CS 696 Emerging Web and Mobile Technologies
Spring Semester, 2011
Doc 20 Some UI
Apr 11, 2011

References

The Design of Everyday Things, Norman, Basic Books, 2002

Optimizing a Screen for Mobile Use, http://www.useit.com/alertbox/mobile-redesign.html

Mobile Design and Development, Brian Fling, http://mobiledesign.org/

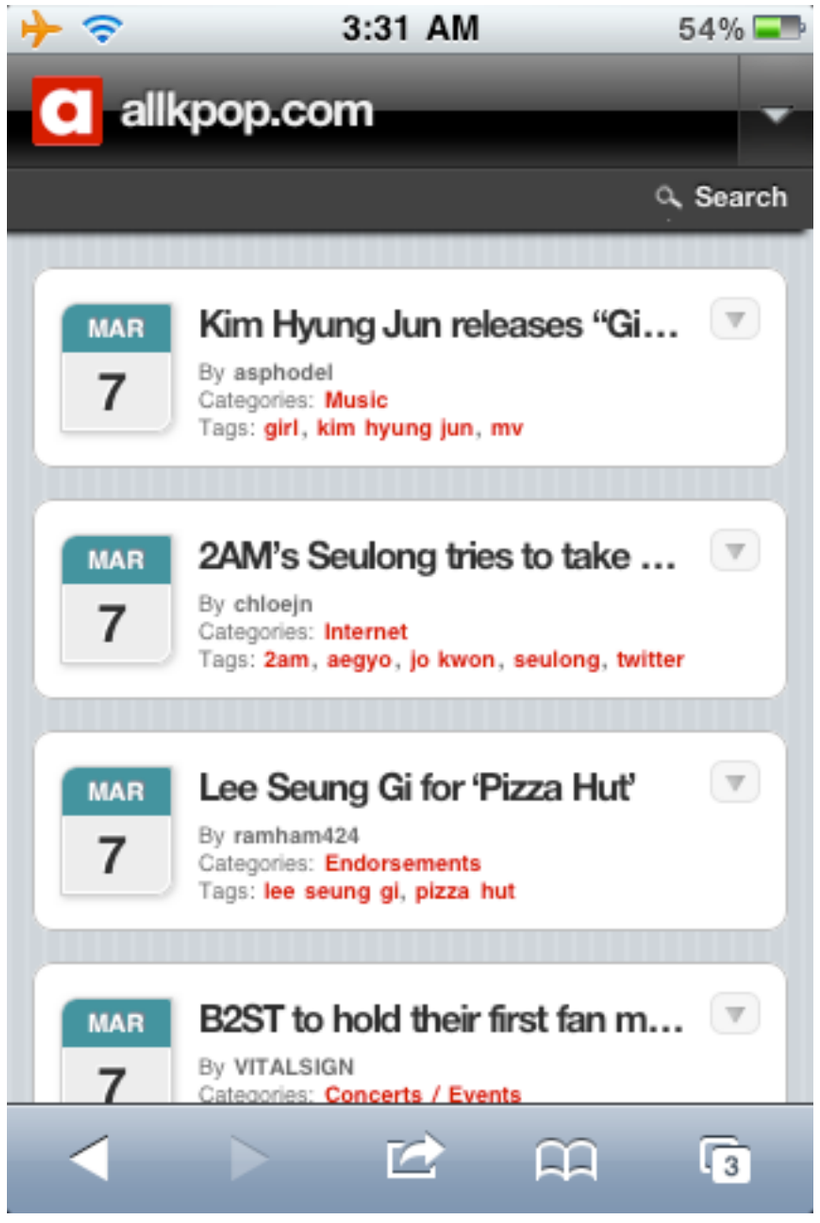
Android Documentation

http://developer.android.com/guide/index.html

Reading Assignments

Some Background

Optimizing a Screen for Mobile Use



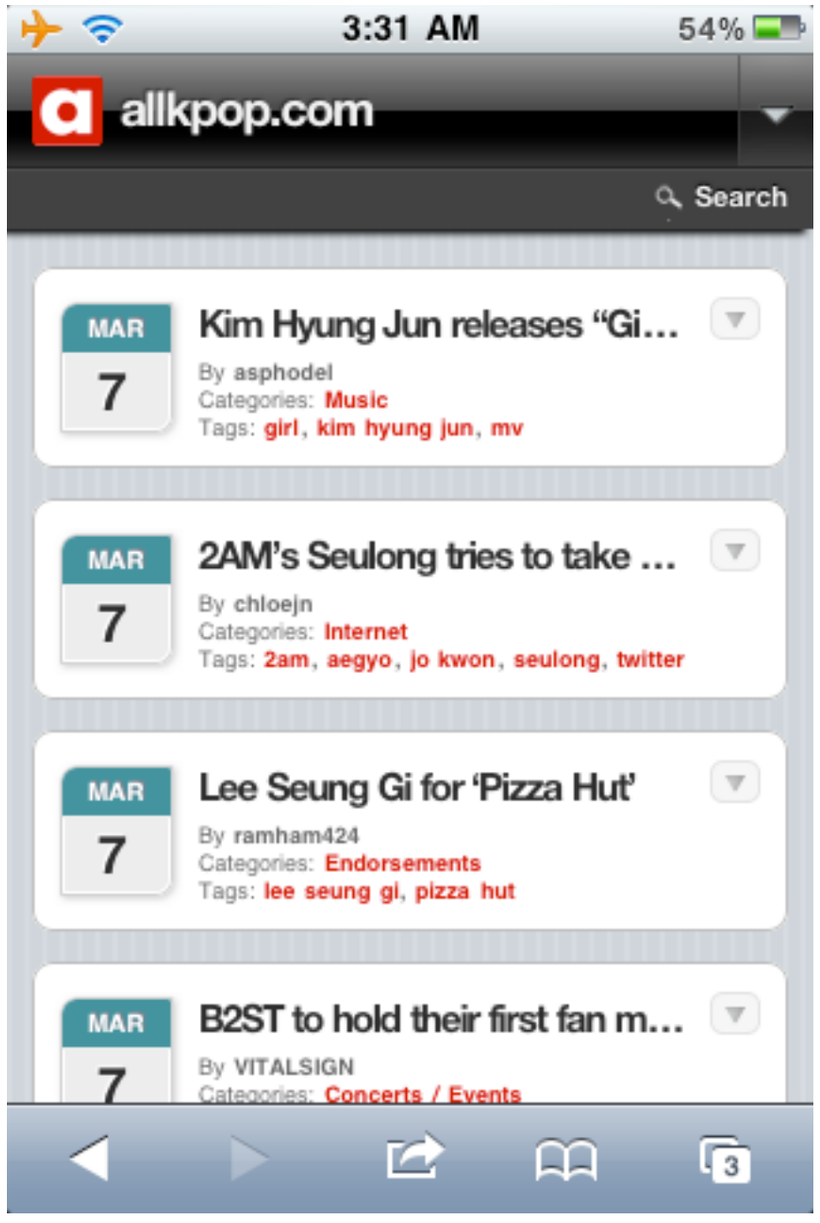
Jakob Nielson - Usability Expert

http://www.useit.com/alertbox/mobile-redesign.html

Redesigned site as exercise

http://www.useit.com/alertbox/mobile-redesign.html

Allkpop mobile Site



The Good

- Separate desktop & mobile versions
- Large touch targets
- Content-carrying keywords

http://www.useit.com/alertbox/mobile-redesign.html

Custom

- Text summaries
- Thumbnail images

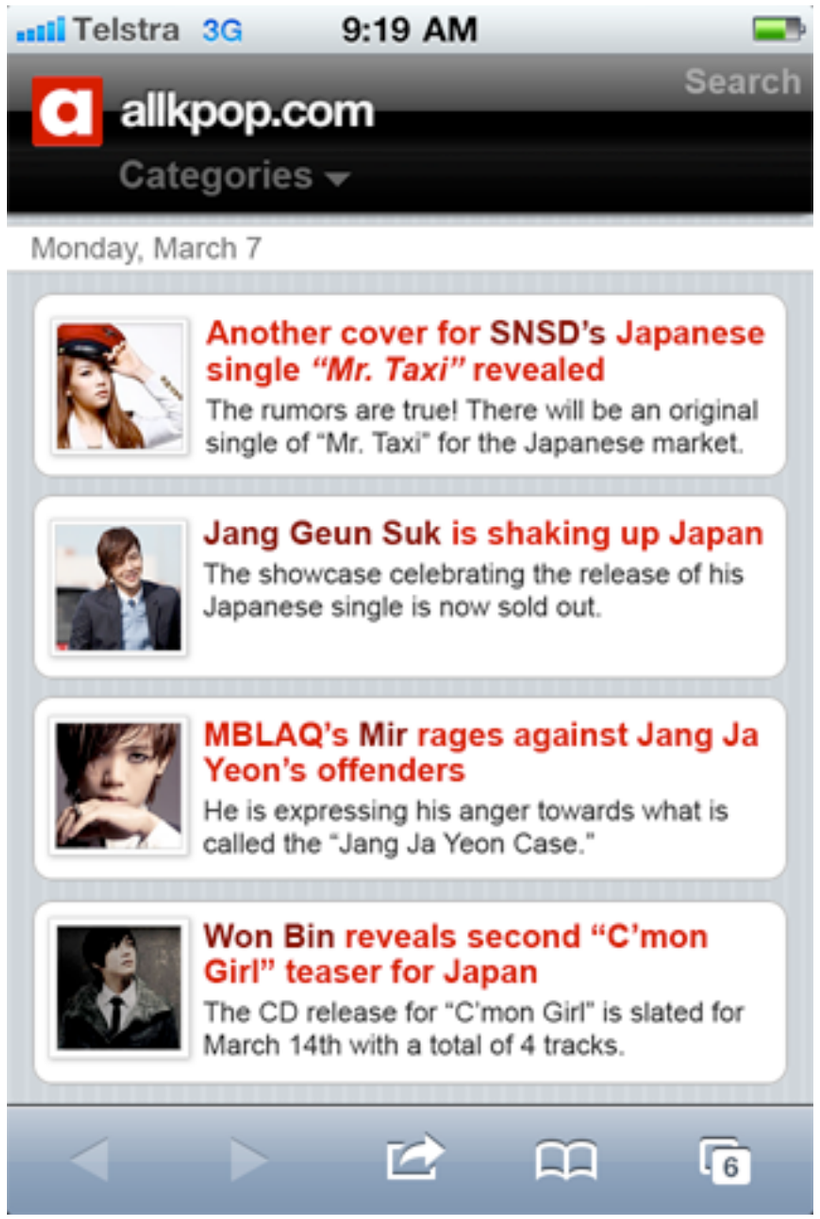
Iteration 2

Added

- Search
- Date

Combined less used items into one menu

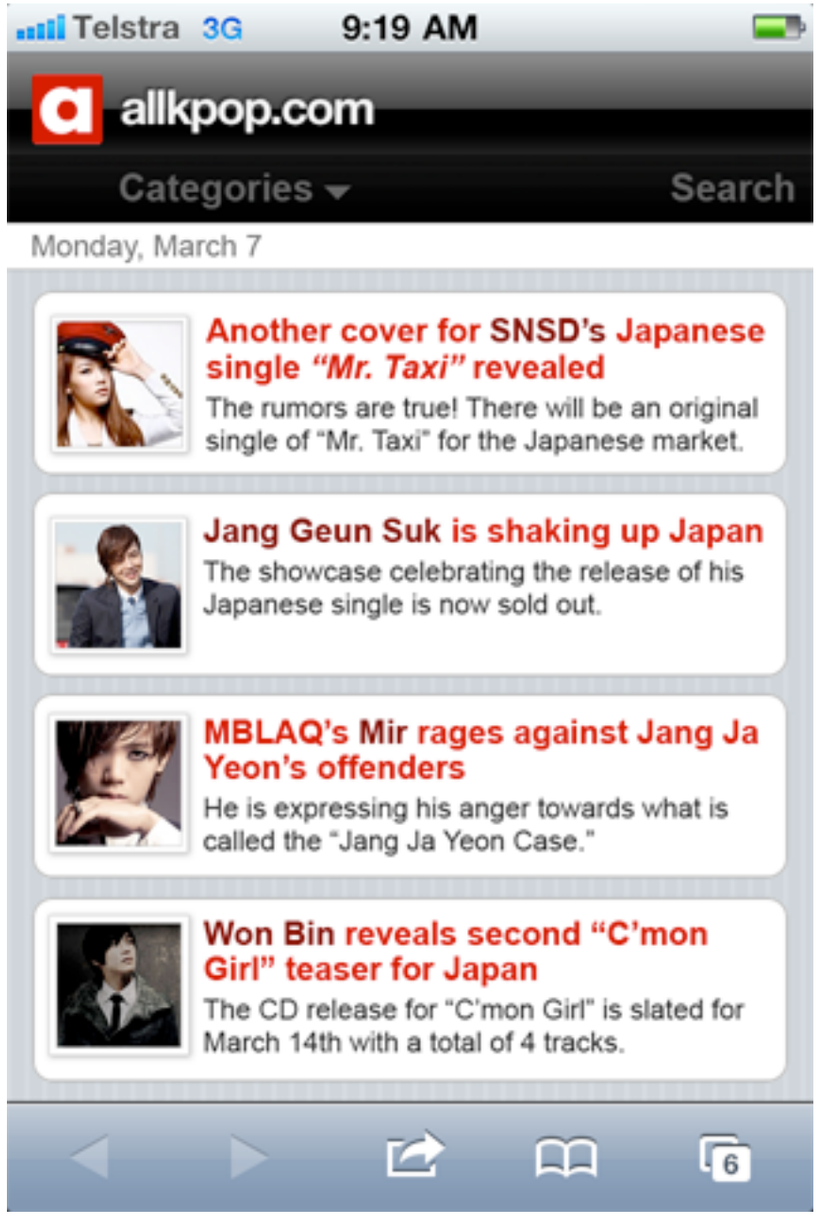
Iteration 3



Menu becomes "Categories"

Moved Search and Categories apart

Final Version



One screen

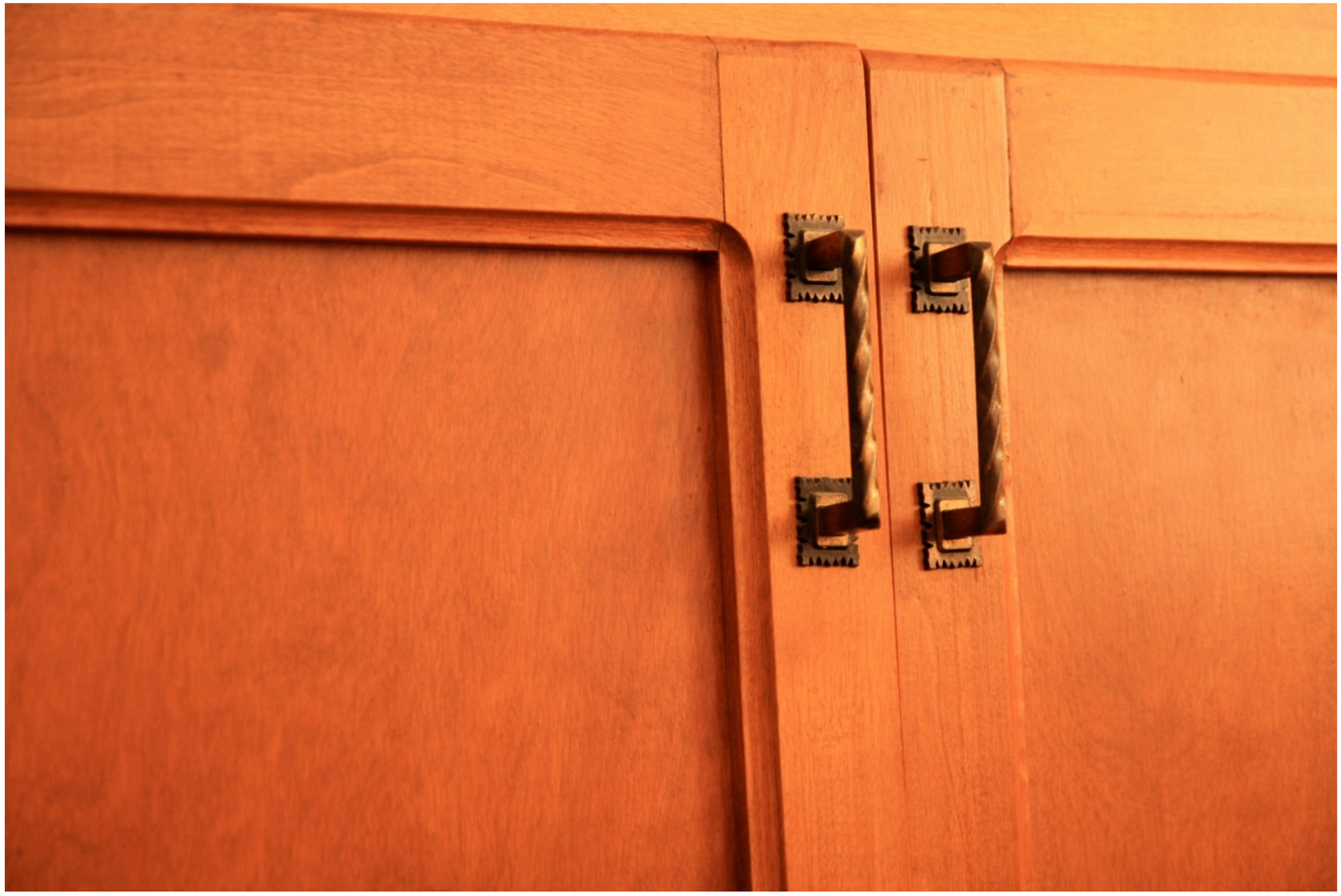
Required 10 design changes to meet usability guidelines

Psychopathology of Things

Affordance
Conceptual Model
Make Things Visible
Mapping
Feedback

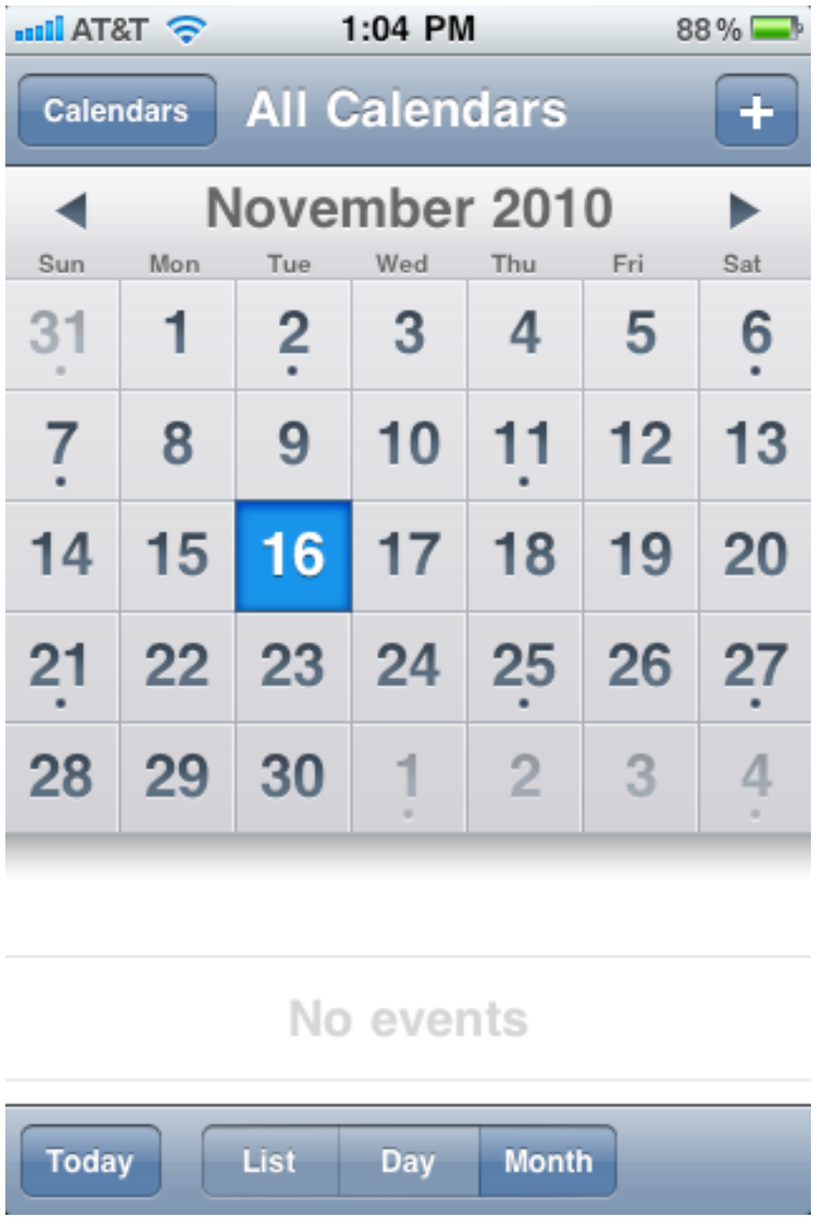
From The Design of Everyday Things, Norman, Basic Books, 2002

Affordance



Affordance

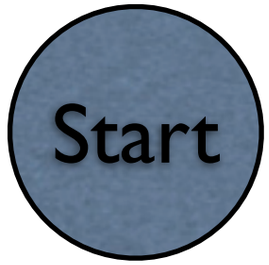
Affordance


AT&T
1:04 PM
88 %
Calendars
All Calendars
+
November 2010
Sun Mon Tue Wed Thu Fri Sat
31 1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 1 2 3 4
No events
Today
List
Day
Month


AT&T
12:19 PM
91 %
La Mesa
H: 70° L: 47°
65°
TUESDAY 70° 47°
WEDNESDAY 72° 47°
THURSDAY 75° 47°
FRIDAY 69° 47°
SATURDAY 65° 45°
SUNDAY 63° 45°
Updated 11/16/10 12:19 PM

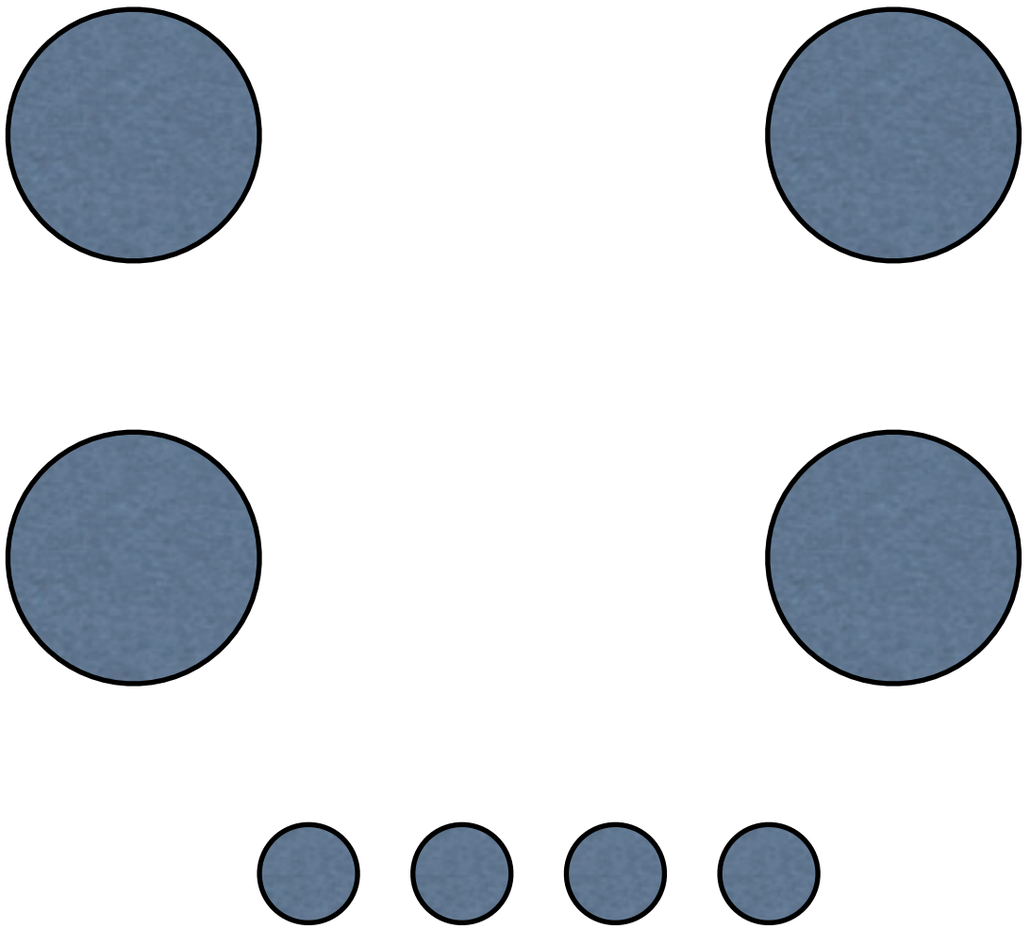
Conceptual Model

Make things Visible

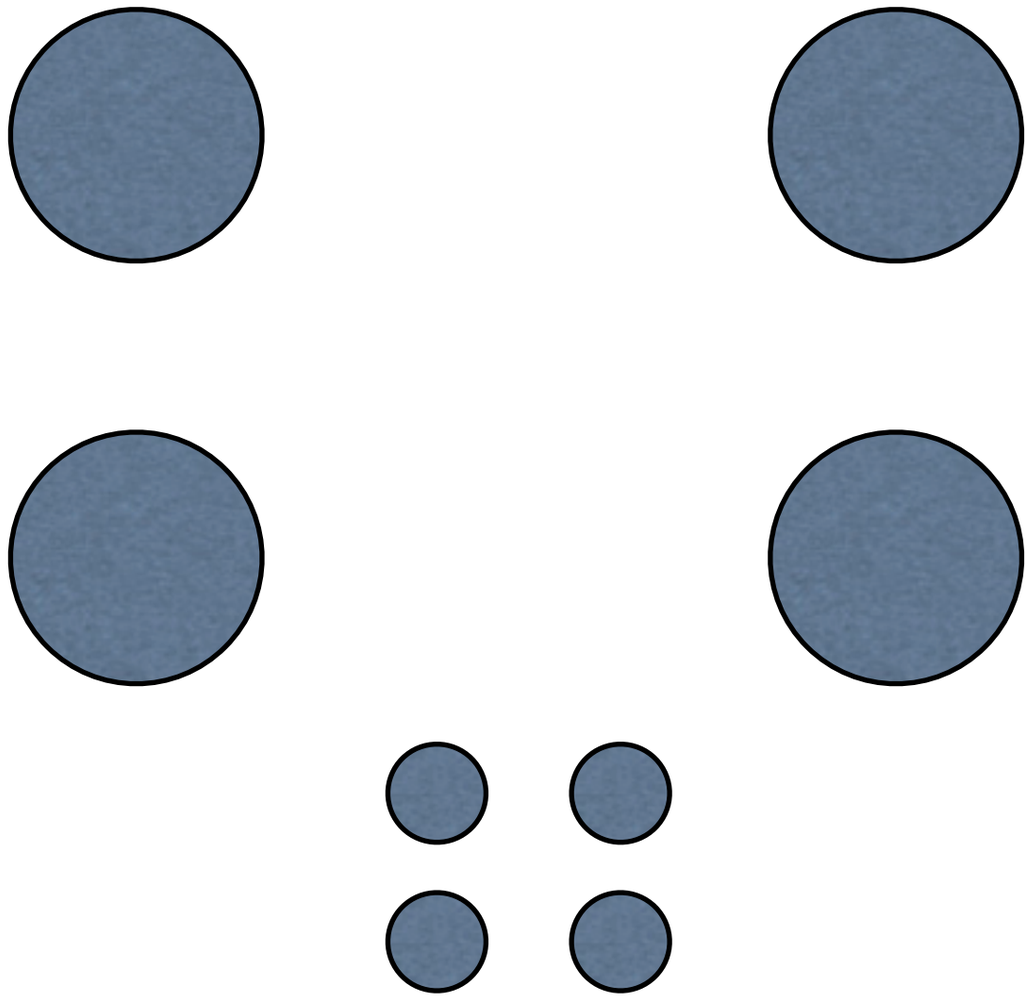


Make things Visible

Mapping



Mapping



Feedback

0.1 seconds

0.1 to 1 seconds

More than one second

Why is Angry Birds so Popular

http://www.mauronewmedia.com/blog/2011/02/why-angry-birds-is-so-successful-a-cognitive-teardown-of-the-user-experience/

Information Architecture

"The truly successful mobile products always have a well-thought-out information architecture"

"information architecture stands apart as being the most crucial element of your product"

Brian Fling

http://mobiledesign.org/mobile_information_architecture

Information Architecture

How the user will get to information or perform tasks within an application

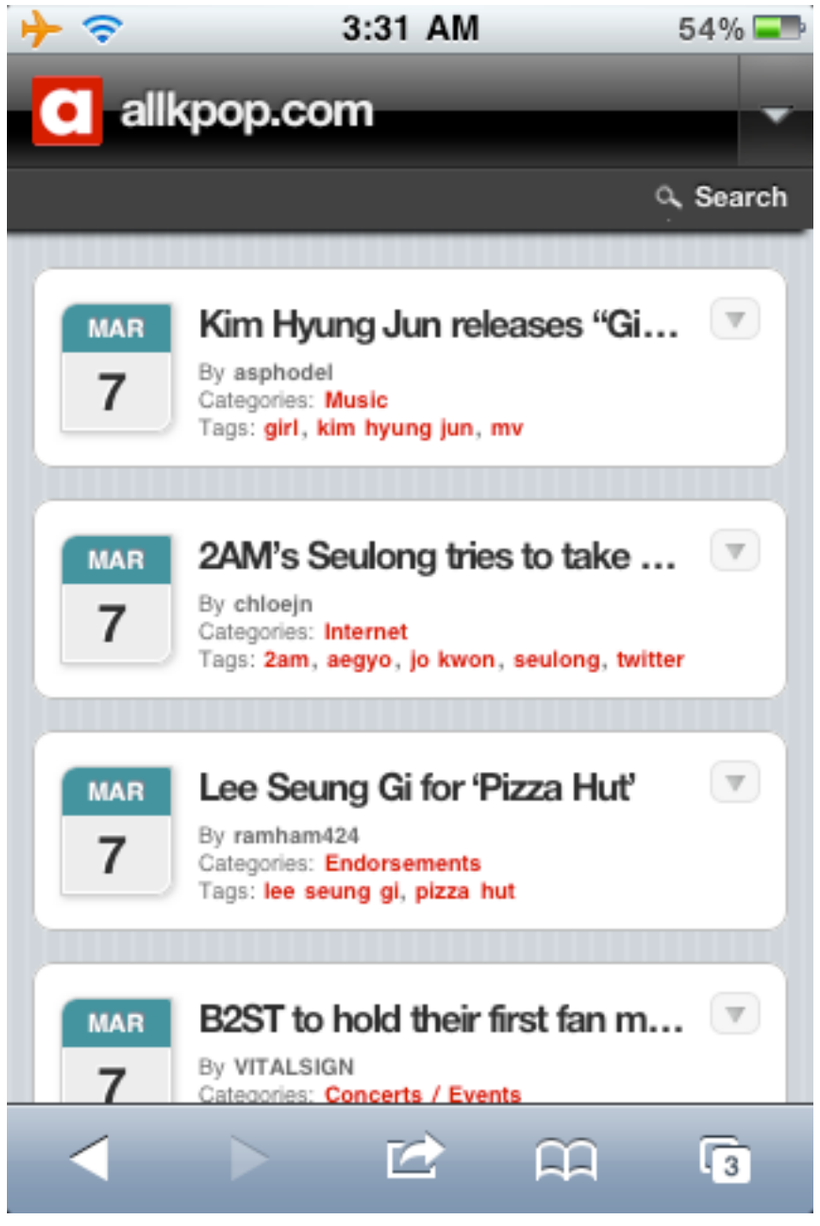
How the user will interact with the application to accomplish her goals

Visual layout of information

The design of the visual paradigms used to create action or understanding

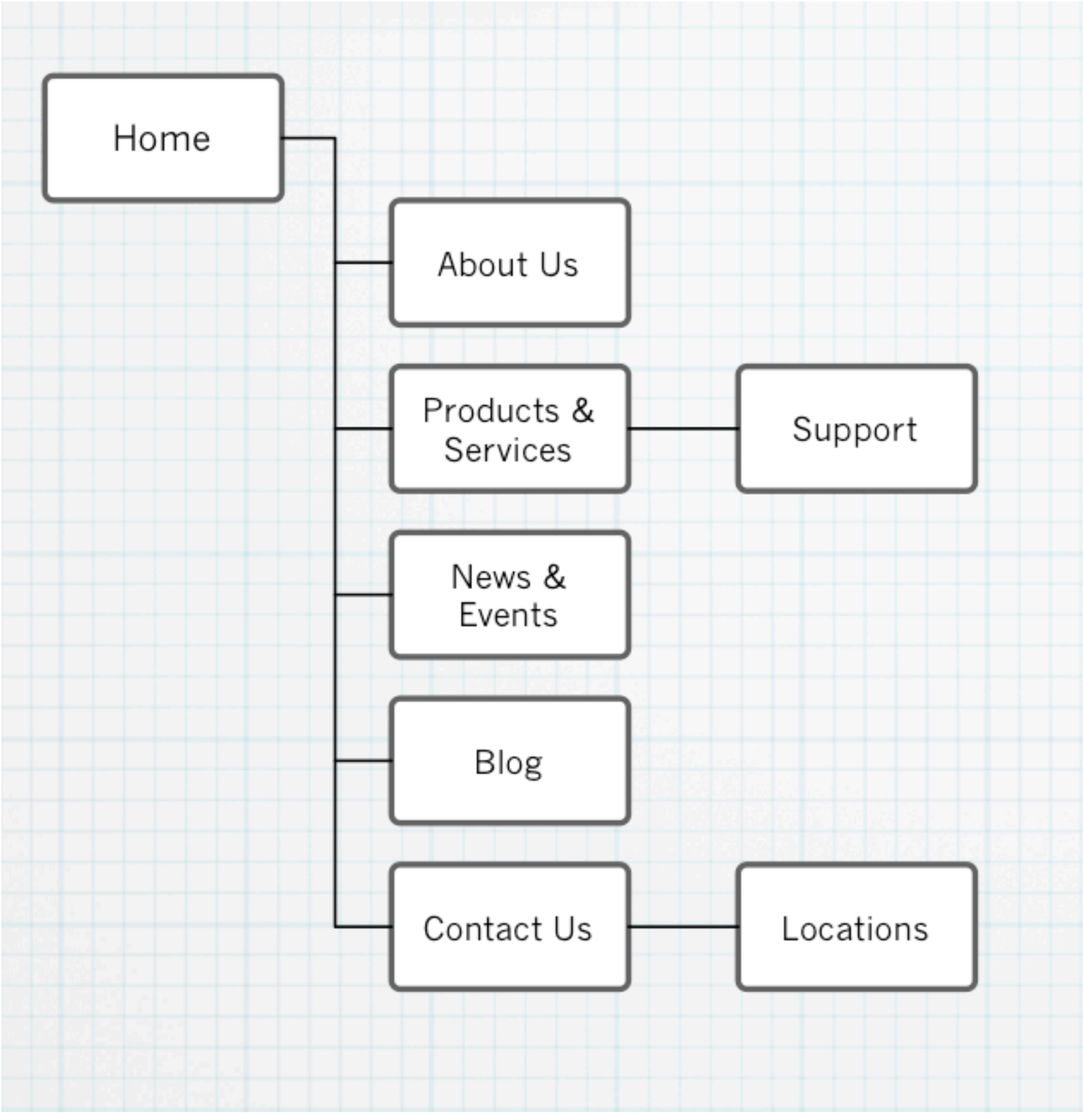
http://mobiledesign.org/what_is_information_architecture

Information Architecture



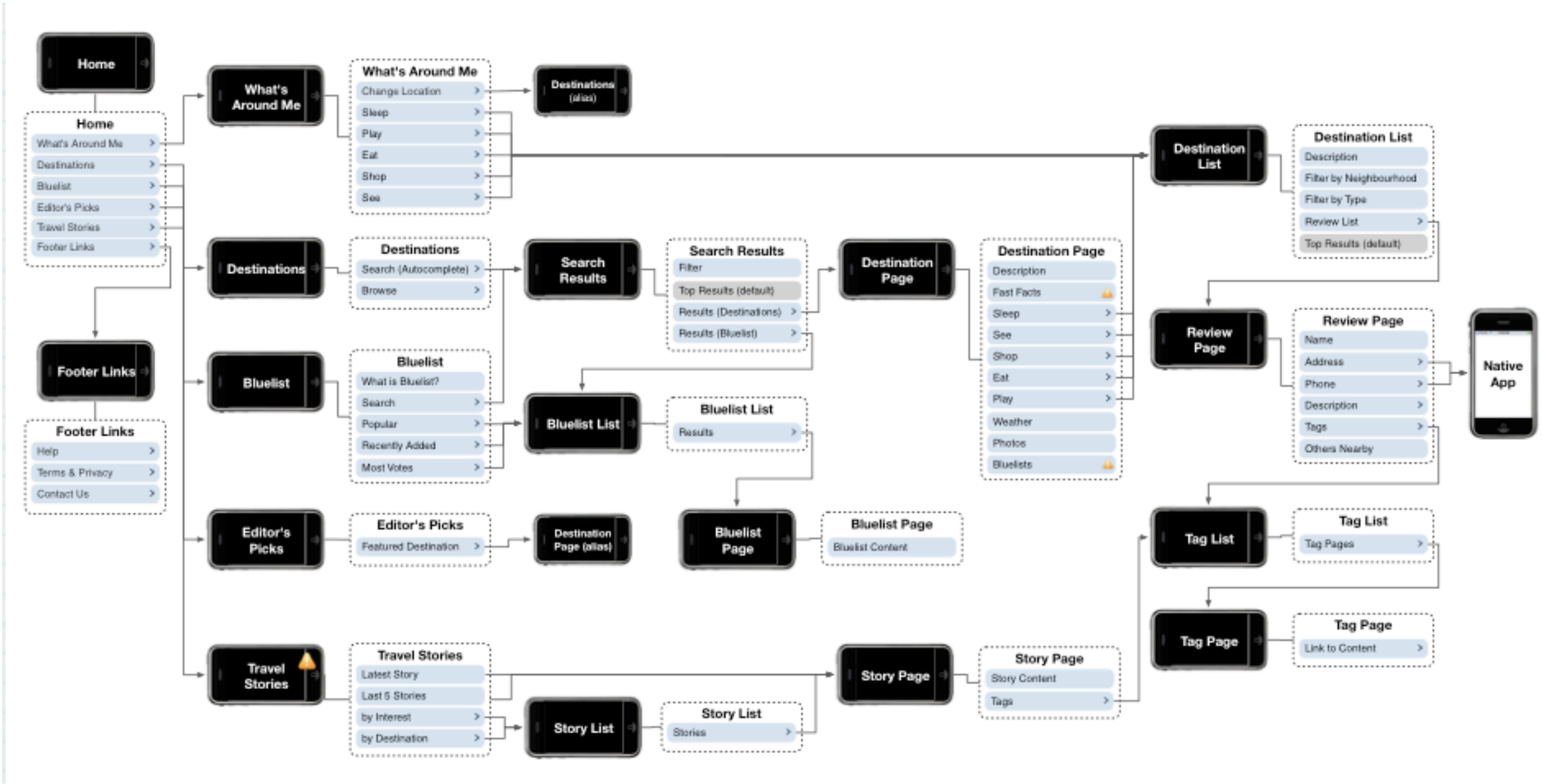
Some Design Tools

Site Map



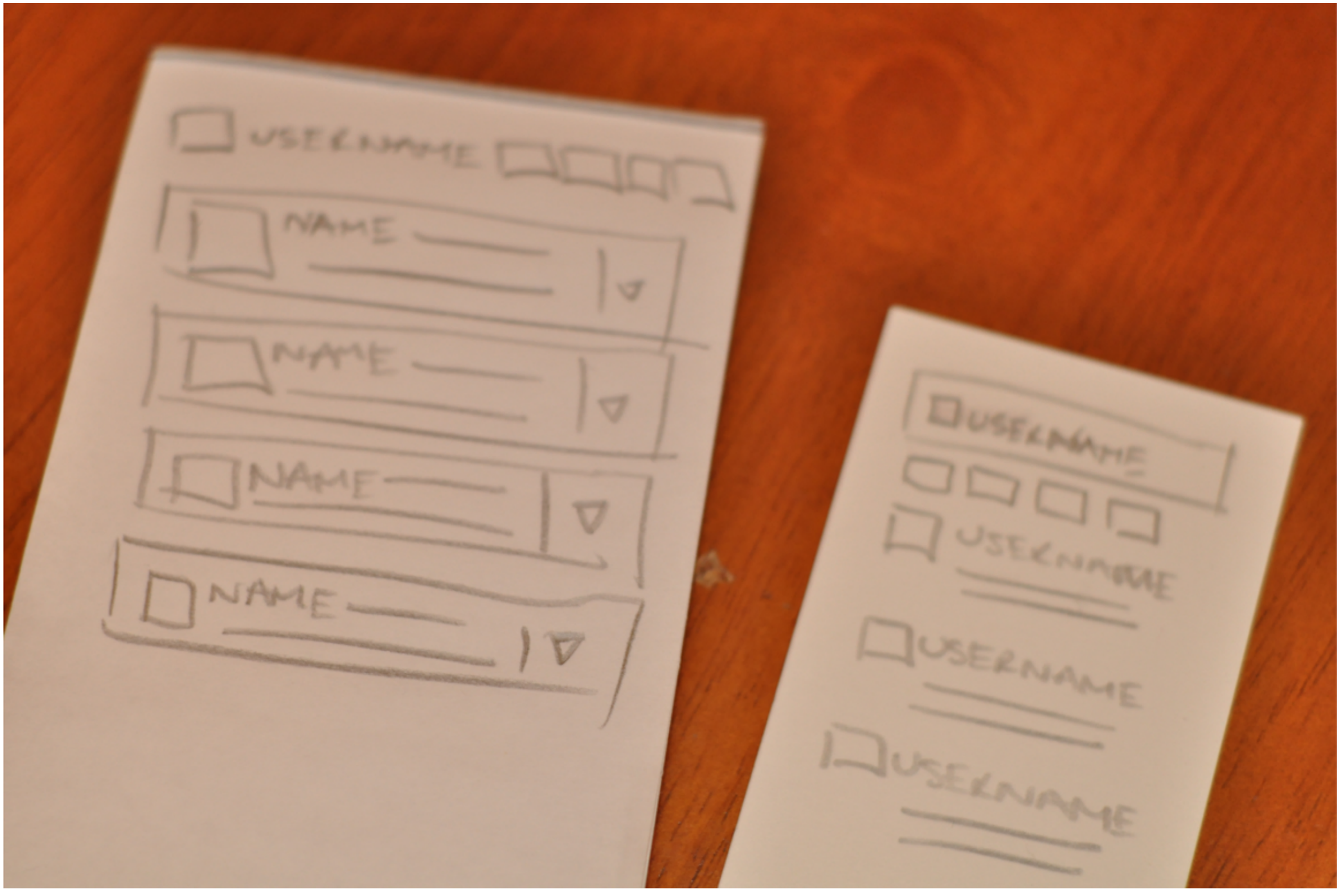
Source http://mobiledesign.org/site_maps

Clickstreams



Source http://mobiledesign.org/clickstreams

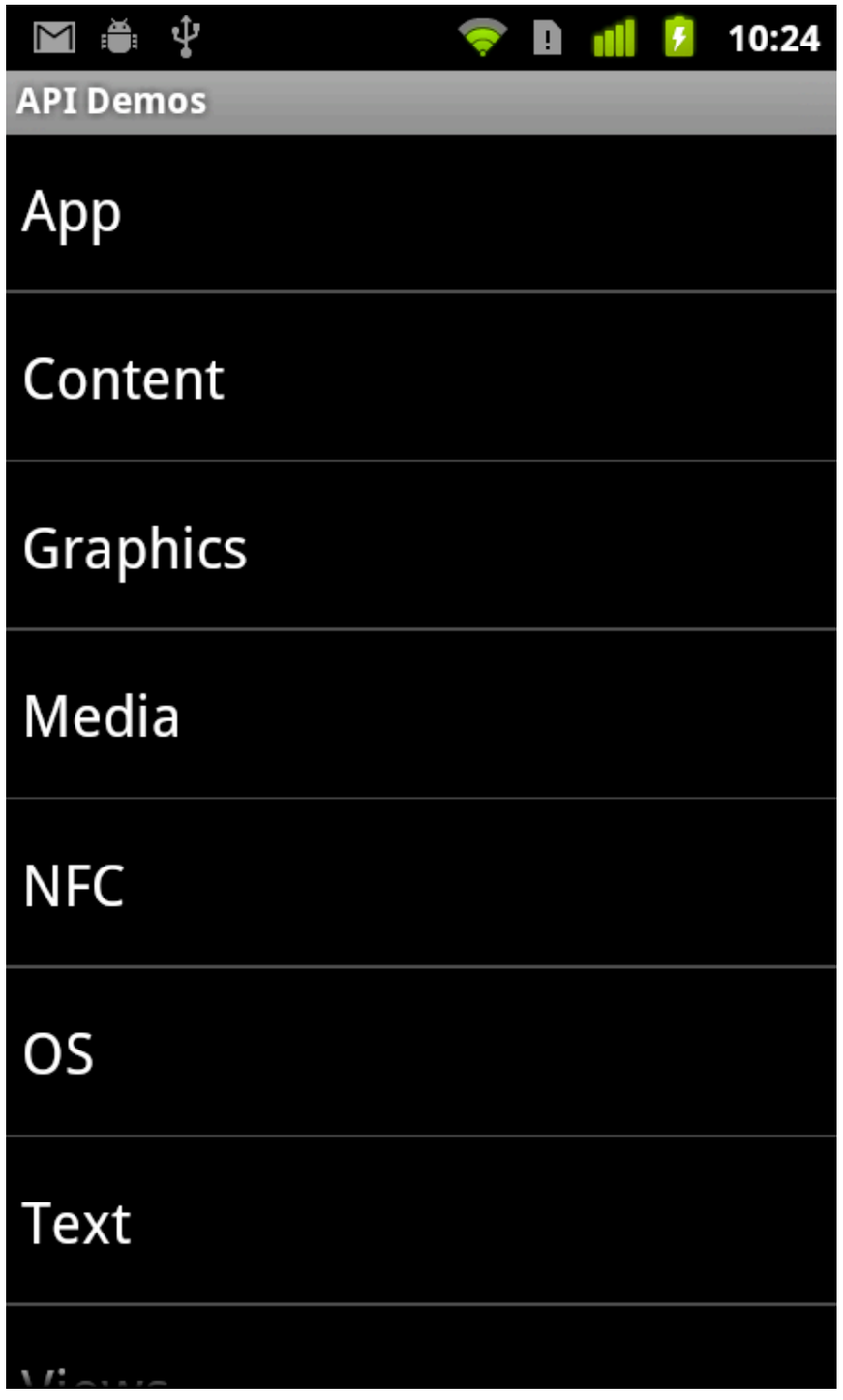
Prototypes



Source http://mobiledesign.org/prototyping

Android UI

Examples



http://developer.android.com/resources/browser.html

Source

androidInstallation/platforms/android-10/samples/ApiDemos

Activity

Code that does some work

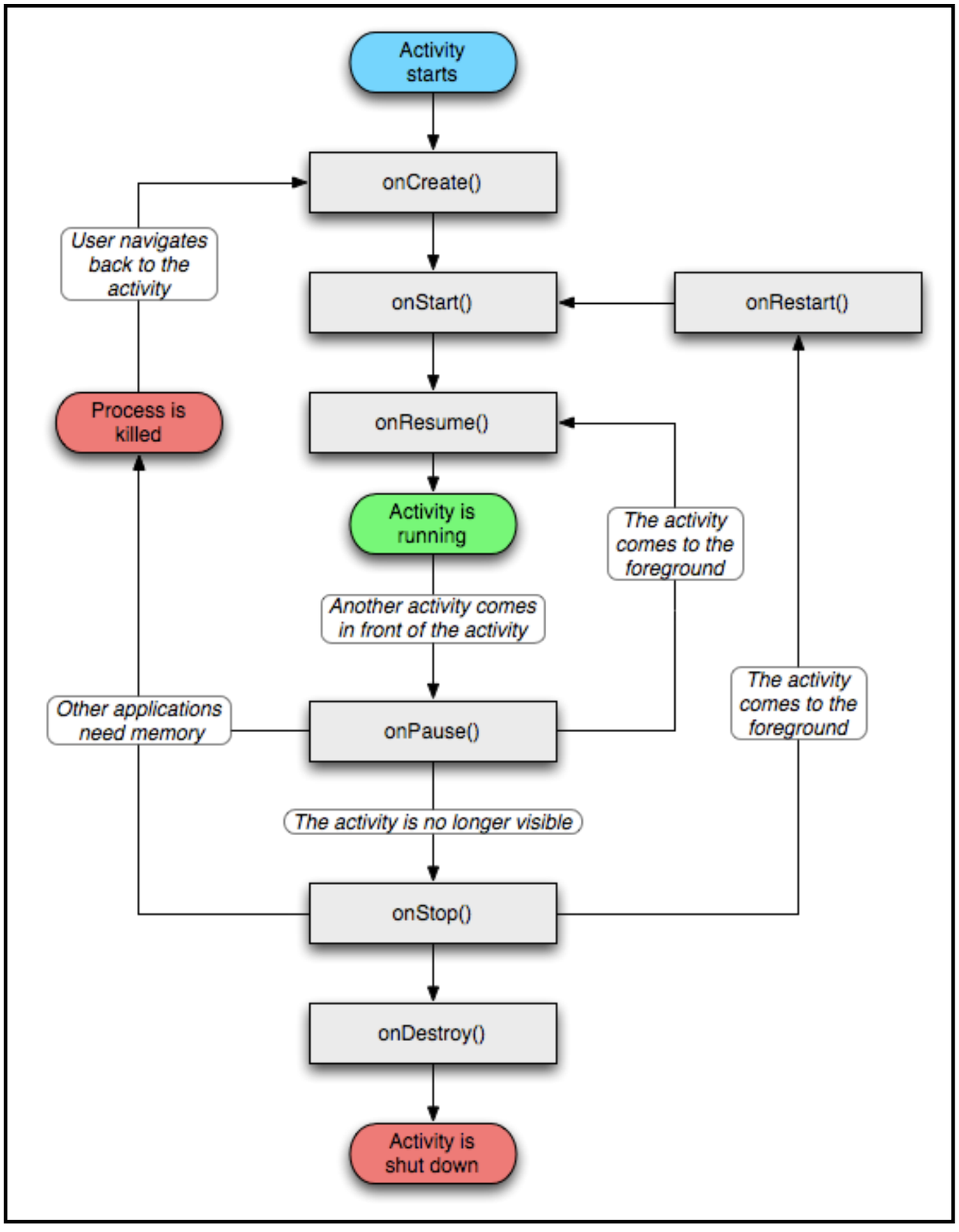
Single, focused thing that a user can do

Usually each screen(View) has its own activity

An application may have multiple screens, hence multiple activities

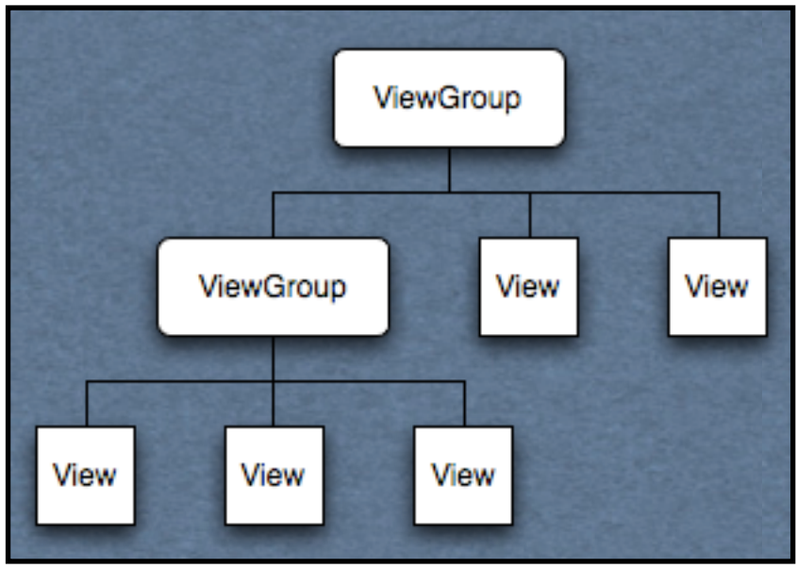
An application runs in its own Linux process

Activities can be viewless


Activity starts
onCreate()
User navigates back to the activity
onStart()
onRestart()
Process is killed
onResume()
Activity is running
The activity comes to the foreground
Another activity comes in front of the activity
Other applications need memory
onPause()
The activity comes to the foreground
The activity is no longer visible
onStop()
onDestroy()
Activity is shut down

Views

View
- Displays content in rectangular area of screen
- Handles
  - Layout, focus, scrolling
  - Keyboard events
  - Gestures



ViewGroups
- Manages set of views and view groups
- Composite pattern

View Documentation

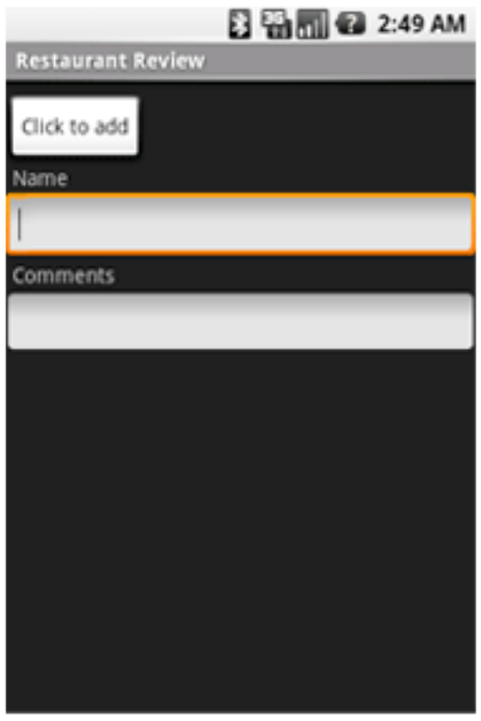
http://developer.android.com/reference/android/view/package-summary.html

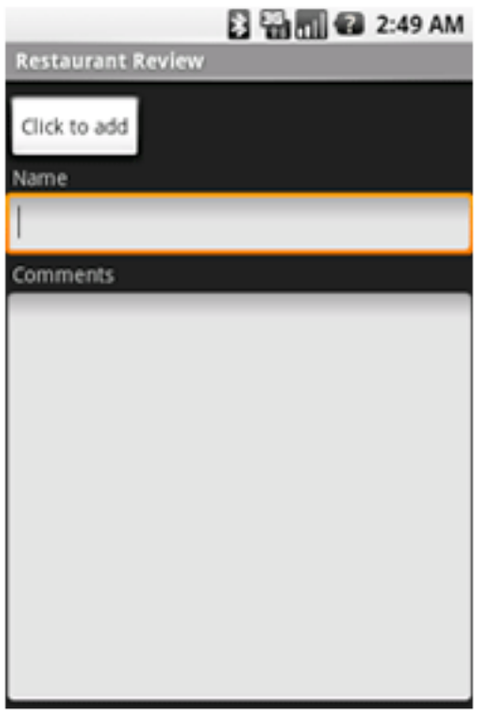
Common Layouts

FrameLayout - displays one child

LinearLayout

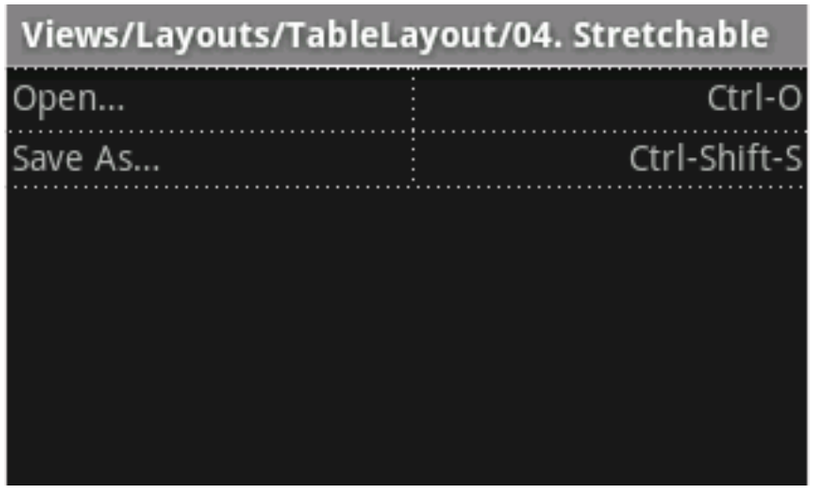
Items stacked vertically or horizontally



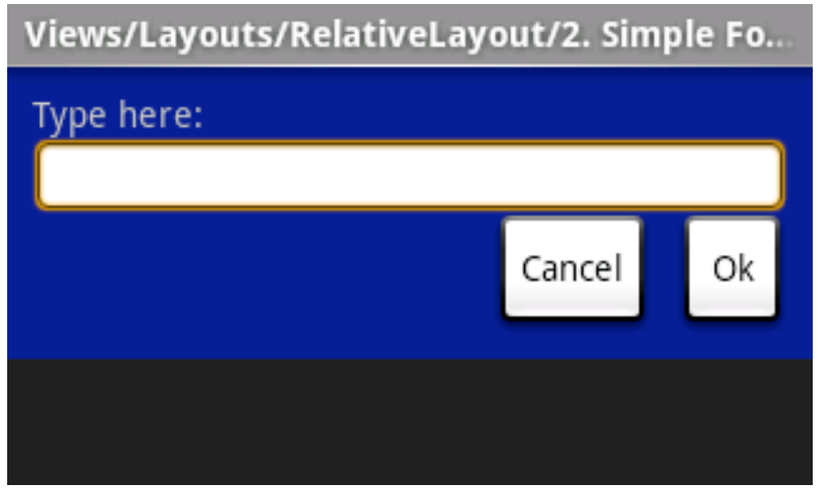


Common Layouts

Tablelayout



RelativeLayout



Layout Documentation with Examples

http://developer.android.com/guide/topics/ui/layout-objects.html

Intents - Calling Activities

Android application consists of multiple actvities

Activity represents one screen or view

Going from one screen to another
- Requires calling activity

Can't call new activity directly
- Use intent to indicate activity to start

Intent Resolution - Explicit Intents

Specify the component (class) an intent is to run

Common way to call your own code

```
Intent go = new Intent();
go.setClassName("edu.sdsu.cs696", "edu.sdsu.cs696.Hello");
startActivity(go);
```

Intent Resolution - Implicit Intents

Provide information about activity you want to run

display web page

System determines which component to run

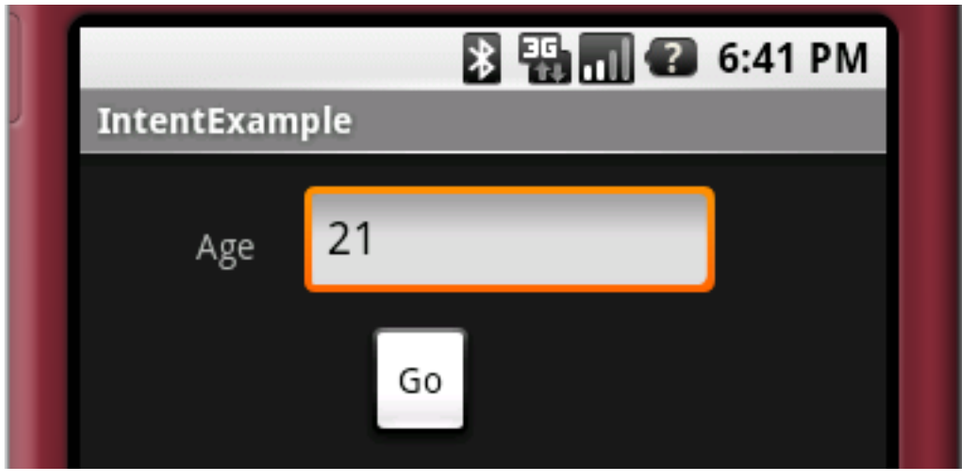
If more than one activity can handle request
User is asked to select

Each activity declares in manifest what it can handle

```
<intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

Intent - Passing Data

IntentExample

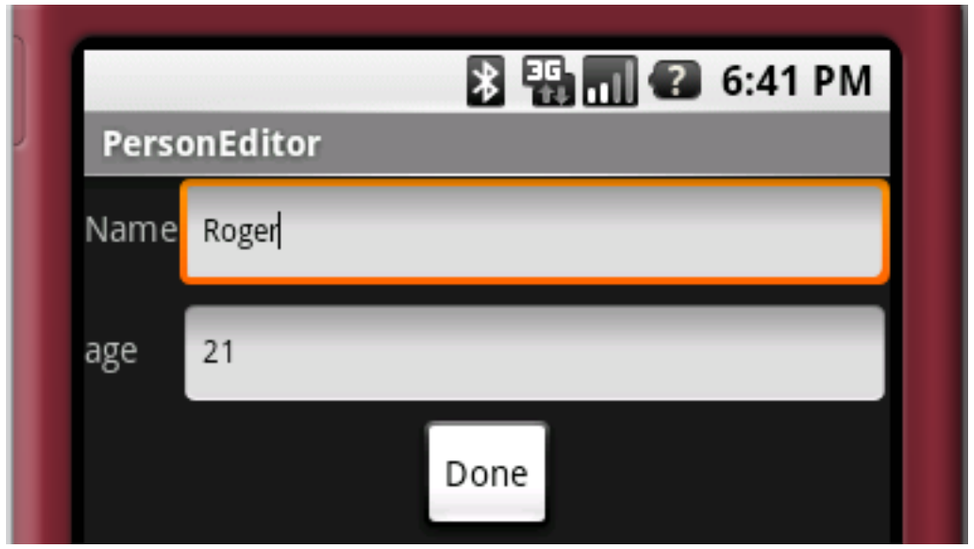


Displays/Edits age

Go button

- Calls PersonEditor
- Passes data
  - Name
  - Age

PersonEditor



Displays/Edits Name and age

Done button

- Returns edited data back
- Age = 0 cancels edit

IntentExample.java

```
public class IntentExample extends Activity implements View.OnClickListener {
    private EditText numberText;
    private static final int INTENT_EXAMPLE_REQUEST = 123;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.intent);
        Button ok = (Button) findViewById(R.id.go);
        ok.setOnClickListener(this);
        numberText = (EditText) this.findViewById(R.id.number);
        numberText.setText("21");

    }
```


When we want a reply back from an Intent request we supply a request number. The request number is return with the answer. That way it is possible to know where the request originated from.

IntentExample.Java continued

Sending the data to PersonEditor

```
public void onClick(View v) {
    Intent go;
    go = new Intent();
    go.setAction("android.intent.action.EDIT");
    go.addCategory("person_editor");
    String newAge = numberText.getText().toString();
    go.putExtra("age", newAge);
    go.putExtra("name", "Roger");
    startActivityForResult(go, INTENT_EXAMPLE_REQUEST);
}
```


The name was sent just to show we can send multiple items. They can be of any base type or serializable. See the putExtra methods in the Intent class.

IntentExample.Java continued

Getting the Results back

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode != INTENT_EXAMPLE_REQUEST) {
        numberText.setText("Not from me");
        return;
    }
    switch (resultCode) {
    case RESULT_OK:
        String editedAge = data.getStringExtra("age");
        numberText.setText(editedAge);
        break;
    case RESULT_CANCELED:
        numberText.setText("Cancelled");
        break;
    }
  }
}
```


When an activity is started after a return from a startActivityForResult request, the method onActivityResult is called. The requestCode is the request code sent in the startActivityForResult method. The result code is the set by the called activity and is RESULT_OK or RESULT_CANCELED by convention. The intent is used to send data back to this activity. RESULT_OK and RESULT_CANCELED are constants defined in the Activity class.

PersonEditor.java

```
public class PersonEditor extends Activity implements View.OnClickListener {
    private EditText ageText;
      private EditText nameText;

      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.person_editor);
            Button done = (Button) findViewById(R.id.edit_done);
            done.setOnClickListener(this);
            ageText = (EditText) this.findViewById(R.id.edit_age);
            nameText = (EditText) this.findViewById(R.id.edit_name);
            Bundle personData = getIntent().getExtras();
            String age = personData.getString("age");
            String name = personData.getString("name");
            if ((age != null) && (name != null)) {
                  ageText.setText(age);
                  nameText.setText(name);
            }
      }
```


Showing how to access the intent that started the activity and extracting the extras from the intent. If the key-value pair was not set in the intent the value will be returned as null.

PersonEditor.java

Returning the data

```
public void onClick(View v) {
    String newAge = ageText.getText().toString();
    Intent result = getIntent();
    result.putExtra("age", newAge);
    if (newAge.equals("0"))
        setResult(RESULT_CANCELED, result);
    else
        setResult(RESULT_OK, result);
    finish();
}
```


setResult(int, Intent) returns information to the calling activity. The first parameter is the result code passed back in onActivityResult. The intent is the intent passed back in onActivityResult.

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="edu.sdsu.cs683.example" android:versionCode="1"
    android:versionName="1.0.0">
    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".IntentExample" android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:label="PersonEditor" android:name="PersonEditor">
            <intent-filter>
                <action android:name="android.intent.action.EDIT"></action>
                <category android:name="person_editor"></category>
                <category android:name="android.intent.category.DEFAULT">
             </category>
            </intent-filter>
        </activity>
    </application>
</manifest>
```


The intent filter for the activity must contain all the categories used by the intent to select the activity. It can contain more categories. The example does not work without the default category.