The Android Linux Kernel (Part 2): Core Kernel IPC & Processing Mechanisms

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

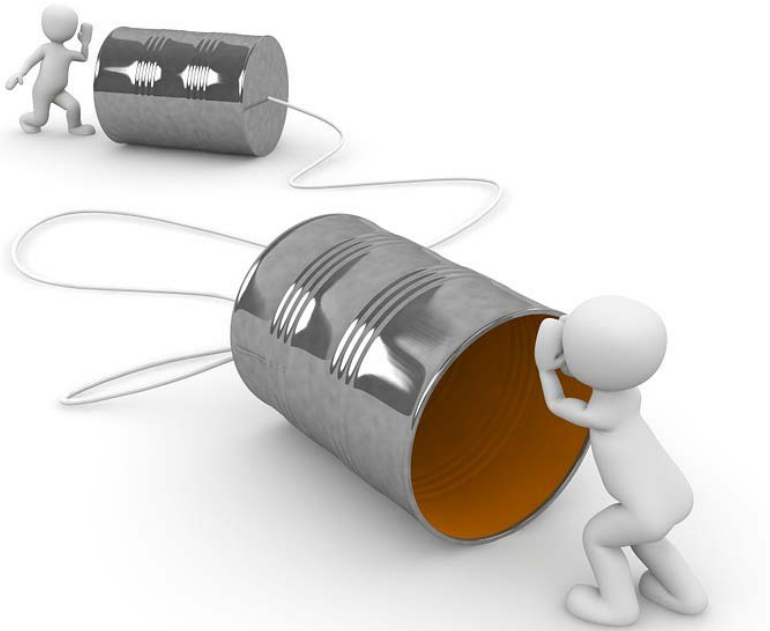
Professor of Computer Science

Institute for Software Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

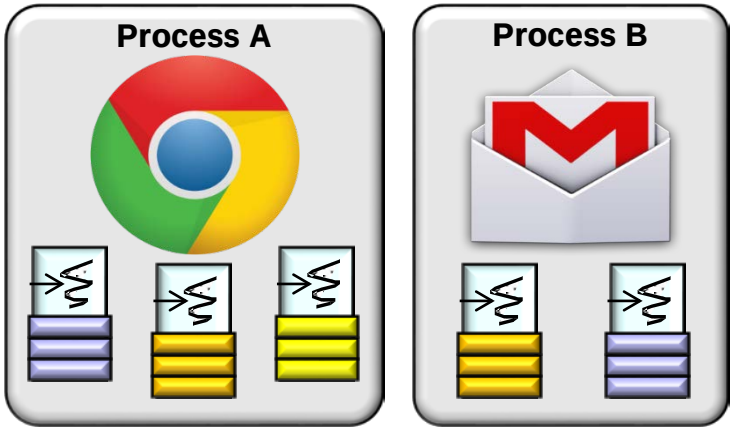
Learning Objectives in this Part of the Lesson

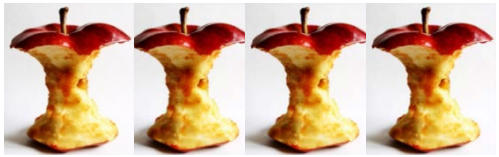
1. Recognize the two types of storage supported by Android Linux
2. Understand Android Linux's local & remote communication mechanisms



Learning Objectives in this Part of the Lesson

1. Recognize the two types of storage supported by Android Linux
2. Understand Android Linux's local & remote communication mechanisms
3. Know how Android Linux's processes & threads mediate access to one or more processor cores

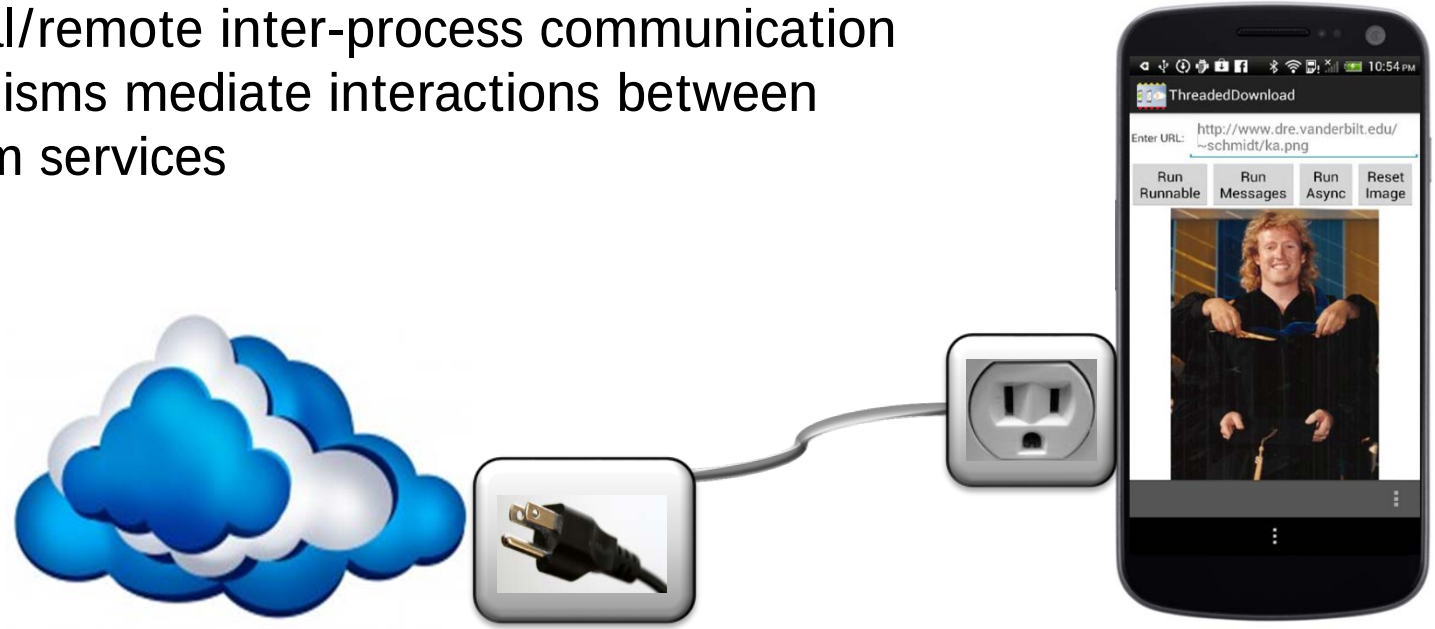




Android Linux Kernel: Local & Remote Inter-Process Communication (IPC)

Android Linux Kernel: Local & Remote IPC

- Android's local/remote inter-process communication (IPC) mechanisms mediate interactions between apps & system services

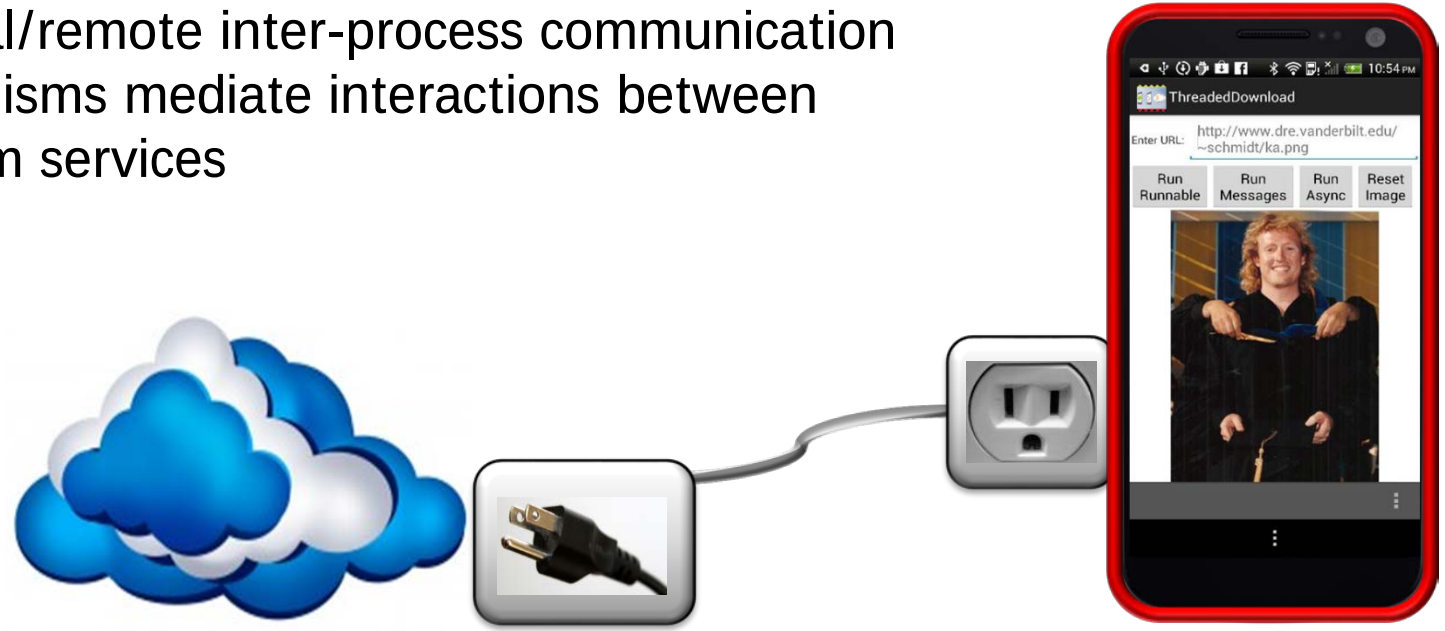


See en.wikipedia.org/wiki/Inter-process_communication

Android Linux Kernel: Local & Remote IPC

C

- Android's local/remote inter-process communication (IPC) mechanisms mediate interactions between apps & system services

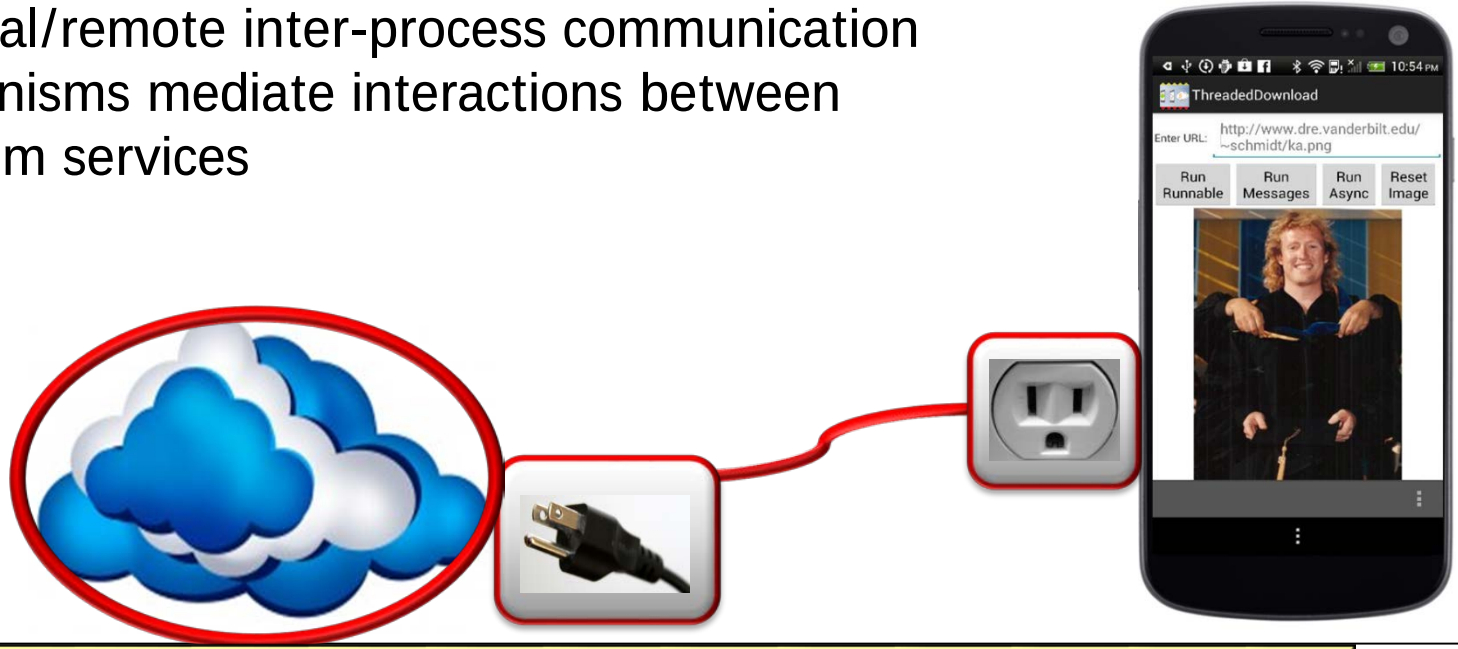


IPC is an essential part of mobile cloud computing (for clients)

Android Linux Kernel: Local & Remote IPC

C

- Android's local/remote inter-process communication (IPC) mechanisms mediate interactions between apps & system services

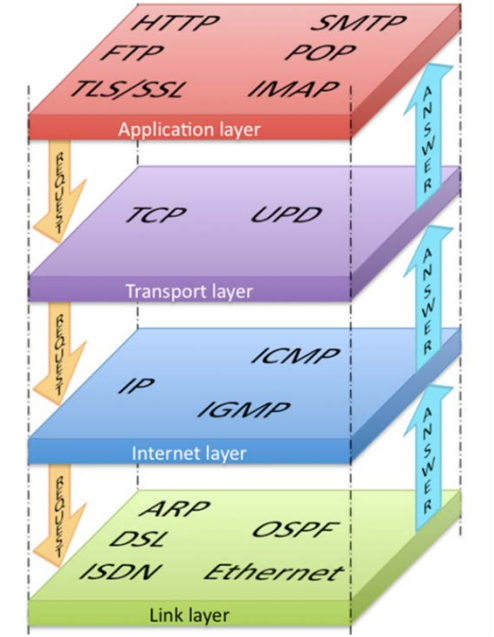


IPC is an essential part of mobile cloud computing (& servers)

Android Linux Kernel: Local & Remote IPC

C

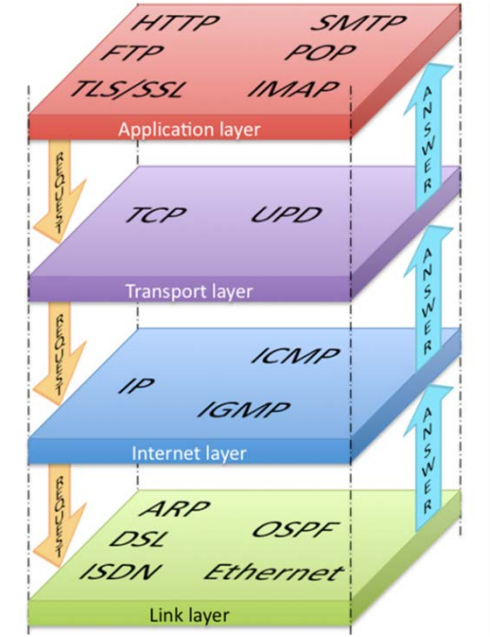
- Android's local/remote inter-process communication (IPC) mechanisms mediate interactions between apps & system services
  - It uses TCP/IP to access the Internet



See en.wikipedia.org/wiki/TCP/IP_model

Android Linux Kernel: Local & Remote IPC

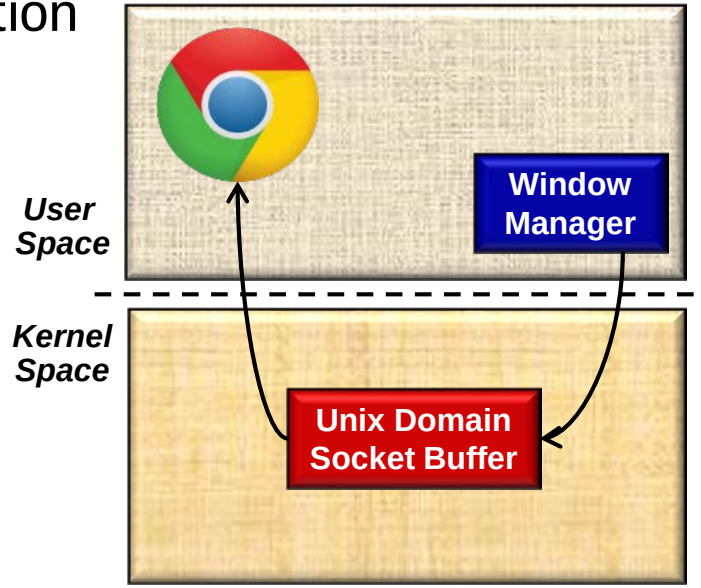
- Android's local/remote inter-process communication (IPC) mechanisms mediate interactions between apps & system services
  - It uses TCP/IP to access the Internet
    - Optimized for LANs & WANs



See en.wikipedia.org/wiki/TCP/IP_model

Android Linux Kernel: Local & Remote IPC

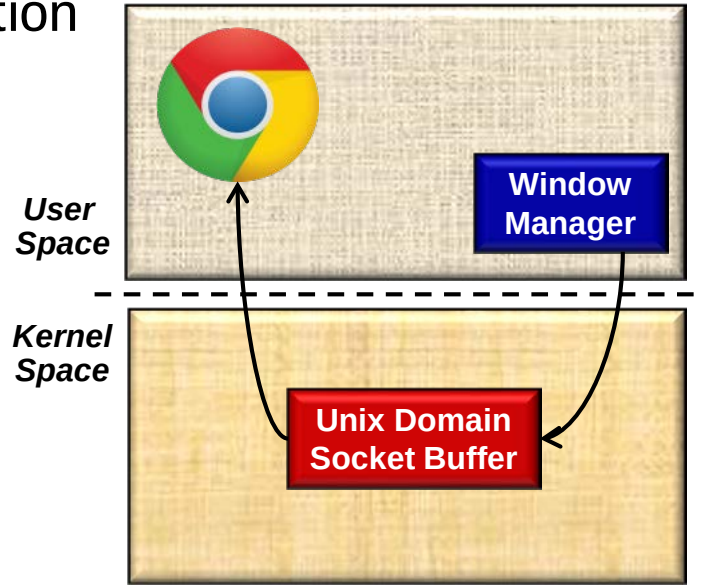
- Android's local/remote inter-process communication (IPC) mechanisms mediate interactions between apps & system services
  - It uses TCP/IP to access the Internet
  - It uses UNIX domain sockets for local communication on a device



See en.wikipedia.org/wiki/Unix_domain_socket

Android Linux Kernel: Local & Remote IPC

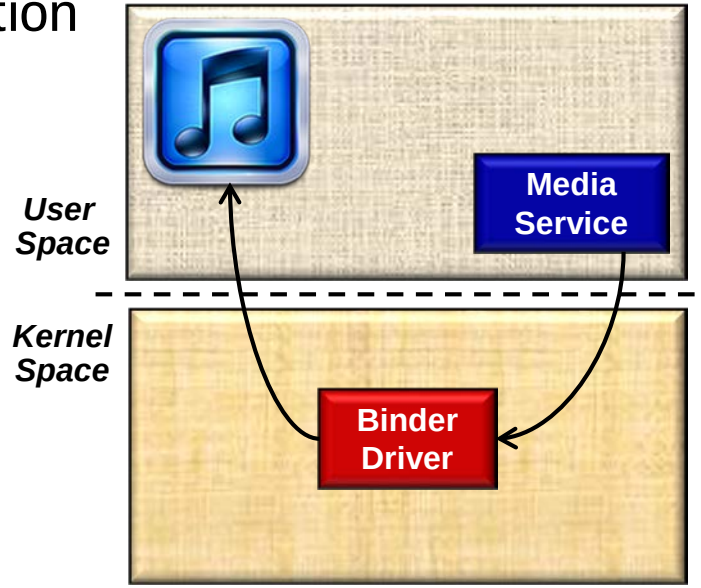
- Android's local/remote inter-process communication (IPC) mechanisms mediate interactions between apps & system services
  - It uses TCP/IP to access the Internet
  - It uses UNIX domain sockets for local communication on a device
    - Optimized for intra-host IPC



See en.wikipedia.org/wiki/Unix_domain_socket

Android Linux Kernel: Local & Remote IPC

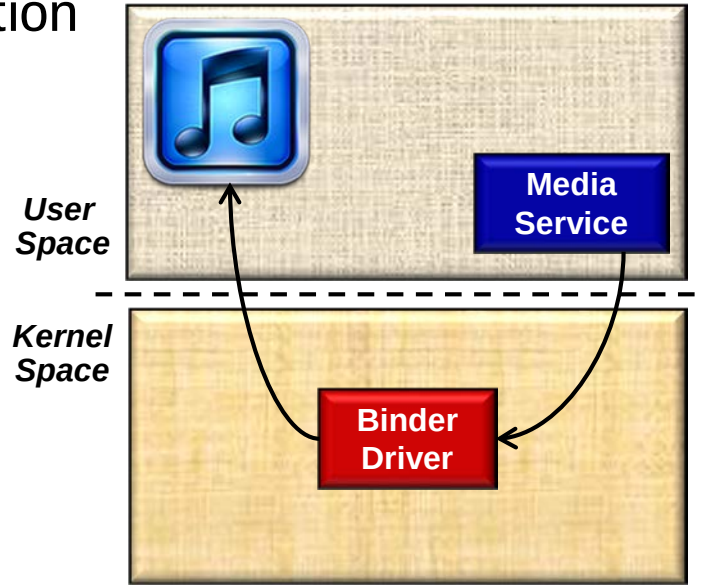
- Android's local/remote inter-process communication (IPC) mechanisms mediate interactions between apps & system services
  - It uses TCP/IP to access the Internet
  - It uses UNIX domain sockets for local communication on a device
  - Its Binder driver supports non-standard message-oriented IPC on a device



See elinux.org/Android_Binder

Android Linux Kernel: Local & Remote IPC

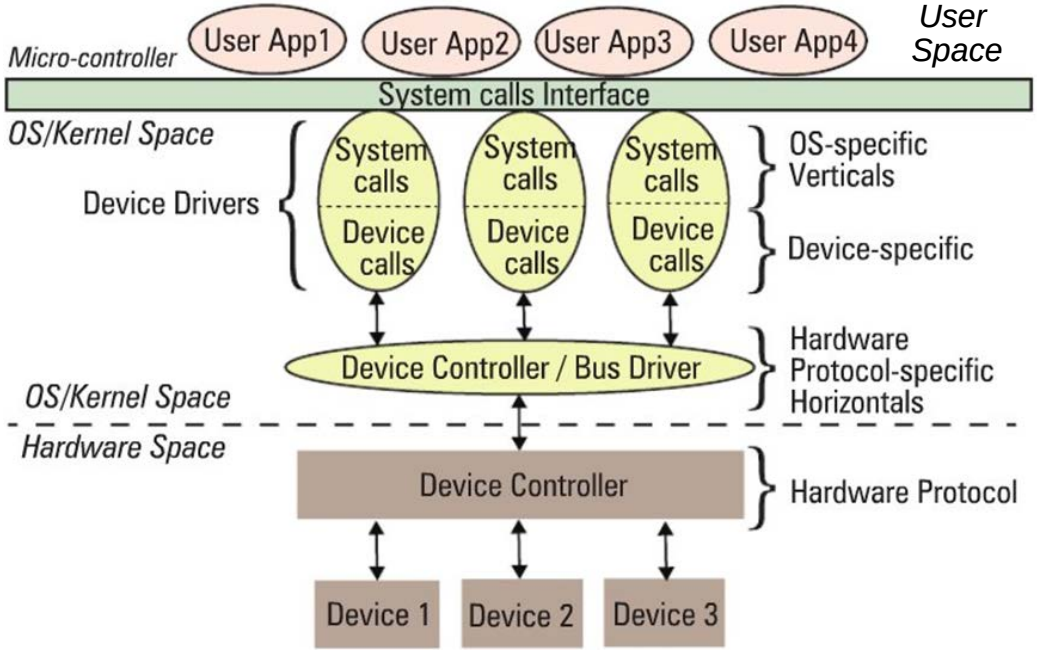
- Android's local/remote inter-process communication (IPC) mechanisms mediate interactions between apps & system services
  - It uses TCP/IP to access the Internet
  - It uses UNIX domain sockets for local communication on a device
  - Its Binder driver supports non-standard message-oriented IPC on a device
    - *Highly* optimized for intra-host IPC



See part 3 of this lesson on "Android Linux Kernel Extensions"

Android Linux Kernel: Local & Remote IPC

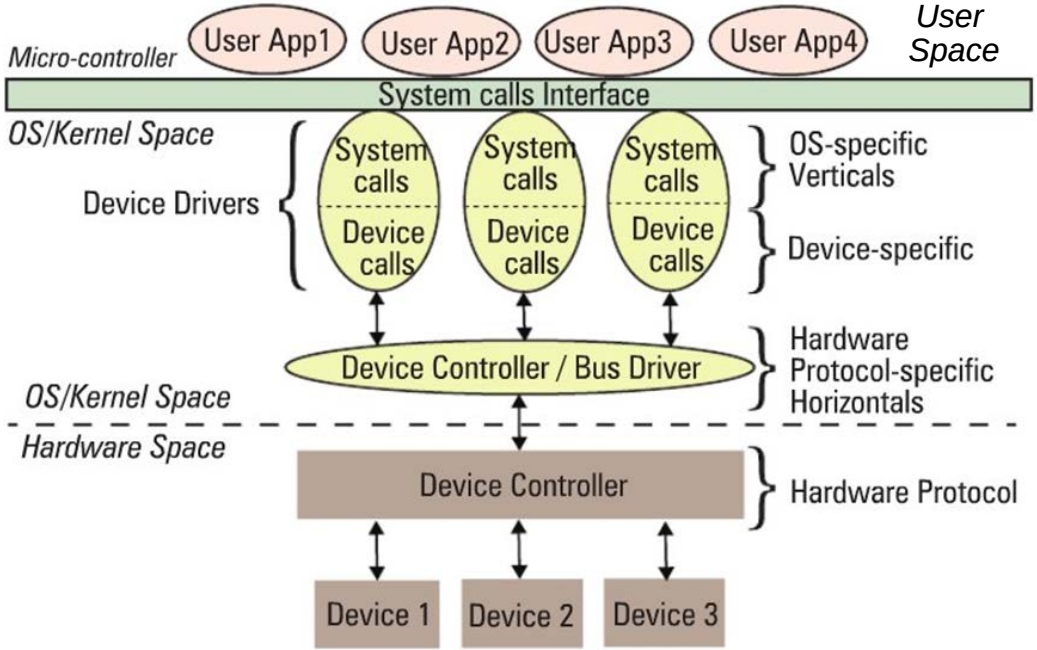
- The device driver framework runs in the kernel & coordinates access to hardware devices



See en.wikipedia.org/wiki/Device_driver

Android Linux Kernel: Local & Remote IPC

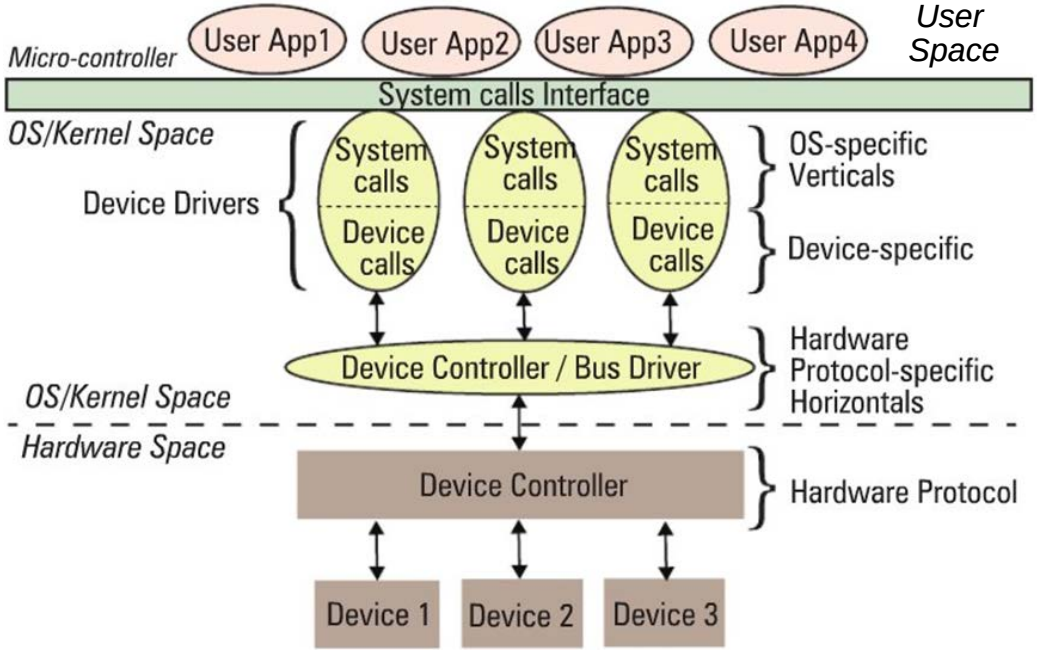
- The device driver framework runs in the kernel & coordinates access to hardware devices, e.g.,
  - Block-oriented devices
    - i.e., transfer data in “chunks”



See en.wikipedia.org/wiki/USB_mass_storage_device_class#Linux

Android Linux Kernel: Local & Remote IPC

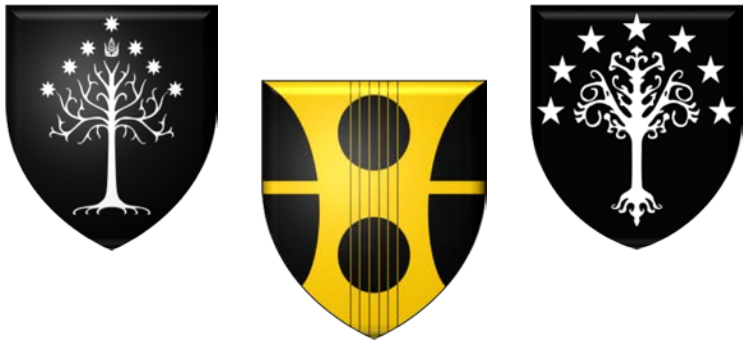
- The device driver framework runs in the kernel & coordinates access to hardware devices, e.g.
  - Block-oriented devices
  - Character-oriented devices
    - i.e., transfer data “byte-by-byte”

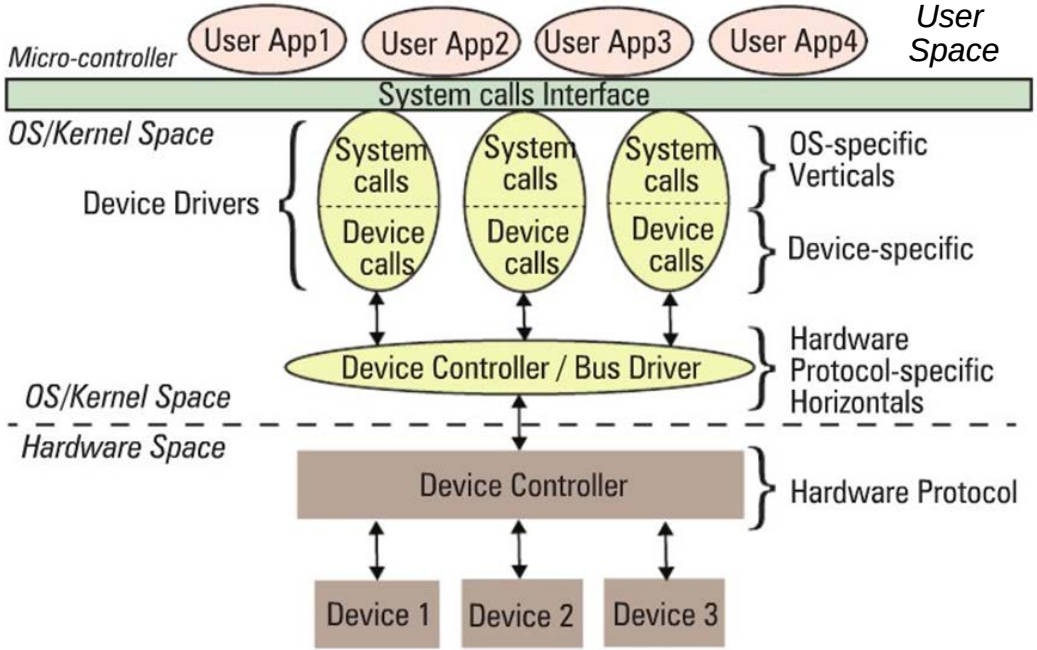


See source.android.com/devices/input/touch-devices.html

Android Linux Kernel: Local & Remote IPC

- The device driver framework shields other parts of the kernel & higher layers of Android from low-level hardware details





Hardware can thus be accessed portably, robustly, & securely

Android Linux Kernel: Local & Remote IPC

- Programming & debugging device drivers is challenging!

Android Linux Kernel: Local & Remote IPC

C

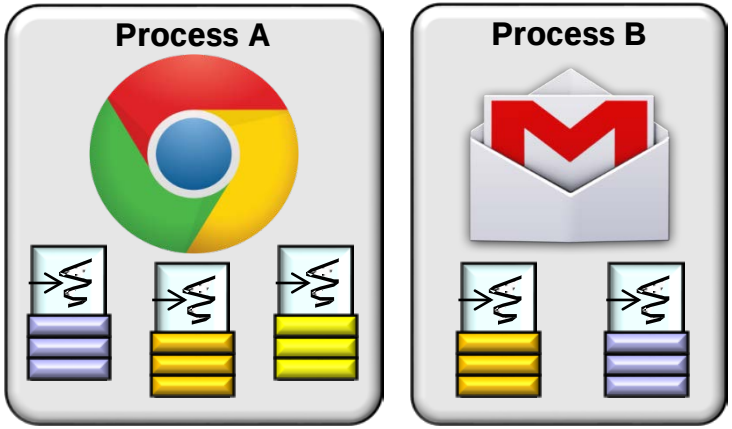
- Programming & debugging device drivers is challenging!
  - Requires low-level system architecture knowledge

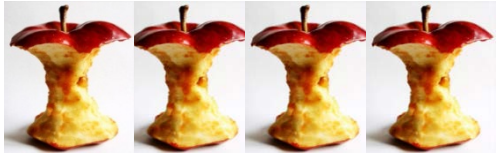
See source.android.com/devices

Android Linux Kernel: Processes & Threads

Android Linux Kernel: Processes & Threads

- The Android Linux kernel supports processes & threads

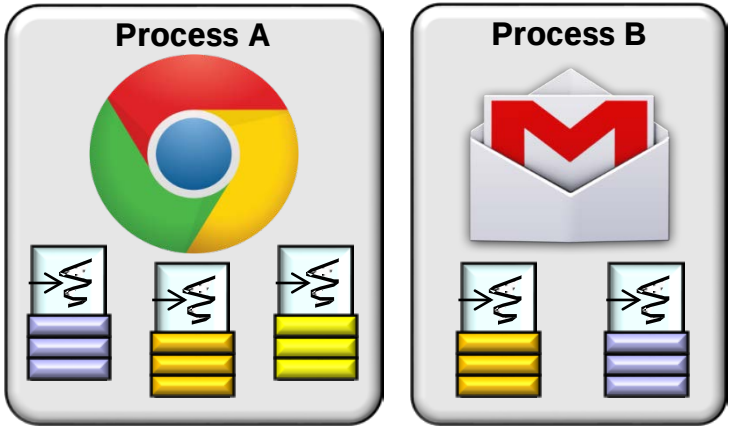


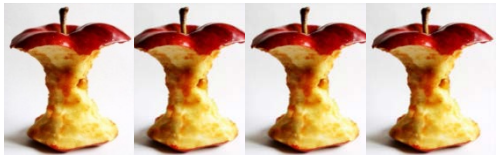


See coltf.blogspot.com/p/android-os-processes-and-zygote.html

Android Linux Kernel: Processes & Threads

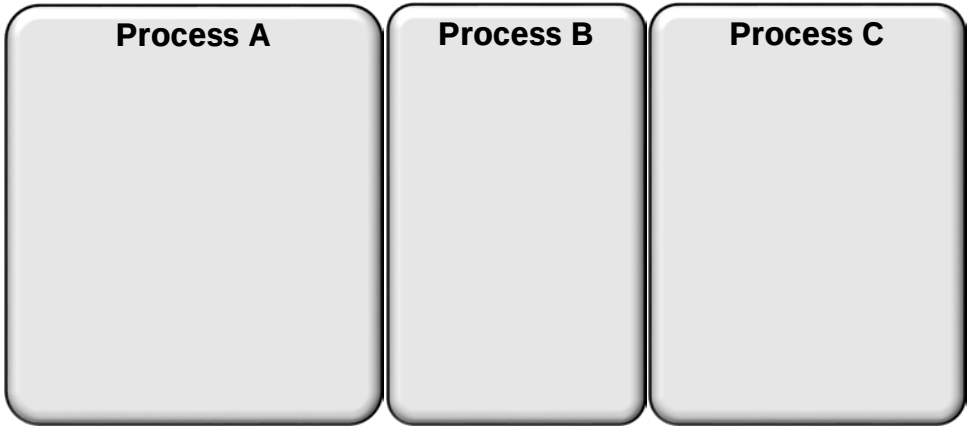
- The Android Linux kernel supports processes & threads
  - Used to encapsulate app instructions & data efficiently, reliably, & securely on one or more processor cores





Android Linux Kernel: Processes & Threads

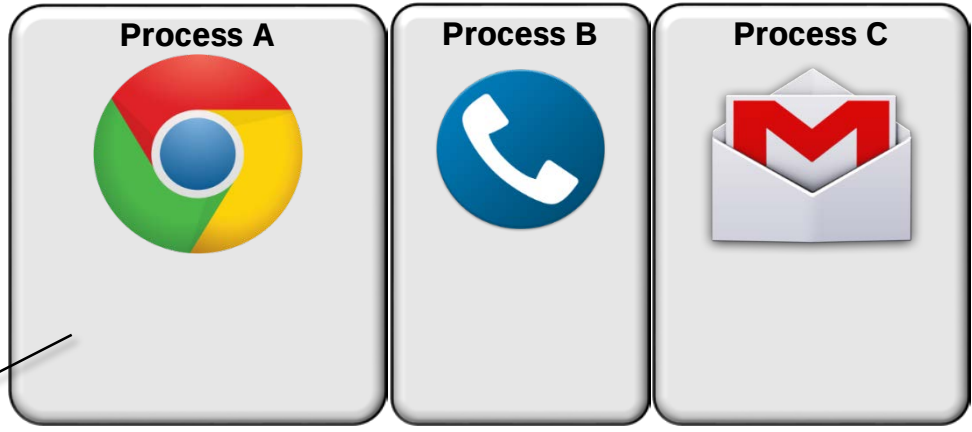
- A process provides a unit of resource allocation & protection



See en.wikipedia.org/wiki/Process_(computing)

Android Linux Kernel: Processes & Threads

- A process provides a unit of resource allocation & protection



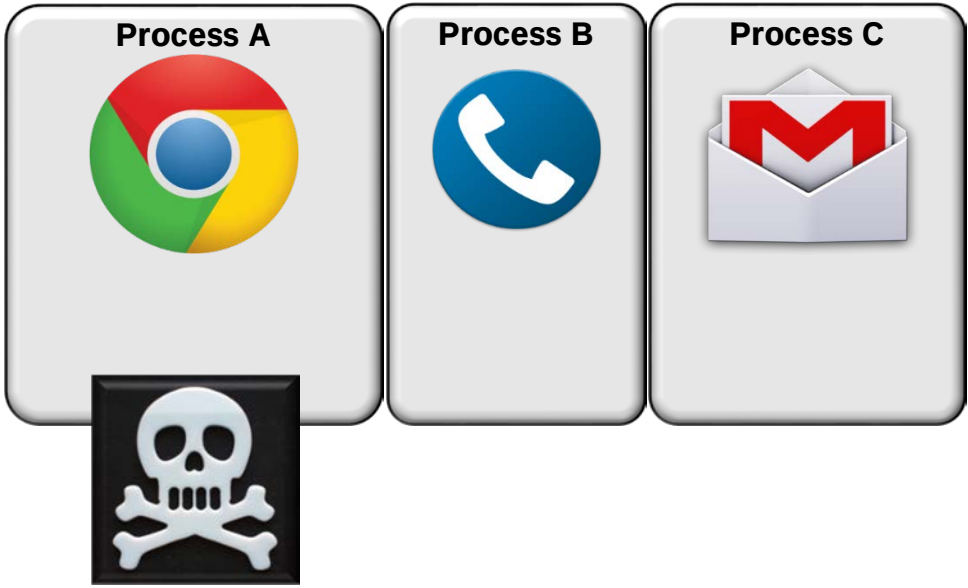
Each Android app typically runs in its own Linux process

See developer.android.com/guide/components/processes-and-threads.html#Processes

Android Linux Kernel: Processes & Threads

C

- A process provides a unit of resource allocation & protection
  - Minimize impact of app failures



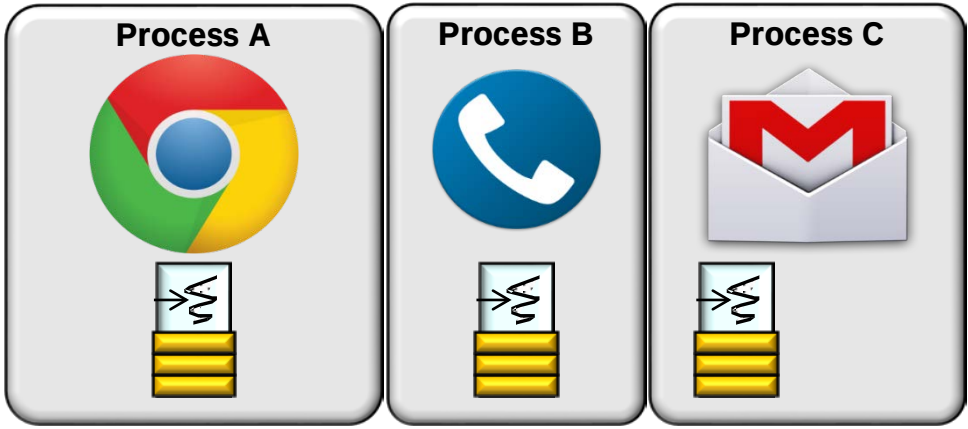
Android Linux Kernel: Processes & Threads

C

- A process provides a unit of resource allocation & protection
  - Minimize impact of app failures
  - Ensure app data is private

Android Linux Kernel: Processes & Threads

- A process provides a unit of resource allocation & protection
  - Minimize impact of app failures
  - Ensure app data is private
  - Contain one thread by default

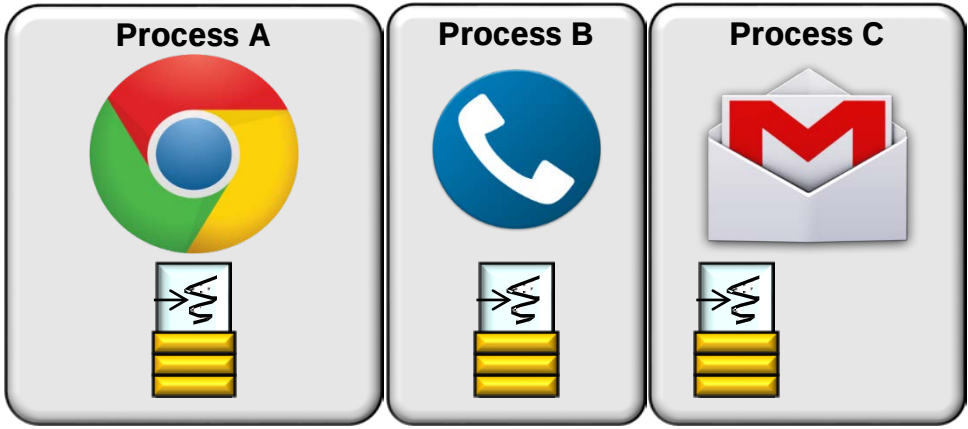


See developer.android.com/guide/components/processes-and-threads.html#Threads

Android Linux Kernel: Processes & Threads

C

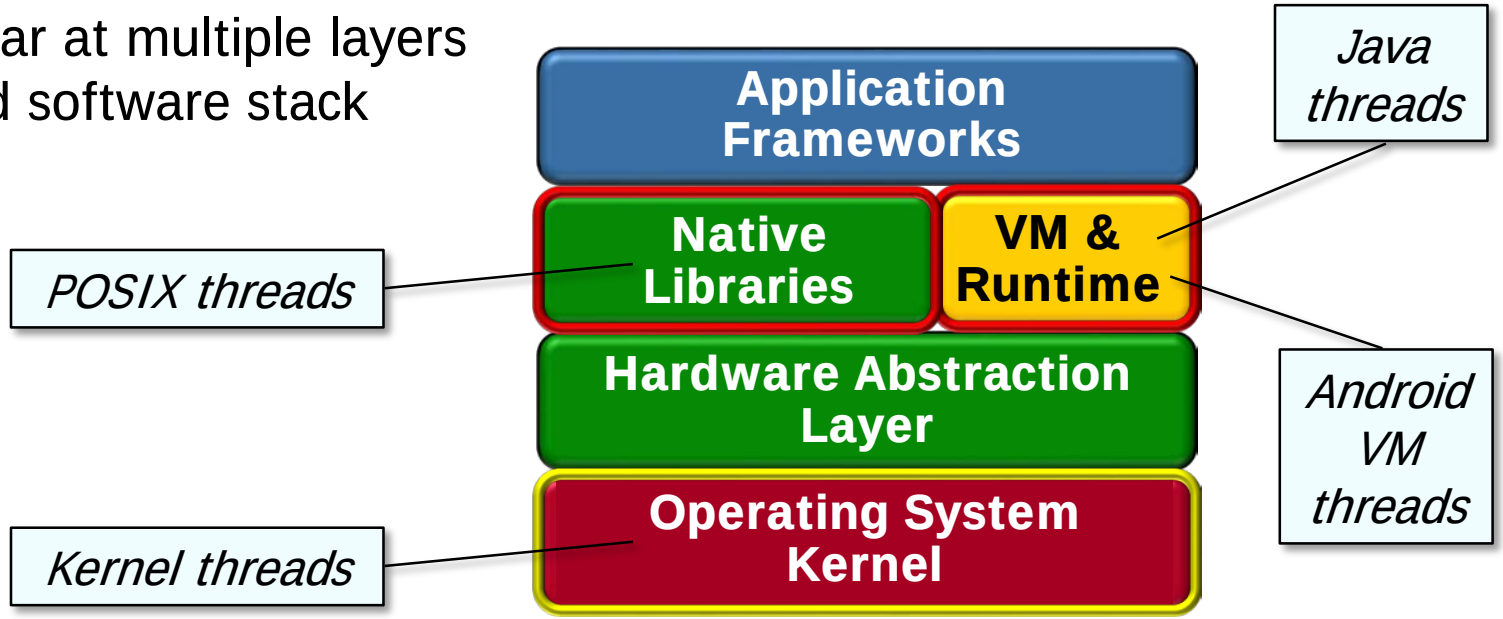
- A process provides a unit of resource allocation & protection
  - Minimize impact of app failures
  - Ensure app data is private
  - Contain one thread by default
    - Dispatches events to widgets & components in Android UI toolkit



See developer.android.com/guide/components/processes-and-threads.html#Threads

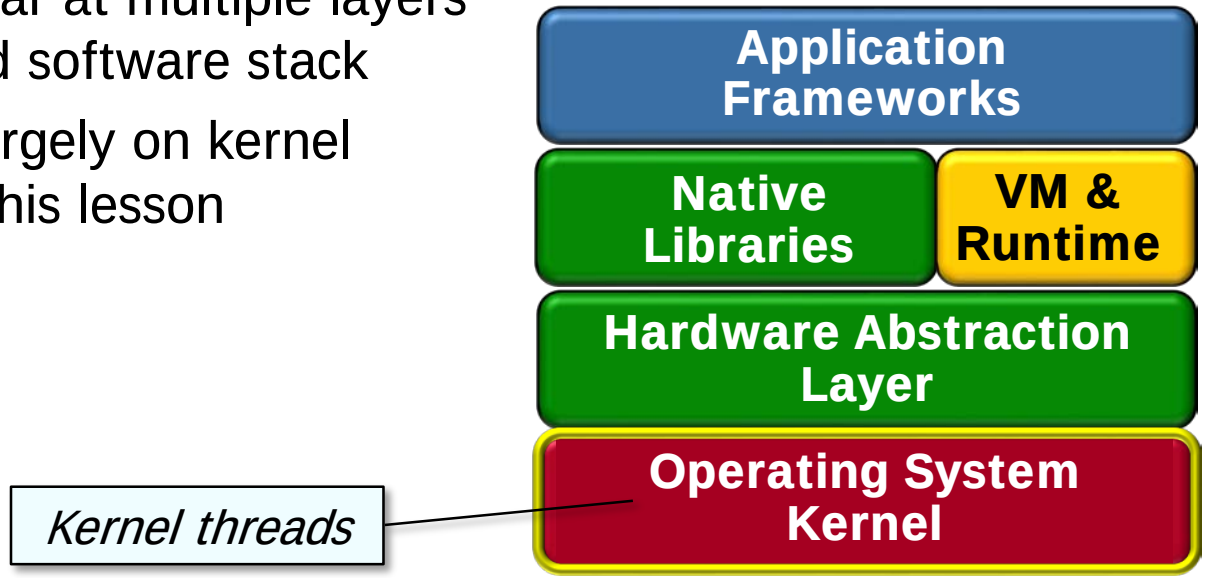
Android Linux Kernel: Processes & Threads

- Threads appear at multiple layers in the Android software stack



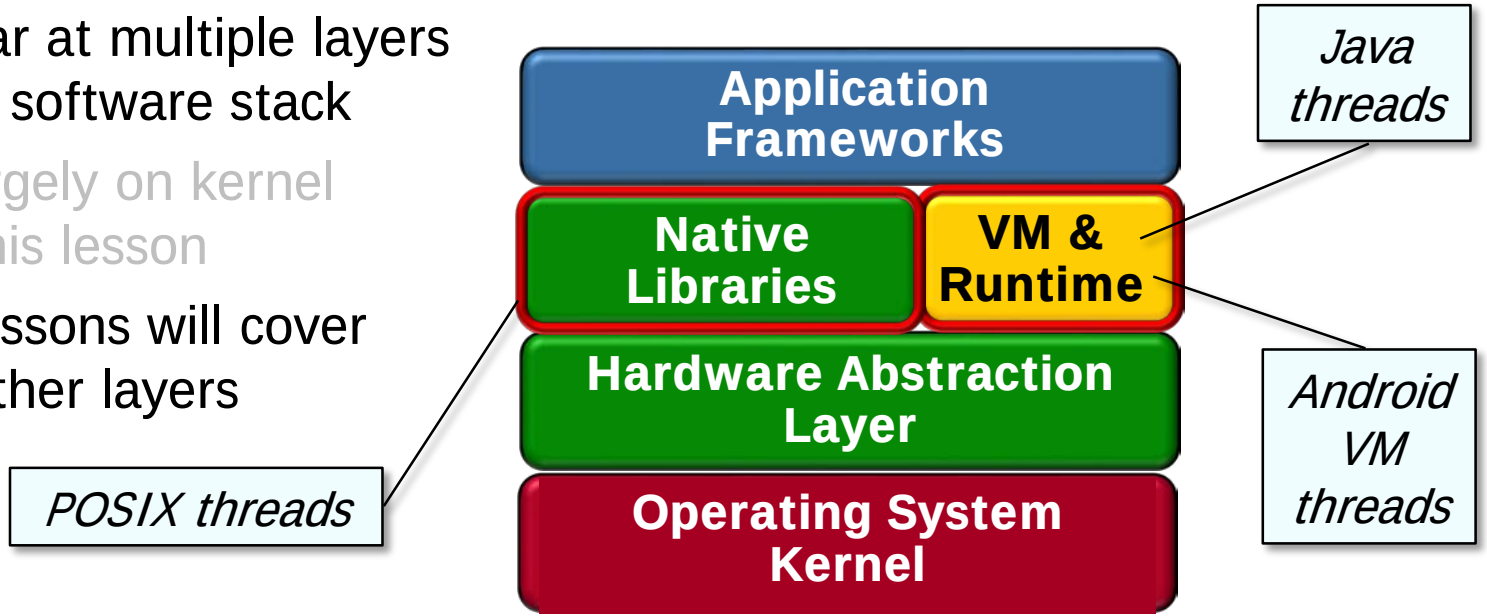
Android Linux Kernel: Processes & Threads

- Threads appear at multiple layers in the Android software stack
  - We focus largely on kernel threads in this lesson



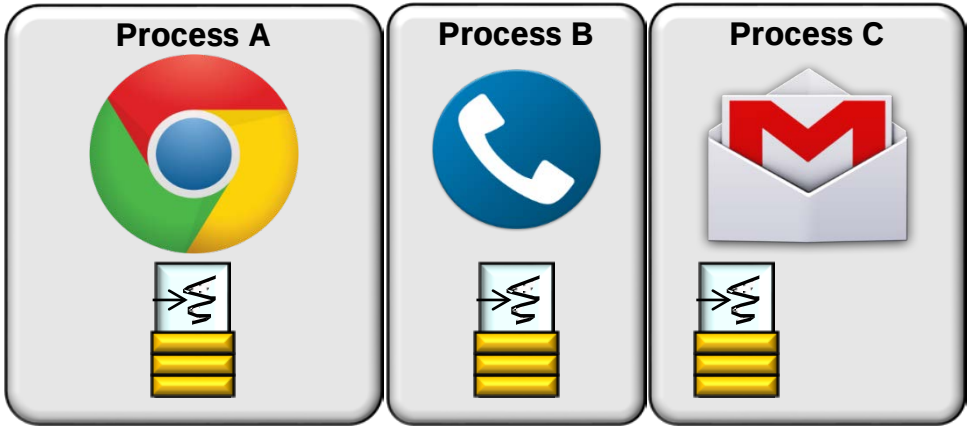
Android Linux Kernel: Processes & Threads

- Threads appear at multiple layers in the Android software stack
  - We focus largely on kernel threads in this lesson
  - Upcoming lessons will cover threads in other layers



Android Linux Kernel: Processes & Threads

- Threads provide units of execution for instruction streams that run on processor cores

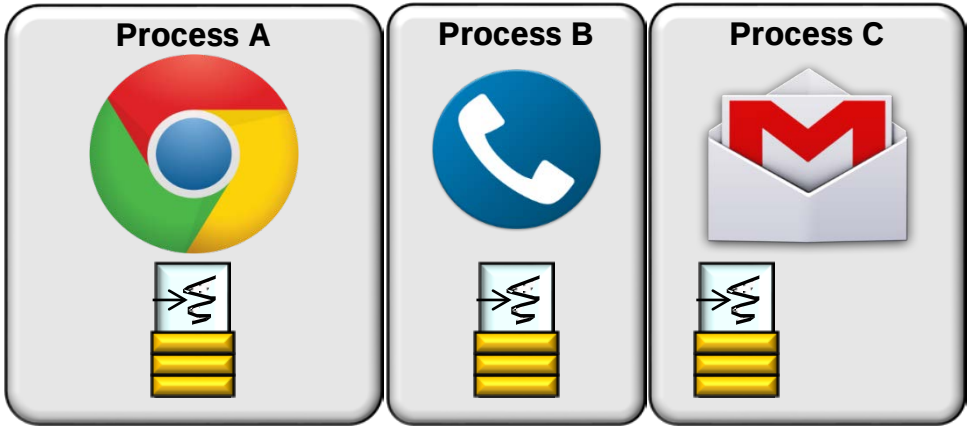


See en.wikipedia.org/wiki/Thread_(computing)

Android Linux Kernel: Processes & Threads

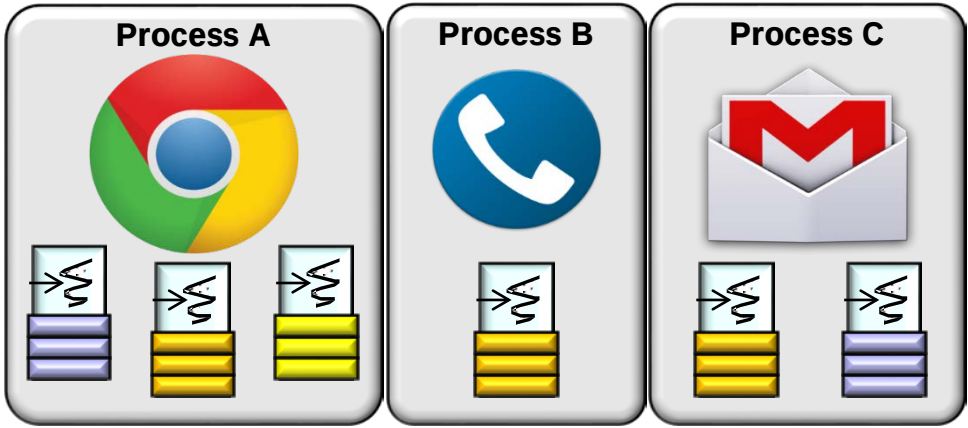
C

- Threads provide units of execution for instruction streams that run on processor cores
  - A Linux process has one thread by default



Android Linux Kernel: Processes & Threads

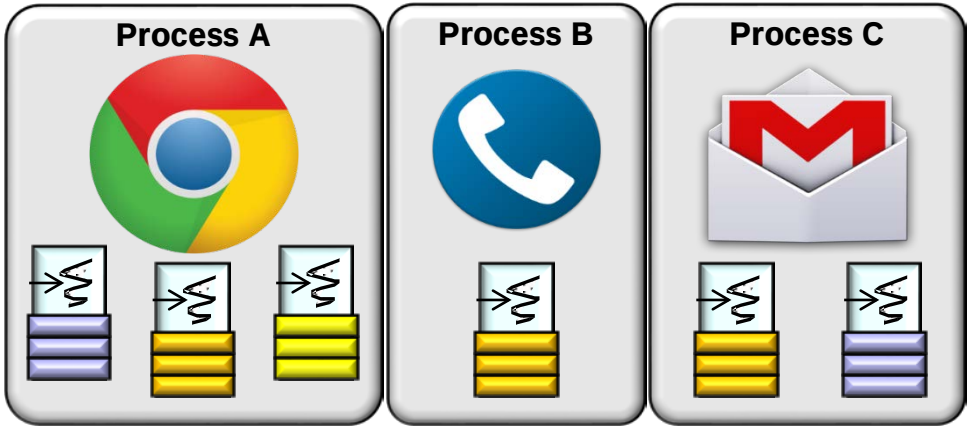
- Threads provide units of execution for instruction streams that run on processor cores
  - A Linux process has one thread by default
  - Apps can create more threads via various API calls



Android Linux Kernel: Processes & Threads

C

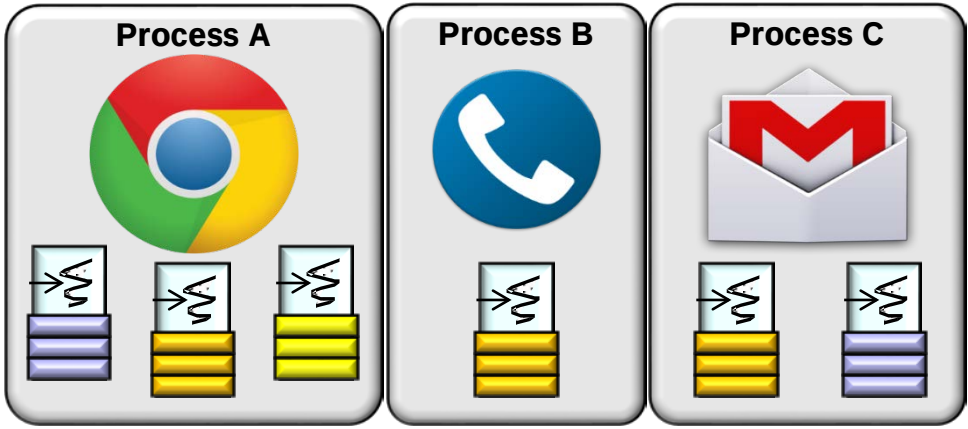
- Threads provide units of execution for instruction streams that run on processor cores
  - A Linux process has one thread by default
  - Apps can create more threads via various API calls
- Threads can run concurrently on one core



Android Linux Kernel: Processes & Threads

C

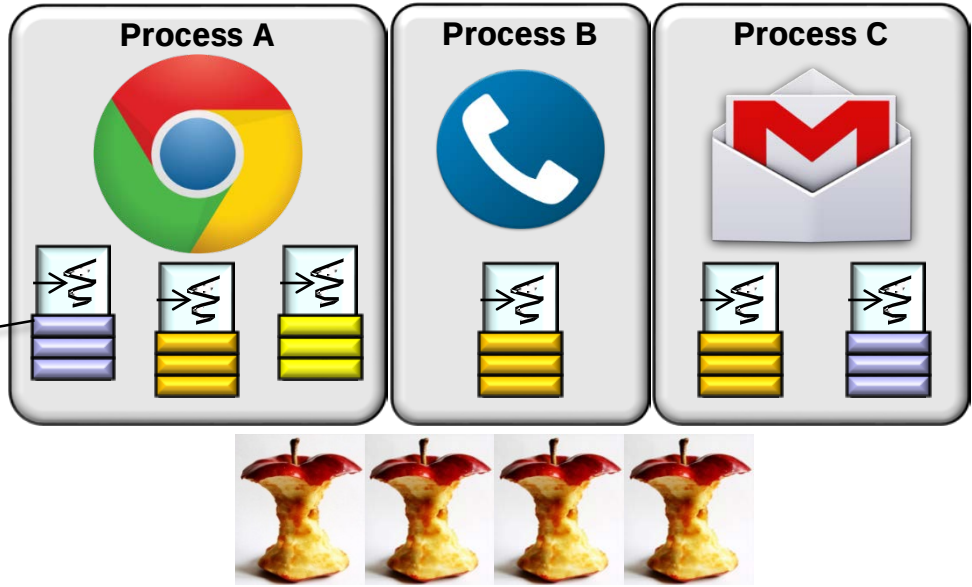
- Threads provide units of execution for instruction streams that run on processor cores
  - A Linux process has one thread by default
  - Apps can create more threads via various API calls
  - Threads can run concurrently on one core
  - They can run in parallel on multiple cores



Android Linux Kernel: Processes & Threads

- Threads provide units of execution for instruction streams that run on processor cores

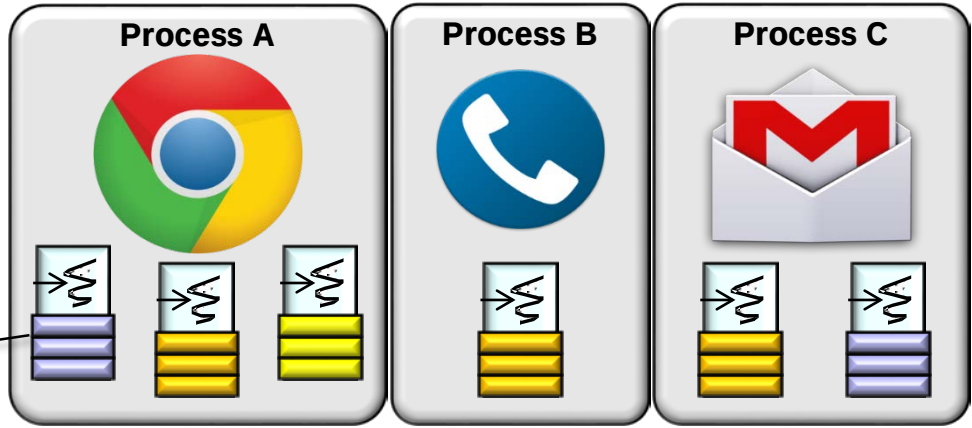
Each thread has a stack, a program counter, & other registers (unique resources)



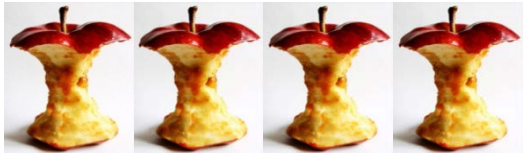
See en.wikipedia.org/wiki/Thread_(computing)#Processes.2C_kernel_threads.2C_user_threads.2C_and_fibers

Android Linux Kernel: Processes & Threads

- Threads provide units of execution for instruction streams that run on processor cores



Open files & memory are shared across threads (shared resources)



See en.wikipedia.org/wiki/Thread_(computing)#Processes.2C_kernel_threads.2C_user_threads.2C_and_fibers

Android Linux Kernel: Processes & Threads

C

- Android Linux kernel threads form the basis for the Java Threads in Android's middleware infrastructure

See developer.android.com/guide/components/processes-and-threads.html#Threads

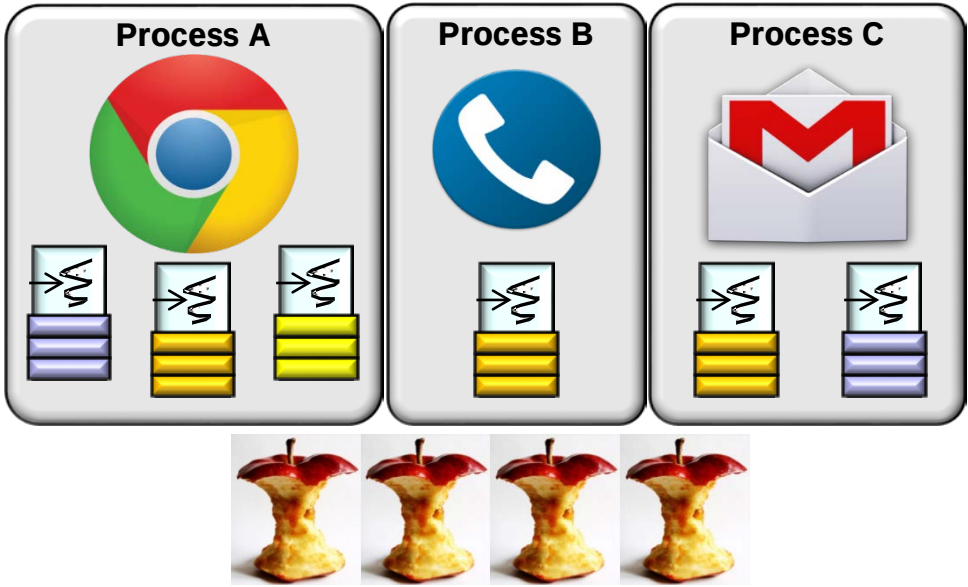
Android Linux Kernel: Processes & Threads

- Processes & threads consume non-trivial amount of system resources

Android Linux Kernel: Processes & Threads

C

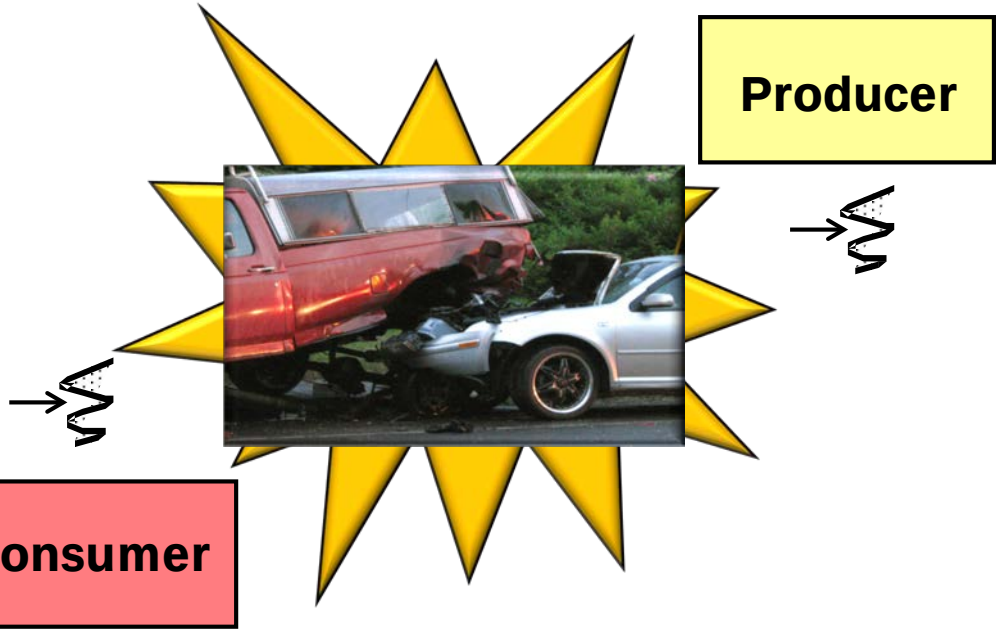
- Programming multi-threaded apps is hard



Android Linux Kernel: Processes & Threads

C

- Programming multi-threaded apps is hard

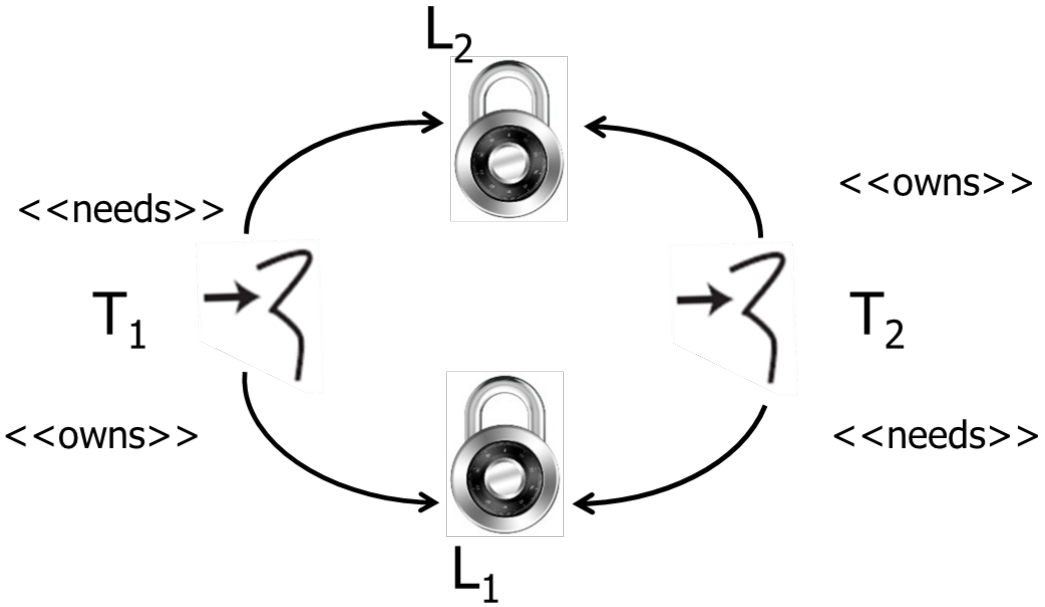


See en.wikipedia.org/wiki/Race_condition

Android Linux Kernel: Processes & Threads

C

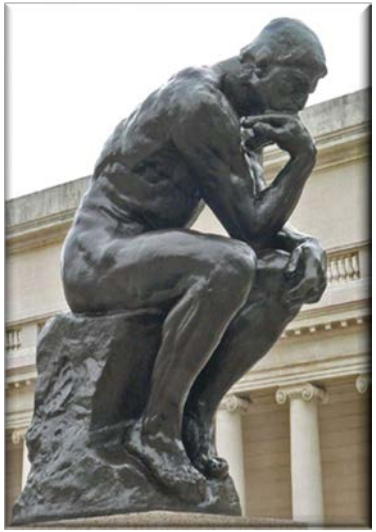
- Programming multi-threaded apps is hard

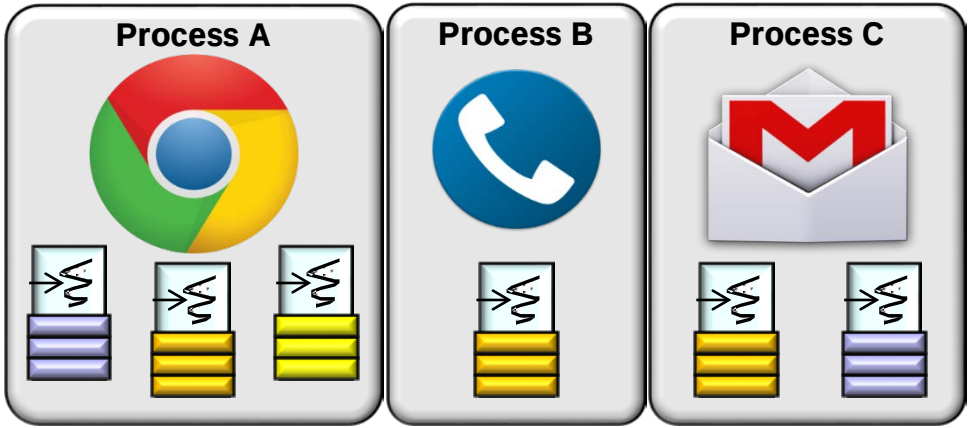


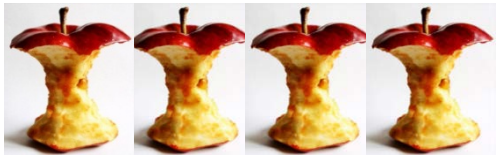
See en.wikipedia.org/wiki/Deadlock

Android Linux Kernel: Processes & Threads

- Programming multi-threaded apps is hard
  - Think deeply about how to program threads/processes







Android Linux Kernel: Processes & Threads

C

- Programming multi-threaded apps is hard
  - Think deeply about how to program threads/processes
  - Consult the Android documentation

Processes and Threads

When an application component starts and the application does not have any other components running, the Android system starts a new Linux process for the application with a single thread of execution. By default, all components of the same application run in the same process and thread (called the "main" thread). If an application component starts and there already exists a process for that application (because another component from the application exists), then the component is started within that process and uses the same thread of execution. However, you can arrange for different components in your application to run in separate processes, and you can create additional threads for any process.

This document discusses how processes and threads work in an Android application.

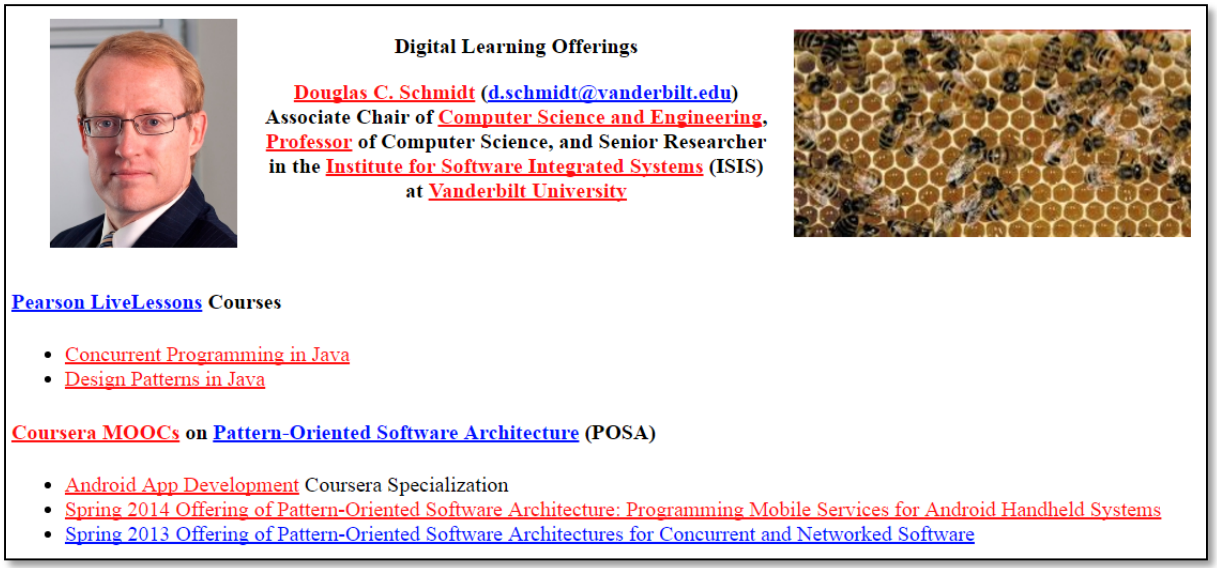
In this document

- Processes
  - Process lifecycle
- Threads
  - Worker threads
  - Thread-safe methods
- Interprocess Communication

See developer.android.com/guide/components/processes-and-threads.html

Android Linux Kernel: Processes & Threads

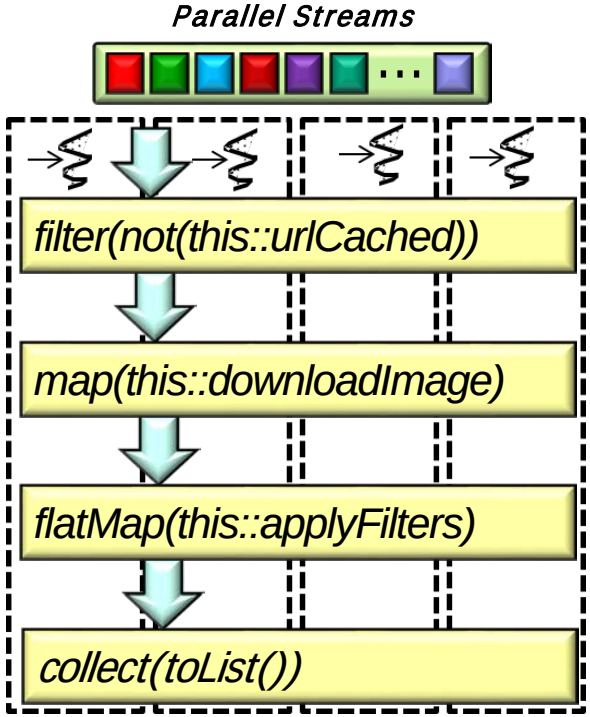
- Programming multi-threaded apps is hard
  - Think deeply about how to program threads/processes
  - Consult the Android documentation
  - & other online resources

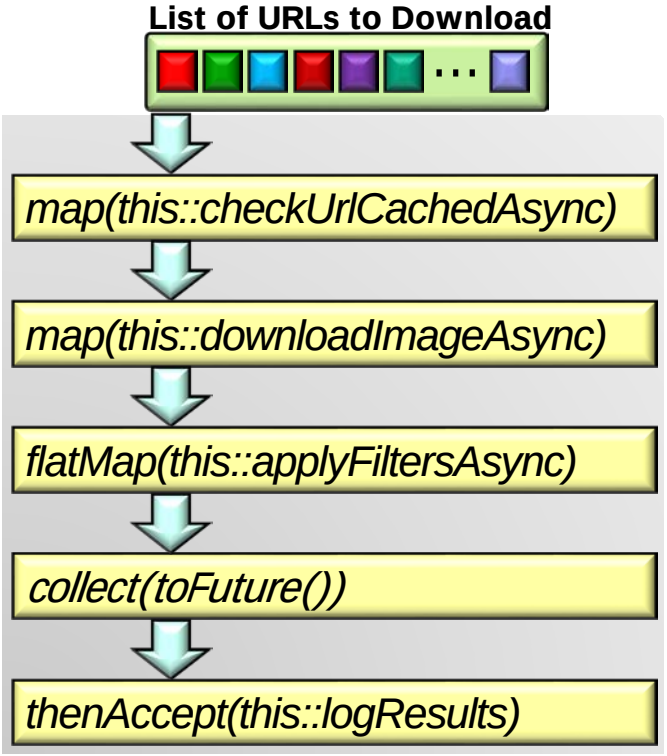


See www.dre.vanderbilt.edu/~schmidt/DigitalLearning

Android Linux Kernel: Processes & Threads

- Java 8 concurrency & parallelism frameworks alleviate many complexities of writing multi-threaded apps





See www.dre.vanderbilt.edu/~schmidt/DigitalLearning

End of the Android Linux Kernel: (Part 2) Core Kernel IPC & Processing Mechanisms