Slide 9.16: Dynamic checkboxes: Java source code (cont.)
Slide 11.1: Android server connection
Home

PRINT VERSION

Dynamic Checkboxes: Java Code (Cont.)

Line 43: final LinkedHashMap<String, String>
Line 44: alphabet = new LinkedHashMap<String, String>( );

- HashMap is hashtable based implementation of the Map interface. The HashMap class is roughly equivalent to Hashtable, except that it is unsynchronized and permits nulls.

- LinkedHashMap differs from HashMap in that it maintains a doubly-linked list running through all of its entries. This linked list defines the iteration ordering, which is normally the order in which keys were inserted into the map (insertion-order).

Line 45: alphabet.put("260", ".NET");

The method put associates the specified value with the specified key in this map. If the map previously contained a mapping for the key, the old value is replaced.

Line 48: Set<?> set = alphabet.entrySet( );

Returns a Set view of the mappings contained in this map.

Line 50: Iterator<?> i = set.iterator( );

Returns an iterator over the elements in this set.

Line 52: while (i.hasNext( )) {

Returns true if the iteration has more elements.

Line 53: @SuppressWarnings("rawtypes")

Indicates that the named compiler warnings should be suppressed in the annotated element.

Line 54: Map.Entry me = (Map.Entry) i.next( );

A map entry is a key-value pair. The method `next` returns the next element in the iteration.

DynamicCheckbox/app/src/main/java/com/ecs/wenchen/dynamiccheckbox/MainActivity.java

```
01 package com.ecs.wenchen.dynamiccheckbox;
02
03 import java.util.Iterator;
04 import java.util.LinkedHashMap;
05 import java.util.Map;
06 import java.util.Set;
07
08 import android.app.Activity;
09 import android.os.Bundle;
10 import android.view.View;
11 import android.widget.Button;
12 import android.widget.CheckBox;
13 import android.widget.LinearLayout;
14 import android.widget.TextView;
15 import android.widget.EditText;
16 import android.graphics.Color;
17
18 public class MainActivity extends Activity {
19   LinearLayout linearBox;
20   CheckBox     checkBox, checkBox1;
21
22   @Override
23   protected void onCreate( Bundle savedInstanceState ) {
24     super.onCreate( savedInstanceState );
25     setContentView( R.layout.activity_main );
26
27     linearBox = (LinearLayout) findViewById( R.id.linearBox );
28     final EditText number = (EditText) findViewById( R.id.number );
29     final EditText title  = (EditText) findViewById( R.id.title );
30     final Button   button = (Button)   findViewById( R.id.add );
31
32     button.setOnClickListener( new View.OnClickListener( ) {
33       public void onClick( View v ) {
34         checkBox1 = new CheckBox( getApplicationContext( ) );
35         checkBox1.setId( Integer.parseInt( number.getText( ).toString( ) ) );
36         checkBox1.setText( title.getText( ).toString( ) );
37         checkBox1.setTextColor( Color.BLACK );
38         checkBox1.setOnClickListener( getOnClickDoSomething( checkBox1 ) );
39         linearBox.addView( checkBox1 );
40       }
41     } );
42
43     final LinkedHashMap<String, String>
44       alphabet = new LinkedHashMap<String, String>( );
45     alphabet.put( "260", ".NET" );
46     alphabet.put( "370", "Computer Architecture" );
47
48     Set<?> set = alphabet.entrySet( );
49     // Get an iterator.
50     Iterator<?> i = set.iterator( );
```

```
51      // Display elements.
52      while ( i.hasNext( ) ) {
53        @SuppressWarnings( "rawtypes" )
54        Map.Entry me = ( Map.Entry ) i.next( );
55        checkBox     = new CheckBox( this );
56        checkBox.setId( Integer.parseInt( me.getKey( ).toString( ) ) );
57        checkBox.setText( me.getValue( ).toString( ) );
58        checkBox.setOnClickListener( getOnClickDoSomething( checkBox ) );
59        linearBox.addView( checkBox );
60      }
61    }
62
63    View.OnClickListener getOnClickDoSomething( final Button button ) {
64      return new View.OnClickListener( ) {
65        public void onClick( View v ) {
66          final TextView tvView = (TextView) findViewById( R.id.textView3 );
67          tvView.setText( button.getId( ) + ": " + button.getText( ).toString( ) );
68        }
69      };
70    }
71  }
```

Slide 9.16: Dynamic checkboxes: Java source code (cont.)
Slide 11.1: Android server connection
Home