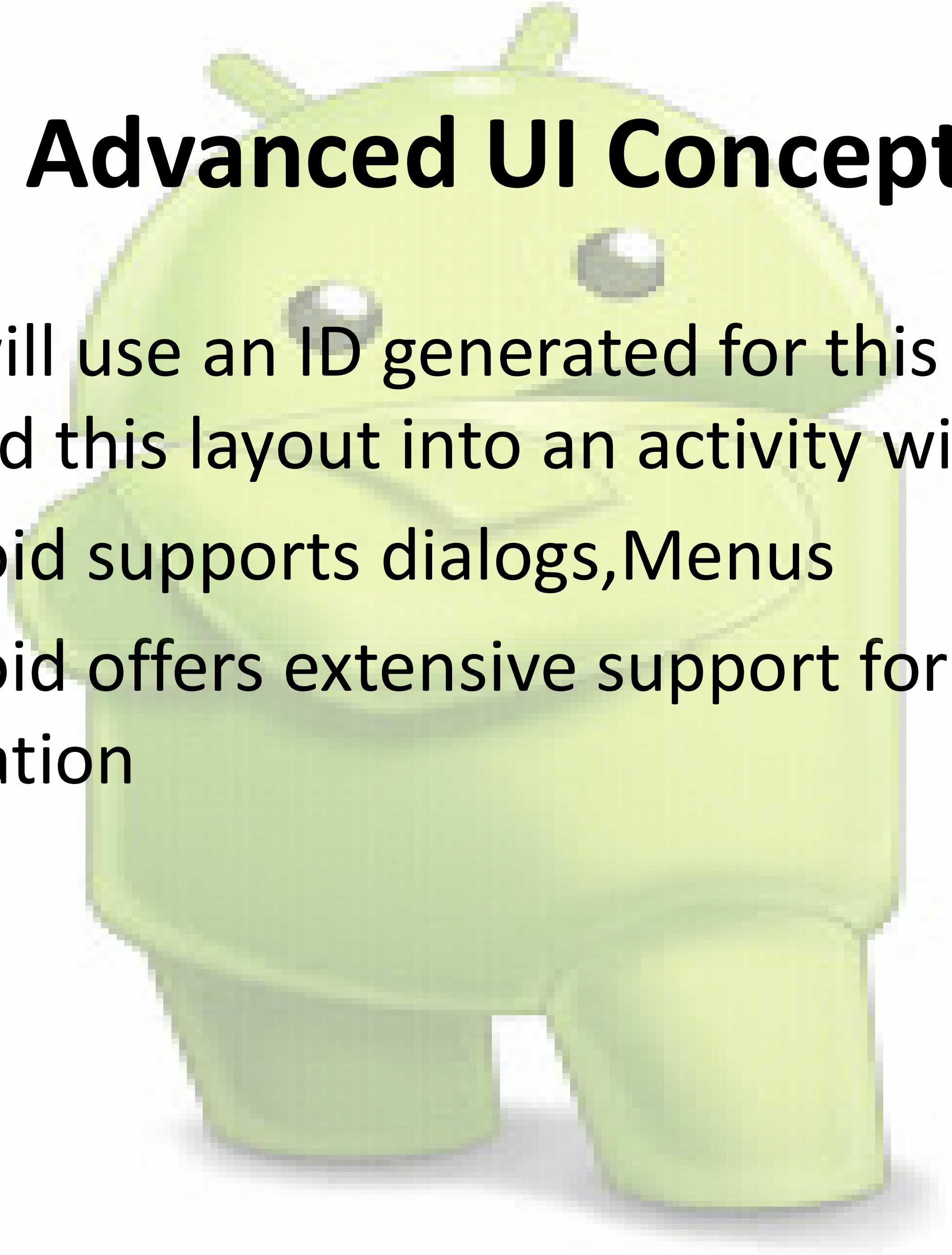
- XML page-layout definitions (similar to HTML web pages) play a critical role in describing the Android UI.

Advanced UI Concepts



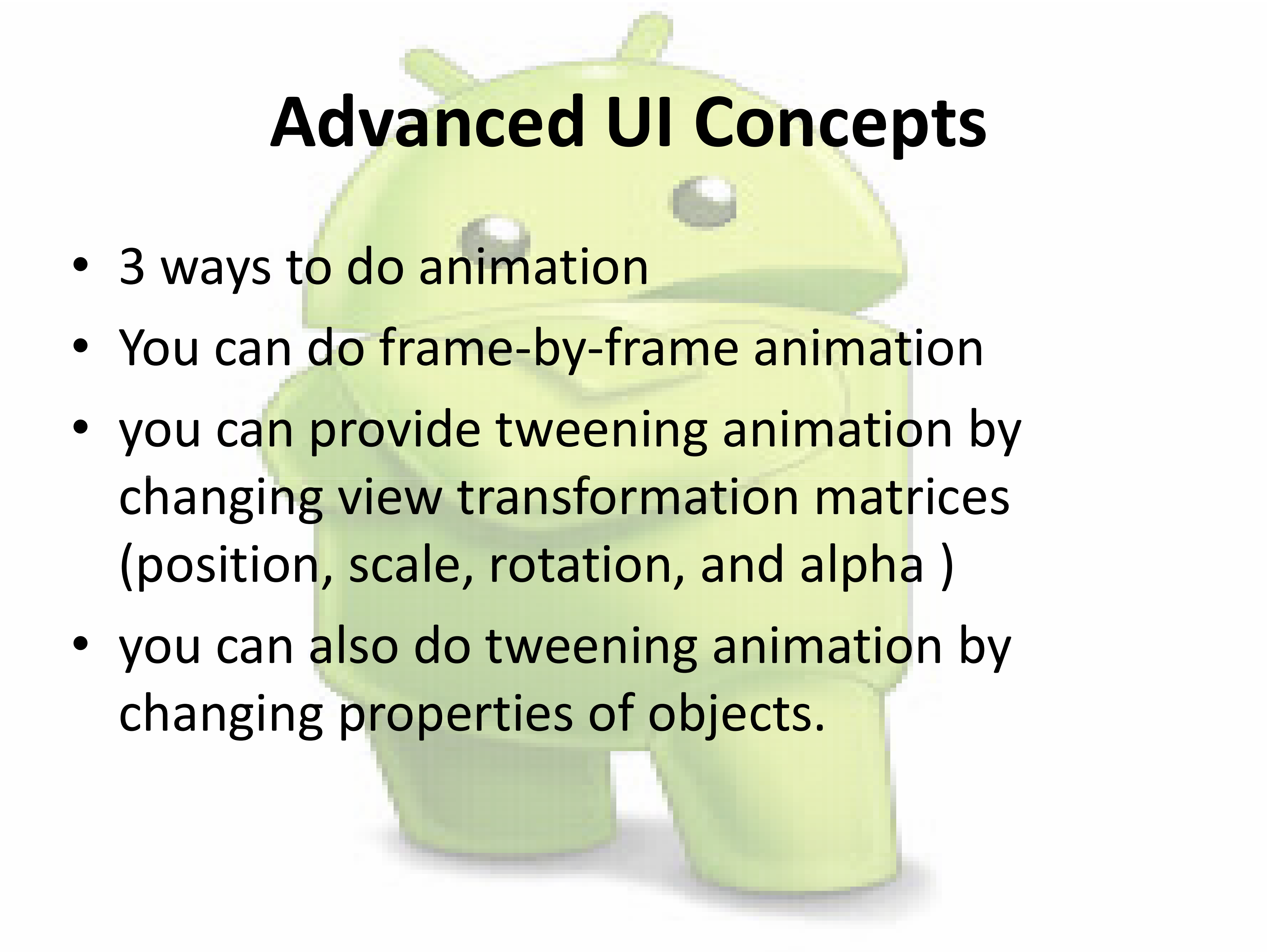
- <?xml version="1.0" encoding="utf-8"?>
- <!-- place it in /res/layout/sample_page1.xml -->
- <!-- will auto generate an id called: R.layout.sample_page1 -->
- <LinearLayout ..some basic attributes..>
- <TextView android:id="@+id/textViewId"
- android:layout_width="fill_parent"
- android:layout_height="wrap_content"
- android:text="@string/hello"
- />
- </LinearLayout>

Advanced UI Concepts



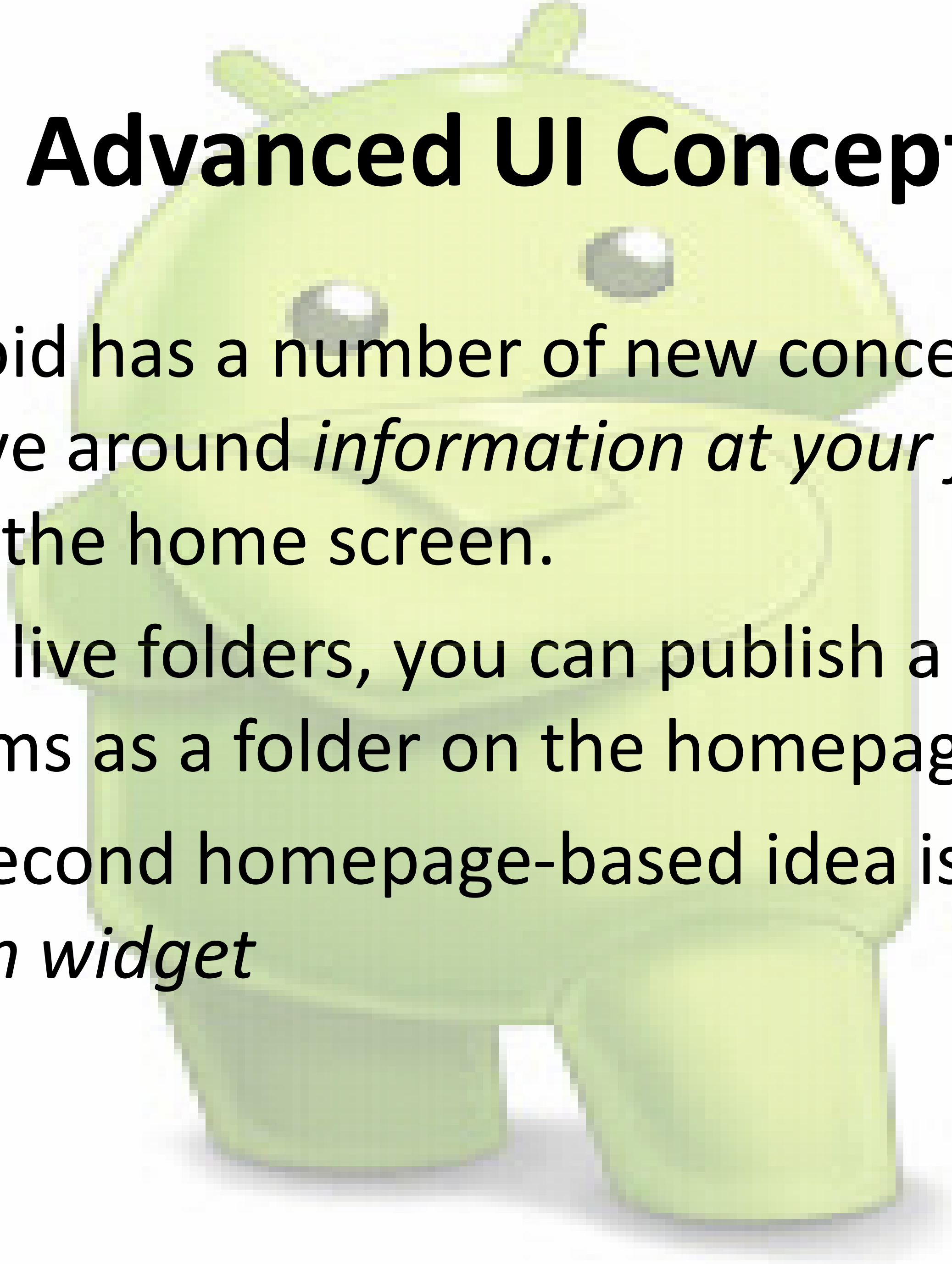
- You will use an ID generated for this XML file to load this layout into an activity window
- Android supports dialogs,Menus
- Android offers extensive support for animation

Advanced UI Concepts



- 3 ways to do animation
- You can do frame-by-frame animation
- you can provide tweening animation by changing view transformation matrices (position, scale, rotation, and alpha)
- you can also do tweening animation by changing properties of objects.

Advanced UI Concepts



- Android has a number of new concepts that revolve around *information at your fingertips* using the home screen.
- Using live folders, you can publish a collection of items as a folder on the homepage.
- The second homepage-based idea is the *home screen widget*