14

Android Persistency: Preferences

Victor Matos
Cleveland State University

Notes are based on:

The Busy Coder's Guide to Android Development
by Mark L. Murphy
Copyright © 2008-2009 CommonsWare, LLC.
ISBN: 978-0-9816780-0-9
&
Android Developers
http://developer.android.com/index.html

Android Data Storage

Android provides several options for you to save persistent application data. The solution you choose depends on your specific needs: private/public, small/large datasets.

Your data storage options are the following:

Shared Preferences	Store private primitive data in key-value pairs.
Internal Storage	Store private data on the device memory.
External Storage	Store public data on the shared external storage.
SQLite Databases	Store structured data in a private database.
Network Connection	Store data on the web with your own network server.
Content Provider	Shared repository globally shared by all apps.

http://developer.android.com/guide/topics/data/data-storage.html

Android Data Storage

Android uses a particular data sharing scheme:

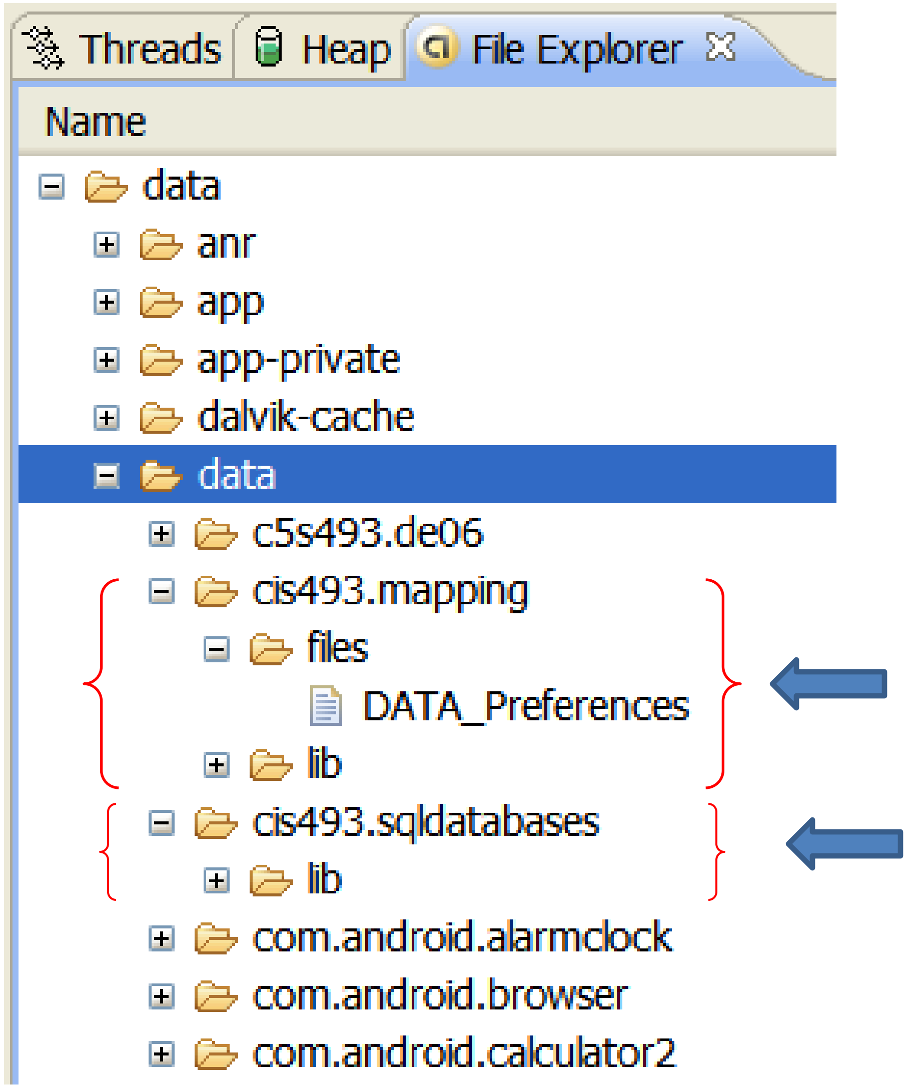
*On Android, all application data held in the device's private memory area is **private** to that application*

Note:
Private memory usually small and different from external storage (SDcards).

http://developer.android.com/guide/topics/data/data-storage.html

Android Data Storage

On Android, all application data (including files) are private to that application.



Threads | Heap | File Explorer

```
Name
data
   anr
   app
   app-private
   dalvik-cache
   data
      c5s493.de06
      cis493.mapping
         files
            DATA_Preferences
         lib
      cis493.sqldatabases
         lib
      com.android.alarmclock
      com.android.browser
      com.android.calculator2
```

http://developer.android.com/guide/topics/data/data-storage.html

Android Data Storage

Content Providers provide a data-layer for non-Sql developers (*to be discussed later*)

Android uses content providers for global data objects, such as

image,
audio,
video files and
personal contact information.

Preferences

Preferences is an Android *lightweight* mechanism to store and retrieve *<key-value>* pairs of primitive data types (also called *Maps*, and *Associative Arrays.*

PREFERENCES are typically used to keep state information and shared data among several activities of an application.

In each entry of the form *<key-value>* the *key* is a string and the *value* must be a primitive data type.

Preferences are similar to Bundles however they are persistent while Bundles are not.

Preferences

Using Preferences API calls

You have three API choices to pick a Preference:

1. **getPreferences()** from within your Activity, to access activity specific preferences

2. **getSharedPreferences()** from within your Activity to access application-level preferences

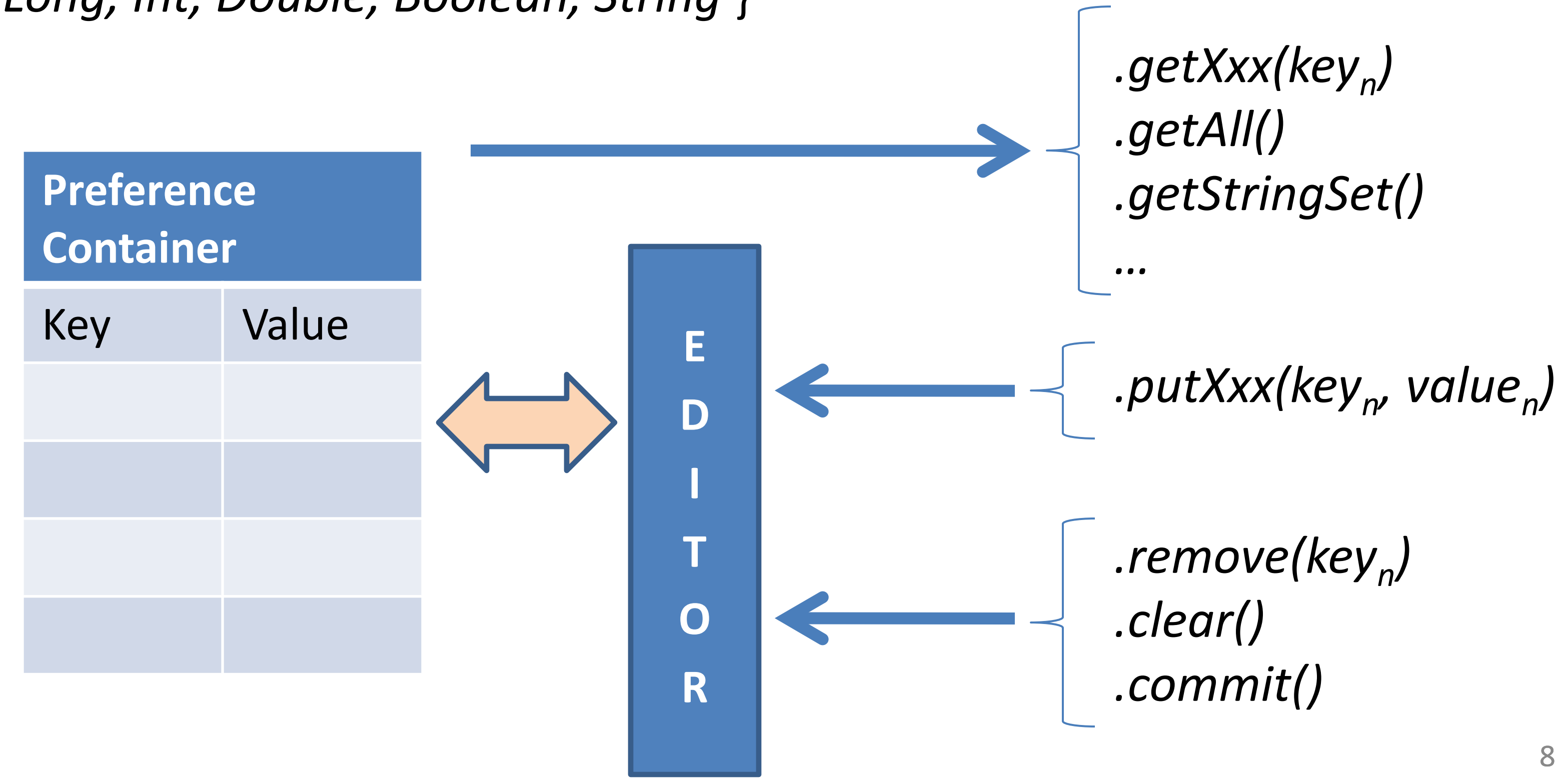
3. **getDefaultSharedPreferences()**, on *PreferencesManager*, to get the shared preferences that work in concert with Android's overall preference framework

Preferences

Using Preferences API calls

All of the *getXXX* Preference methods return a Preference object whose contents can be manipulated by an *editor* that allows *putXxx… and getXxx…* commands to place data in and out of the Preference container.

Xxx = { *Long, Int, Double, Boolean, String* }



Preferences

Example1

1. In this example a persistent *SharedPreferences* object is created at the end of an activity lifecycle. It contains some *formatting* specifications made by the user to define aspects of the graphical interface.

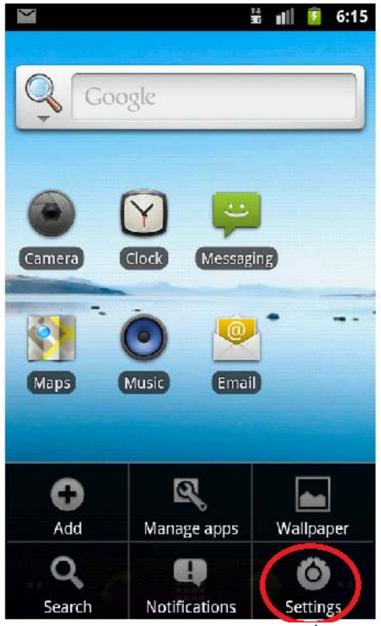
2. When re-executed, it finds the saved *Preference* and uses its persistent data to reproduce the UI according to the specifications previously given by the user.

Warning
Make sure you test from a ‘fresh’ configuration. If necessary use DDMS and *delete* existing Preferences held in the application’s name-space.

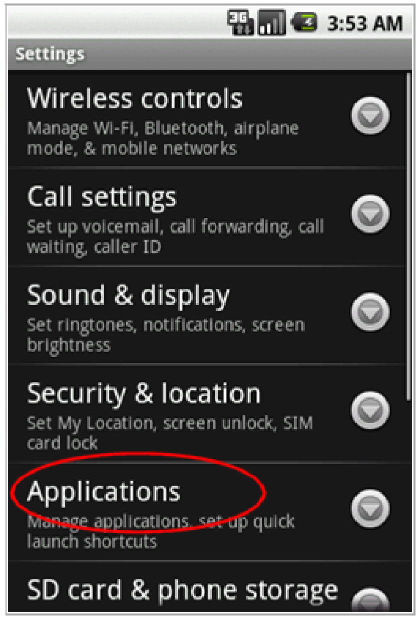
Preferences

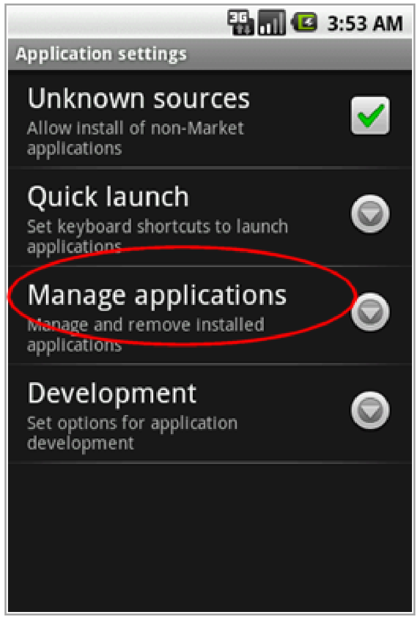
Example1

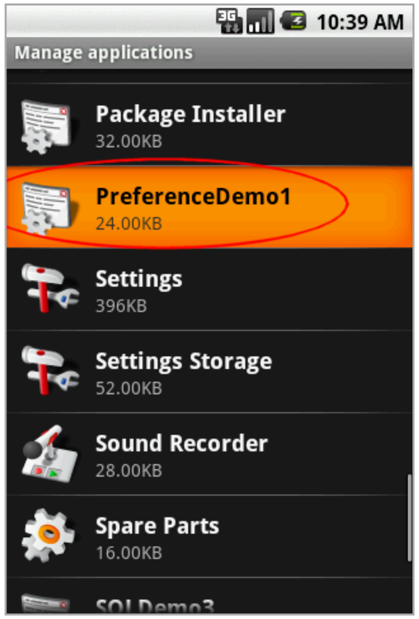
Warning
Make sure you test from a ‘fresh’ configuration. Next images illustrate the process of removing existing traces of an application from the phone’s system area using device’s Application Manager



Menu
Button


3:53 AM
Settings
Wireless controls
Manage Wi-Fi, Bluetooth, airplane mode, & mobile networks
Call settings
Set up voicemail, call forwarding, call waiting, caller ID
Sound & display
Set ringtones, notifications, screen brightness
Security & location
Set My Location, screen unlock, SIM card lock
Applications
Manage applications, set up quick launch shortcuts
SD card & phone storage

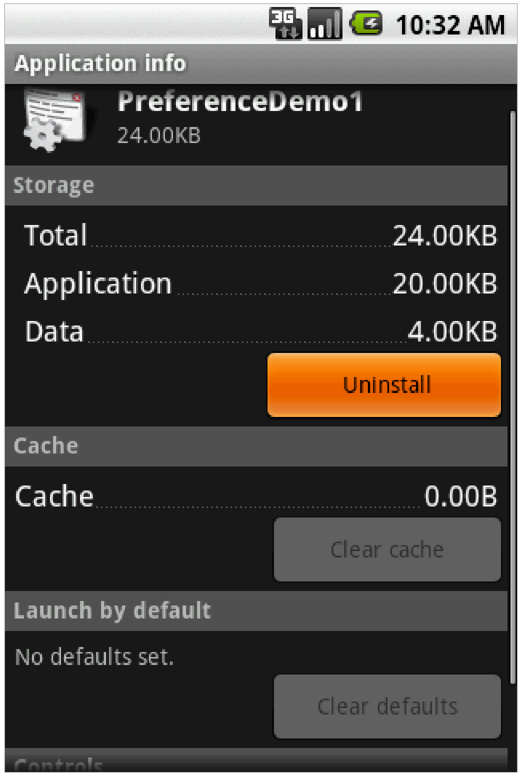

3:53 AM
Application settings
Unknown sources ✓
Allow install of non-Market applications
Quick launch
Set keyboard shortcuts to launch applications
Manage applications
Manage and remove installed applications
Development
Set options for application development


10:39 AM
Manage applications
Package Installer
32.00KB
PreferenceDemo1
24.00KB
Settings
396KB
Settings Storage
52.00KB
Sound Recorder
28.00KB
Spare Parts
16.00KB
SQLDemo3

Preferences

Example1. *cont.*

Warning
Make sure you test from a ‘fresh’ configuration. Next images illustrate the process of removing existing traces of an application from the phone’s system area using device’s Application Manager


10:32 AM

Application info

PreferenceDemo1
24.00KB

Storage

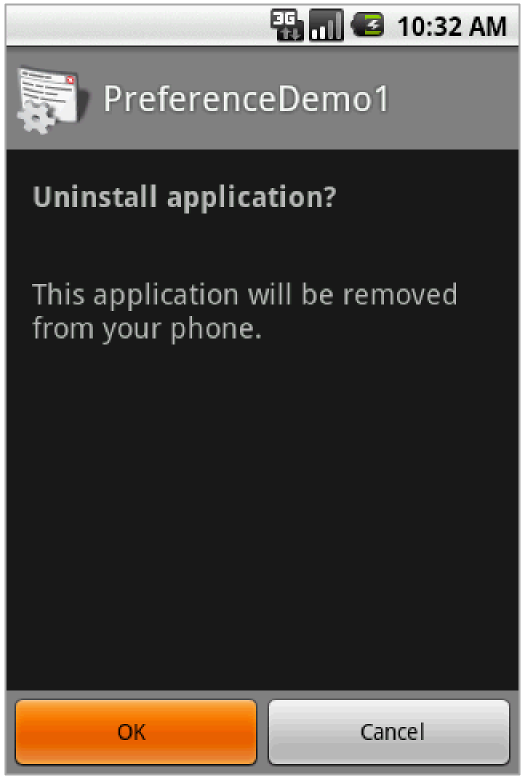
Total 24.00KB
Application 20.00KB
Data 4.00KB

Cache

Cache 0.00B

Launch by default

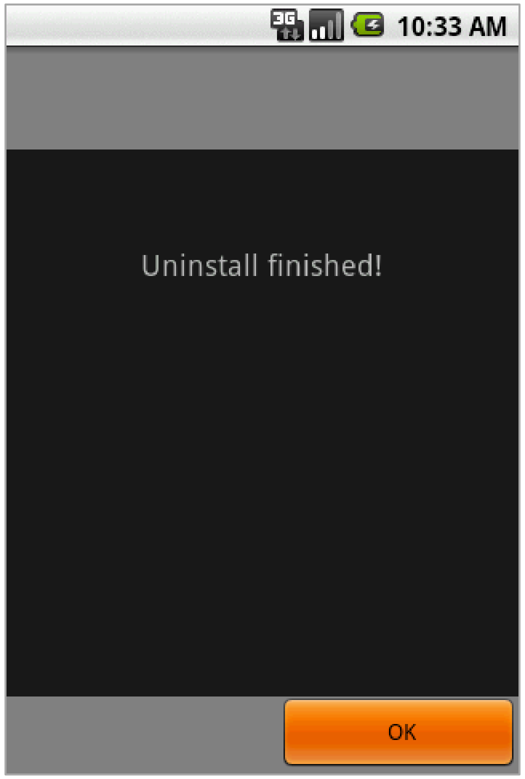
No defaults set.


10:32 AM

PreferenceDemo1

Uninstall application?

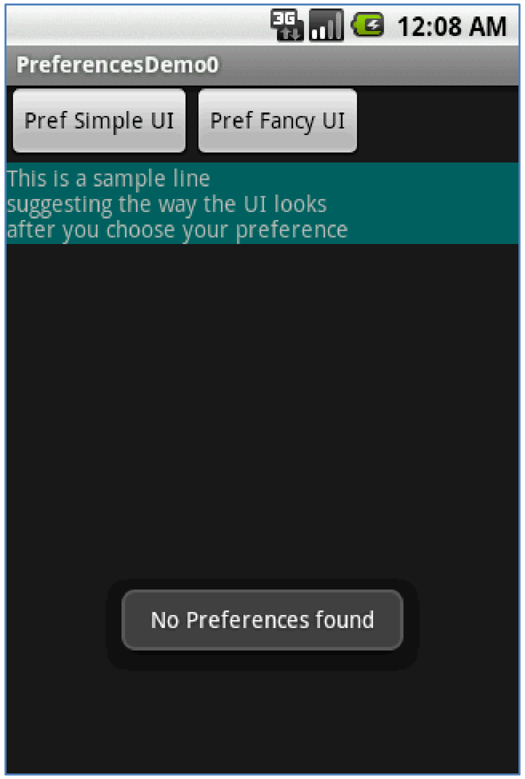
This application will be removed from your phone.


10:33 AM

Uninstall finished!

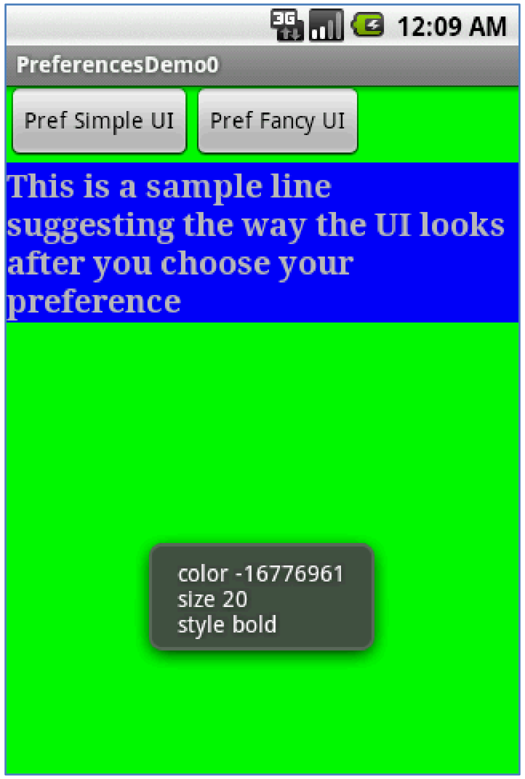
Preferences

Example1: Saving/Retrieving a SharedPreference Object holding UI user choices.



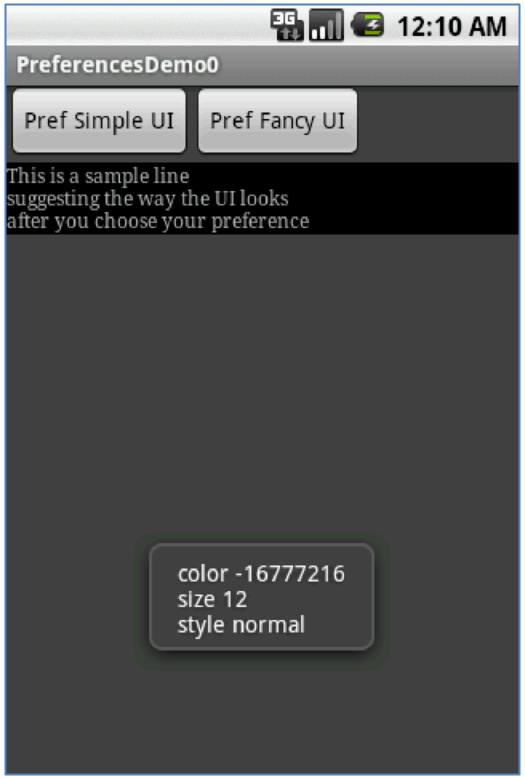
This is a sample line
suggesting the way the UI looks
after you choose your preference

No Preferences found



This is a sample line
suggesting the way the UI looks
after you choose your
preference

color -16776961
size 20
style bold



This is a sample line
suggesting the way the UI looks
after you choose your preference

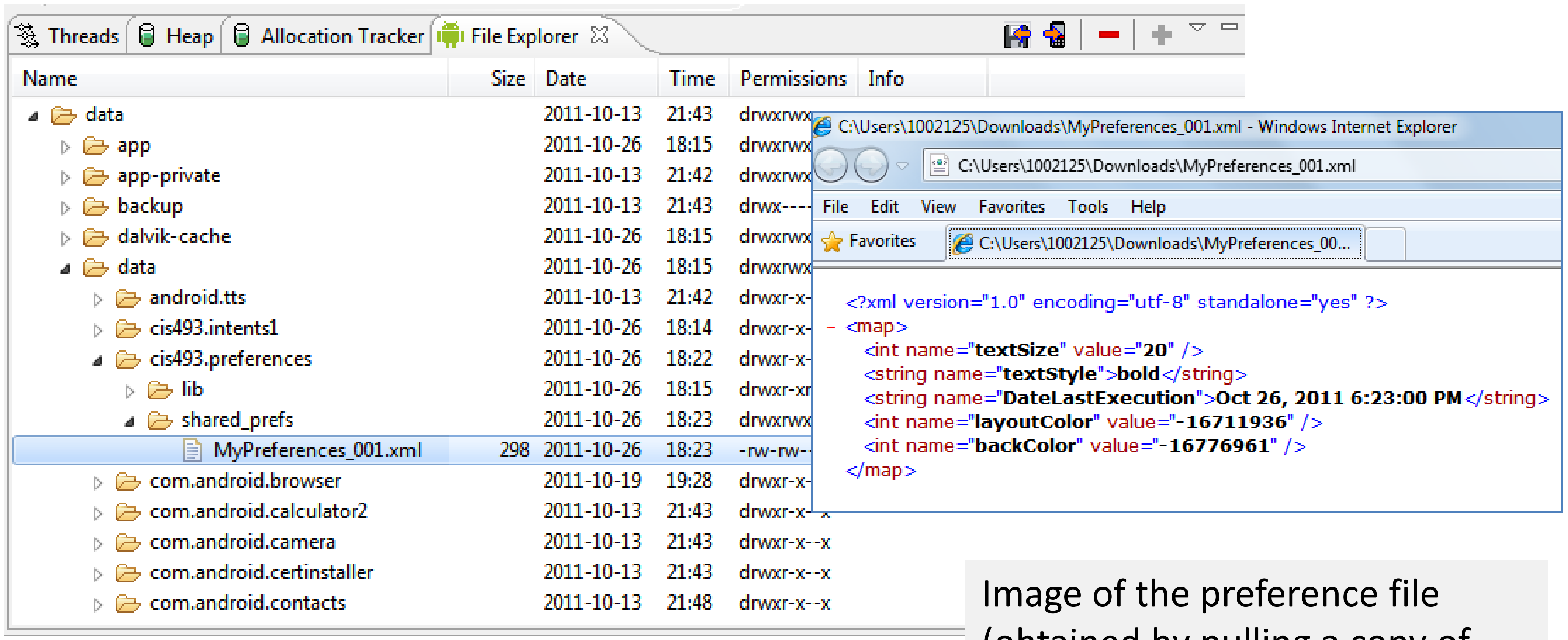
color -16777216
size 12
style normal

Initial UI with no choices made/save yet.

Images of the choices made by the user regarding the looks of the UI. The 'green screen' corresponds to the fancy layout, the 'grey screen' is the simple choice. Data is saved into the SharedPreference object: *myPreferences_001*.

Preferences

Example1: Saving/Retrieving a SharedPreference Object


Threads
Heap
Allocation Tracker
File Explorer
C:\Users\1002125\Downloads\MyPreferences_001.xml - Windows Internet Explorer
C:\Users\1002125\Downloads\MyPreferences_001.xml
File
Edit
View
Favorites
Tools
Help
Favorites
C:\Users\1002125\Downloads\MyPreferences_00...

Name	Size	Date	Time	Permissions	Info
data		2011-10-13	21:43	drwxrwx	
app		2011-10-26	18:15	drwxrwx	
app-private		2011-10-13	21:42	drwxrwx	
backup		2011-10-13	21:43	drwx----	
dalvik-cache		2011-10-26	18:15	drwxrwx	
data		2011-10-26	18:15	drwxrwx	
android.tts		2011-10-13	21:42	drwxr-x-	
cis493.intents1		2011-10-26	18:14	drwxr-x-	
cis493.preferences		2011-10-26	18:22	drwxr-x-	
lib		2011-10-26	18:15	drwxr-xr	
shared_prefs		2011-10-26	18:23	drwxrwx	
MyPreferences_001.xml	298	2011-10-26	18:23	-rw-rw-	
com.android.browser		2011-10-19	19:28	drwxr-x-	
com.android.calculator2		2011-10-13	21:43	drwxr-x--x	
com.android.camera		2011-10-13	21:43	drwxr-x--x	
com.android.certinstaller		2011-10-13	21:43	drwxr-x--x	
com.android.contacts		2011-10-13	21:48	drwxr-x--x	

```
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
- <map>
    <int name="textSize" value="20" />
    <string name="textStyle">bold</string>
    <string name="DateLastExecution">Oct 26, 2011 6:23:00 PM</string>
    <int name="layoutColor" value="-16711936" />
    <int name="backColor" value="-16776961" />
  </map>
```

Image of the preference file (obtained by pulling a copy of the file out of the device).

Using DDMS to explore the Device’s memory map. Observe the choices made by the user are saved in the *data/data/Shared_prefs/* folder as an XML file.

Preferences

Example1: Saving/Retrieving a SharedPreference Object

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
android:id="@+id/linLayout1Vertical"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:orientation="vertical"
xmlns:android="http://schemas.android.com/apk/res/android" >

<LinearLayout
android:id="@+id/linLayout2Horizontal"
android:layout_width="fill_parent"
android:layout_height="wrap_content“ >
      <Button
            android:id="@+id/btnPrefSimple"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Pref Simple UI“  />
      <Button
            android:id="@+id/btnPrefFancy"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Pref Fancy UI“ />
</LinearLayout>

<TextView
      android:id="@+id/txtCaption1"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:background="#ff006666"
      android:text="This is some sample text “ />

</LinearLayout>
```

Preferences

Example1: Saving/Retrieving a SharedPreference Object

```
package cis493.preferences;
import ...

public class PreferenceDemo0 extends Activity implements OnClickListener {
     Button btnSimplePref;
     Button btnFancyPref;
     TextView txtCaption1;
     Boolean fancyPrefChosen = false;
     View    myLayout1Vertical;

     final int mode = Activity.MODE_PRIVATE;
     final String MYPREFS = "MyPreferences_001";

     // create a reference to the shared preferences object
     SharedPreferences mySharedPreferences;

     // obtain an editor to add data to my SharedPreferences object
     SharedPreferences.Editor myEditor;
```

File creation modes:
MODE_APPEND
MODE_

Preferences

Example1: Saving/Retrieving a SharedPreference Object

```
@Override
public void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.main);
     myLayout1Vertical = (View)findViewById(R.id.linLayout1Vertical);
     txtCaption1 = (TextView) findViewById(R.id.txtCaption1);
     txtCaption1.setText("This is a sample line \n“
                        + "suggesting the way the UI looks \n"
                        + "after you choose your preference");
     // create a reference & editor for the shared preferences object
     mySharedPreferences = getSharedPreferences(MYPREFS, 0);
     myEditor = mySharedPreferences.edit();
     // has a Preferences file been already created?
     if (mySharedPreferences != null
        && mySharedPreferences.contains("backColor")) {
        // object and key found, show all saved values
            applySavedPreferences();
        } else {
            Toast.makeText(getApplicationContext(),
                          "No Preferences found", 1).show();
     }
     btnSimplePref = (Button) findViewById(R.id.btnPrefSimple);
     btnSimplePref.setOnClickListener(this);
     btnFancyPref = (Button) findViewById(R.id.btnPrefFancy);
     btnFancyPref.setOnClickListener(this);
}// onCreate
```

Preferences

Example1: Saving/Retrieving a SharedPreference Object

```
@Override
public void onClick(View v) {
    // clear all previous selections
    myEditor.clear();

    // what button has been clicked?
    if (v.getId() == btnSimplePref.getId()) {
        myEditor.putInt("backColor", Color.BLACK);// black background
        myEditor.putInt("textSize", 12);   // humble small font
    } else { // case btnFancyPref
        myEditor.putInt("backColor", Color.BLUE); // fancy blue
        myEditor.putInt("textSize", 20);   // fancy big
        myEditor.putString("textStyle", "bold");  // fancy bold
        myEditor.putInt("layoutColor", Color.GREEN);//fancy green
    }
    myEditor.commit();
    applySavedPreferences();
}
```

Preferences

Example1: Saving/Retrieving a SharedPreference Object

```
@Override
protected void onPause() {
    // warning: activity is on its last state of visibility!.
    // It's on the edge of being killed! Better save all current
    // state data into Preference object (be quick!)
    myEditor.putString("DateLastExecution", new Date().toLocaleString());
    myEditor.commit();
    super.onPause();
}
```

Preferences

Example1: Saving/Retrieving a SharedPreference Object

```
public void applySavedPreferences() {
     // extract the <key/value> pairs, use default param for missing data
     int backColor = mySharedPreferences.getInt("backColor",Color.BLACK);
     int textSize = mySharedPreferences.getInt("textSize", 12);
     String textStyle = mySharedPreferences.getString("textStyle", "normal");
     int layoutColor = mySharedPreferences.getInt("layoutColor",Color.DKGRAY);
     String msg = "color " + backColor + "\n"
                + "size "   + textSize    + "\n"
                + "style "  + textStyle;
     Toast.makeText(getApplicationContext(), msg, 1).show();

     txtCaption1.setBackgroundColor(backColor);
     txtCaption1.setTextSize(textSize);
     if (textStyle.compareTo("normal")==0){
        txtCaption1.setTypeface(Typeface.SERIF,Typeface.NORMAL);
     }
     else {
        txtCaption1.setTypeface(Typeface.SERIF,Typeface.BOLD);
     }
     myLayout1Vertical.setBackgroundColor(layoutColor);
}// applySavedPreferences

}//class
```

Preferences

Example2

1. In this example a persistent *SharedPreferences* object is created at the end of an activity lifecycle. It contains data (name, phone, credit, etc. of a fictional customer)

2. The process is interrupted using the “*Back Button*” and re-executed later.

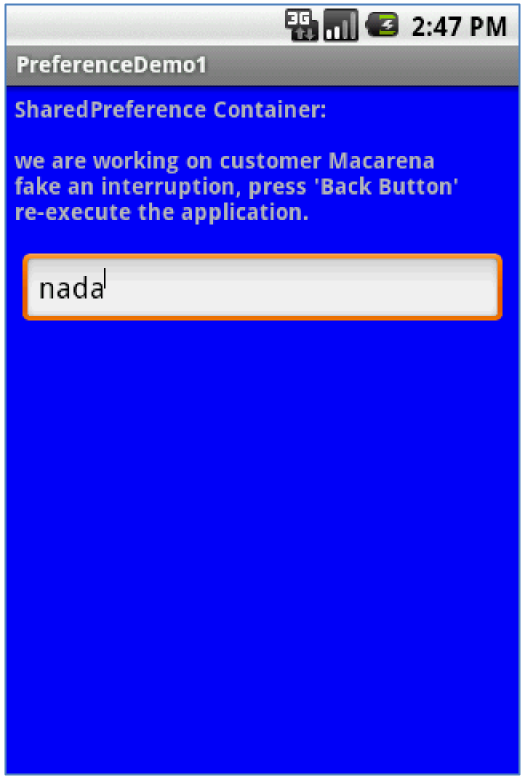
3. Just before been killed, the state of the running application is saved in the designated *Preference* object.

4. When re-executed, it finds the saved *Preference* and uses its persistent data.

Warning
Make sure you test from a ‘fresh’ configuration. If necessary use DDMS and *delete* existing Preferences held in the application’s name-space.

Preferences

Example2: Saving/Retrieving a SharedPreference Object containing 'business' data.



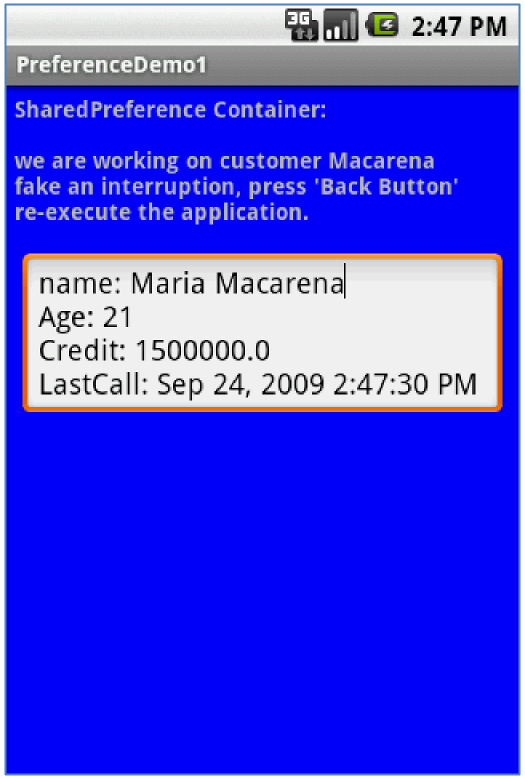
2:47 PM

PreferenceDemo1

SharedPreference Container:

we are working on customer Macarena
fake an interruption, press 'Back Button'
re-execute the application.

nada



2:47 PM

PreferenceDemo1

SharedPreference Container:

we are working on customer Macarena
fake an interruption, press 'Back Button'
re-execute the application.

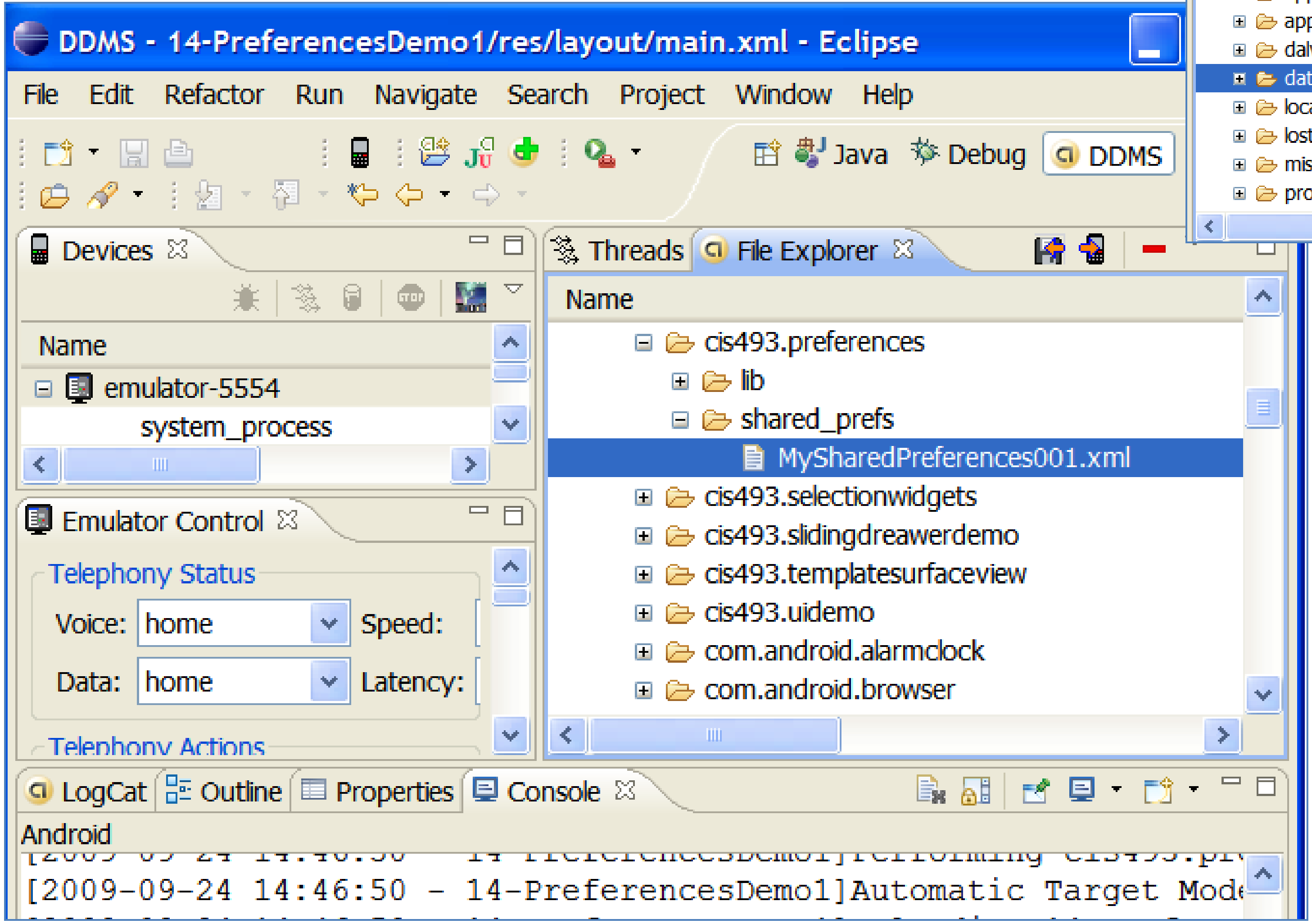
name: Maria Macarena
Age: 21
Credit: 1500000.0
LastCall: Sep 24, 2009 2:47:30 PM

Image of the data held in the SharedPreferences object displayed the first time the Activity **Preferences1** is executed.

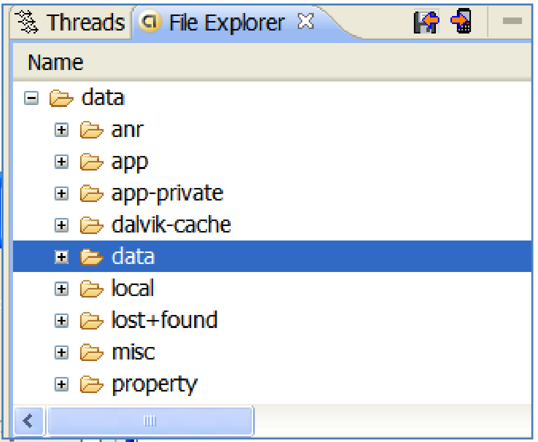
Image of the saved Preference data displayed the second time the Activity **Preferences1** is executed.

Preferences

Example2: Saving/Retrieving a SharedPreference Object



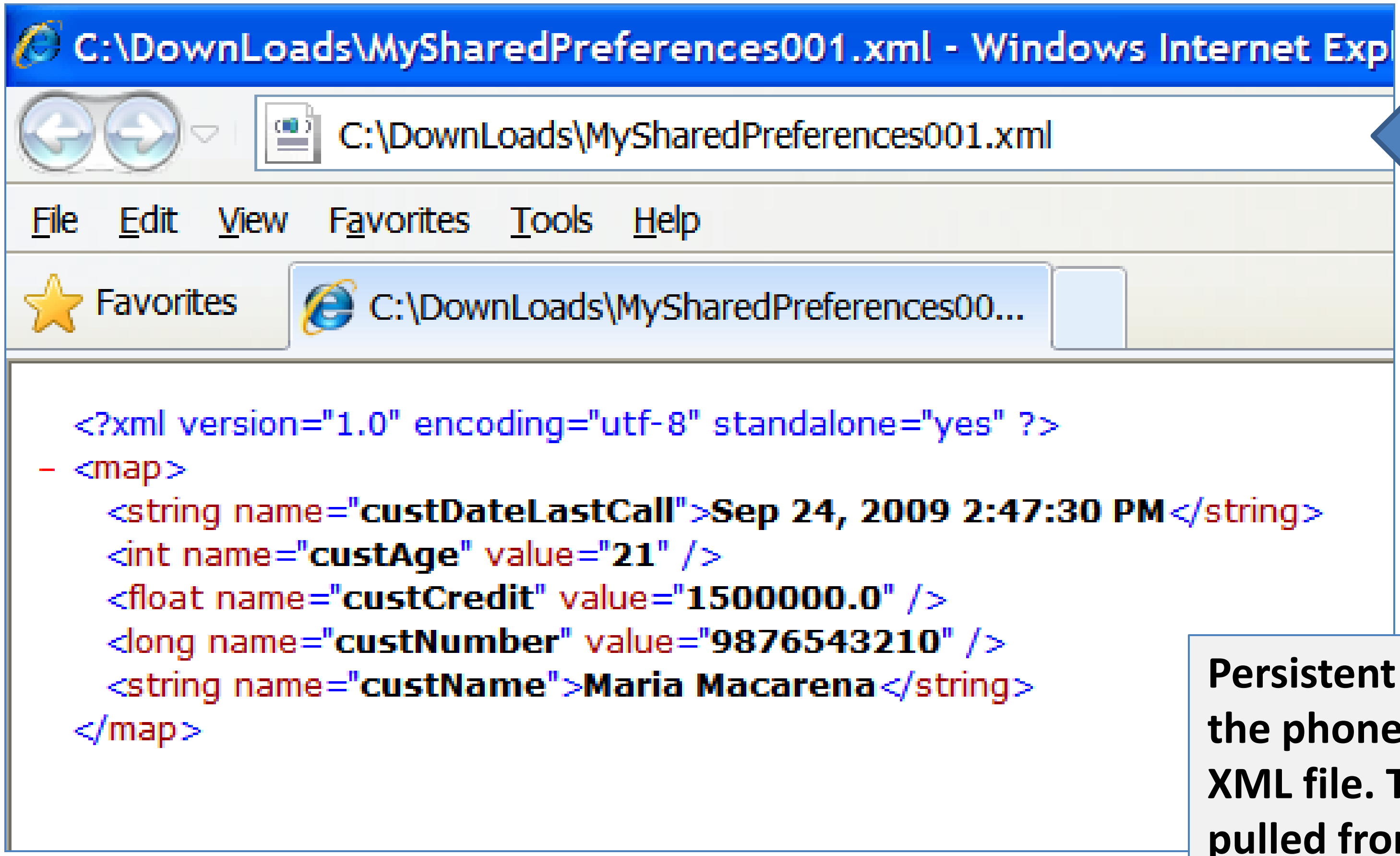
```
[2009-09-24 14:46:50 - 14-PreferencesDemo1]Automatic Target Mode
```



Use DDMS to see persistent data set

Preferences

Example2: Saving/Retrieving a SharedPreference Object


C:\DownLoads\MySharedPreferences001.xml - Windows Internet Expl
C:\DownLoads\MySharedPreferences001.xml
File Edit View Favorites Tools Help
Favorites
C:\DownLoads\MySharedPreferences00...

```
  <?xml version="1.0" encoding="utf-8" standalone="yes" ?>
- <map>
    <string name="custDateLastCall">Sep 24, 2009 2:47:30 PM</string>
    <int name="custAge" value="21" />
    <float name="custCredit" value="1500000.0" />
    <long name="custNumber" value="9876543210" />
    <string name="custName">Maria Macarena</string>
  </map>
```

Persistent data is saved in the phone's memory as an XML file. This image was pulled from the device using DDMS.

Preferences

Example2: Saving/Retrieving a SharedPreference Object

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
android:id="@+id/linLayou1"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:background="#ff0000ff"
android:orientation="vertical"
xmlns:android="http://schemas.android.com/apk/res/android"
>
<TextView
    android:id="@+id/captionBox"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="SharedPreferences Container: Customer Data"
    android:layout_margin="5px" android:textStyle="bold">
</TextView>
<EditText
    android:id="@+id/txtPref"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10px"
    >
</EditText>
</LinearLayout>
```

Preferences

Example2: Saving/Retrieving a SharedPreference Object

```
package cis493.preferences;

import java.util.Date;

import android.app.Activity;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.widget.*;

public class Preference1 extends Activity {
    public static final String MYPREFS = "MySharedPreferences001";
    //this data values describe a typical customer record
    String custName = "n.a.";
    int    custAge = 0;
    float  custCredit = 0;
    long   custNumber = 0;
    String custDateLastCall;

    TextView captionBox;
    EditText txtPref;
    final int mode = Activity.MODE_PRIVATE;
```

Preferences

Example2: Saving/Retrieving a SharedPreference Object

```
@Override
  public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.main);
      txtPref = (EditText)findViewById(R.id.txtPref);
      captionBox = (TextView) findViewById(R.id.captionBox);
      captionBox.setText("SharedPreference Container: \n\n"+
              "we are working on customer Macarena \n" +
              "fake an interruption, press 'Back Button' \n" +
              "re-execute the application.");

      //create a reference to the shared preferences object
      int mode = Activity.MODE_PRIVATE;
      SharedPreferences mySharedPreferences = getSharedPreferences(MYPREFS, mode);
      //is there an existing Preferences from previous executions of this app?
      if (mySharedPreferences != null  &&
          mySharedPreferences.contains("custName")) {
          //object and key found, show all saved values
          showSavedPreferences();
      }
      else
      {
          txtPref.setText("nada");
      }
  }//onCreate
```

Preferences

Example2: Saving/Retrieving a SharedPreference Object

```
@Override
protected void onPause() {
   //warning: activity is on last state of visibility! We are on the
   //edge of been killed! Better save current state in Preference object
   savePreferences();
   super.onPause();
}

protected void savePreferences(){
    //create the shared preferences object
    SharedPreferences mySharedPreferences =
                           getSharedPreferences(MYPREFS, mode);

    //obtain an editor to add data to (my)SharedPreferences object
    SharedPreferences.Editor myEditor = mySharedPreferences.edit();

    //put some <key/value> data in the preferences object
    myEditor.putString("custName", "Maria Macarena");
    myEditor.putInt("custAge", 21);
    myEditor.putFloat("custCredit", 1500000.00F);
    myEditor.putLong("custNumber", 9876543210L);
    myEditor.putString("custDateLastCall", new Date().toLocaleString());
    myEditor.commit();
}//savePreferences
```

Preferences

Example2: Saving/Retrieving a SharedPreference Object

```
  public void showSavedPreferences() {
        //retrieve the SharedPreferences object

        SharedPreferences mySharedPreferences =
                                   getSharedPreferences(MYPREFS, mode);

        //extract the <key/value> pairs, use default param for missing data
        custName = mySharedPreferences.getString("custName", "defNameValue");
        custAge = mySharedPreferences.getInt("custAge", 18);
        custCredit = mySharedPreferences.getFloat("custCredit", 1000.00F);
        custNumber = mySharedPreferences.getLong("custNumber", 1L);
        custDateLastCall = mySharedPreferences.getString("custDateLastCall",
                                                   new Date().toLocaleString());
        //show saved data on screen
        String msg = "name: " + custName + "\nAge: " + custAge +
                     "\nCredit: " + custCredit +
                     "\nLastCall: " + custDateLastCall;
        txtPref.setText(msg);
    }//loadPreferences

}//Preferences1
```

Preferences

Questions ?