Xamarin Android - Usando a classe Timer

Neste artigo vou mostrar como usar a classe **Timer** do namespace **System.Timers** em um aplicação Xamarin Android usando o Visual Studio 2015 e a linguagem C#.

O namespace **System.Timers** fornece o componente **Timer**, que permite disparar um evento em um intervalo especificado.

O componente **Timer** é um temporizador baseado no servidor, que permite especificar um intervalo recorrente no qual o evento **Timer.Elapsed** é gerado no seu aplicativo.

O Timer baseado em servidor foi projetado para uso com threads de trabalho em um ambiente multithreaded. Temporizadores de servidor podem se mover entre threads para manipular o evento **Timer.Elapsed** disparado, resultando em mais precisão do que o componente Timer do Windows em levantar o evento no tempo.

As principais propriedades do componente **Timer** são:

Propriedade	Descrição
AutoReset	Obtem ou define um valor indicando se o Timer irá disparar o evento Timer.Elapsed a cada vez que intervalo especificado decorrer ou somente após a primeira vez.
Enabled	Obtém ou define um valor indicando se o Timer irá disparar o evento Timer.Elapsed.
Interval	Obtém ou define o intervalo no qual o evento Timer.Elapsed será disparado.

O componente possui o evento **Elapsed** que ocorre quando um intervalo decorre.

Recursos usados:

- Visual Studio Community 2017 ou Xamarin Studio
- Xamarin
- Emulador Android virtual ou físico (veja como emular usando o Vysor)

Criando o projeto no VS Community 2017

Abra o VS 2017 Community e clique em **New Project;**

Selecione a linguagem Visual C# e o template **Android -> Blank App(Android)**

Informe o nome **Droid_Timer** e clique no botão **OK**;

Abra o arquivo **Main.axml** na pasta **Resources/layout** e no modo **Designer.**

A seguir inclua os seguintes controles a partir da **ToolBox**:

- **1 TextView - id = txtvTitulo**
- **1 CheckBox - id = chkAtivar**
- **1 Button - id = btnInicia**
- **1 Button - id = btnCancela**
- **1 TextView - id = txtvContador**

Abaixo vemos o leiaute no emulador do Xamarin e ao lado o respectivo código XML gerado :


```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
   android:background="#460063"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <TextView
      android:id="@+id/txtvTitulo"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:gravity="center_horizontal"
      android:autoLink="web"
      android:textStyle="bold"
      android:text="macoratti.net" />
   <CheckBox
      android:id="@+id/chkAtivar"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Ativar o Relógio" />
   <Button
      android:id="@+id/btnInicia"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="Iniciar" />
   <Button
      android:id="@+id/btnCancela"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="Cancelar" />
   <TextView
      android:id="@+id/txtvContador"
      android:textSize="25sp"
      android:background="@android:color/holo_orange_dark"
      android:textColor="@android:color/Black"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:gravity="center_horizontal"
      android:text="Relógio" />
</LinearLayout>
```

Agora vamos definir o código no arquivo **MainActivity.cs** vinculado a nossa view **Main.axml**.

Abra o arquivo **MainActivity.cs** e altere o código desse arquivo conforme abaixo:

```
using Android.App;
using Android.Widget;
using Android.OS;
using System.Timers;
using System;

namespace Droid_Timer
{
    [Activity(Label = "Droid_Timer", MainLauncher = true, Icon = "@drawable/icon")]
    public class MainActivity : Activity
    {
        Timer timer = null;
        TextView txtContador = null;

        protected override void OnCreate(Bundle bundle)
        {
            base.OnCreate(bundle);

            // Set our view from the "main" layout resource
            SetContentView (Resource.Layout.Main);

            txtContador = FindViewById<TextView>(Resource.Id.txtvContador);
            var btnIniciar = FindViewById<Button>(Resource.Id.btnInicia);
            var btnCancelar = FindViewById<Button>(Resource.Id.btnCancela);
            var chkAtivar = FindViewById<CheckBox>(Resource.Id.chkAtivar);

            btnIniciar.Click += delegate
            {
                timer = new Timer();
                if (chkAtivar.Checked == true)
                {
                    timer.Interval = 1000;
                    timer.Elapsed += Timer_Elapsed;
                    timer.Start();
                }
            };

            btnCancelar.Click += BtnCancelar_Click;
        }

        private void BtnCancelar_Click(object sender, EventArgs e)
        {
            Cancelar();
        }

        private void Timer_Elapsed(object sender, ElapsedEventArgs e)
        {
            DateTime dt = DateTime.Now;
            string formato = "dd/MM/yyyy HH:mm:ss";
            RunOnUiThread(() => { txtContador.Text = dt.ToString(formato); });
        }

        private void Cancelar()
        {
            timer.Enabled = false;
            timer.Dispose();
        }
    }
}
```

Vamos entender o código nos concentrando apenas no controle Timer usado:

1- Definimos uma variável do tipo Timer :

```
Timer timer = null;
```

3- No evento **Click** do botão - **btnIniciar** - definimos o código que vai verificar se o **Timer** será ativado ou não.

Na ativação definimos a propriedade Interval igual a 1000 milisegundos (1 s) e o evento **Elapsed** que vai acionar o método **Timer_Elapsed**;

```
btnIniciar.Click += delegate
{
    timer = new Timer();
    if (chkAtivar.Checked == true)
    {
        timer.Interval = 1000;
        timer.Elapsed += Timer_Elapsed;
        timer.Start();
    }
};
```

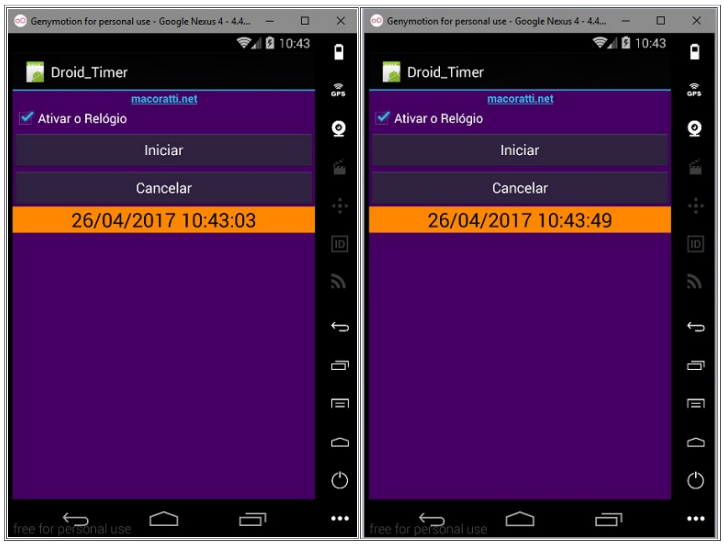
3 - No código do método **Timer_Elapsed** definimos a data e hora atual para ser exibida no formato especificado na TextView **txtContador**:

```
private void Timer_Elapsed(object sender, ElapsedEventArgs e)
{
    DateTime dt = DateTime.Now;
    string formato = "dd/MM/yyyy HH:mm:ss";
    RunOnUiThread(() => { txtContador.Text = dt.ToString(formato); });
}
```

Observe que usamos o método **RunOnUIThread**() usando expressões lambdas para exibir as mensagens. Esse método executa a ação especificada na thread da UI. Se a thread atual for a thread da UI, então a ação é executada imediatamente. Se a thread atual não for a thread da interface do usuário, a ação é enviada para a fila de eventos da thread.

Executando o projeto usando o emulador do **Xamarin Android Player** e emulando o **Genymotion** iremos obter o seguinte resultado:

Na primeira figura ativamos o relógio usando o Timer e na segunda ao clicar no botão **Cancelar** paramos a execução do Timer:



Este é um exemplo bem básico que mostra o uso da classe **Timer** em uma aplicação Xamarin Android.

Pegue o projeto aqui : **Droid_Timer.zip** (sem as referências)

"Em quem (Jesus) temos a redenção pelo seu sangue, a remissão das ofensas, segundo as riquezas da sua graça," Efésios 1:7

Referências:

- Seção VB .NET do Site Macoratti.net
- Super DVD .NET - A sua porta de entrada na plataforma .NET
- Super DVD Vídeo Aulas - Vídeo Aula sobre VB .NET, ASP .NET e C#
- Super DVD C# - Recursos de aprendizagens e vídeo aulas para C#
- Seção C# do site Macoratti.net
- Seção ASP .NET do site Macoratti .net
- Curso Básico VB .NET - Vídeo Aulas
- Curso C# Básico - Vídeo Aulas
- Curso Fundamentos da Programação Orientada a Objetos com VB .NET NEW
- **Macoratti .net | Facebook**
- **macoratti - YouTube**
- **Jose C Macoratti (@macorati) | Twitter**
- VB.NET 2005 - Controles - Macoratti.net
- Seção de Jogos do site Macoratti .net
- Xamarim - Desenvolvimento Multiplataforma com C# ... - Macoratti.net
- Xamarin - Apresentando Xamarin.Forms - Macoratti.net
- Xamarin.Forms - Olá Mundo - Criando sua primeira ... - Macoratti.net
- Xamarin.Forms - Olá Mundo - Anatomia da aplicação - Macoratti.net
- https://developer.xamarin.com/api/type/Android.App.AlertDialog/

- Xamarin Android - Tratando eventos de forma declarativa
- Seção Mobile/Xamarin do site Macoratti .net
- https://developer.android.com/reference/android/app/Activity.html
- https://developer.xamarin.com/api/type/Android.Widget.ProgressBar/

José Carlos Macoratti