Kentico

Paralelní a asynchronní programování

Zdeněk Jurka

Kentico

Obsah přednášky

- Paralelní vs. Asynchronní programování
- Thread
- ThreadPool
- TPL
- Async await
- Synchronizace
- PLINQ

Kentico

Paralelní vs. Asynchronní programování

- **Paralelní programování**
  - Dělání více věcí najednou
- **Multithreading**
  - Forma paralelního programování využívající vlákna
- **Asynchronní programování**
  - Druh paralelního programování, využívajícího callbacks k redukci čekání a blokování vláken

Kentico

Thread

- Vlastní zásobník
- Sdílí haldu
- V C# reprezentováno třídou `Thread`
- Můžeme nastavit jméno, prioritu, appartment state, …
- NEPOUŽÍVAT přímo až na specifické případy ☺

Kentico

ThreadPool

- `ThreadPool` – pool (návrhový vzor object pool) vláken, recykluje vlákna
- `ThreadPool.QueueUserWorkItem` – naplánuje úlohu, pool rozhodne kdy ji pustí
- Nemůžeme nastavit vlastnosti jako u `Thread` (jméno, …)

Kentico

TPL

- Task parallel library
- Představeno v .NET 4.0
- Podpora pro task based paralelní operace
- Použito v PLINQ
- Použito v async await
- Většina IO operací ve frameworku má task based operace

Kentico

TPL - Třída Parallel

- `Parallel.Invoke` – paralelní spuštění akcí
- `Parallel.For` – paralelní cyklus for
- `Parallel.ForEach` – paralelní cyklus foreach

Kentico

TPL - Třída Task

- Reprezentuje úlohu, která může být naplánováno pro běh na jiném vlákně
- `Task.Run` - naplánuje úlohu ke běhu
- `Task<T>` - Pokud úloha vrací hodnotu
- `Task.Result` – výsledek úlohy
- `Task.Wait` – počká na ukončení úlohy

Kentico

TPL - Task factory

- Můžeme spouštět task přímo
- `TaskFactory.StartNew` – naplánuje novou úlohu
- `Task.Factory` – default factory
- Můžeme vytvořit instance s různými plánovači, parametry pro spuštění, ...

Kentico

TPL – Čekání na dokončení více úloh

- `Task.WaitAll` – metoda počká na ukončení všech úloh
- `Task.WaitAny` – metoda počká na první ukončenou úlohu
- `Task.WhenAll` – vrátí úlohu která skončí až skončí všechny
- `Task.WhenAny` – vrátí úlohu která skončí až skončí první

Kentico

TPL – Řetězení úloh

- Můžeme vytvořit řetězec úloh
- `Task.ContinueWith`

Kentico

Task – Zpracování výjimek

- Problém jak přenést výjimku zpět do původního vlákna
- `Task.Exception` - výjimka pokud nějaká nastala
- `Task.Status` je nastaven na faulted...
- `Task.Wait, Task.Result, Task.WaitAny, Task.WaitAll` - throws `AggregateException`

Kentico

TPL - Task cancelation

- `CancellationTokenSource.Token`
- `CancellationTokenSource.Cancel`
- `CancellationToken.IsCancellationRequested`
- `CancellationToken.ThrowIfCancellationRequested`
- Vytvořený token se předá do operace která podporuje zrušení
- Zrušení vede k vyjímce `OperationCanceledException`

Kentico

TPL – Signalizace průběhu operace

- `IReportProgress` – předá se operaci
- `ReportProgress` – poskytuje implementaci která eventem informuje o změně průběhu operace

Kentico

Async await

- `async await` klíčová slova
- `async` metoda musí vracet `Task/Task<T>/void`
- Měla by mít Async suffix
- Měla by vracet `Task` by měl být vracen pouze na vrcholu volání např. event handler jinak `Task<TResult>`
- `await` vyhodí první výjimku

Kentico

Synchronizační primitiva - locking

Konstrukt	Účel	Mezi procesy	Overhead
lock	Zaručí výlučný přístup k bloku kódu	-	20 ns
Mutex		Ano	1000 ns
SemaphoreSlim	Zaručí že k bloku kódu má přístup maximálně definovaný počet vláken/procesů	-	200 ns
Semaphore		Ano	1000 ns
ReaderWriterLockSlim	Umožní několik producentů a jednoho konzumenta	-	40 ns
ReaderWriterLock		-	100 ns

Kentico

Synchronizační primitiva - signalling

Konstrukt	Účel	Mezi procesy	Overhead
AutoResetEvent	Při signalizaci umožní průchod jednomu vláknu	Ano	1000 ns
ManualResetEvent	Umožní průchod dokud není explicitně resetováno	Ano	1000 ns
ManualReserEventSlim		-	40 ns
CountdownEvent	Umožní průchod po přijetí definovaného počtu signálů	-	40 ns

Kentico

Synchronizace pomocí System.Collections.Concurrent

- `BlockingCollection<T>` - jako interní data store může sloužit cokoliv z následujících (`IProducerConsumerCollection`)
- `ConcurrentQueue<T>`
- `ConcurrentStack<T>`
- `ConcurrentBag<T>`

Kentico

PLINQ

- Paralelní implementace LINQ
- Pokud uzná za vhodné vyhodnocuje výrazy paralelně
- Operátory
  - `AsParallel`
  - `AsOrdered/AsUnordered`
  - `WithCancellation`
  - `WithDegreeOfParallelism`
  - `WithExecutionMode`
- Další operátory obdobné jako v normálním LINQ

Kentico

Ing. Zdeněk Jurka

Kentico software
Nové sady 25
Brno
Česká republika
E-mail: zdenekj@kentico.com

Kentico

Reference

- Cleary, Stephen. Concurrency in C# Cookbook
- http://www.albahari.com/threading/