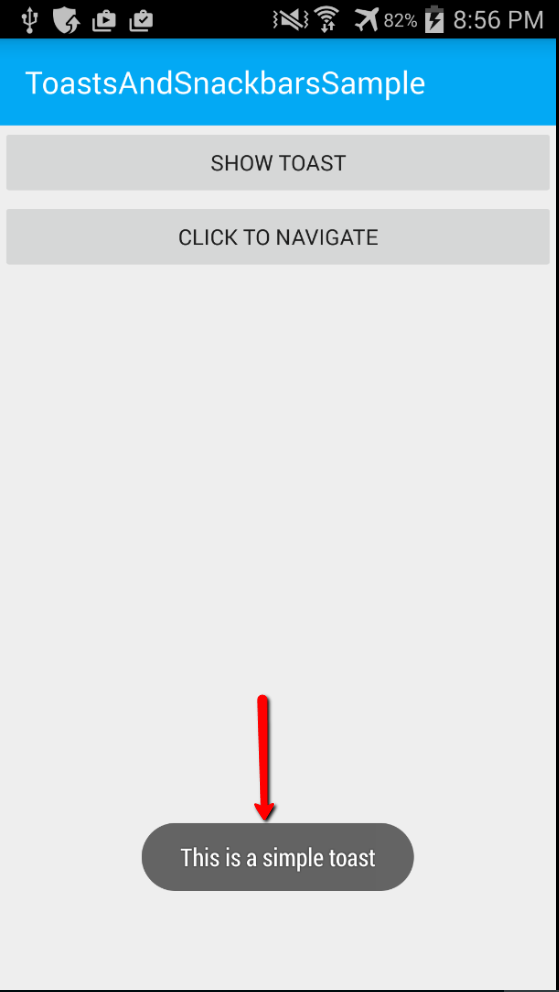
Day 24: Toasts and Snackbars

Today, we will explore common ways to show feedback to users in Android. The most common way in Android being Toasts and Snackbars is the new way of showing alerts to users that can be acted upon. Snackbars require the Design Support Library.

# Toasts

Toast provides a simple way to show feedback to users after an operation is complete. Toast appears on the bottom of the screen, only taking up the same that is required for filling the message. Let’s see how a simple Toast looks like –



Let’s see how to create this Toast –

We will be calling the **MakeText** static method on the Toast class from the Android.Widget namespace to make a Toast. The MakeText method takes 3 parameters: First, it requires the current Android Context (this), then the string of text to be shown to the user as feedback and finally, how long the Toast should be visible. The length of the duration of Toast is controlled by the ToastLength enum, which has Long and Short as the options. Let’s see code in action now –

|  |
| --- |
| using System;  using Android.App;  using Android.OS;  using Android.Widget;  namespace ToastsAndSnackbarsSample  {  [Activity(Label = "ToastsAndSnackbarsSample", MainLauncher = true, Icon = "@drawable/icon")]  public class MainActivity : BaseActivity  {  protected override int LayoutResource  {  get { return Resource.Layout.main; }  }  protected override void OnCreate(Bundle bundle)  {  base.OnCreate(bundle);  // Get our button from the layout resource,  // and attach an event to it  var showToastButton = FindViewById<Button>(Resource.Id.showToastButton);  showToastButton.Click += ShowToastButton\_Click;  SupportActionBar.SetDisplayHomeAsUpEnabled(false);  SupportActionBar.SetHomeButtonEnabled(false);  }  private void ShowToastButton\_Click(object sender, EventArgs e)  {  Toast.MakeText(this, "This is a simple toast", ToastLength.Long).Show();  }  }  } |

Gist file link: <https://gist.github.com/vkoppaka/491caccfc8c2fb1040a5>

Finally, to show the Toast, we call .Show on the Toast that we just created.

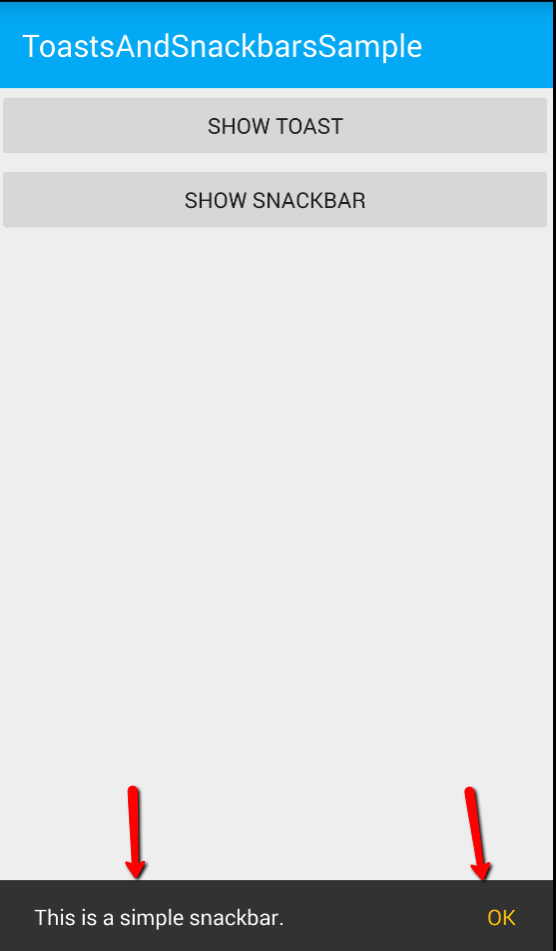
One drawback on Toasts, is that they are purely informational. You cannot act on a Toast and do additional actions with it. Which is where our next topic, Snackbars come in.

# Snackbars

Snackbars are part of the Android Design Support Library. To use Snackbars, or for that matter any widget from the support design library, you need to first install the Xamarin Android Support Design Library

|  |
| --- |
| Install-Package Xamarin.Android.Support.Design |

Once installed, we will be able to use Snackbars. Let’s see how a Snackbar based notification (feedback) looks like –



Let’s see how to create this Snacbar –

We will be calling the **Make** static method on the Snackbar class from the Android.Support.Design.Widget namespace to make a Snackbar. The Make method takes 3 parameters: First, it requires the Android View, the parent view, on where the Snackbar would be displaying from. Then the string of text to be shown to the user as feedback and finally, how long the Snackbar should be visible. The length of the duration of Snackbar is controlled by the Snackbar.LengthLong or Snackbar.LengthShort constants. What sets the Snackbar apart from Toast is the **SetAction** method which lets us set actionable buttons on the Snackbar where we can call back to our code to either undo an item or doing anything we like.

Let’s see code in action now –

|  |
| --- |
| using System;  using Android.App;  using Android.OS;  using Android.Support.Design.Widget;  using Android.Widget;  namespace ToastsAndSnackbarsSample  {  [Activity(Label = "ToastsAndSnackbarsSample", MainLauncher = true, Icon = "@drawable/icon")]  public class MainActivity : BaseActivity  {  protected override int LayoutResource  {  get { return Resource.Layout.main; }  }  protected override void OnCreate(Bundle bundle)  {  base.OnCreate(bundle);  // Get our button from the layout resource,  // and attach an event to it  var snackbarButton = FindViewById<Button>(Resource.Id.showSnackbarButton);  snackbarButton.Click += SnackbarButton\_Click;  SupportActionBar.SetDisplayHomeAsUpEnabled(false);  SupportActionBar.SetHomeButtonEnabled(false);  }  private void SnackbarButton\_Click(object sender, EventArgs e)  {  var linearLayout = FindViewById<LinearLayout>(Resource.Id.main\_content);  Snackbar.Make(linearLayout, "This is a simple snackbar.", Snackbar.LengthLong)  .SetAction("OK", action => { })  .Show();  }  }  } |

Gist file link: <https://gist.github.com/vkoppaka/b774dde34fa97c558406>

Finally, to show the Snackbar, we call .Show on the Snackbar that we just created.

That’s it for today, see you all tomorrow.