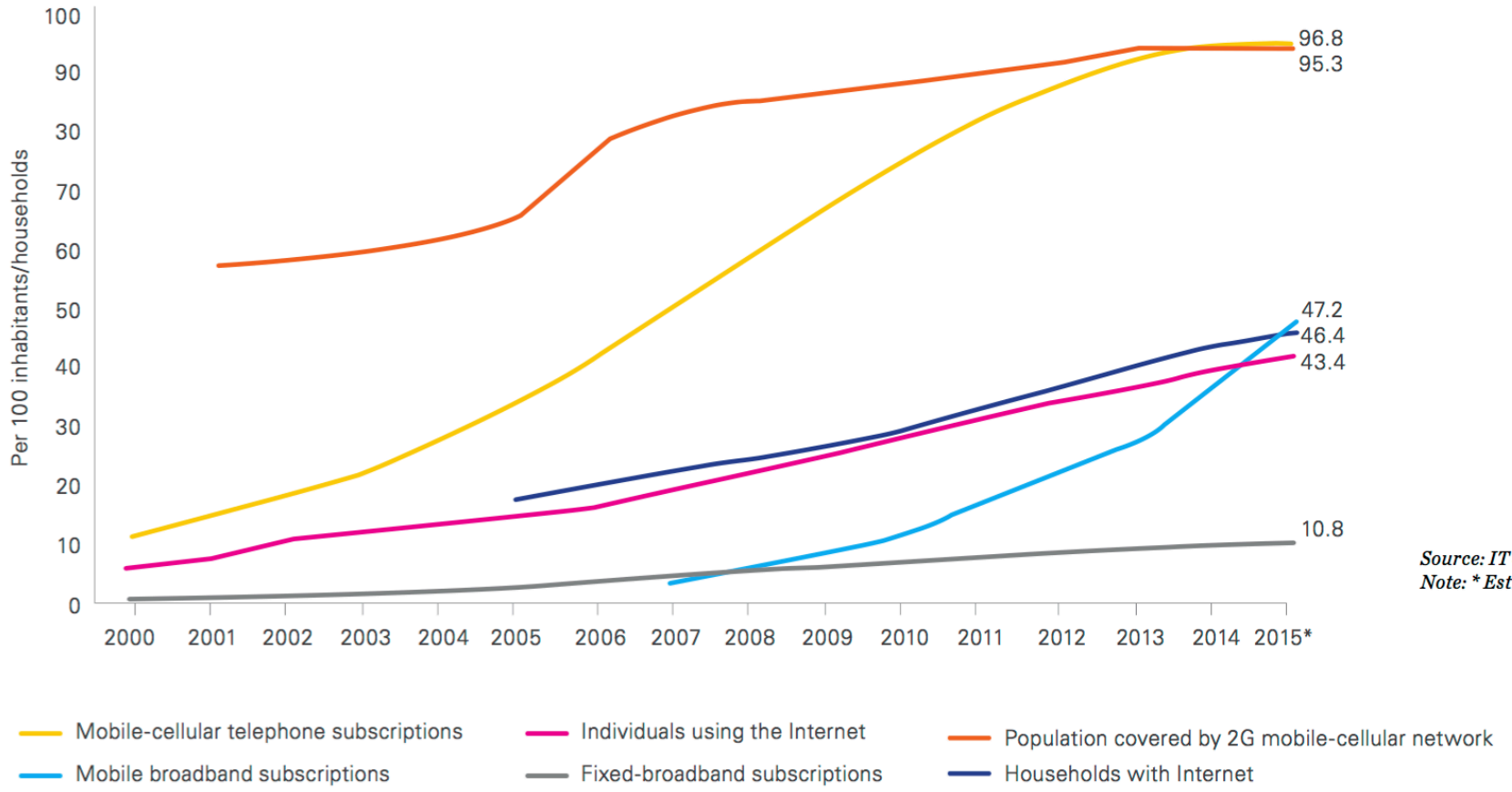
# 95-702 Distributed Systems

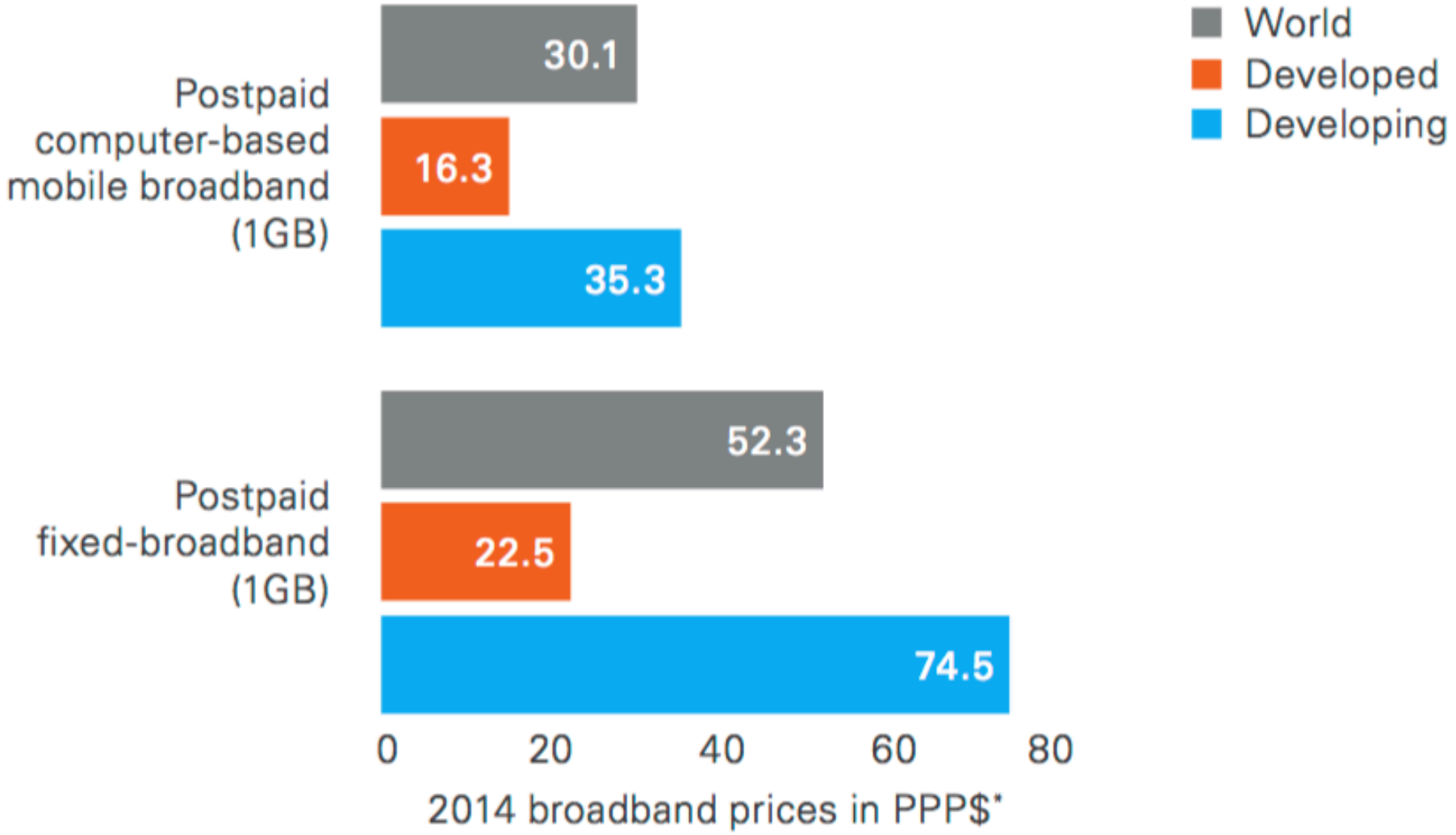
## Mobile & Android

# ITU Access Statistics



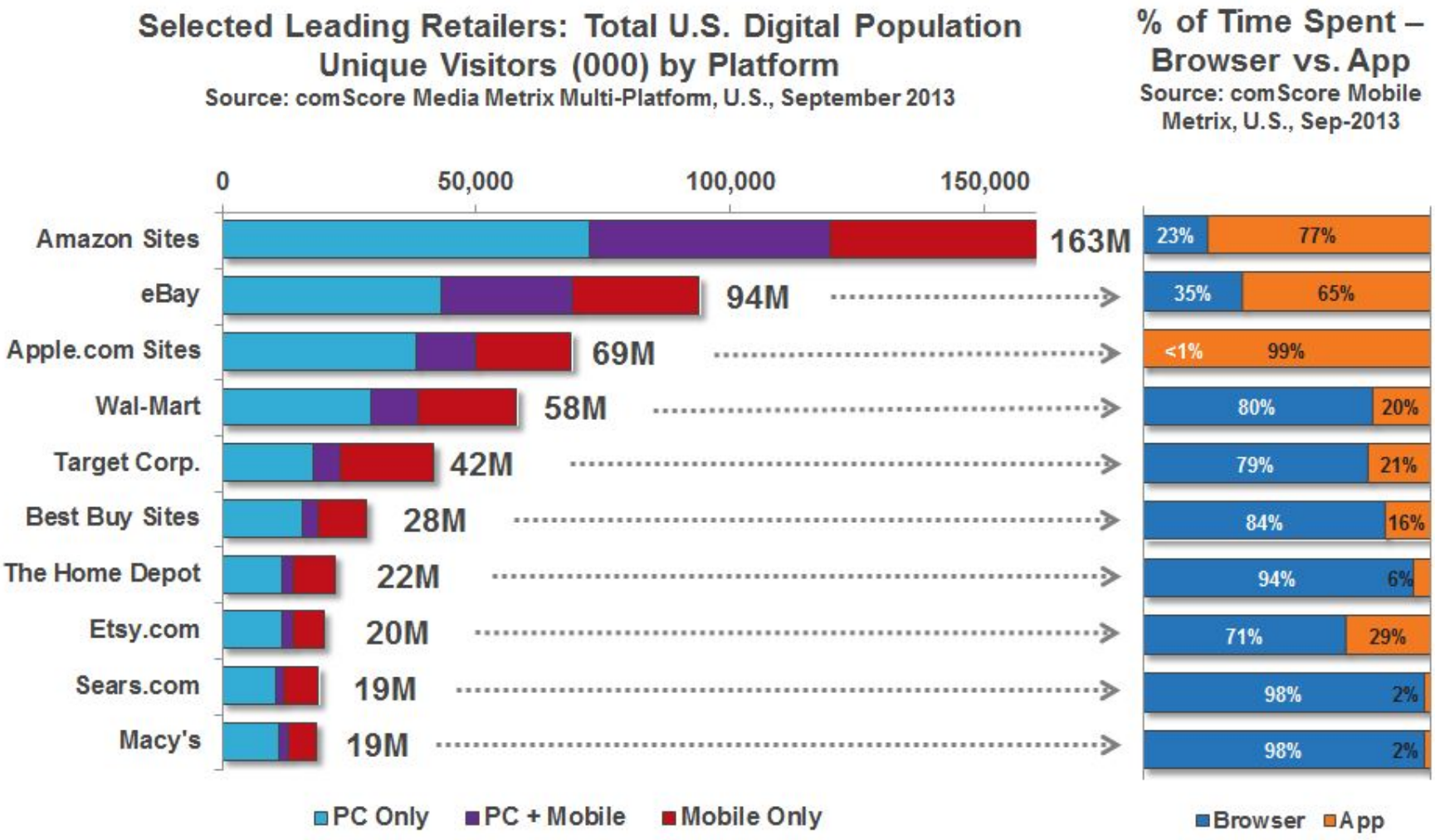

Source: http://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2015.pdf

# Mobile broadband is cheaper than fixed



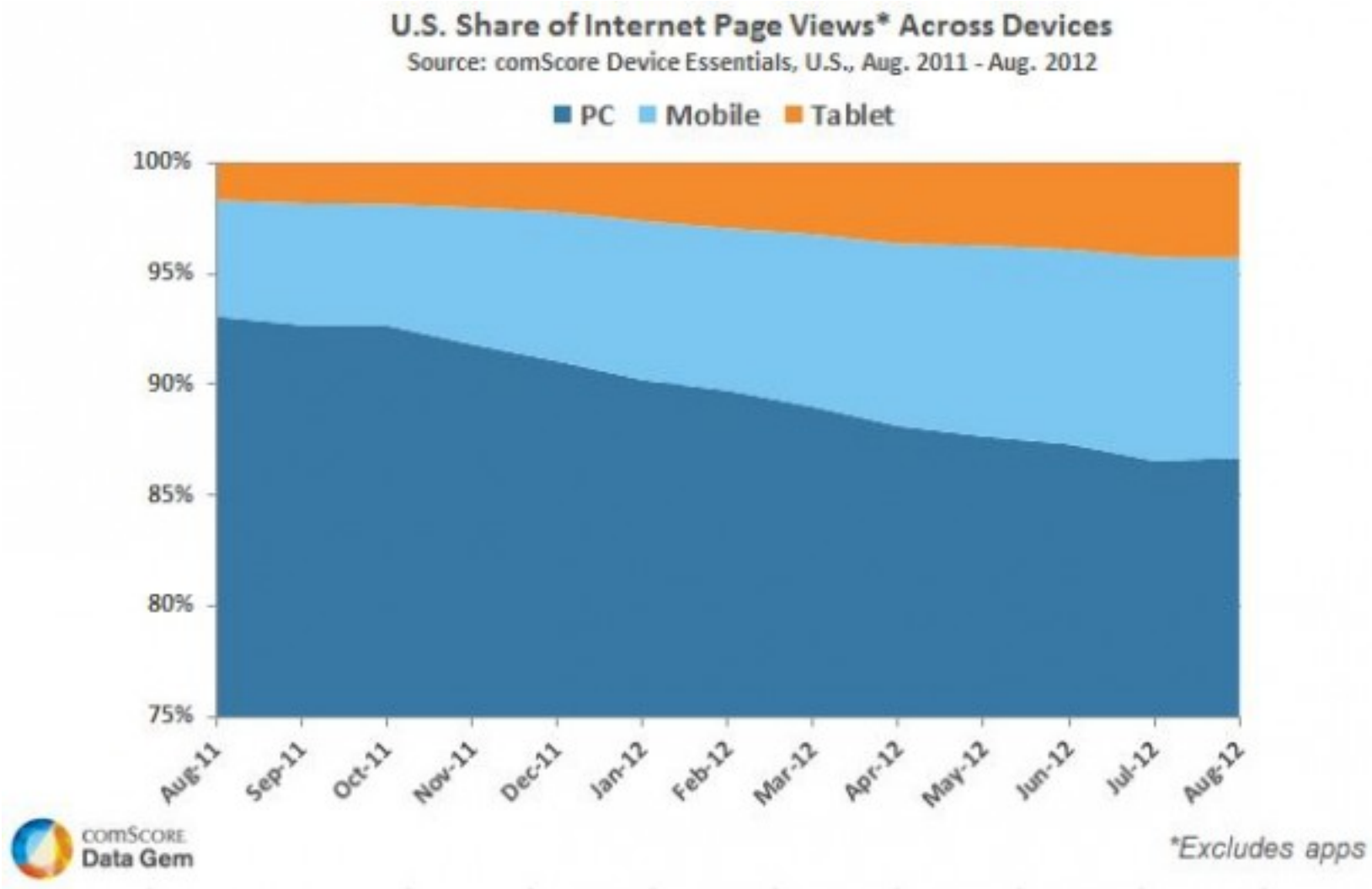

Source: http://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2015.pdf

# How do users access leading businesses?



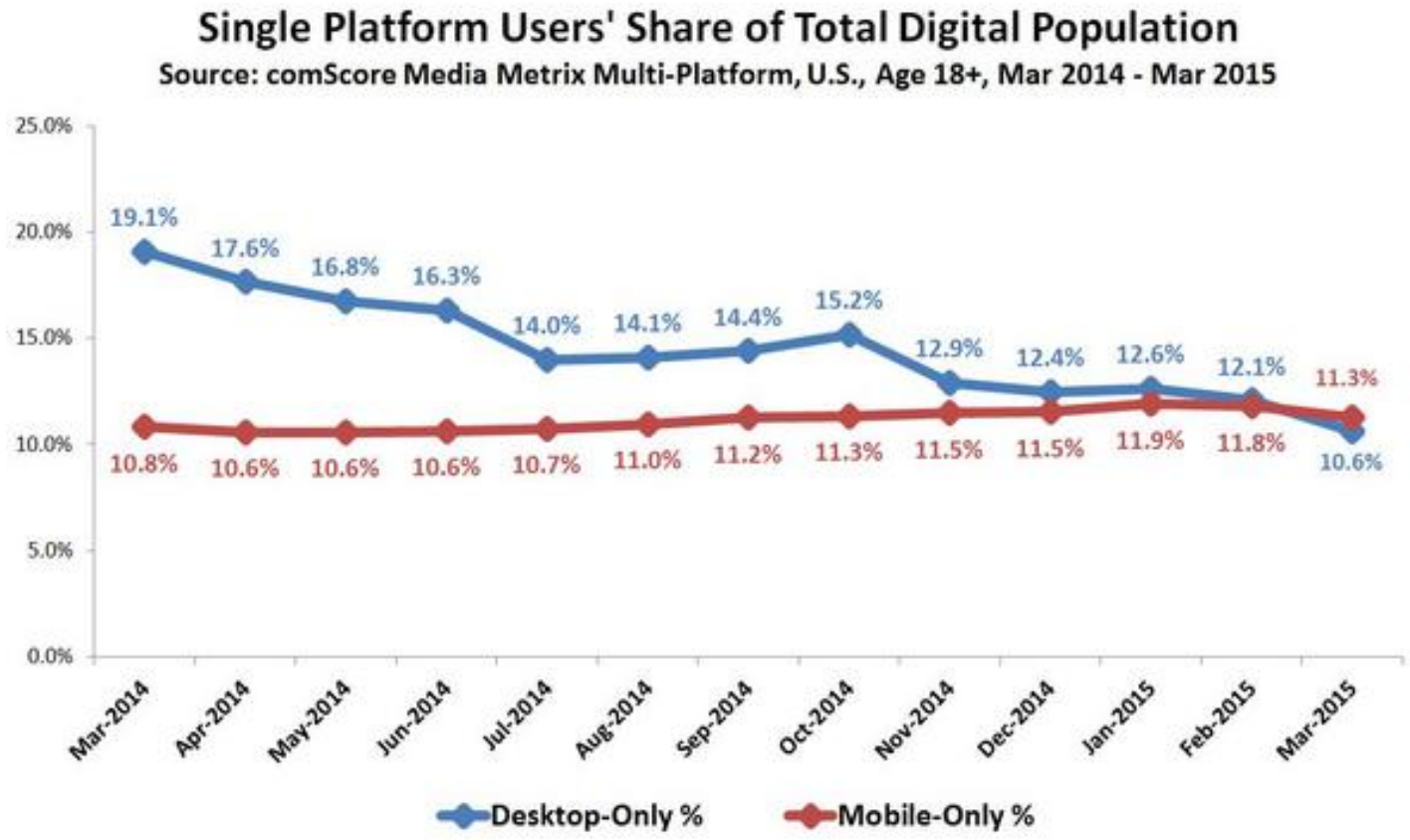

Source: http://www.comscoredatamine.com/2013/12/amazon-ebay-drive-heavy-app-usage-while-multi-channel-retailers-rely-on-mobile-web/

# Change of device use




Source:http://www.comscoredatamine.com/2012/10/mobile-phones-and-tablets-now-account-for-1-in-8-u-s-internet-page-views/

# US Mobile-Only Internet Uses > Desktop




Source: http://www.comscore.com/Insights/Blog/Number-of-Mobile-Only-Internet-Users-Now-Exceeds-Desktop-Only-in-the-U.S

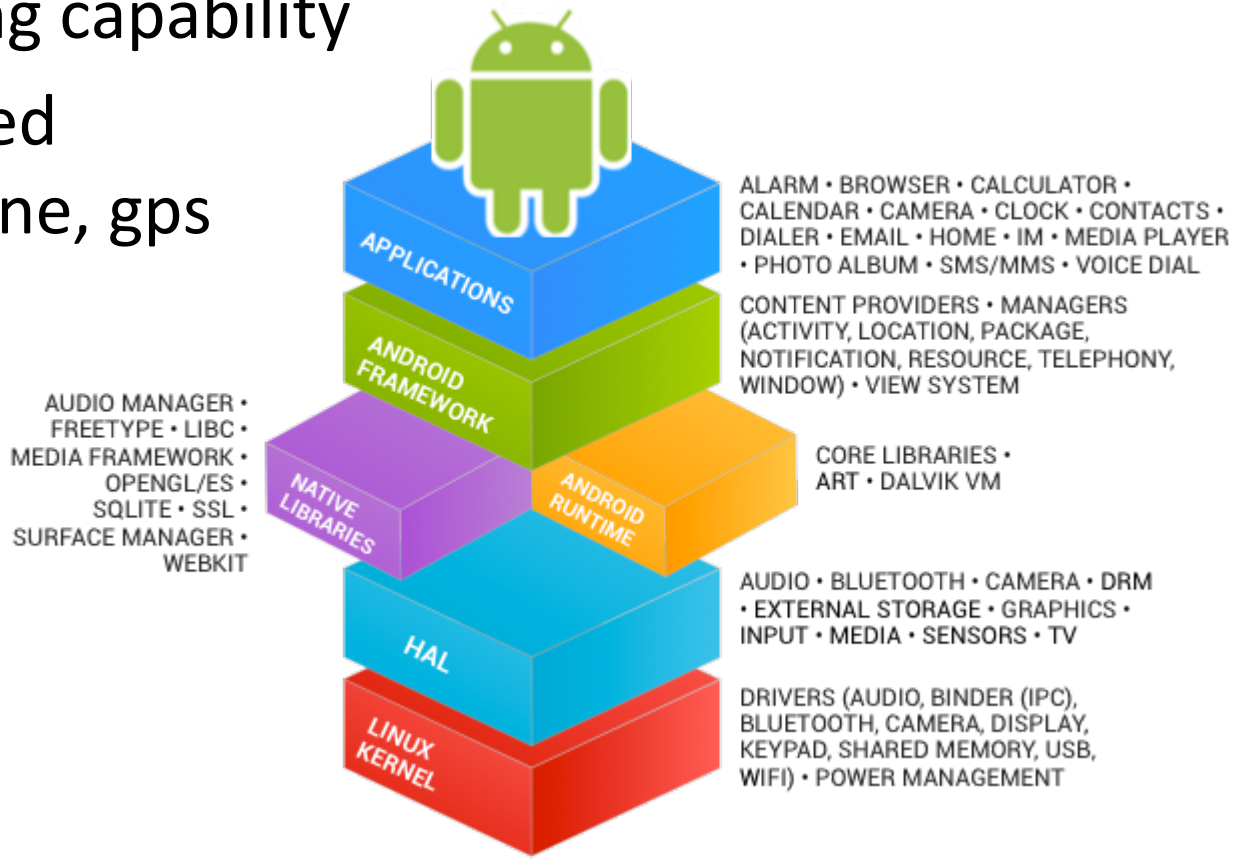
# Bottom Line

- In the foreseeable future, mobile devices will be the most prevalent way that people access the Internet, and therefore, distributed systems.

- Learning goals for today's class:
  1. To understand mobile phones as components in a distributed system.
     - I.e. as a distributed system is an application or service built from a set of heterogeneous computing systems that communicate by passing messages, a mobile phone is often one of those computing systems
  2. To create a simple native mobile application that communicates with other systems in a distributed system. (In doing so, demystify mobile apps: they are just java programs like others we've built.)

# Android architecture

- An Android phone is just:
  - a computer running Linux
  - with networking capability
  - some specialized hardware (phone, gps camera)
  - and a software stack to support writing applications



# Flow of control

| Type of application | Class? | Method(s) to implement? | Configuration information file? |
|---|---|---|---|
| Plain Java Application | | | |
| Web Application | | | |
| Android Application | | | |

# Flow of control

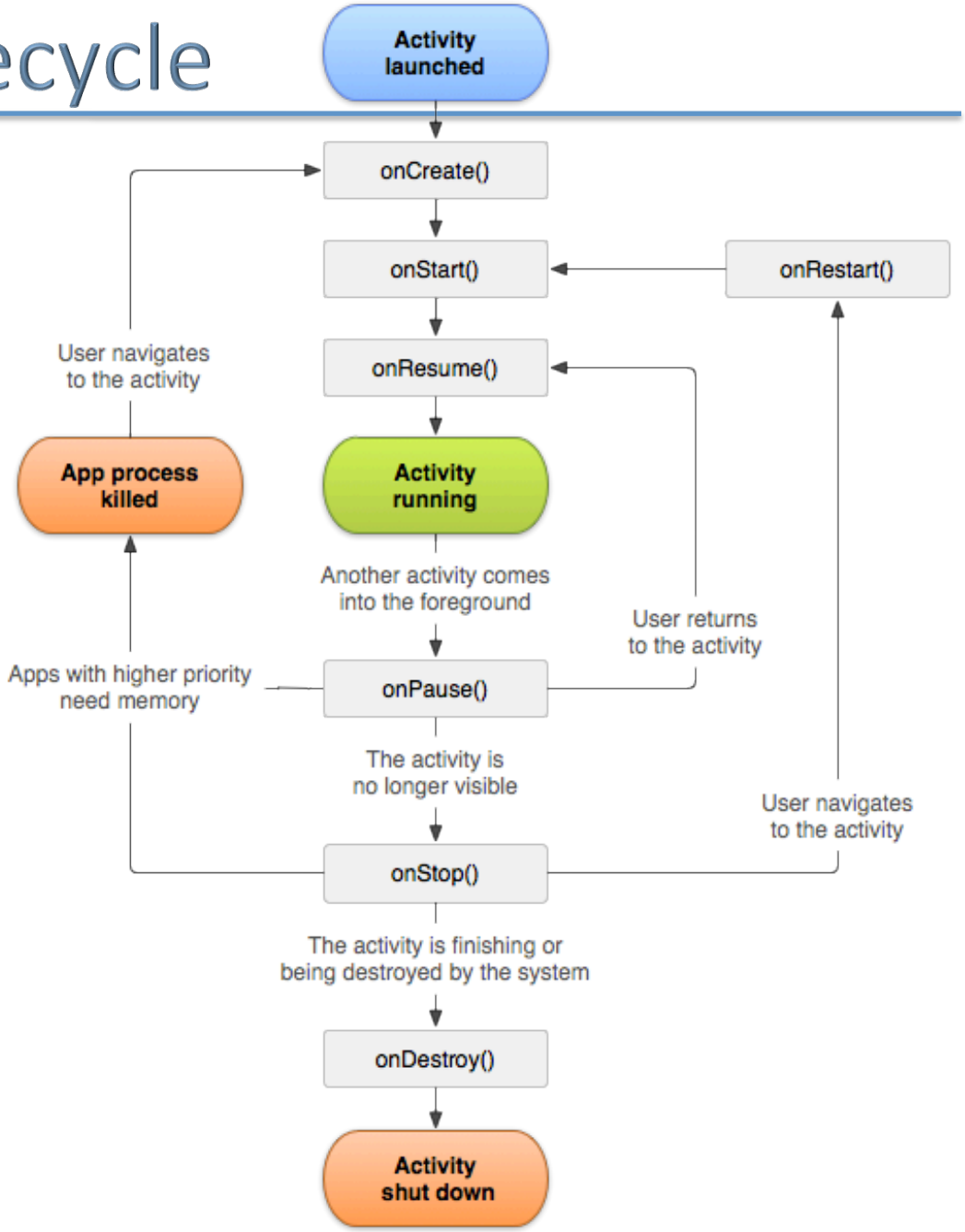
| Type of application | Class? | Method(s) to implement? | Configuration information file? |
|---|---|---|---|
| Plain Java Application | Any POJO | main() | None |
| Web Application | | | |
| Android Application | | | |

# Flow of control

| Type of application | Class? | Method(s) to implement? | Configuration information file? |
|---|---|---|---|
| Plain Java Application | Any POJO | main() | None |
| Web Application | Extend HttpServlet | doGet(), init(), doPost(), etc. | web.xml |
| Android Application | | | |

# Android activity lifecycle

- An Android app with a user interface is called an Activity
- Android apps extend Activity or a subclass such as AppCompatActivity
- A basic Android app needs to only implement the onCreate() method




Source: https://developer.android.com/reference/android/app/Activity.html

# *Which* Activity should be executed on launch?

## AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="ds.cmu.edu.helloandroid">
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name"
            android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

ds.cmu.edu.helloandroid.MainActivity

ds.cmu.edu.helloandroid.MainActivity.java

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        …
```

# Flow of control

| Type of application | Class? | Method(s) to implement? | Configuration information file? |
|---|---|---|---|
| Plain Java Application | Any POJO | main() | None |
| Web Application | Extend HttpServlet | doGet(), init(), doPost(), etc. | web.xml |
| Android Application | Extend Activity (or one of its subclasses such as AppCompatActivity) | onCreate(), onPause(), etc. | AndroidManifest.xml |

# Android app user interface

- An Android app's UI is defined
  - in xml files
  - within the *res* directory
- Two editing modes:
  - Design (WYSIWYG)
  - Text (xml)
- The complete *res* directory is compiled from xml into Java classes that define the *R* object
- Therefore *R.* refers to UI elements defined in the *res* directory
  - If *R* is undefined, it means the *res* xml has an error.

# Android *res* components

- See Table 1 at:
  https://developer.android.com/guide/topics/resources/providing-resources.html

- Some key elements
  - layout
    - How UI are arranged (linear, grid, etc.)
  - drawable
    - Picture (bitmap) files (jpg, png, etc.
  - menu
    - Application menus
  - values
    - Constant values such as static strings, colors, etc.

# Accessing the UI from the Activity

- The global variable R refers to all UI resources.
- To set the UI layout:
  - setContentView(R.layout.*activity_main*);
    - activity_main.xml would be in the res/layout directory
- To find UI elements, use findViewById:
  - Button submitButton = (Button)findViewById(R.id.*submit*);
    - This would refer to a Button in the UI with the ID == submit

# User interface interactivity problem

- Roles
  - User (person 1)
  - UI (person 2)
- Scenario
  - User tosses UI balls, which they process quickly and pass back.
  - If one of the balls is something that takes a long time, then subsequent balls tossed from User are dropped while UI completes it.
- Have you experienced this?
- How can you solve it?

# User interface interactivity problem

- New role
  - Helper (person 3)
- UI can pass off requests that will take a long time to Helper
- When Helper is finished, it can pass the results back to UI
  - The screen is a data structure, you don't want multiple threads changing it.
  - All UI changes should be done by the UI only

# Network latency

Example comparison of latency, scaled to if a CPU cycle took 1 second

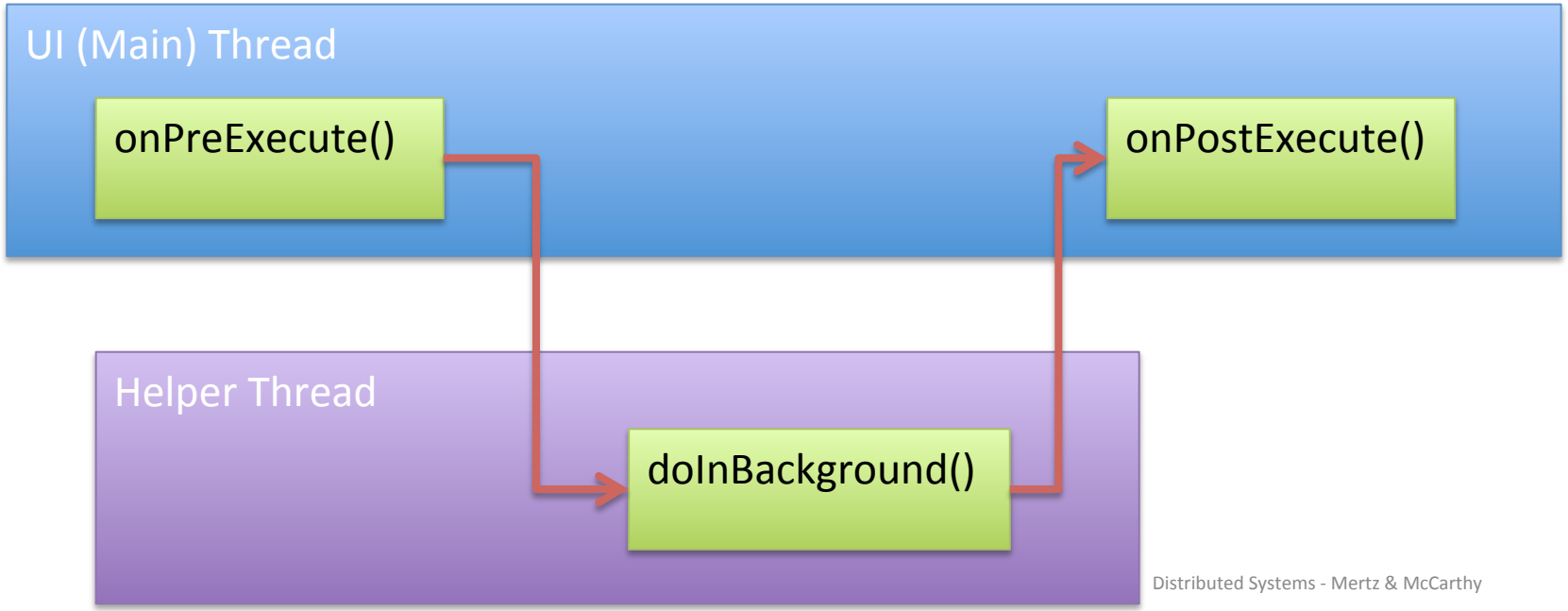
| Event | Latency | Scaled |
|---|---|---|
| 1 CPU Cycle | 0.3nsec | 1 sec |
| Main Memory | 120nsec | 6 min |
| Solid State Disk | 50 – 150 µsec | 2 – 6 days |
| Rotational Disk | 1 – 10 ms | 1 – 12 months |
| Internet SF to NYC | 40 ms | 4 years |

Source: Systems Performance: Enterprise and the Cloud 1st Edition by Brendan Gregg, ISBN-13: 978-0133390094

You do NOT want your UI waiting for 4 years before it can respond to your scroll gesture or button click!

# AsyncTask

- AsyncTask is a class that makes doing work in a helper thread easy.
  - Create a class that extends AsyncTask
  - Implement doInBackground and onPostExecute
  - Instantiate the class, and call its execute method



# Getting Started with Android Studio

- The Android developers have a set of videos.
- One is how to get started with Android Studio.
  - https://www.youtube.com/watch?v=Z98hXV9GmzY