

Mobile Programming

Ian Batten

igb@batten.eu.org

Today's Content

- Setting up the school machines for development
- Making sure your phone works
- Anatomy of an Application
- Debugging Techniques

School Machines

- You can now do Android development on a school machine (tested in UG04 this morning!)
- Add to .login:

```
module load Ant
```

- Add to .cshrc:

```
set android = /bham/pd/packages/android-sdk-linux_r22  
set path = ( $android/tools $android/platform-tools $path)
```

Test Your Phone

- Set up a null application (see last week's slides)
- Build with `ant debug`
- Install with `adb -r bin/Appname-debug.apk`
- Run it! But...

Making sure it works

- Your phone needs to be in Developer mode (seven taps on the software build version, or some other phone-specific hack).
- You might need to un-tick and re-tick the “Allow USB Debugging” option to get the authorisation dialogue if your phone has been used to develop with other machines.

Moving Source

- If you want to move a source hierarchy between machines (say, your own and the school's), be aware of:
- `project.properties`: you might need to change “`target=android-19`” to match a local app
- `local.properties`: “`sdk.dir=/Android/sdk`”

The Activity Class

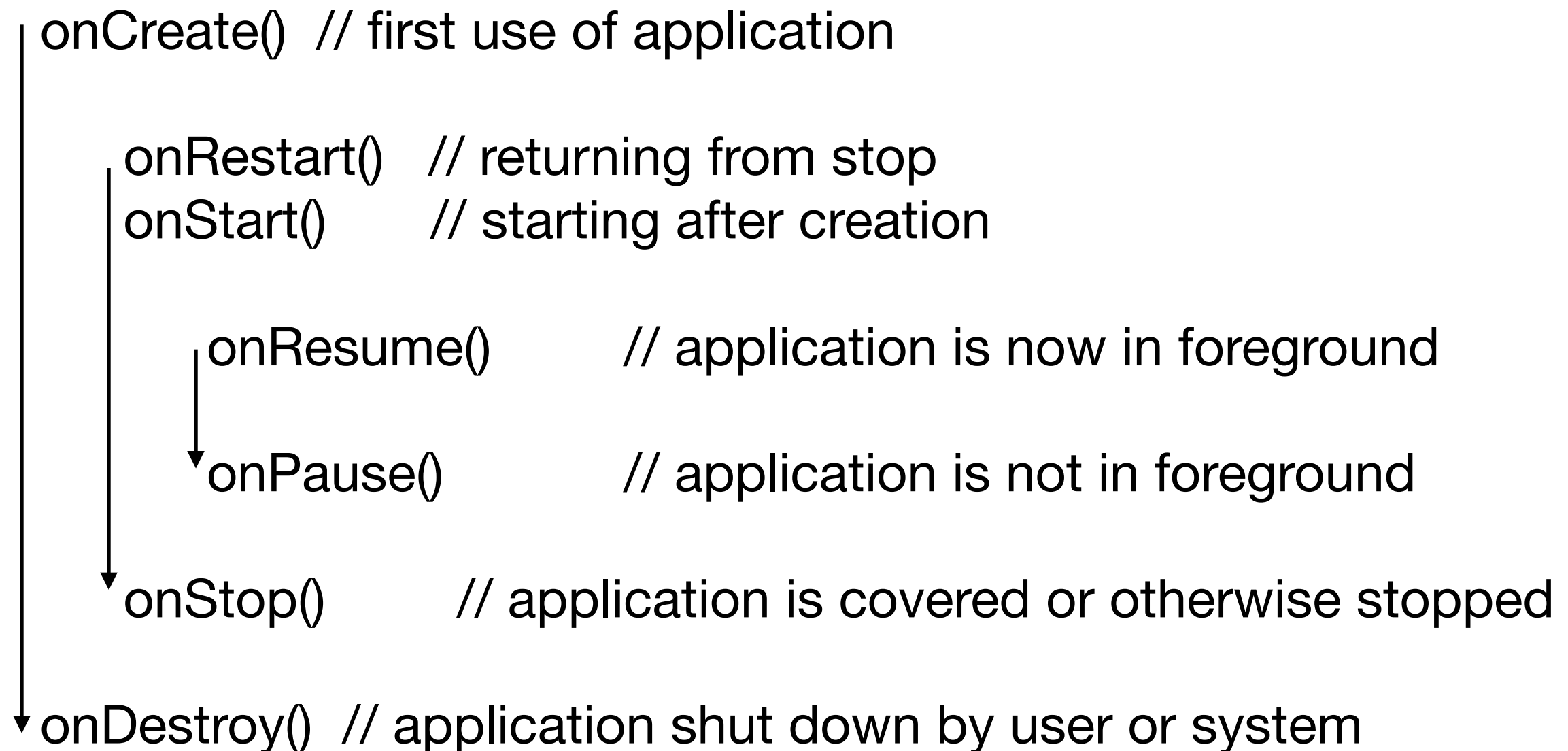
```
public class Activity extends ApplicationContext {  
    protected void onCreate(Bundle savedInstanceState);  
  
    protected void onStart();  
  
    protected void onRestart();  
  
    protected void onResume();  
  
    protected void onPause();  
  
    protected void onStop();  
  
    protected void onDestroy();  
}
```

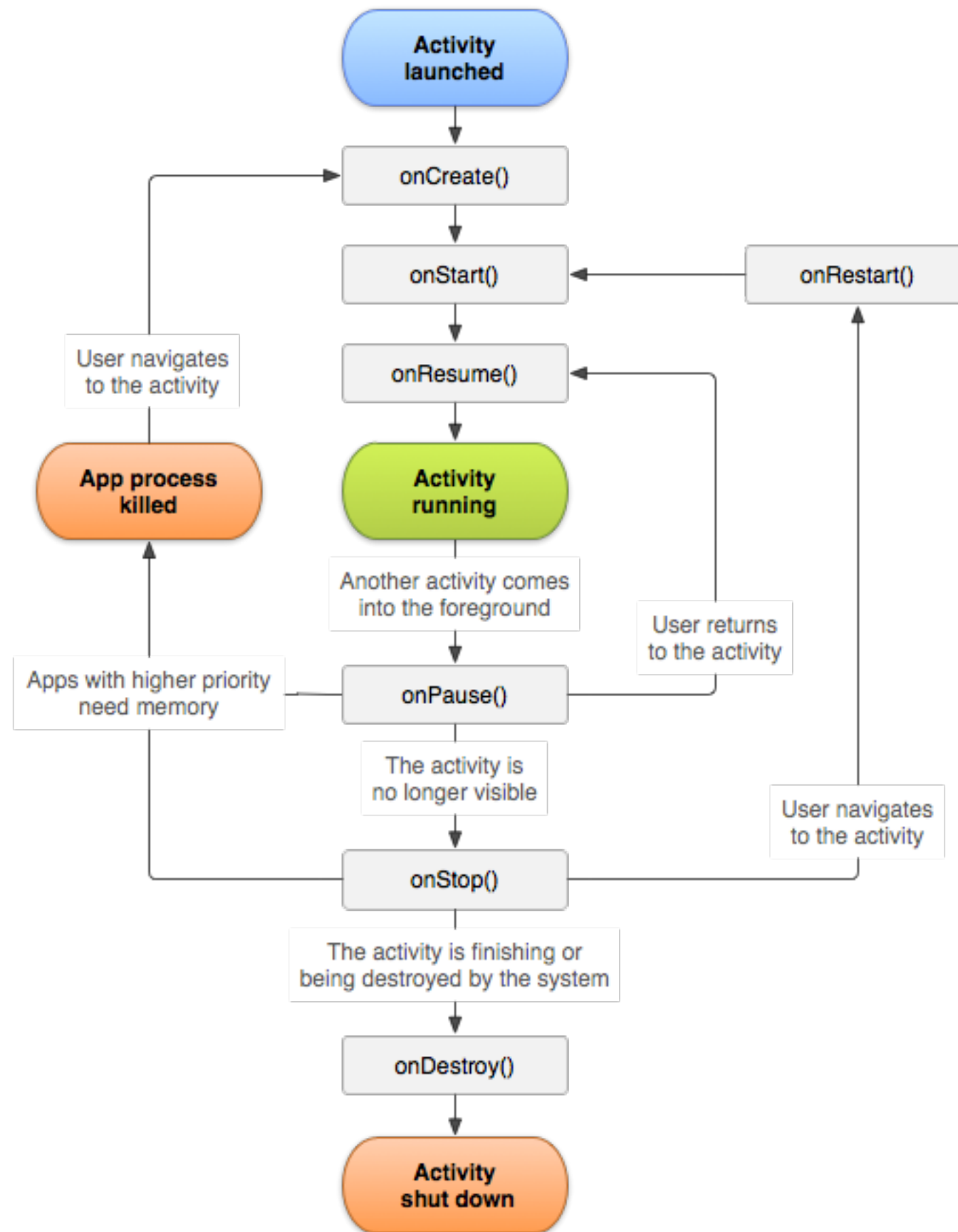
Crucial Methods

- All these methods need to call their superclass before running custom code
- `onCreate()` is called when an application starts up. It paints the initial screen, registers handlers for events where necessary and then waits for more events. A simple application only needs to override this.
- `onPause()` is called when the application loses control, and is normally used to save state (note that it will be called when you flip from landscape to portrait). Most applications will need to override this to save data or stop sensor inputs.

Lifecycle

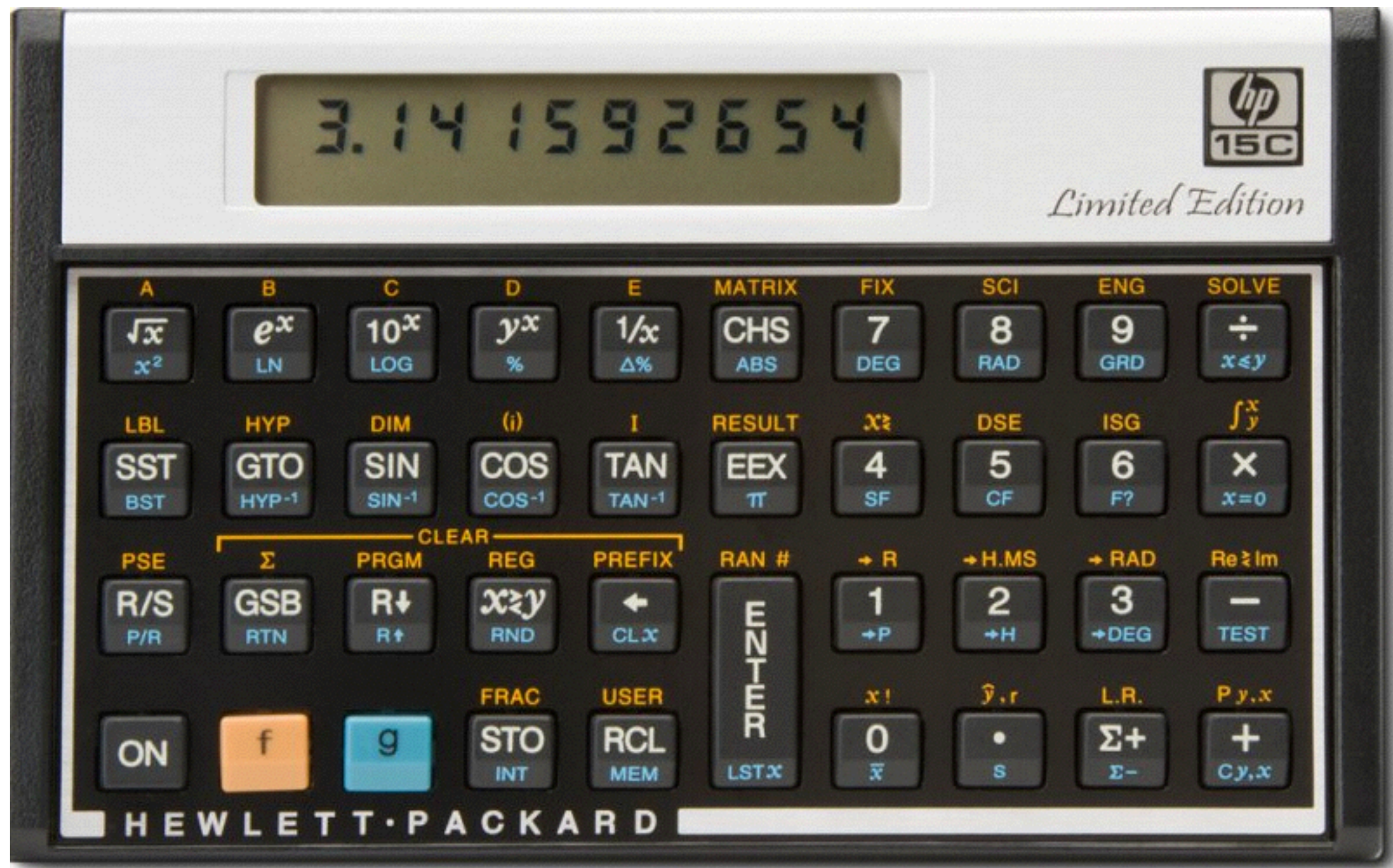
- The methods actually form pairs:





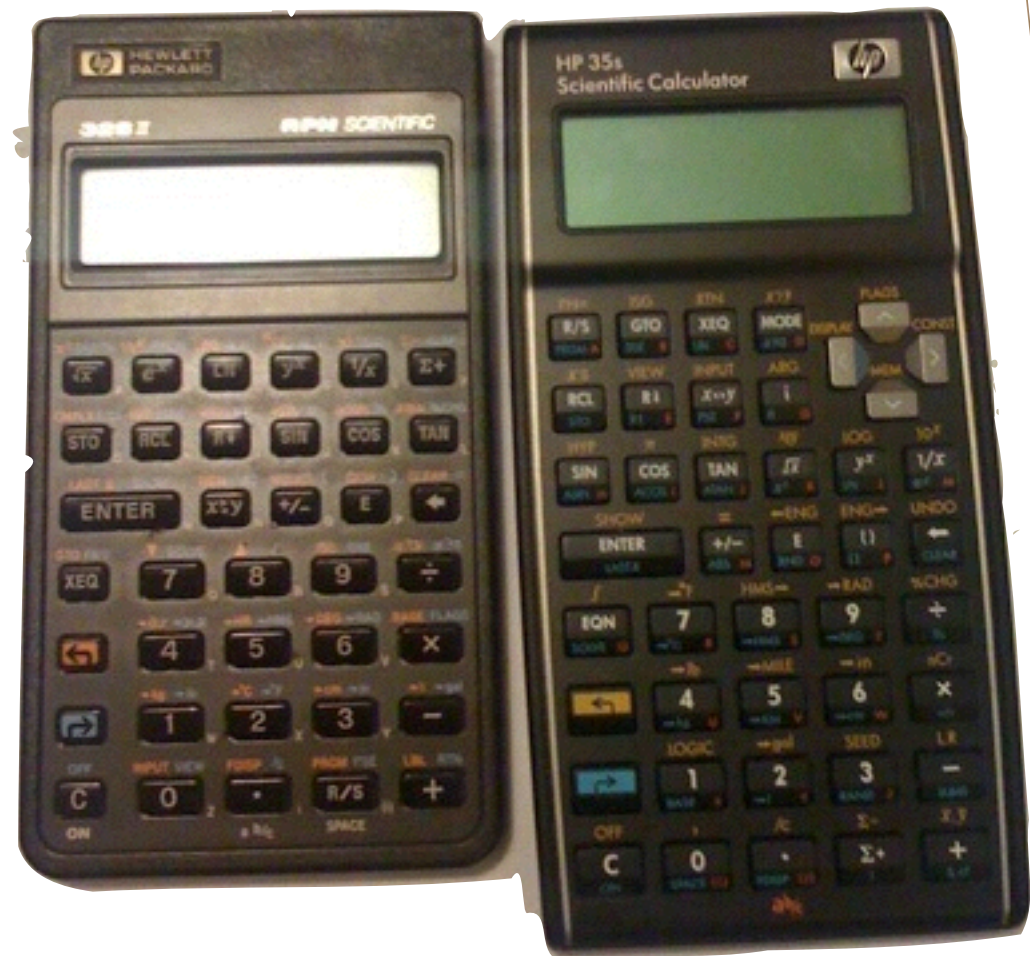
A Real Application

Reverse Polish Notation Calculators



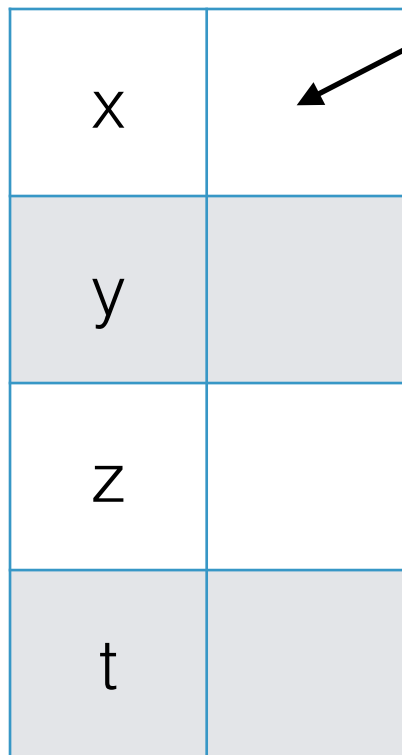
Confession

- I own several RPN Calculators, mostly HP
- Which I meant to bring in today, but didn't...
- HP15C, HP28-II, HP-35S, a credit-card sized HP15C clone...
- A friend has 4 HP16C “Programmers’ Calculators” which will “see him out” (he is sixty)
- The HP12C is a staple of Wall Street, although oddly not shown amongst the cocaine madness of “Wolf of Wall Street”. Still in production after thirty years!



RPN In Action

Display



x	
y	
z	
t	

RPN In Action

x	
y	
z	
t	

5

x	5
y	
z	
t	

RPN In Action

x	
y	
z	
t	

5

x	5
y	
z	
t	

4

x	4
y	5
z	
t	

RPN In Action

x	
y	
z	
t	

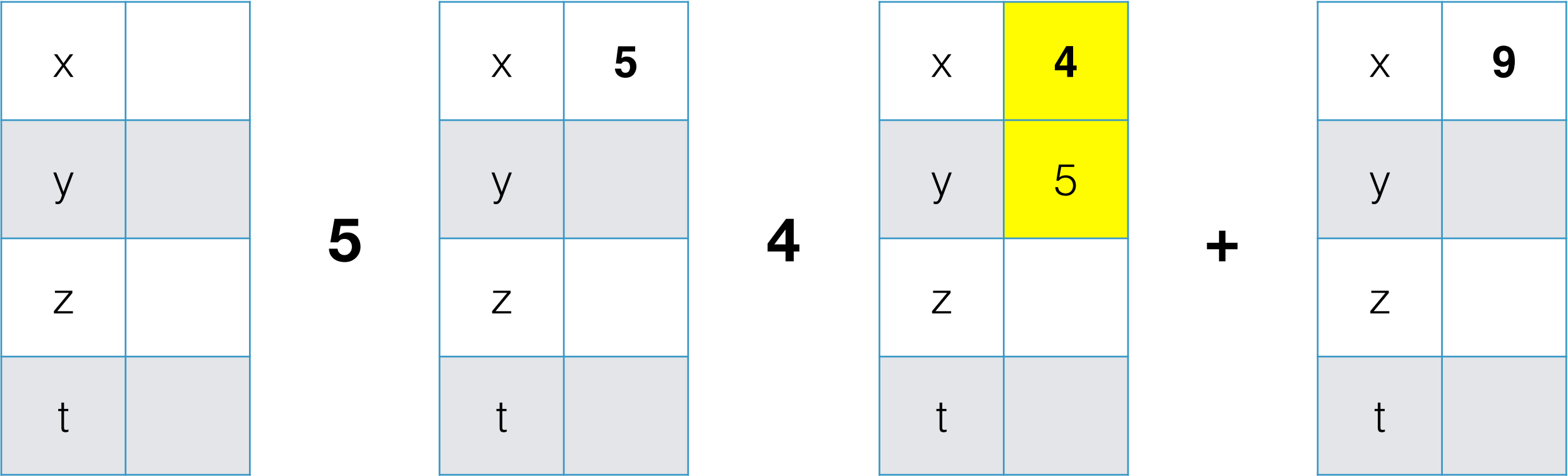
5

x	5
y	
z	
t	

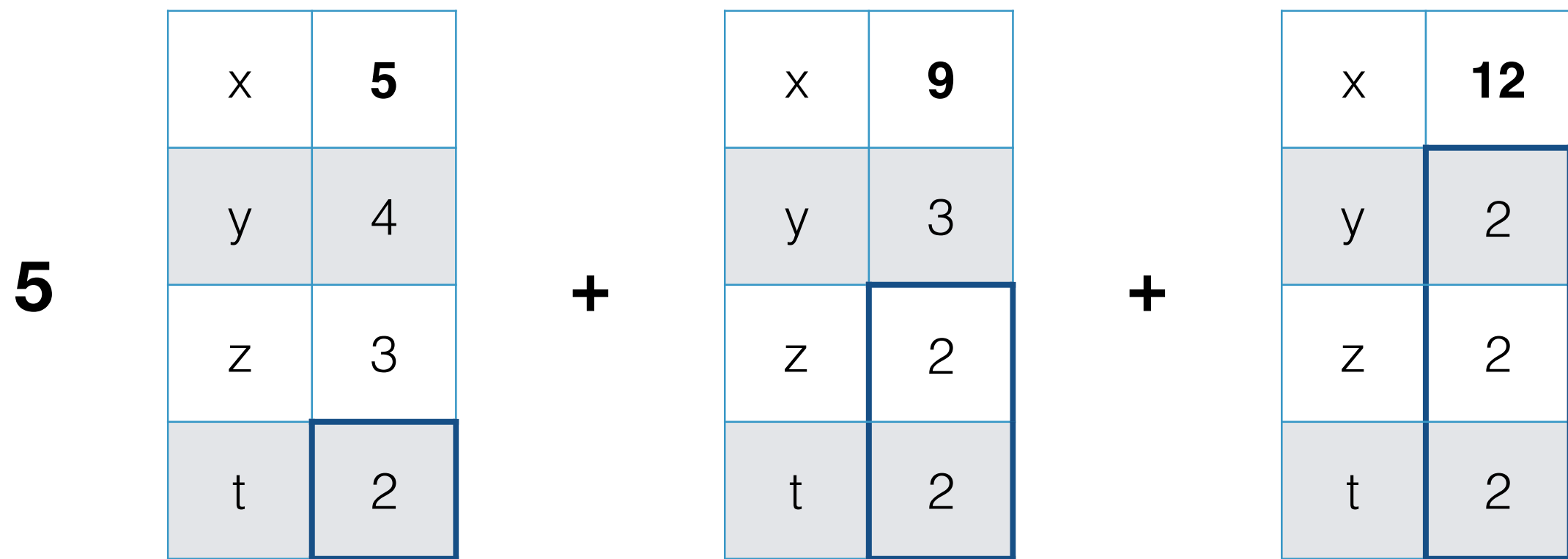
4

x	4
y	5
z	
t	

RPN In Action



Oddities



HP Manuals put the stack the other way around, with “t” at the “top” and operations taking place at the bottom. Hence “lift” is what we would call “push” and “drop” is what we would call “pop”

Source for this

- <http://www.batten.eu.org/~igb/Calc.zip>

Buttons on Screen

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".Home"
    android:background="#fff" >

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textColor="#f00"
        android:text="@string/disp"
        android:id="@+id/display"
        android:hint="@string/dispHint" />
    <LinearLayout android:orientation="horizontal"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:gravity="center"
        android:paddingTop="20dp">
        <Button
            android:layout_width="55dp"
            android:layout_height="wrap_content"
            android:id="@+id/seven"
            android:text="@string/seven"
            android:onClick="ButtonOnClick"
        />
```

Key Caps

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="app_name">MainCalc</string>
  <string name="one">1</string>
  <string name="two">2</string>
  <string name="three">3</string>
  <string name="four">4</string>
  <string name="five">5</string>
  <string name="six">6</string>
  <string name="seven">7</string>
  <string name="eight">8</string>
  <string name="nine">9</string>
  <string name="zero">0</string>
  <string name="add">+</string>
  <string name="sub">-</string>
  <string name="mul">*</string>
  <string name="div">/</string>
  <string name="cancel">CLx</string>
  <string name="point">.</string>
  <string name="enter">enter</string>
  <string name="disp">0</string>
  <string name="dispHint">0</string>
</resources>
```

For our code...

```
public class Calc extends Activity {
    TextView disp;
    BigDecimal zero = (new BigDecimal (0)).setScale (8);

    BigDecimal x = zero, y = zero, z = zero, t = zero;
    int inputScale = 0;
    boolean next_number = false;
    boolean last_was_op = false;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.calc);

        disp = (TextView) findViewById(R.id.display);
        update_display ();
    }

    public void stack_lift () {
        t = z; z = y; y = x;
    }

    public void stack_drop () {
        y = z; z = t;
    }
}
```

Event

```
public void ButtonOnClick(View arg0) {  
    switch(arg0.getId()){  
        case R.id.one:  
            add_to_display (1);  
            break;  
        case R.id.two:  
            add_to_display (2);  
            break;  
            // code deleted  
        case R.id.mul:  
            operation (y.multiply (x, MathContext.DECIMAL128));  
            break;  
        case R.id.div:  
            try {  
                operation (y.divide (x, MathContext.DECIMAL128));  
            } catch (Exception e) {  
                x = zero;  
            }  
            break;  
    }  
}
```

Bugs

- Suppose we remove the try/catch around division, and divide by zero. We get the helpful message “Unfortunately, MainCalc has stopped” when we type “6 enter 0 /”.
- “adb logcat” is your friend. It will show the crash in real time.


```
W/dalvikvm(23231): threadid=1: thread exiting with uncaught exception (group=0x41b02ba8)
E/AndroidRuntime(23231): FATAL EXCEPTION: main
E/AndroidRuntime(23231): Process: com.example.calc, PID: 23231
E/AndroidRuntime(23231): java.lang.IllegalStateException: Could not execute method of the activity
E/AndroidRuntime(23231):     at android.view.View$1.onClick(View.java:3823)
E/AndroidRuntime(23231):     at android.view.View.performClick(View.java:4438)
E/AndroidRuntime(23231):     at android.view.View$PerformClick.run(View.java:18422)
E/AndroidRuntime(23231):     at android.os.Handler.handleCallback(Handler.java:733)
E/AndroidRuntime(23231):     at android.os.Handler.dispatchMessage(Handler.java:95)
E/AndroidRuntime(23231):     at android.os.Looper.loop(Looper.java:136)
E/AndroidRuntime(23231):     at android.app.ActivityThread.main(ActivityThread.java:5017)
E/AndroidRuntime(23231):     at java.lang.reflect.Method.invokeNative(Native Method)
E/AndroidRuntime(23231):     at java.lang.reflect.Method.invoke(Method.java:515)
E/AndroidRuntime(23231):     at com.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:779)
E/AndroidRuntime(23231):     at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:595)
E/AndroidRuntime(23231):     at dalvik.system.NativeStart.main(Native Method)
E/AndroidRuntime(23231): Caused by: java.lang.reflect.InvocationTargetException
E/AndroidRuntime(23231):     at java.lang.reflect.Method.invokeNative(Native Method)
E/AndroidRuntime(23231):     at java.lang.reflect.Method.invoke(Method.java:515)
E/AndroidRuntime(23231):     at android.view.View$1.onClick(View.java:3818)
E/AndroidRuntime(23231):     ... 11 more
E/AndroidRuntime(23231): Caused by: java.lang.ArithmeticException: Division by zero
E/AndroidRuntime(23231):     at java.math.BigDecimal.divide(BigDecimal.java:1199)
E/AndroidRuntime(23231):     at java.math.BigDecimal.divide(BigDecimal.java:1279)
E/AndroidRuntime(23231):     at com.example.calc.Calc.ButtonOnClick(Calc.java:141)
E/AndroidRuntime(23231):     ... 14 more
W/ActivityManager( 769): Force finishing activity com.example.calc/.Calc
```

Adding our own logging

```
import android.util.Log;
```

```
case R.id.div:
    try {
        operation (y.divide (x, MathContext.DECIMAL128));
    } catch (Exception e) {
        Log.w ("calc", "caught exception " + e.getMessage ());
        x = zero;
    }
    break;
```

It's a mess, but...

I/GAV2 (23583): Thread[GAThread,5,main]: No campaign data found.

D/dalvikvm(23583): GC_CONCURRENT freed 325K, 3% free 17261K/17620K, paused 1ms+1ms, total 16ms

W/calculator (23633): caught exception Division by zero

W/GAV2 (23583): Thread[Service Reconnect,5,main]: Service unavailable (code=1), using local store.

W/ActivityManager(769): Unable to start service Intent { act=com.google.android.gms.analytics.service.START cmp=com.google.android.gms/.analytics.service.AnalyticsService (has extras) } U=0: not found