Parallel Programming Library (PPL) con Delphi

Fernando Rizzato
Lead Software Consultant, *Latin America*

AGENDA

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

COMPUTACIÓN PARALELA

- La capacidad para realizar tareas en paralelo, aprovechando el procesamiento simultáneo en los dispositivos y ordenadores con múltiples CPU
- Normalmente los dispositivos y computadoras hoy en día tienen múltiples CPU, incluso su teléfono!
- Pero cuando se trata de la programación para obtener todos los beneficios de estos núcleos es un poco complicado, o requiere mucho tiempo, y exigirá código extra...
- Así fue hasta la llegada de la nueva Parallel Programming Library (PPL) para Delphi y C ++ Builder!

TTHREAD CLASS (SYSTEM.CLASSES.TTHREAD)

- **TThread** es una clase abstracta que permite la creación de líneas separadas que se ejecuta en una aplicación
- Enfoque tradicional - presente desde las primeras versiones
- Sigue siendo útil para algunas situaciones 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**
  - Ejecución paralela para el *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 proporciona una clase para crear y gestionar la interacción con las instancias 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 un ITask capaz de correr en un segmento paralelo que devuelve un tipo específico cuando sea necesario. El tipo es especificado por el parámetro de tipo genérico T

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

DEMOS

RECURSOS ADICIONALES

- Documentación
  - 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 ADICIONALES

- 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

GRACIAS!

Preguntas?

Me puedes encontrar en:
@FernandoRizzato
fernando.rizzato@embarcadero.com

Síguenos en
fb.com/EMBTLatAm