Lecture Input

DT228/3

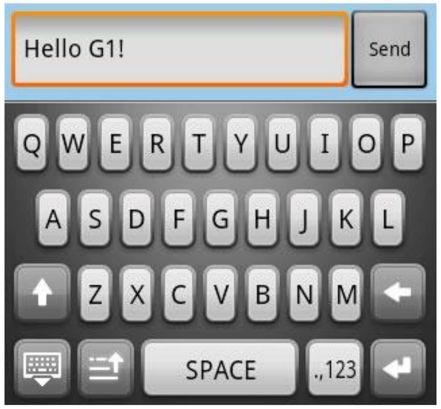
Dr. Susan McKeever

Input

- The purpose of this module is to:
  - Give you tips on how to improve the user experience when they have to type in data using the inputType attribute in XML
  - Covers "drop down" and "autocomplete" widget examples
  - Also, a note on image storage too.

Inputting Data

- Getting data from user needed for most apps…
- Android 1.5 introduced the Input Method Framework **(IMF)** (includes *software keyboard*)
- Can use IMF to tailor software keyboards for your app



Golden Rule of input

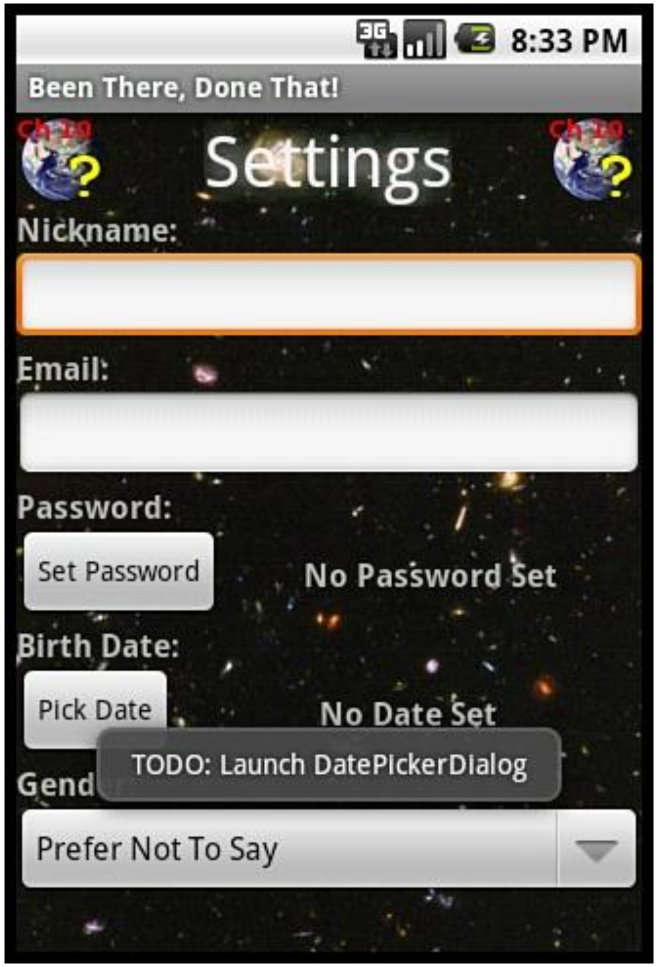
- Your user will of course need to input data sometimes.
- **Never ever (ever) get the user to type letters or characters on the keyboard unless there's no other option!!**

Example of user input

If the user HAS to enter data,

ALWAYS simplify data entry as much as youE.g.

- Validate email format
- Prevent letters going into number field (e.g. telephone number)
- Dropdown boxes..



TODO: Launch DatePickerDialog

Taking input

- If you’re expecting the user to enter data (<editText>) , the more help they get the better

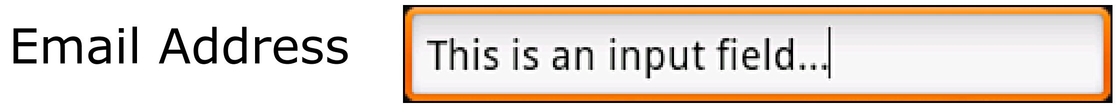
- Android lets you say what type of data they are entering.. And then modifies the displayed keyboard to support this (e.g. numbers only)

  - E.g. If field is an email, keyboard will show the @ modifier on the keyboard
  - Numeric field, the keyboard will only display numbers.. Etc.

Use XML attributes to help particular “inputType” attribute

```
<TextView
      android:text="Email address:“ />

<EditText
      android:inputType="text|textEmailAddress"/>
```



Email Address This is an input field...

- See “input” example in emulator

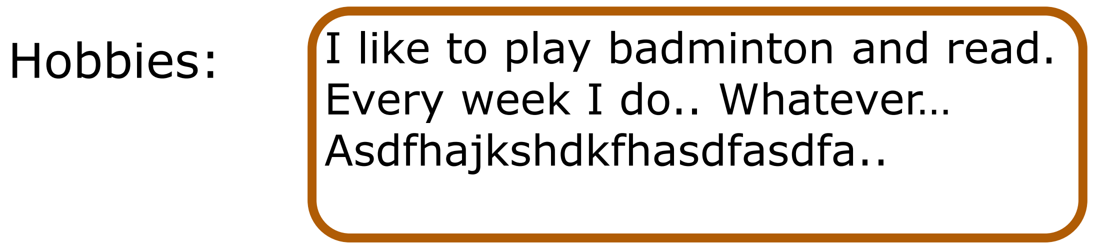
Types of input that input method editor will adjust to..

- Text (default)
- Number
- Phone
- datetime
- date
- Time
- (Don’t forget XXX Picker classes for specific input e.g. DatePicker)

Entering multiple lines or not.

- Use edittext attributes ..

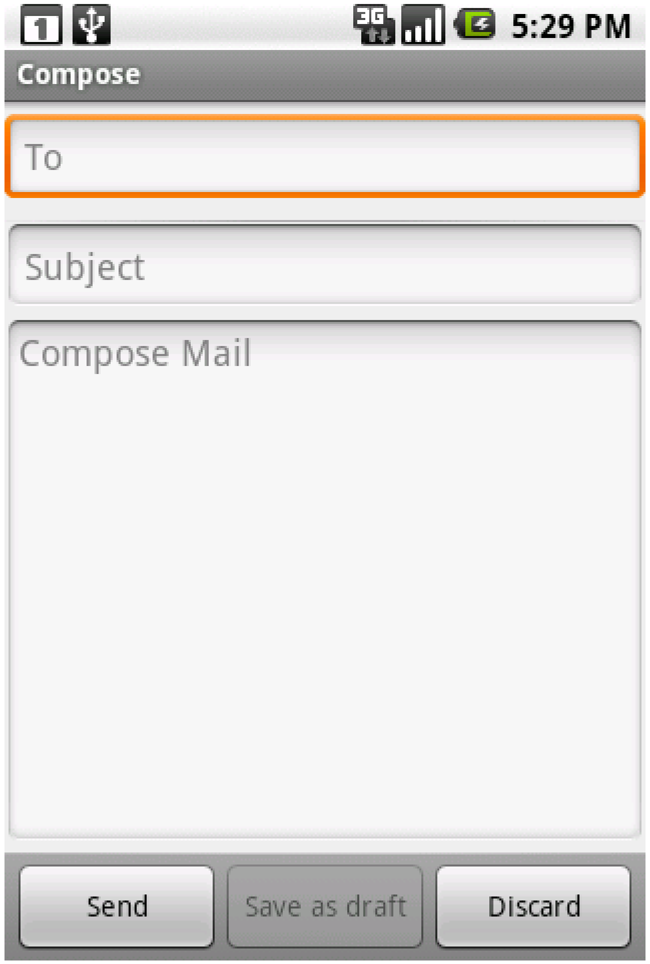
```
<TextView
          android:text=“Hobbies:"/>
<EditText
   android:inputType="text|textMultiLine|textAutoCorrect“
   android:minLines="3“
/>
```

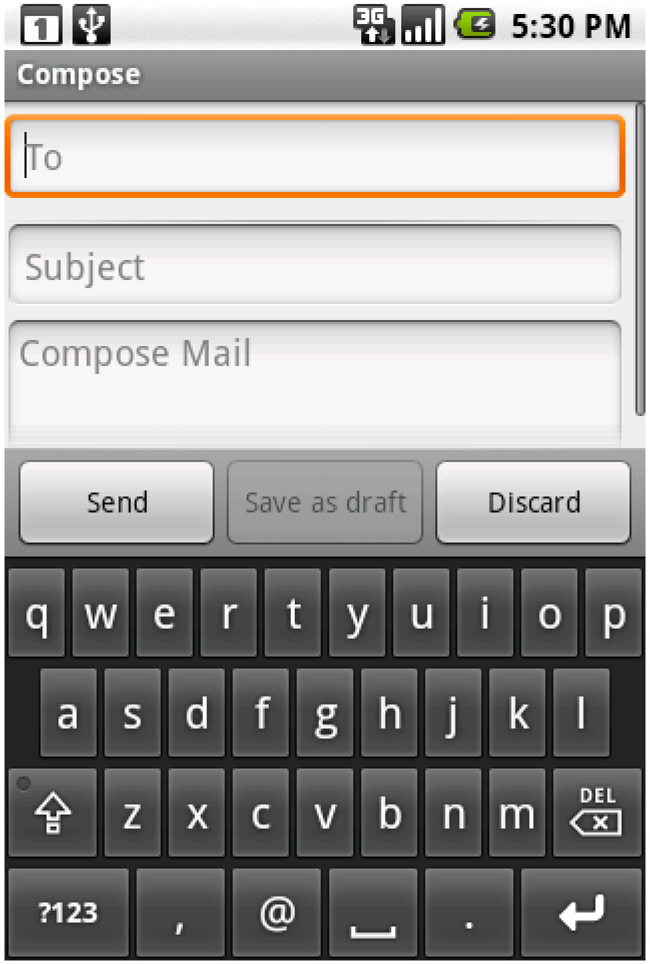


Hobbies: I like to play badminton and read.
Every week I do.. Whatever...
Asdfhajkshdkfhasdfasdfa..

Fitting the soft keyboard

- If your screen needs the keyboard.. Need to make sure your screen isn't compromised





Fitting the soft keyboard

3 possibilities

1. “Pan” your activity – Keyboard will cover part of your screen but keep current focus shown (nuisance if another field is then under keyboard)
2. “Resize” your activity – i.e. shrink it (see previous slide)
3. In landscape mode.. Shows the keyboard full screen – completely hiding your activity.. Biggest possible keyboard

Fitting the soft keyboard

- Android will make the choice it thinks best.. Depending on what's on your activity (screen)

- To control explicitly.. Need to use an attribute in the **AndroidManifest.xml** file for the activity
  - Every activity has an entry in manifest

- The attribute is called

  `android:windowSoftInputMode`

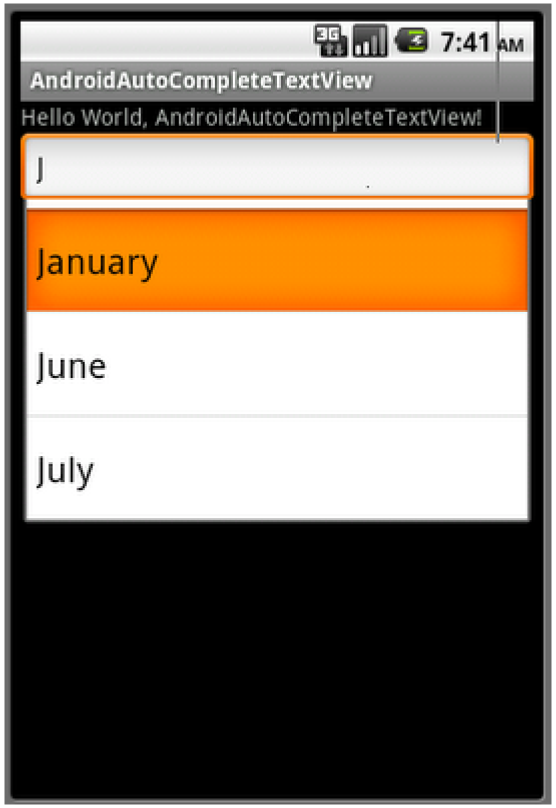
Fitting the soft keyboard

- **Sample** values from API for attribute `android:windowSoftInputMode`

"adjustResize"	The activity's main window is always resized to make room for the soft keyboard on screen.
"adjustPan"	The activity's main window is not resized to make room for the soft keyboard. Rather, the contents of the window are automatically panned so that the current focus is never obscured by the keyboard and users can always see what they are typing. This is generally less desirable than resizing, because the user may need to close the soft keyboard to get at and interact with obscured parts of the window.
"stateAlwaysHidden"	The soft keyboard is always hidden when the activity's main window has input focus.

Autocomplete

- Sometimes want system to “guess” what’s being typed.
  - i.e. like predictive text on sms
- To implement:
  - Need candidate list of “guesses”
  - Have a way of declaring that a particular input field is going to do autcomplete


7:41 AM
AndroidAutoCompleteTextView
Hello World, AndroidAutoCompleteTextView!
J
January
June
July

Autocomplete – main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        >
        <TextView
                android:id="@+id/selection"
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                />
        <AutoCompleteTextView android:id="@+id/edit"
                        android:layout_width="fill_parent"
                        android:layout_height="wrap_content"
                        android:completionThreshold="3"/>
</LinearLayout>
```

- **What is the meaning of android:completionThreshold attribute?**

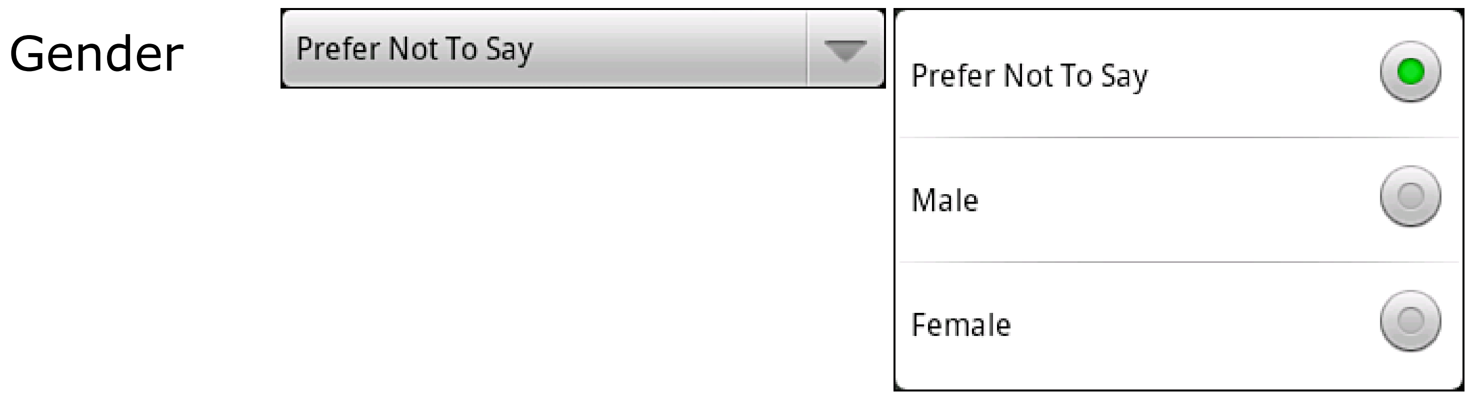
```
public class AutoCompleteDemo extends Activity
{
  AutoCompleteTextView edit;
  // candidate "guess" list here for illustration ..
  //  But better external
  String[]  countries={"  "Afghanistan", "Albania",  etc…"};

  @Override
   public void onCreate(Bundle icicle)
  {
       super.onCreate(icicle);
       setContentView(R.layout.main);
       edit=(AutoCompleteTextView)findViewById(R.id.edit);
       edit.addTextChangedListener(this); // if you want something to happen
       edit.setAdapter(new ArrayAdapter<String>(this,
       android.R.layout.simple_dropdown_item_1line,countries);
   }
       // whatever other code you want…
}
```

- **What's this code doing?**

Spinners

- Androids name for Drop Down boxes
- Good for getting user to pick something without typing..



Spinners

- Has its own tag that goes in the XML file
  `<Spinner>`
- Need to give them data (for the choices)
- Usual technique for data supply
  - Create an adapter
  - Link it to a data source
  - And assign it to a spinner

- See code attached to the lecture

Storing images in Android

- Android supports the following image files:
  - .png, .jpg.gif, …
  - Preferred form is .PNG

- Store the images in Res/Drawable
- (Note: app launcher icons go in Res/mipmaps)

- Can also have Drawablehdpi, mdpi, ldpi etc. Android will dynamically pick those depending on screen dpi

Using images in Android

To display an image, need as usual to add the right tag to your layout XML file (e.g. main.xml)

```
<LinearLayout.. Etc etc.
<ImageView android:id="@+id/test_image"
    android:src="@drawable/test"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    />
```

- What's the name of the image file?
- Where should the image file be stored?
- To use this image, how would you access it from your java code in your activity?

To display an image, need as usual to add the right tag to your layout XML file (e.g. main.xml)

Using images in Android

```
public class TestImages extends Activity
{

/** Called when the activity is first created. */

@Override
public void onCreate(Bundle savedInstanceState)
{
   super.onCreate(savedInstanceState);
   setContentView(R.layout.main);
   ImageView image = (ImageView) findViewById(R.id.test_image);
    image.setImageResource(R.drawable.test2);

} }
```

- **What’s this activity doing?**