



# Handling Screen Rotations and Other App Restarts

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

**Customized Java EE Training:** <http://courses.coreservlets.com/>  
Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.



**For live Android training, please see courses at <http://courses.coreservlets.com/>.**

**Taught by the author of *Core Servlets and JSP*, *More Servlets and JSP*, and this Android tutorial. Available at public venues, or customized versions can be held on-site at your organization.**



- Courses developed and taught by Marty Hall
    - Android development, JSF 2, servlets/JSP, Ajax, jQuery, Java 6 programming, custom mix of topics
    - Ajax courses can concentrate on 1 library (jQuery, Prototype/Scriptaculous, Ext-JS, Dojo, etc.) or survey several
  - Courses developed and taught by coreservlets.com experts (edited by Marty)
    - Spring, Hibernate/JPA, EJB3, GWT, RESTful and SOAP-based Web Services
- Contact [hall@coreservlets.com](mailto:hall@coreservlets.com) for details**

# Topics in This Section

- **Motivation**
- **Saving data**
- **The Bundle class**
- **Retrieving data**
- **Activity lifecycle**

5

© 2011 Marty Hall



## Overview

**Customized Java EE Training:** <http://courses.coreservlets.com/>  
Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

# Big Idea

- **Android can shut down and restart your app**
  - When you rotate the screen
  - When you change languages
  - When app is in background and Android is short on memory
  - When you hit the Back button
- **Problem**
  - You risk losing user changes
- **Solution**
  - Save data in a Bundle in onSaveInstanceState
  - Read data out of Bundle in onRestoreInstanceState (or in onCreate)
    - Does not handle Back button scenario.  
App restarts from scratch with no saved data in that case.

7

# Reminder: Preventing Screen Rotations

- **Issue**
  - Screen rotations usually require a new layout
  - They also cause the app to be shutdown and restarted
    - Handling this is the topic of this lecture
- **Problem**
  - What if you do not have landscape layout?
  - Or have not yet handled shutdown and restart?
- **Solution**
  - Put an entry in AndroidManifest.xml saying that app runs only in portrait mode (or only in landscape mode).

```
<activity android:name=".YourActivity"
        android:label="@string/app_name"
        android:screenOrientation="portrait">
```

8



# Steps

**Customized Java EE Training:** <http://courses.coreservlets.com/>  
Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

## Summary: Saving Data

- **Override onSaveInstanceState**
  - And pass the Bundle to the superclass method

```
protected void onSaveInstanceState(Bundle outState) {  
    super.onSaveInstanceState(outState);  
    outState.putBlah(someData);  
}
```
- **Called**
  - When user rotates screen
  - When user changes language
  - When app is hidden and Android needs the memory
- **Not called**
  - When user hits Back button
- **Note**
  - Superclass method automatically stores state of GUI widgets (EditText data, CheckBox state, etc.)

# Summary: Restoring Data

- **Override onRestoreInstanceState**

- Pass Bundle to superclass method
- Look for data by name, check for null, use the data

```
protected void onRestoreInstanceState(Bundle savedInstanceState) {  
    super.onRestoreInstanceState(savedInstanceState);  
    SomeType data = savedInstanceState.getBlah(key);  
    if (data != null) { doSomethingWith(data); }  
}
```

- **Called**

- Any time app is restarted after onSaveInstanceState

- **Note**

- The same Bundle is passed to onCreate.
- Superclass method automatically restores widget state

11

# The Bundle Class: Details

- **Putting data in a Bundle**

- putBoolean, putBooleanArray, putDouble, putDoubleArray, putString, putStringArray, 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.
- putSerializable, putParcelable
  - Lets you store custom objects. Note that ArrayList and most other builtin Java types are already Serializable

- **Retrieving data from a Bundle**

- getBoolean, getBooleanArray, getDouble, getDoubleArray, getString, getStringArray, etc.
  - No typecast required on retrieval. Numbers are 0 if no match.
- getSerializable, getParcelable
  - Typecast required on retrieval. Values are null if no match.

12



## Example

**Customized Java EE Training:** <http://courses.coreservlets.com/>  
Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

## Loan Payment Comparisons

- **Idea**
  - User can enter various interest rates to compare the resultant monthly and total payments for a \$100K, 30 year, fixed-rate loan
  - Different layouts for portrait and landscape modes
  - Uses the PaymentInfo class that was shown in the first Intents lecture
- **Approach**
  - Keep an ArrayList of PaymentInfo
    - Also put results in a TableView
  - In onSaveInstanceState, store the ArrayList
  - In onRestoreInstanceState, retrieve the ArrayList
    - Also put results in a TableView

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

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://..."
    android:orientation="vertical" ...>
    <TextView (introductory text) .../>
    <LinearLayout (horizontal) ...>
        <TextView (prompt) .../>
        <EditText (interest rate input) ...>...</EditText>
    </LinearLayout>
    <Button (when pressed, adds row to table) ... />
    <TableLayout ... >
        (Has heading and divider, but no body rows.)
    </TableLayout>
</LinearLayout>
```

15

## Layout File: Landscape Mode (res/layout-land/main.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://..." (horizontal) ...>
    <LinearLayout (vertical) ...>
        <TextView (introductory text) .../>
        <LinearLayout (horizontal) ...>
            <TextView (prompt) .../>
            <EditText (interest rate input) ...>...</EditText>
        </LinearLayout>
        <Button (when pressed, adds row to table) ... />
    </LinearLayout>
    <TableLayout ... >
        (Has heading and divider, but no body rows.)
    </TableLayout>
</LinearLayout>
```

16



# Values Files

- **res/values/strings.xml**
  - Defines intro text, prompt, table headings, and button label.
    - Used in both portrait and landscape mode.
- **res/values/dimens.xml**
  - Gives font sizes.
    - Used in portrait mode.
- **res/values-land/dimens.xml**
  - Gives font sizes.
    - Used in landscape mode.

17

# Core Java Code

```
public class RotationsActivity extends Activity {
    private final static double LOAN_AMOUNT = 100000;
    private final static long LOAN_PERIOD = 360; // In months
    private TableLayout mPaymentsTable;
    private EditText mInterestRateField;
    private float mTableSize;
    private ArrayList<PaymentInfo> mRowData =
        new ArrayList<PaymentInfo>();

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        mPaymentsTable = (TableLayout)findViewById(R.id.payment_table);
        mInterestRateField =
            (EditText)findViewById(R.id.interest_rate_field);
        Resources resources = getResources();
        mTableSize = resources.getDimension(R.dimen.table_body_size);
    }
}
```

Looks up the ID of the table so that it can later programmatically add rows to it. Note that the PaymentInfo data is declared as ArrayList instead of List. This is because we will later store it in a Bundle, and ArrayList is Serializable whereas List is not.

18



## Core Java Code (Continued)

```
// Attached to the Button via the android:onClick attribute

public void addComparisonRow(View clickedButton) {
    double interestRate = 5.0;
    try {
        interestRate = Double.parseDouble
            (mInterestRateField.getText().toString());
    } catch(Exception e) {}
    PaymentInfo info =
        new PaymentInfo(LOAN_AMOUNT, interestRate, LOAN_PERIOD);
    mRowData.add(info);
    addRow(info);
    mInterestRateField.setText("");
}
```

Reads the textfield value, converts to a number (5.0 default if field is empty or value is illegal), computes payment info, stores payment info in an ArrayList, then calls code to add data to bottom of table.

19

## Core Java Code (Continued)

```
private void addRow(PaymentInfo info) {
    TableRow row = new TableRow(this);
    row.addView(makeColumn
        (info.getFormattedAnnualInterestRateInPercent()));
    row.addView(makeColumn(info.getFormattedMonthlyPayment()));
    row.addView(makeColumn(info.getFormattedTotalPayments()));
    mPaymentsTable.addView(row);
}

private TextView makeColumn(String text) {
    TextView col = new TextView(this);
    col.setGravity(Gravity.RIGHT);
    col.setTextSize(mTableSize);
    col.setText(text);
    return(col);
}
```

Creates a TableRow and inserts it into bottom of TableView.

20

# Results (Without Saving State)

User enters some interest rates  
to get this result

Rotations Example

Compare interest rates: enter an interest rate in percent, then press button to see what the monthly payment and total payments would be. All examples assume a \$100,000 principal with a 30-year fixed-rate loan.

Interest Rate:

Add Comparison

| Interest Rate | Monthly Payment | Total Payments |
|---------------|-----------------|----------------|
| 2.50%         | \$395.12        | \$142,243.52   |
| 5.00%         | \$536.82        | \$193,255.78   |
| 7.50%         | \$699.21        | \$251,717.22   |
| 10.00%        | \$877.57        | \$315,925.77   |

After entering the data as on the left,  
user rotates the screen. Data is lost!

Rotations Example

Compare interest rates: enter an interest rate in percent, then press button to see what the monthly payment and total payments would be. All examples assume a \$100,000 principal with a 30-year fixed-rate loan.

Interest Rate:

Add Comparison

21

# Java Code: Saving State

```
@Override
protected void onSaveInstanceState(Bundle outState) {
    super.onSaveInstanceState(outState);
    outState.putSerializable("rowData", mRowData);
}
```

ArrayList is already Serializable. But, I had to make PaymentInfo (the data inside) Serializable as well.

22

# Java Code: Saving State

```
@Override
protected void onRestoreInstanceState(Bundle savedInstanceState) {
    super.onRestoreInstanceState(savedInstanceState);
    @SuppressWarnings("unchecked")
    ArrayList<PaymentInfo> rowData =
        (ArrayList<PaymentInfo>)
            savedInstanceState.getSerializable("rowData");
    if (rowData != null) {
        mRowData = rowData;
        for(PaymentInfo info: rowData) {
            addRow(info);
        }
    }
}
```

Since generics are implemented by erasure (all work done at compile time, no work done at run time), casting to `ArrayList<Blah>` will give a warning that Java cannot verify that the list contains only `Blah`s. So, suppress that warning.

23

# Java Code: PaymentInfo

```
public class PaymentInfo implements Serializable {
    private final double mLoanAmount,
                        mAnnualInterestRateInPercent,
                        mMonthlyPayment, mTotalPayments;

    private final long mLoanPeriodInMonths;
    private final String mCurrencySymbol;

    // Calculates and stores monthly payment and total payments
    // Has methods to print out values with nice formatting

}
```

Custom data stored in a `Bundle` must be `Serializable` or `Parcelable`.

24

# Results (Saving State)

User enters some interest rates  
to get this result

Rotations Example

Compare interest rates: enter an interest rate in percent, then press button to see what the monthly payment and total payments would be. All examples assume a \$100,000 principal with a 30-year fixed-rate loan.

Interest Rate:

Add Comparison

| Interest Rate | Monthly Payment | Total Payments |
|---------------|-----------------|----------------|
| 2.50%         | \$395.12        | \$142,243.52   |
| 5.00%         | \$536.82        | \$193,255.78   |
| 7.50%         | \$699.21        | \$251,717.22   |
| 10.00%        | \$877.57        | \$315,925.77   |

After entering the data as on the left, user  
rotates the screen. Data is preserved!

Rotations Example

Compare interest rates: enter an interest rate in percent, then press button to see what the monthly payment and total payments would be. All examples assume a \$100,000 principal with a 30-year fixed-rate loan.

Interest Rate:

Add Comparison

| Interest Rate | Monthly Payment | Total Payments |
|---------------|-----------------|----------------|
| 2.50%         | \$395.12        | \$142,243.52   |
| 5.00%         | \$536.82        | \$193,255.78   |
| 7.50%         | \$699.21        | \$251,717.22   |
| 10.00%        | \$877.57        | \$315,925.77   |

25

© 2011 Marty Hall



## More Lifecycle Events

Customized Java EE Training: <http://courses.coreservlets.com/>

Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

# Overview

- **Idea**

- You can respond to situations other than app being killed and app being restarted

- **Situations**

- Another Activity partially obscures Activity
  - Use onPause
- Activity is no longer visible at all
  - Use onStop
- A stopped Activity is restarted
  - Use onStart (every time including first) or onRestart (every time except first)
- An Activity is killed for good (no saved state Bundle)
  - Use onDestroy

27

# Lifecycle Summary

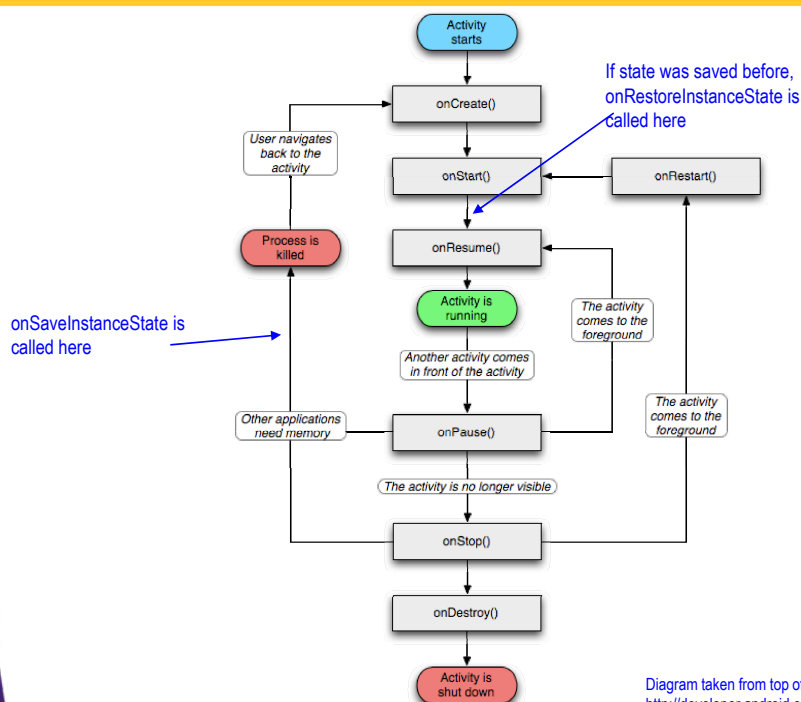


Diagram taken from top of  
<http://developer.android.com/reference/android/app/Activity.html>

28



# Wrap-Up

**Customized Java EE Training:** <http://courses.coreservlets.com/>  
Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

## More Reading

- **JavaDoc: Activity**
  - <http://developer.android.com/reference/android/app/Activity.html>
    - Introductory parts give lots of details
- **Chapters**
  - Handling Activity Lifecycle Events     *and*
  - Handling Rotation
    - From *The Busy Coder's Guide to Android Development* by Mark Murphy.
      - <http://commonsware.com/Android/>

# Summary

- **Save data in onSaveInstanceState**
  - Can put individual pieces of data in the Bundle, or can add a composite data structure.
  - Custom classes must implement Serializable or Parcelable
- **Load data in onRestoreInstanceState or in onCreate**
  - Look in Bundle for property of given name
  - For Object types, check for null
  - For number types, check for 0

31

© 2011 Marty Hall



## Questions?

**Customized Java EE Training:** <http://courses.coreservlets.com/>  
Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.