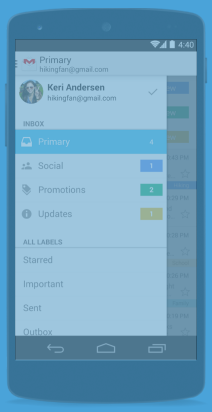

Xamarin Evolve 2014
Homeward Bound:
Implementing Android
Navigation Patterns
Michael Stonis
Eight-Bot, Inc.
@michaelstonis


Xamarin Evolve 2014

Common Android Navigation Patterns

Stack Navigation

Tab Navigation



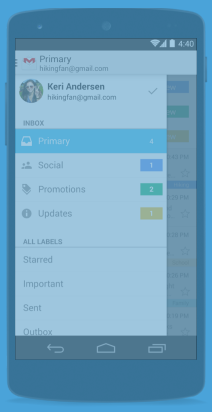
Navigation Drader

Xamarin Evolve 2014

Common Android Navigation Patterns

Stack Navigation

Tab Navigation



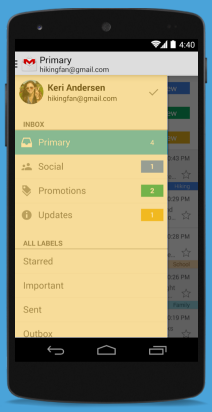
Navigation Drader

Xamarin Evolve 2014

Common Android Navigation Patterns

Stack Navigation

Tab Navigation



Navigation Drawer

Xamarin Evolve 2014

Android 4+ and the Action Bar

Action "Ice Cream" Bar

- Android 4.0 brings the Action Bar to phones

- The Action Bar provides context sensitive navigation and menus

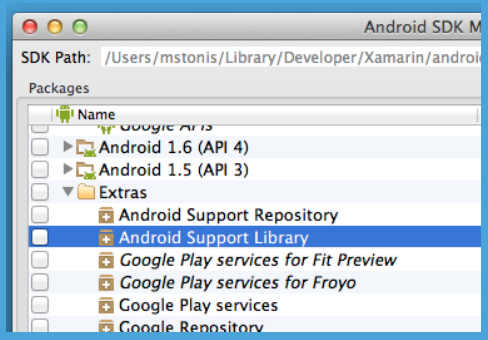
- The Action Bar can be added to Android devices all the way back to 2.1

Xamarin Evolve 2014

Android Support Library

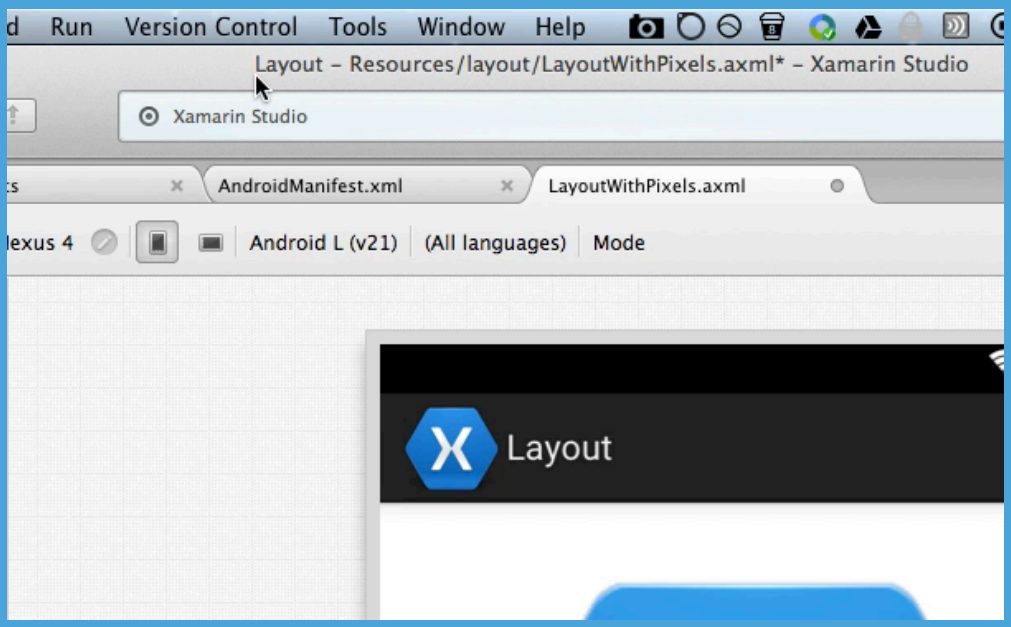
New Android Features for Old Devices

- Brings new Features to Android 1.6+



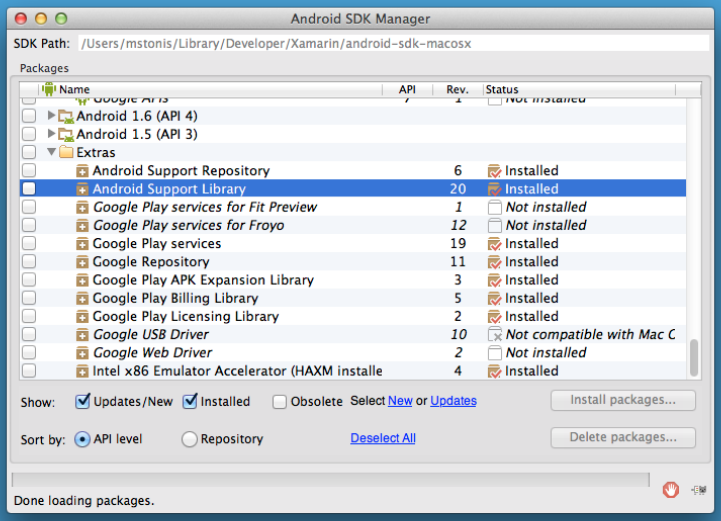
- New Components Include
  - Action Bar
  - Fragments
  - Rich Notifications

- Early Design and Pre-Release Features

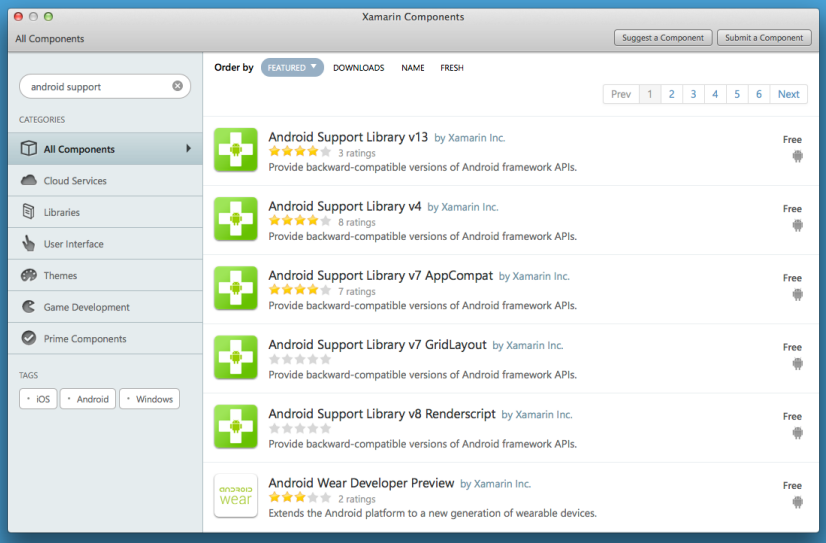


Xamarin Evolve 2014

Android Support Library



Android Support Library



Stack Navigation

Xamarin Evolve 2014

Stack Navigation

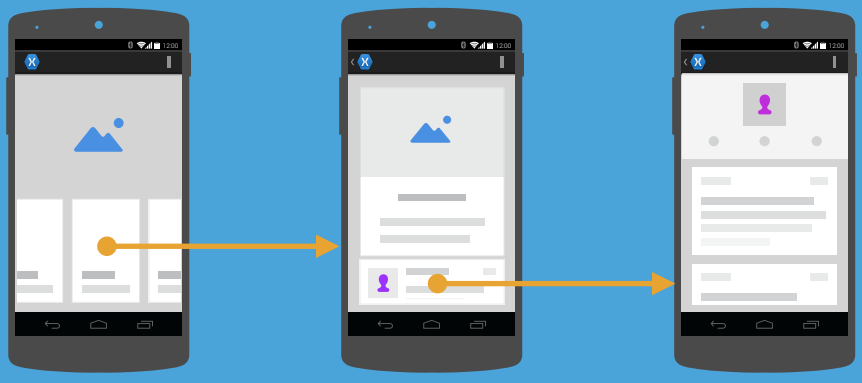
The standard navigation

- Forward Navigation
- Back Navigation
- Up Navigation

Xamarin Evolve 2014

Forward Navigation

Forward Navigation



Forward Navigation

Moving From One Interface to Another

- Navigating forward will add the previous activity onto the back stack
- This is most commonly performed using an *Intent*

Starting an Activity with an Intent

```
var intent = new Intent(this, typeof(SessionActivity));
intent.PutExtra("Title", title);
StartActivity(intent);
```

Starting an Activity with an Intent

```
var intent = new Intent(this, typeof(SessionActivity));
intent.PutExtra("Title", title);
StartActivity(intent);
```


Starting an Activity with an Intent

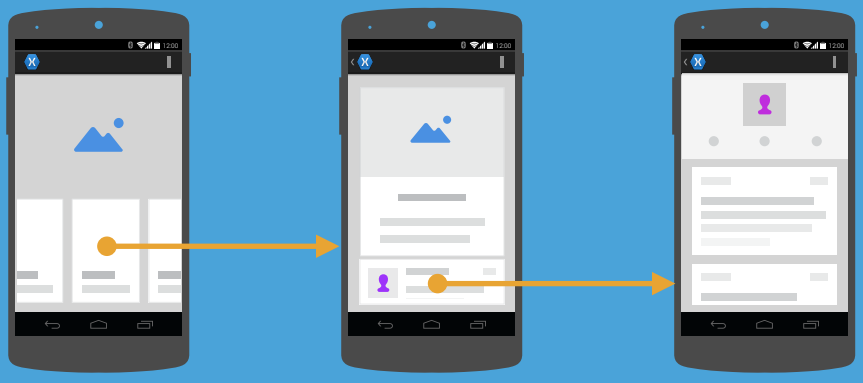
```
var intent = new Intent(this, typeof(SessionActivity));
intent.PutExtra("Title", title);
StartActivity(intent);
```

Xamarin Evolve 2014

Back Navigation

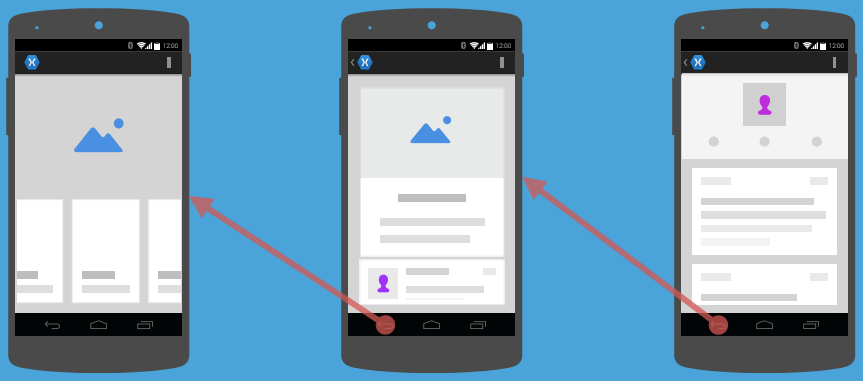
Xamarin Evolve 2014

Back Navigation



Xamarin Evolve 2014

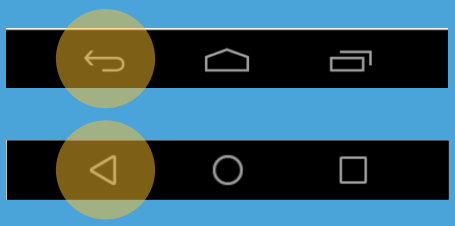
Back Navigation



Back Navigation

Moving back to the last Activity

- All Android devices provide either hardware or software-rendered back buttons



- Screen progression tracked by default, so no additional work to implement

Up Navigation

Xamarin Evolve 2014

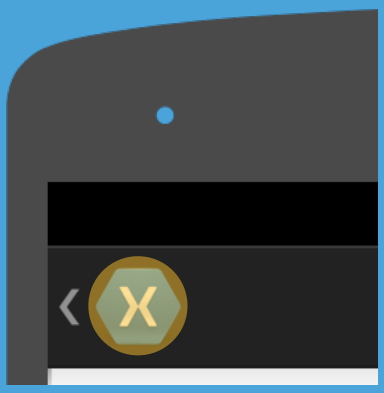
Up Navigation

Xamarin Evolve 2014

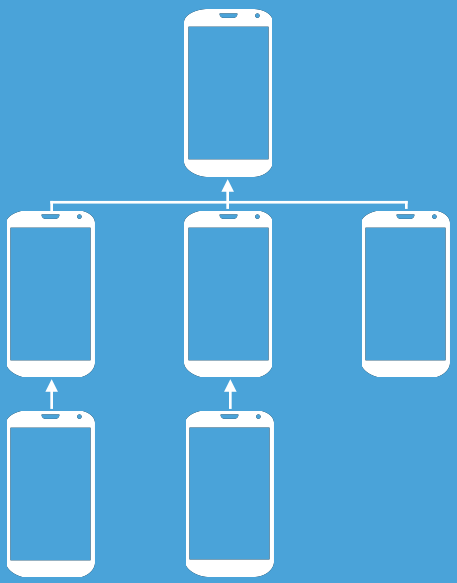
Up Navigation

Navigating Back Up the Application's Hierarchy

- Navigation back to your App's main or home screen
- Consistent navigation for your applications hierarchy
- Your interface will need an Action Bar to properly implement it



App Hierarchy



Specifying Activity Parents

```
[Activity (Label = "Session", ParentActivity = typeof(SessionsActivity))]
//Add Metadata to support Android 4.0 and lower
[MetaData("android.support.PARENT_ACTIVITY", Value = “com.xamarin.university.SessionsActivity")]
public class SessionActivity : Activity
{
  protected override void OnCreate (Bundle bundle)
  {
```


Specifying Activity Parents

```
[Activity (Label = "Session", ParentActivity = typeof(SessionsActivity))]
//Add Metadata to support Android 4.0 and lower
[MetaData("android.support.PARENT_ACTIVITY", Value = “com.xamarin.university.SessionsActivity")]
public class SessionActivity : Activity
{
  protected override void OnCreate (Bundle bundle)
  {
```

Xamarin Evolve 2014

Adding Up Navigation

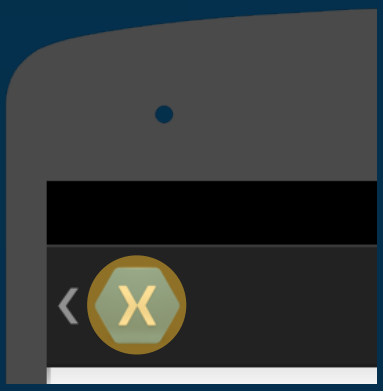
```
protected override void OnCreate (Bundle bundle)
{
  base.OnCreate (bundle);
  SetContentView (Resource.Layout.Main);

  ActionBar.SetDisplayHomeAsUpEnabled (true);
  ActionBar.SetHomeButtonEnabled (true);
}
```

Responding to Up Navigation

```
public override bool OnOptionsItemSelected (IMenuItem item)
{
  switch (item.ItemId) {
    case Android.Resource.Id.Home:
      NavUtils.NavigateUpFromSameTask (this);
    default:
      break;
  }
}
```

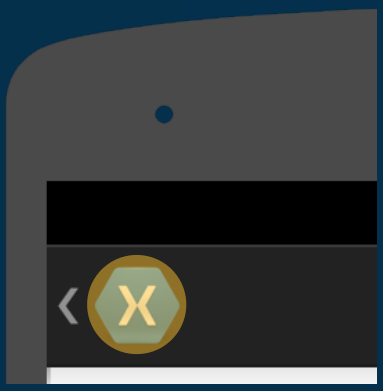
Responding to Up Navigation

```
public override bool OnOptionsItemSelected (IMenuItem item)
{
  switch (item.ItemId) {
    case Android.Resource.Id.Home:
      NavUtils.NavigateUpFromSameTask (this);
    default:
      break;
  }
}
```



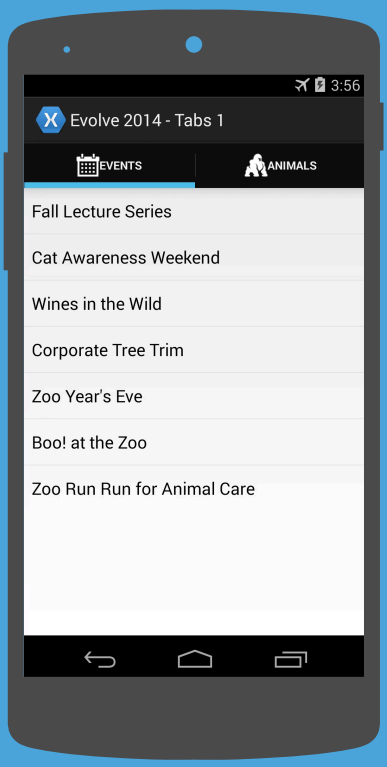
Responding to Up Navigation

```
public override bool OnOptionsItemSelected (IMenuItem item)
{
  switch (item.ItemId) {
    case Android.Resource.Id.Home:
      NavUtils.NavigateUpFromSameTask (this);
    default:
      break;
  }
}
```



Tab Navigation

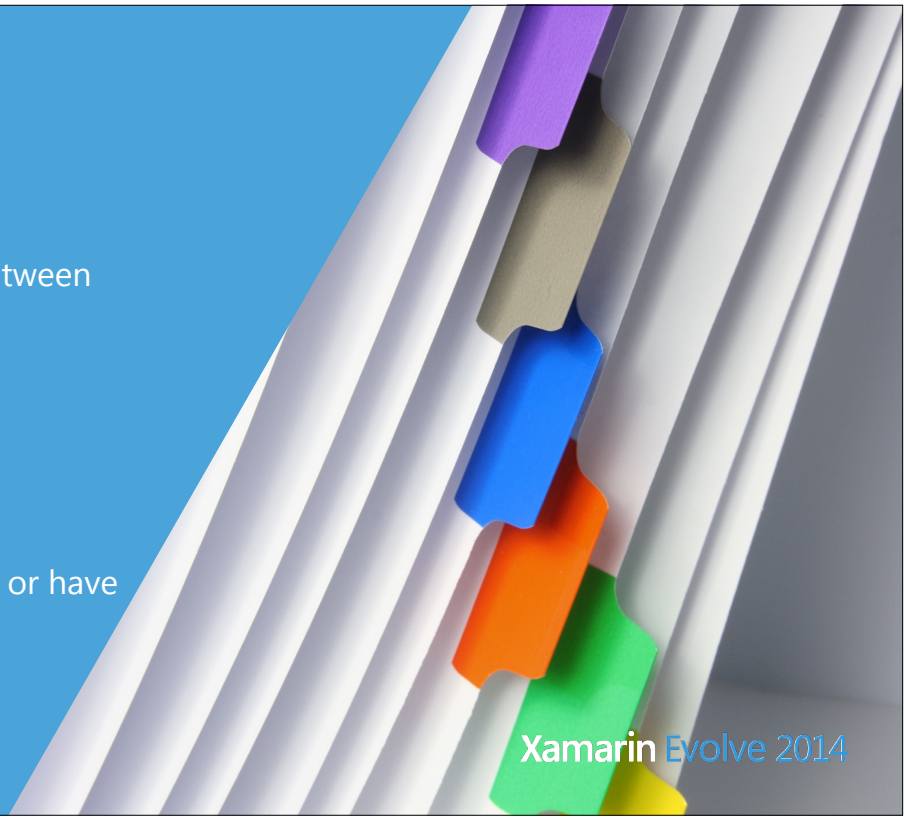
Demo



Tab Navigation

The quick navigation

- Quick, persistent navigation between application functionality
- Intended for 3-4 content tabs
- Will need to target Android 4+ or have the support library installed



Tab Navigation

Required Steps

- Implement the ITabListener Interface
- Add tabs to our Action Bar
- Listen for tab switched tabs

Adding Tab Functionality

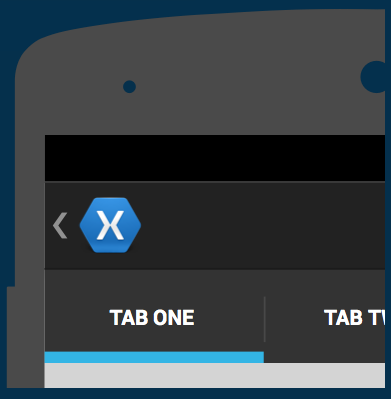
```
public class MainActivity : ActionBarActivity, ActionBar.ITabListener
{

  public void OnTabSelected (ActionBar.Tab tab, FragmentTransaction ft)
  {
    Log.Debug (Tag, "The tab {0} as been selected.", tab.Text);
  }

  public void OnTabUnselected (ActionBar.Tab tab, FragmentTransaction ft)
  {
    Log.Debug (Tag, "The tab {0} as been unselected.", tab.Text);
  }

  public void OnTabReselected (ActionBar.Tab tab, FragmentTransaction ft)
  {
    Log.Debug (Tag, "The tab {0} was re-selected.", tab.Text);
  }
```

Adding Tab Functionality

```
public class MainActivity : ActionBarActivity, ActionBar.ITabListener
{

  public void OnTabSelected (ActionBar.Tab tab, FragmentTransaction ft)
  {
    Log.Debug (Tag, "The tab {0} as been selected.", tab.Text);
  }

  public void OnTabUnselected (ActionBar.Tab tab, FragmentTransaction ft)
  {
    Log.Debug (Tag, "The tab {0} as been unselected.", tab.Text);
  }

  public void OnTabReselected (ActionBar.Tab tab, FragmentTransaction ft)
  {
    Log.Debug (Tag, "The tab {0} was re-selected.", tab.Text);
  }
```

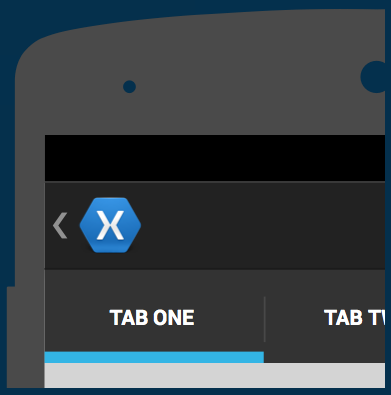

Adding Tabs

```
void AddTabToActionBar (int labelResourceId, int iconResourceId)
{
  var tab =
    ActionBar.NewTab ()
        .SetText (labelResourceId)
        .SetIcon (iconResourceId)
        .SetTabListener (this);


  ActionBar.AddTab (tab);
}
```



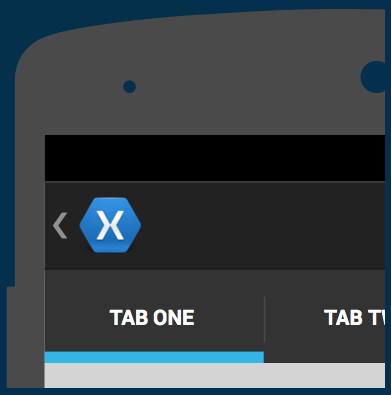
Adding Tabs

```
void AddTabToActionBar (int labelResourceId, int iconResourceId)
{
  var tab =
    ActionBar.NewTab ()
        .SetText (labelResourceId)
        .SetIcon (iconResourceId)
        .SetTabListener (this);


  ActionBar.AddTab (tab);
}
```



Adding Tabs

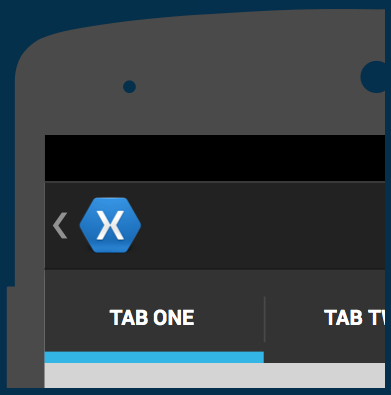
```
void AddTabToActionBar (int labelResourceId, int iconResourceId)
{
  var tab =
    ActionBar.NewTab ()
        .SetText (labelResourceId)
        .SetIcon (iconResourceId)
        .SetTabListener (this);


  ActionBar.AddTab (tab);
}
```



Xamarin Evolve 2014

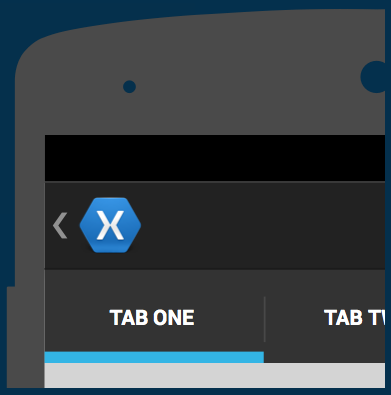
Adding Tabs

```
void AddTabToActionBar (int labelResourceId, int iconResourceId)
{
  var tab =
    ActionBar.NewTab ()
        .SetText (labelResourceId)
        .SetIcon (iconResourceId)
        .SetTabListener (this);

  ActionBar.AddTab (tab);
}
```



Adding Tabs

```
void AddTabToActionBar (int labelResourceId, int iconResourceId)
{
  var tab =
    ActionBar.NewTab ()
        .SetText (labelResourceId)
        .SetIcon (iconResourceId)
        .SetTabListener (this);

  ActionBar.AddTab (tab);
}
```



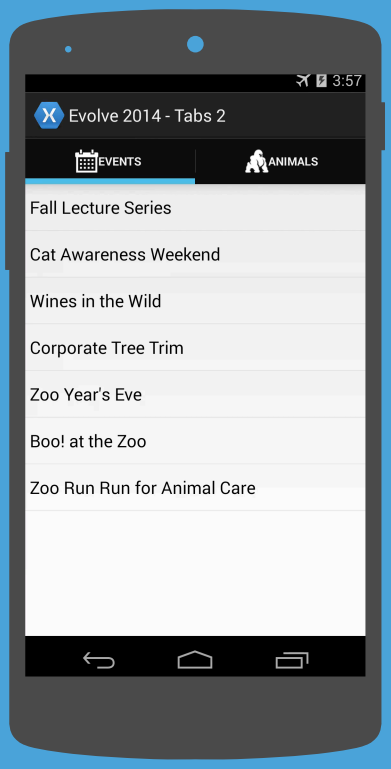
Xamarin Evolve 2014

Switching Tabs

```
public void OnTabReselected (ActionBar.Tab tab, FragmentTransaction ft)
{
  Log.Debug (Tag, "The tab {0} was re-selected.", tab.Text);
}

public void OnTabSelected (ActionBar.Tab tab, FragmentTransaction ft)
{
  SupportFragmentManager.PopBackStackImmediate ();

  Fragment frag = _fragments [tab.Position];
  ft.Replace (Resource.Id.content_frame, frag);

  Log.Debug (Tag, "The tab {0} has been selected.", tab.Text);
}

public void OnTabUnselected (ActionBar.Tab tab, FragmentTransaction ft)
{
  // perform any extra work associated with saving fragment state here.
  Log.Debug (Tag, "The tab {0} as been unselected.", tab.Text);
}
```


Demo

Xamarin Evolve 2014



Xamarin Evolve 2014

ViewPager for Tabs

Required Steps

- Add a ViewPager element to your Layout XML
- Create an Adapter to host your different content
- Implement the IOnPageChangeListener Interface

Building a Better Tab Host

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.view.ViewPager
  xmlns:android="http://schemas.android.com/apk/res/android"
    android:minWidth="25px"
    android:minHeight="25px"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/content_frame" />
```

Create a tab Adapter

```
class TabAdapter : FragmentStatePagerAdapter
{

  Fragment[] _fragments;

  public TabAdapter(FragmentManager fragmentManager) : base(fragmentManager){

    _fragments = new Fragment[] { /* List of Fragments */ } ;
  }

  public override int Count {
    get { return _fragments.Length; }
  }

  public override Fragment GetItem (int position)
  {
    if (position < 0 ||  position > _fragments.Length - 1)
      return null;

    return _fragments [position];
  }

}
```


Create a tab Adapter

```
class TabAdapter : FragmentStatePagerAdapter
{

  Fragment[] _fragments;

  public TabAdapter(FragmentManager fragmentManager) : base(fragmentManager){

    _fragments = new Fragment[] { /* List of Fragments */ } ;
  }

  public override int Count {
    get { return _fragments.Length; }
  }

  public override Fragment GetItem (int position)
  {
    if (position < 0 ||  position > _fragments.Length - 1)
      return null;

    return _fragments [position];
  }

}
```

Xamarin Evolve 2014

Implement ViewPager in Activity

```
public class MainActivity : ActionBarActivity, ActionBar.ITabListener, ViewPager.IOnPageChangeListener
{

  public void OnPageScrollStateChanged (int state)
  {
  }

  public void OnPageScrolled (int position, float positionOffset, int positionOffsetPixels)
  {
  }

  public void OnPageSelected (int position)
  {
    ActionBar.SetSelectedNavigationItem (position);
  }
```


Implement ViewPager in Activity

```
public class MainActivity : ActionBarActivity, ActionBar.ITabListener, ViewPager.IOnPageChangeListener
{

  public void OnPageScrollStateChanged (int state)
  {
  }

  public void OnPageScrolled (int position, float positionOffset, int positionOffsetPixels)
  {
  }

  public void OnPageSelected (int position)
  {
    ActionBar.SetSelectedNavigationItem (position);
  }
```

Xamarin Evolve 2014

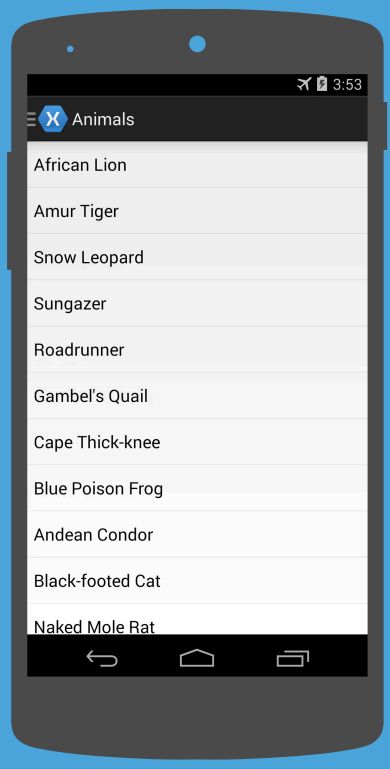
Update the Tab Listeners

```
public void OnTabSelected (ActionBar.Tab tab, FragmentTransaction ft)
{
  viewPager.SetCurrentItem (tab.Position, true);
}
```


Navigation Drawer

Demo

Xamarin Evolve 2014



Xamarin Evolve 2014

Navigation Drawer

The hidden navigation

- Fast, non-intrusive navigation between main application pages
- Intended for 3+ main pages
- Can be used for navigation and for contextual menus

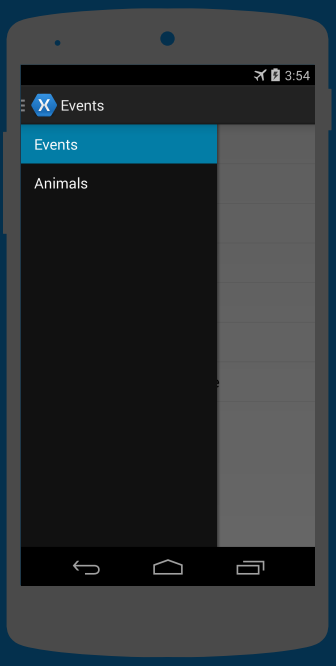
Navigation Drawer

Required Steps

- Update our application's layout to support a navigation drawer
- Configure the navigation drawer in our activity
- Synchronize our Navigation Drawer
- Respond to drawer selections

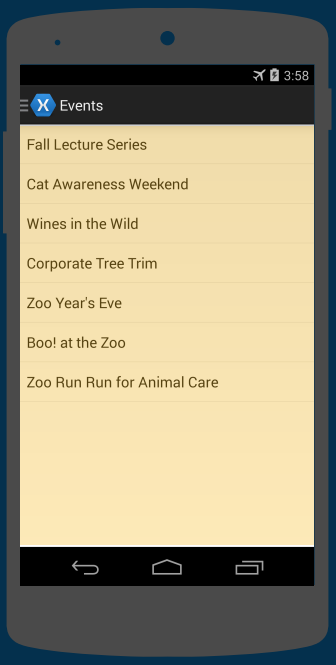
Layout Setup

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.DrawerLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:local="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <FrameLayout
        android:id="@+id/content_frame"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <ListView
        android:id="@+id/flyout"
        android:divider="@android:color/transparent"
        android:dividerHeight="0dp"
        android:choiceMode="singleChoice"
        android:layout_width="240dp"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        android:background="#111" />

</android.support.v4.widget.DrawerLayout>
```



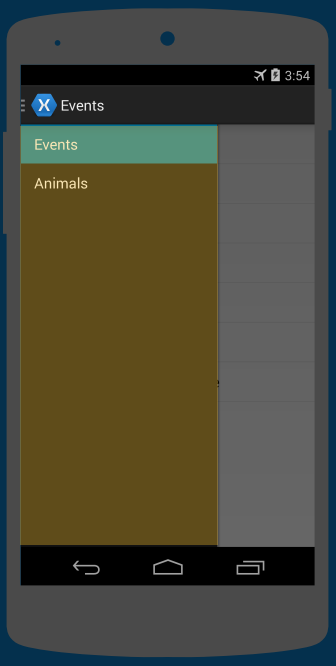
Layout Setup

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.DrawerLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:local="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <FrameLayout
        android:id="@+id/content_frame"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <ListView
        android:id="@+id/flyout"
        android:divider="@android:color/transparent"
        android:dividerHeight="0dp"
        android:choiceMode="singleChoice"
        android:layout_width="240dp"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        android:background="#111" />

</android.support.v4.widget.DrawerLayout>
```



Layout Setup

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.DrawerLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:local="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <FrameLayout
        android:id="@+id/content_frame"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <ListView
        android:id="@+id/flyout"
        android:divider="@android:color/transparent"
        android:dividerHeight="0dp"
        android:choiceMode="singleChoice"
        android:layout_width="240dp"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        android:background="#111" />

</android.support.v4.widget.DrawerLayout>
```

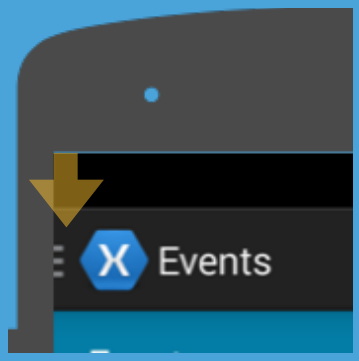


Drawer Setup

```
public class MainActivity : ActionBarActivity, IFragmentNavigation
{
  DrawerLayout drawerLayout;
  ActionBarDrawerToggle drawerToggle;

  protected override void OnCreate (Bundle bundle)
  {
    base.OnCreate (bundle);
    SetContentView (Resource.Layout.Main);

    drawerLayout = FindViewById<DrawerLayout> (Resource.Id.drawer_layout);

    drawerToggle = new ActionBarDrawerToggle (this,
      drawerLayout, Resource.Drawable.ic_drawer,
      Resource.String.drawer_open, Resource.String.drawer_close);

    drawerLayout.SetDrawerShadow (Resource.Drawable.drawer_shadow, (int)GravityFlags.Start);

    drawerLayout.SetDrawerListener (drawerToggle);
```

Navigation Drawer

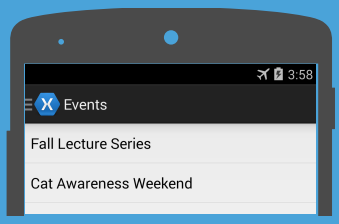
Closed



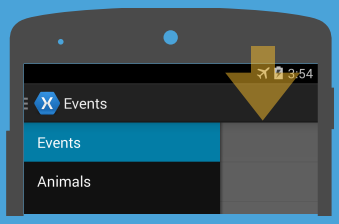
Open

Drawer Setup

```
protected override void OnCreate (Bundle bundle)
{
  base.OnCreate (bundle);
  SetContentView (Resource.Layout.Main);

  drawerToggle = new ActionBarDrawerToggle (this,
    drawerLayout, Resource.Drawable.ic_drawer,
    Resource.String.drawer_open, Resource.String.drawer_close);

  drawerLayout = FindViewById<DrawerLayout> (Resource.Id.drawer_layout);
  drawerLayout.SetDrawerShadow (Resource.Drawable.drawer_shadow, (int)GravityFlags.Start);
  drawerList = FindViewById<ListView> (Resource.Id.flyout);
  drawerList.Adapter = new ArrayAdapter<string> (this, Resource.Layout.drawer_list_item, sections);


  drawerLayout.SetDrawerListener (drawerToggle);
```



Closed



Open

Drawer Setup

```
protected override void OnCreate (Bundle bundle)
{
  base.OnCreate (bundle);
  SetContentView (Resource.Layout.Main);

  drawerToggle = new ActionBarDrawerToggle (this,
    drawerLayout, Resource.Drawable.ic_drawer,
    Resource.String.drawer_open, Resource.String.drawer_close);

  drawerLayout = FindViewById<DrawerLayout> (Resource.Id.drawer_layout);
  drawerLayout.SetDrawerShadow (Resource.Drawable.drawer_shadow, (int)GravityFlags.Start);
  drawerList = FindViewById<ListView> (Resource.Id.flyout);
  drawerList.Adapter = new ArrayAdapter<string> (this, Resource.Layout.drawer_list_item, sections);


  drawerLayout.SetDrawerListener (drawerToggle);
```

Make sure the Action Bar is Enabled

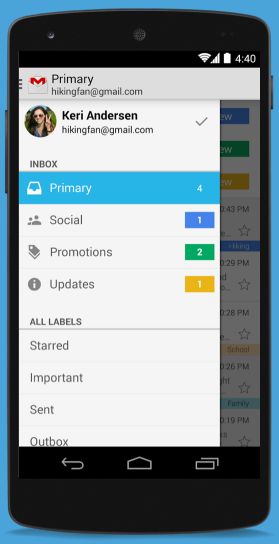
```
  drawerToggle = new ActionBarDrawerToggle (this,
    drawerLayout, Resource.Drawable.ic_drawer,
    Resource.String.drawer_open, Resource.String.drawer_close);

  drawerLayout.SetDrawerListener (drawerToggle);

  drawerList.ItemClick += (object sender, AdapterView.ItemClickEventArgs e) => ListItemClicked
(e.Position);
  ListItemClicked (0);

  ActionBar.SetDisplayHomeAsUpEnabled (true);
  ActionBar.SetHomeButtonEnabled (true);
}
```


Responding to Selections

```
public override bool OnOptionsItemSelected (IMenuItem item)
{
  if(drawerToggle.DrawerIndicatorEnabled && drawerToggle.OnOptionsItemSelected(item))
    return true;

    return base.OnOptionsItemSelected (item);
}
```

Quick Tips

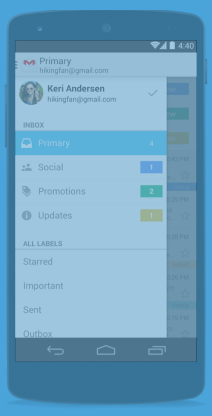
- On first run, consider starting your app with the Navigation Drawer displayed
- Make your drawer listings descriptive
  - Icons
  - Images
  - Counters
- The Android Developer Center has Navigation Drawer Icons



Common Android Navigation Patterns

Stack Navigation

Tab Navigation

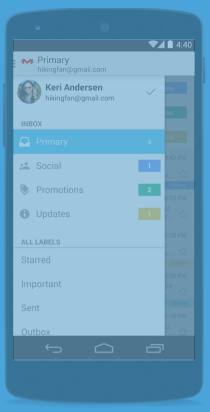


Navigation Drawer

Common Android Navigation Patterns

Stack Navigation

Tab Navigation



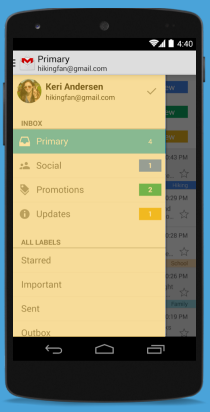
Navigation Drawer

Xamarin Evolve 2014

Common Android Navigation Patterns

Stack Navigation

Tab Navigation



Navigation Drawer

Xamarin Evolve 2014

Questions?

Xamarin Evolve 2014

Xamarin Evolve 2014

Homeward Bound: Implementing Android Navigation Patterns

Michael Stonis
@michaelstonis