Parallel Programming Library (PPL) com Delphi

Fernando Rizzato
Lead Software Consultant, *Latin America*

AGENDA

- Computação Paralela
- TThread class
- System.Threading unit
  - TParallel.For
  - TTask (ITask)
  - TTask.IFuture

COMPUTAÇÃO PARALELA

- Capacidade de executar tarefas em paralelo, aproveitando o processamento simultâneo em dispositivos e computadores com múltiplas **CPU**
- Tipicamente dispositivos e computadores hoje têm múltiplas **CPU**, até mesmo o seu telefone!
- Mas quando se trata de programação para obter todos os benefícios desses núcleos, é um pouco complicado ou demorado, e vai exigir código extra...
- Isso até a chegada da nova **Parallel Programming Library (PPL)** para Delphi e C++ Builder!!!

TTHREAD CLASS (SYSTEM.CLASSES.TTHREAD)

- **TThread** é uma classe abstrata que permite a criação de linhas separadas de execução em um aplicativo
- Abordagem tradicional, presente desde as primeiras versões
- Segue útil para algumas situações específicas

```
program SleepSortDemo;

{$APPTYPE CONSOLE}

uses
  Windows, SysUtils, Classes, SyncObjs;

type
  TSleepThread = class(TThread)
  private
    FValue: Integer;
    FLock: TCriticalSection;
  protected
    constructor Create(AValue: Integer; ALock: TCriticalSection);
    procedure Execute; override;
  end;
```

SYSTEM.THREADING UNIT

- **TParallel.For**
  - Execução paralela para *loop* fácil de usar

```
const
 Max = 50000; // 50K

for I := 1 to Max do
 begin
   if IsPrime (I) then
     Inc (Tot);
 end;
```

```
TParallel.For(1, Max, procedure (I: Integer)
 begin
   if IsPrime (I) then
     TInterlocked.Increment (Tot);
 end);
```

SYSTEM.THREADING UNIT

- **TTask (ITask)**
  - TTask fornece uma classe para criar e gerenciar interação com instâncias de ITask

```
procedure TForm1.Button1Click(Sender: TObject);
var
 aTask: ITask;
begin
 aTask := TTask.Create (procedure ()
   begin
     sleep (3000); // 3 seconds
     ShowMessage ('Hello');
   end);
 aTask.Start;
end;
```

SYSTEM.THREADING UNIT

- **TTask.IFuture**
  - IFuture implementa uma ITask capaz de executar em um segmento paralelo que retorna um tipo específico quando necessário. O tipo é especificado pelo parâmetro de tipo genérico T

```
FutureObject := TTask.Future<Integer>(function: Integer
    begin
        Sleep(3000);
        Result := 16;
    end);
// …
MyValue := FutureObject.Value;
```

DEMOS

RECURSOS ADICIONAIS

- Documentação
  - http://docwiki.embarcadero.com/Libraries/Berlin/en/System.Classes.TThread
  - http://docwiki.embarcadero.com/RADStudio/Berlin/en/Writing_the_Thread_Function
  - http://docwiki.embarcadero.com/RADStudio/Berlin/en/Using_the_Parallel_Programming_Library
  - http://docwiki.embarcadero.com/RADStudio/Berlin/en/Parallel_Programming_Library_Tutorials

RECURSOS ADICIONAIS

- Blogs
  - http://www.malcolmgroves.com/blog/?cat=123
  - https://delphiaball.co.uk/2014/09/05/quick-introduction-to-parallel-programming-with-xe7/
  - http://robstechcorner.blogspot.com.br/2015/02/tpl-ttask-example-in-how-not-to-use.html
  - http://www.stevemaughan.com/delphi/delphi-parallel-programming-library-memory-managers/
  - https://community.embarcadero.com/blogs/entry/developer-skill-sprints-week-3-fast-code-faster-with-parallel-programming-library

OBRIGADO!

Perguntas?

Você pode me encontrar em:
@FernandoRizzato
fernando.rizzato@embarcadero.com

Siga-nos em
fb.com/DelphiBrasil
fb.com/EmbarcaderoBR