Show a Continuous Progress Bar While Processing on a Background Thread

Start

```
//File:Window.xaml.cs
using System.ComponentModel;
using System.Threading;
using System.Windows;
using System.Windows.Input;

namespace WpfApplication1
{
public partial class Window1 : Window
{
private BackgroundWorker worker = new BackgroundWorker();

public Window1()
{
InitializeComponent();
worker.WorkerSupportsCancellation = true;
worker.DoWork += new DoWorkEventHandler(worker_DoWork);
worker.RunWorkerCompleted += new
RunWorkerCompletedEventHandler(worker_RunWorkerCompleted);
}

private void button_Click(object sender, RoutedEventArgs e)
{
if (!worker.IsBusy)
```

```
{
this.Cursor = Cursors.Wait;
progressBar.IsIndeterminate = true;
button.Content = “Cancel”;
worker.RunWorkerAsync();
}
else
{
worker.CancelAsync();
}
}

private void worker_RunWorkerCompleted(object sender,
RunWorkerCompletedEventArgs e)
{
this.Cursor = Cursors.Arrow;

if (e.Error != null)
{
MessageBox.Show(e.Error.Message);
}

button.Content = “Start”;
progressBar.IsIndeterminate = false;
}

private void worker_DoWork(object sender, DoWorkEventArgs e){
for (int i = 1; i <= 100; i++) { if
(worker.CancellationPending) break; Thread.Sleep(50); } } } }
[/csharp]
```