

# Android Tutorial: Intents Part I -- Invoking Activities by Class Name

## Example Target Activity: Loan Calculator

- **Inputs**

- Loan amount
- Interest rate (as a percent)
- Loan period in months

- **Outputs**

- Monthly payment
  - Both are in same units (e.g., dollars) as the loan amount
- Total payments over life of loan

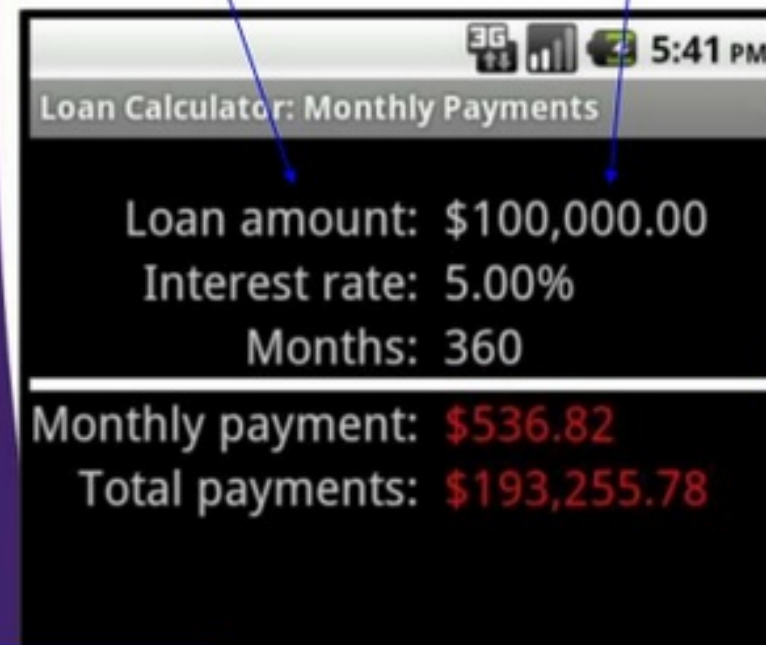
- **Defaults**

- Unless values are passed in from other Activity, uses default values for all inputs



Entries in first column are right-aligned.

Entries in second column are left-aligned. They are also given ids so that the Java code can insert the text.



This is a View with android:column\_span="2" and a fixed height.

TableLayout

## XML: Layout File: First Row (res/layout/loan\_payment.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://..."
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:stretchColumns="1">
    <TableRow android:layout_marginTop="20dp">
        <TextView android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textColor="@color/default_foreground"
            android:textSize="@dimen/font_size"
            android:text="@string/loan_amount_prompt"
            android:gravity="right"/>
        <TextView android:id="@+id/loan_amount"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textColor="@color/default_foreground"
            android:textSize="@dimen/font_size"
            android:gravity="left"/>
    </TableRow>
```



Second and third rows are very similar. Bottom two rows are almost the same except for a different text color for the second column.

## XML: Layout File: Divider (res/layout/loan\_payment.xml)

```
<TableRow>
    <TextView android:layout_span="2"
        android:layout_width="match_parent"
        android:layout_height="5dp"
        android:background="@color/divider_background"/>
</TableRow>
```



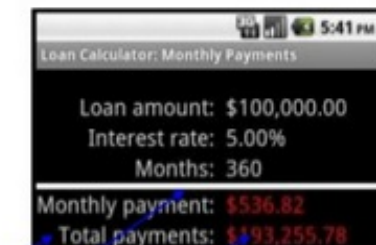
## XML: Strings File (res/values/strings.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Intent Filters and Activity Switching</string>
    <string name="loan_calculator_app_name">
        Loan Calculator: Monthly Payments
    </string>
    <string name="tabs_app_name">Tabbed Windows</string>
    <string name="loan_amount_prompt">Loan amount: &#160;&#160;</string>
    <string name="interest_rate_prompt">Interest rate: &#160;&#160;</string>
    <string name="loan_period_prompt">Months: &#160;&#160;</string>
    <string name="monthly_payment_prompt">Monthly payment: &#160;&#160;</string>
    <string name="total_payments_prompt">Total payments: &#160;&#160;</string>
</resources>
```



## XML: Colors File (res/values/colors.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <!-- Used inside loan_payments.xml. -->
    <color name="default_foreground">#d3d3d3</color>
    <color name="divider_background">#ffffff</color>
    <color name="result_foreground">#ff0000</color>
</resources>
```






```

public class LoanCalculatorActivity extends Activity {
    private double mLoanAmount=100000,
                mAnnualInterestRateInPercent=5.0;
    private long mLoanPeriodInMonths=360; // 30 years
    private String mCurrencySymbol = "$";

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.loan_payments);
        setInputsFromExtras();
        setInputsFromUri();
        calculateAndSetOutputValues();
    }

```



```

private void calculateAndSetOutputValues() {
    PaymentInfo paymentInfo =
        new PaymentInfo(mLoanAmount, mAnnualInterestRateInPercent,
                        mLoanPeriodInMonths, mCurrencySymbol);

    TextView loanAmountDisplay = (TextView)findViewById(R.id.loan_amount);
    loanAmountDisplay.setText(paymentInfo.getFormattedLoanAmount());
    TextView interestRateDisplay =
        (TextView)findViewById(R.id.interest_rate);
    interestRateDisplay.setText
        (paymentInfo.getFormattedAnnualInterestRateInPercent());
    TextView loanPeriodDisplay = (TextView)findViewById(R.id.loan_period);
    loanPeriodDisplay.setText(paymentInfo.getFormattedLoanPeriodInMonths());
    TextView monthlyPaymentDisplay =
        (TextView)findViewById(R.id.monthly_payment);
    monthlyPaymentDisplay.setText(paymentInfo.getFormattedMonthlyPayment());
    TextView totalPaymentsDisplay =
        (TextView)findViewById(R.id.total_payments);
    totalPaymentsDisplay.setText(paymentInfo.getFormattedTotalPayments());
}

```

## Java (PaymentInfo.java)

```

public class PaymentInfo {
    private final double mLoanAmount, mAnnualInterestRateInPercent,
                        mMonthlyPayment, mTotalPayments;
    private final long mLoanPeriodInMonths;
    private final String mCurrencySymbol;

    public PaymentInfo(double loanAmount, double annualInterestRateInPercent,
                        long loanPeriodInMonths, String currencySymbol) {
        mLoanAmount = loanAmount;
        mAnnualInterestRateInPercent = annualInterestRateInPercent;
        mLoanPeriodInMonths = loanPeriodInMonths;
        mCurrencySymbol = currencySymbol;
        mMonthlyPayment = LoanUtils.monthlyPayment(loanAmount,
                                                    annualInterestRateInPercent,
                                                    loanPeriodInMonths);
        mTotalPayments = mMonthlyPayment * mLoanPeriodInMonths;
    }
}

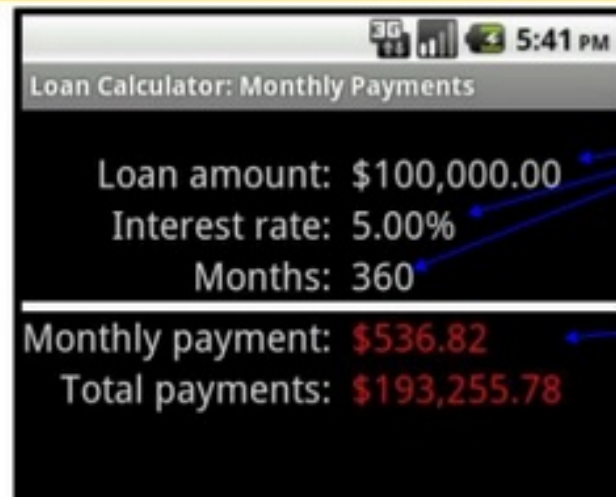
```

## Java (LoanUtils.java)

```

public class LoanUtils {
    public static double monthlyPayment(double loanAmount,
                                        double annualInterestRateInPercent,
                                        long loanPeriodInMonths) {
        if (annualInterestRateInPercent <= 0) {
            annualInterestRateInPercent = 0.0000001;
        }
        double monthlyInterestRate = annualInterestRateInPercent / 1200.0;
        double numerator = loanAmount * monthlyInterestRate;
        double denominator =
            1 - Math.pow(1 + monthlyInterestRate, -1 * loanPeriodInMonths);
        return (numerator / denominator);
    }
}

```



For now, these values are fixed to the initial values set for the instance variables of the `LoanCalculatorActivity`. However, the upcoming sections will show how to pass values from another Activity to this one.

Computed by `LoanUtils`.  
Formatted by `PaymentInfo`.

# Invoking Activities by Class Name

- **Idea**

- Specify class name of new Activity
  - New Activity must be in same project as original Activity

- **Syntax**

- Java (original Activity)

```
Intent activityIntent = new Intent(this, NewActivity.class);
startActivity(activityIntent);
```
- XML (`AndroidManifest.xml`)

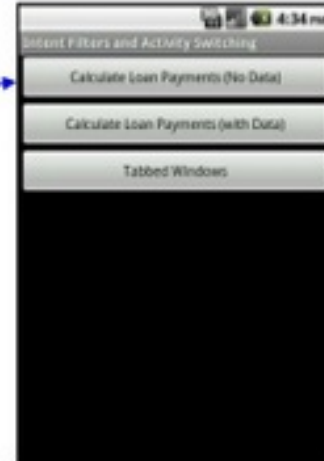
```
<activity android:name=".NewActivity"
    android:label="@string/some_app_name">
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />
        <category android:name="android.intent.category.DEFAULT"/>
    </intent-filter>
</activity>
```



# Example: Invoking Loan Calculator with Default Values

- **Initial Activity**

- Has Button that, when pressed, invokes the loan calculator activity
  - No data is sent to the loan calculator, so it will use default values for the loan amount, interest rate, etc.



- **Approach**

- Create Intent referring to LoanCalculatorActivity.class
  - Thus, the two Activities must be in same project
- Call startActivity
- Put entry for LoanCalculatorActivity in AndroidManifest.xml
  - So that the initial Activity has permission to invoke the loan calculator

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://..."
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <Button android:text="Calculate Loan Payments (No Data)"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:onClick="showLoanPayments1"/>
    ...
</LinearLayout>
```



# XML: Manifest File Template (AndroidManifest.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.coreservlets.intentfilter1"
    android:versionCode="1"
    android:versionName="1.0">
    <uses-sdk android:minSdkVersion="8" />

    <application android:icon="@drawable/icon"
        android:label="@string/app_name">
        <activity android:name=".IntentFilter1Activity"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        ...
    </application>
</manifest>
```

This part is generated automatically when you build a new Android project in Eclipse.

Means that this is the Action that runs when you run the project.

Means that this app gets an icon on the screen of your Android device.

Other Activities will be declared here. See next slide for LoanCalculatorActivity.

# XML: Manifest File Action Declaration

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.coreservlets.intentfilter1"
    android:versionCode="1"
    android:versionName="1.0">
    <uses-sdk android:minSdkVersion="8" />

    <application android:icon="@drawable/icon"
        android:label="@string/app_name">
        ... <!-- Declaration for IntentFilter1Activity on previous slide -->
        <activity android:name=".LoanCalculatorActivity"
            android:label="@string/loan_calculator_app_name">
            <intent-filter>
                <action android:name="android.intent.action.VIEW" />
                <category android:name="android.intent.category.DEFAULT" />
                ... <!-- "data" entry shown later; not used in this example -->
            </intent-filter>
        </activity>
        ...
    </application>
</manifest>
```

Use the fully-qualified name if the new Activity is in a different package than the main one (i.e., the one listed at the top in the "manifest" start tag).

Means that this Action displays data to the user, but is not launched as the initial Activity of the project.

Means that this Action can be the default for certain types of data (shown later).

The "data" tag of the land the "activity" tag for the tabbed windows Activity are both shown later.

You virtually always set action.VIEW and category.DEFAULT for Activities that will be invoked by other Activities.

# XML: Strings File (res/values/strings.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">
        Intent Filters and Activity Switching
    </string>
    ...
</resources>
```

```
<Button android:text="Calculate Loan Payments (No Data)"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:onClick="showLoanPayments1" />
```



# Sending Data via the “Extras” Bundle

- **Idea**

- Attach a Bundle (like a Map – see next slides) to the Intent. The Bundle will contain data to be used by the new Activity.

- **Syntax**

- Java (original Activity)

```
Intent activityIntent = new Intent(this, NewActivity.class);
Bundle newActivityInfo = new Bundle();
newActivityInfo.putBlah(...); // putDouble, putString, etc.
activityIntent.putExtras(newActivityInfo);
startActivity(activityIntent);
```

- Java (new Activity)

```
Intent intent = getIntent();
Bundle info = intent.getExtras();
if (info != null) { /* Retrieve vals with info.getBlah(...) */ }
```

- **Putting data in a Bundle**

- putBoolean, putBooleanArray, putDouble, putDoubleArray, putString, putStringArray, putStringArrayList etc.
  - These all take keys and values as arguments. The keys must be Strings. The values must be of the standard types (int, double, etc.) or array of them.
    - You can also make a custom class that implements Serializable or Parcelable, then store instance of that class with putSerializable or putParcelable
  - Methods return void, so you cannot chain as with the putExtra method of Intent.

- **Retrieving data from a Bundle**

- getBoolean, getBooleanArray, getDouble, getDoubleArray, getString, getStringArray, getStringArrayList, etc.
  - These take keys (Strings) as arguments. No typecast required on retrieval.

# Option 1: Attaching Entire Bundle to Intent

- **Idea**

- Make a Bundle, add it all at once to Intent.
  - Instantiate a Bundle, then use the Bundle's *putBlah* method (one such method for each standard type). Then, attach Bundle to Intent with Intent's *putExtras* method.

- **Syntax**

```
Bundle newActivityInfo = new Bundle();
newActivityInfo.putDouble("key1", someDouble);
newActivityInfo.putString("key2", someString);
...
yourIntent.putExtras(newActivityInfo);
```

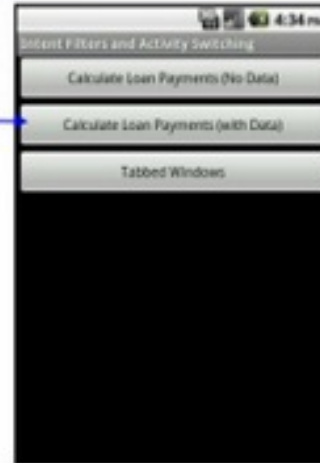
- Note that it is *putExtras*, not *putExtra*



## Example: Invoking Loan Calculator with Custom Values

- **Initial Activity**

- Has Button that, when pressed, invokes the loan calculator activity
  - Creates randomized data and sends it to the loan calculator, which retrieves the values and uses them for its calculations.



- **Approach**

- Create Intent referring to LoanCalculatorActivity.class
- Create Bundle with 3 values
  - Loan amount, interest rate, loan period
- Attach Bundle to Intent with putExtras
- Call startActivity
- Put entry for LoanCalculatorActivity in manifest

## XML: Layout File (res/layout/main.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://..."
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    ...
    <Button
        android:text="Calculate Loan Payments (with Data)"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:onClick="showLoanPayments2"/>

    ...
</LinearLayout>
```

First button shown earlier

# XML: Manifest File Action Declaration

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.coreservlets.intentfilter1"
    android:versionCode="1"
    android:versionName="1.0">
    <uses-sdk android:minSdkVersion="8" />

    <application android:icon="@drawable/icon"
        android:label="@string/app_name">
        ... <!-- Declaration for IntentFilter1Activity on previous slide -->
        <activity android:name=".LoanCalculatorActivity"
            android:label="@string/loan_calculator_app_name">
            <intent-filter>
                <action android:name="android.intent.action.VIEW" />
                <category android:name="android.intent.category.DEFAULT" />
                ... <!-- "data" entry shown later; not used in this example -->
            </intent-filter>
        </activity>
        ...
    </application>
</manifest>
```

## Java (LoanBundler.java)

```
public class LoanBundler {
    public static Bundle makeLoanInfoBundle(double loanAmount,
                                           double annualInterestRateInPercent,
                                           long loanPeriodInMonths,
                                           String currencySymbol) {
        Bundle loanInfo = new Bundle();
        loanInfo.putDouble("loanAmount", loanAmount);
        loanInfo.putDouble("annualInterestRateInPercent",
                           annualInterestRateInPercent);
        loanInfo.putLong("loanPeriodInMonths", loanPeriodInMonths);
        loanInfo.putString("currencySymbol", currencySymbol);
        return(loanInfo);
    }
}
```

## Java (IntentFilter1Activity.java)

```
public class IntentFilter1Activity extends Activity {
    ...

    public void showLoanPayments2(View clickedButton) {
        Intent activityIntent =
            new Intent(this, LoanCalculatorActivity.class);
        activityIntent.putExtra
            (LoanBundler.makeRandomizedLoanInfoBundle());
        startActivity(activityIntent);
    }
    ...
}
```

## Java (LoanBundler.java, Continued)

[illegible]



# Java (LoanCalculatorActivity.java)

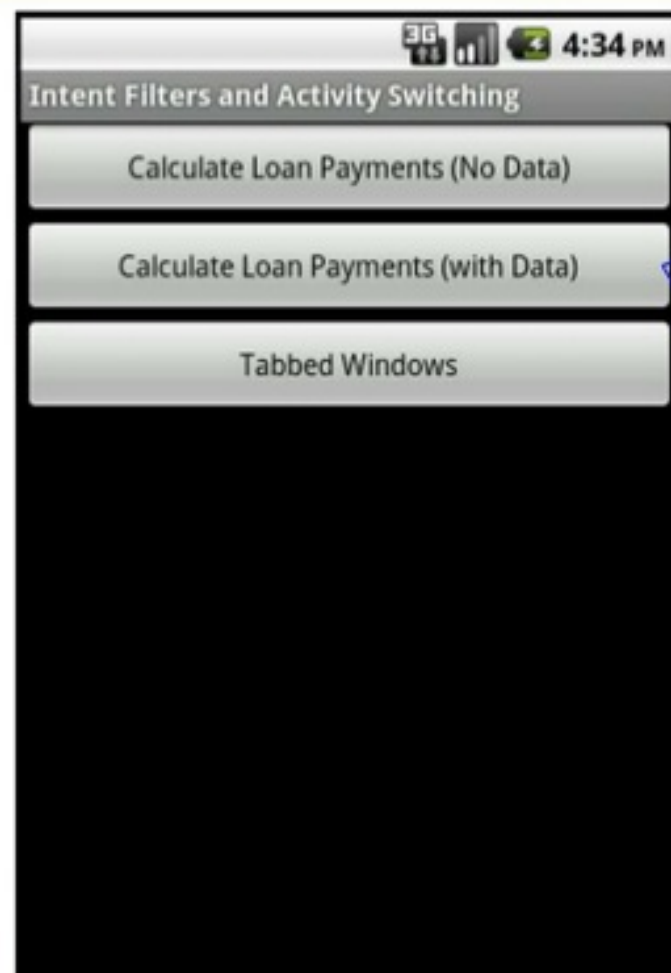
```
public class LoanCalculatorActivity extends Activity {
    private double mLoanAmount=100000,
                mAnnualInterestRateInPercent=5.0;
    private long mLoanPeriodInMonths=360; // 30 years
    private String mCurrencySymbol = "$";

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.loan_payments);
        setInputsFromExtras();
        setInputsFromUri();
        calculateAndSetOutputValues();
    }
}
```

# Java (LoanCalculatorActivity, Continued)

```
private void setInputsFromExtras() {
    Intent intent = getIntent();
    Bundle loanInfo = intent.getExtras();
    if (loanInfo != null) {
        double loanAmount = loanInfo.getDouble("loanAmount");
        double annualInterestRateInPercent =
            loanInfo.getDouble("annualInterestRateInPercent");
        long loanPeriodInMonths =
            loanInfo.getLong("loanPeriodInMonths");
        String currencySymbol =
            loanInfo.getString("currencySymbol");
        setInputs(loanAmount, annualInterestRateInPercent,
            loanPeriodInMonths, currencySymbol);
    }
}

private void setInputs(double loanAmount,
    double annualInterestRateInPercent,
    long loanPeriodInMonths,
    String currencySymbol) {
    if (loanAmount > 0) {
        mLoanAmount = loanAmount;
    }
    if (annualInterestRateInPercent > 0) {
        mAnnualInterestRateInPercent =
            annualInterestRateInPercent;
    }
    if (loanPeriodInMonths > 0) {
        mLoanPeriodInMonths = loanPeriodInMonths;
    }
    if (currencySymbol != null) {
        mCurrencySymbol = currencySymbol;
    }
}
```





Action	URI	Meaning
Intent.ACTION_CALL	tel: <i>phone_number</i>	Opens phone application and calls <i>phone_number</i> .
Intent.ACTION_DIAL	tel: <i>phone_number</i>	Opens phone application and dials (but doesn't call) <i>phone_number</i> .
Intent.ACTION_DIAL	voicemail:	Opens phone application and dials (but doesn't call) the voice mail number.
Intent.ACTION_VIEW	geo: <i>lat,long</i>	Opens the maps application centered on ( <i>lat, long</i> ).
Intent.ACTION_VIEW	geo:0,0?q= <i>address</i>	Opens the maps application centered on the specified address.
Intent.ACTION_VIEW	http:// <i>url</i> https:// <i>url</i>	Opens the browser application to the specified address.
Intent.ACTION_WEB_SEARCH	<i>plain_text</i>	Opens the browser application and uses Google search for given string.


Table adapted from Section 4.1.5 of *Android in Action* by Ahlson et al.

Android Tutorial: Intents, Intent Filters, and Invoking Activities -- Part III: Invoking Activities with Tabbed Windows

<http://www.slideshare.net/martyhall/android-tutorial-intents-intent-filters-and-invoking-activities-part-iii-invoking-activities-with-tabbed-windows>

Android Tutorial: Intents, Intent Filters, and Invoking Activities -- Part II: .....  
Marty Hall

<http://www.slideshare.net/martyhall/android-tutorial-intents-intent-filters-and-invoking-activities-part-ii-using-u>



© 2012 Marty Hall

# Intents, Intent Filters, and Invoking Activities: Part III: Using Tabs

Originals of Slides and Source Code for Examples:  
<http://www.coreservlets.com/android-tutorial/>

**Customized Java EE Training: <http://courses.coreservlets.com/>**  
Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

# Java (IntentFilter1Activity.java)

```
public class IntentFilter1Activity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }

    public void showLoanPayments1(View clickedButton) {
        Intent activityIntent =
            new Intent(this, LoanCalculatorActivity.class);
        startActivity(activityIntent);
    }

    ...
}
```

```
application android:icon="@drawable/icon"
    android:label="@string/app_name">
... <!-- Declaration for IntentFilter1Activity on previous slide -->
<activity android:name=".LoanCalculatorActivity"
    android:label="@string/loan_calculator_app_name">
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />
        <category android:name="android.intent.category.DEFAULT" />
        ... <!-- "data" entry shown later; not used in this example -->
    </intent-filter>
</activity>
```

one listed at the top in the "manifest" start tag).

Means that this Action displays data

Means that this

