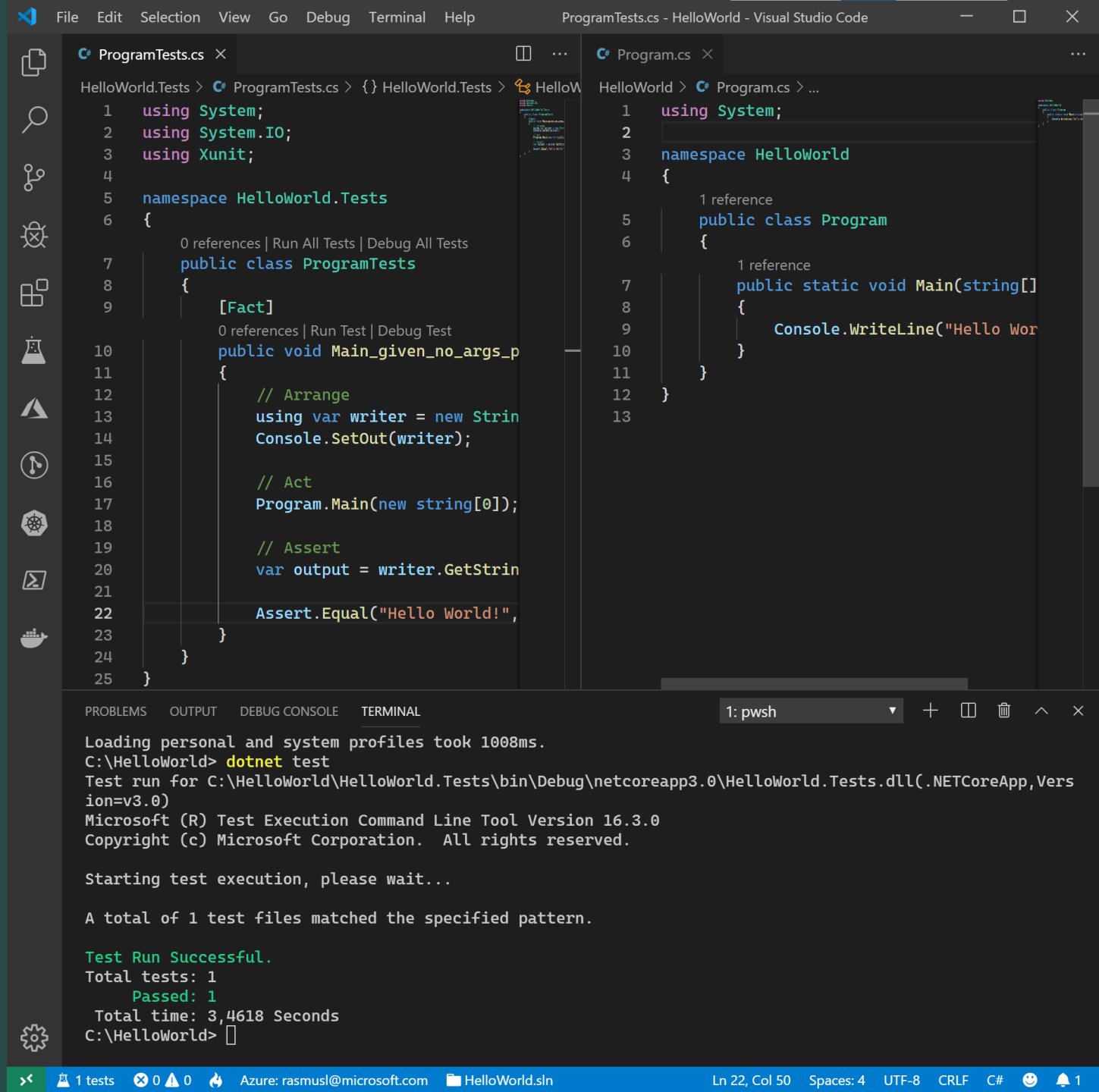
C#
Asynchronous and Parallel Programming

Rasmus Lystrøm
Associate Professor
ITU
rnie@itu.dk



Agenda

Dictionary

Threads

Task Parallel Library

Asynchronous Programming

Async ≠ Parallel ≠ Threads

Concurrency

Concurrency I

A property of systems in which several computations are executing **simultaneously**, and potentially interacting with each other. The computations may be executing on multiple cores in the same chip, preemptively time-shared threads on the same processor, or executed on physically separated processors.

Concurrency II

Multiple tasks which start, run, and complete in overlapping time periods, in no specific order

Parallelism

Parallelism

When multiple tasks OR several parts of a unique task literally run at the same time, e.g. on a multi-core processor.

Multithreading

Multithreading

Software implementation which allows different threads to be executed concurrently.

A multithreaded program appears to be doing several things at the same time even when it’s running on a single-core machine.

Compare to chatting with different people through various IM windows; although you’re switching back and forth, the net result is that you’re having multiple conversations at the same time.

Asynchronous methods

Asynchronous methods

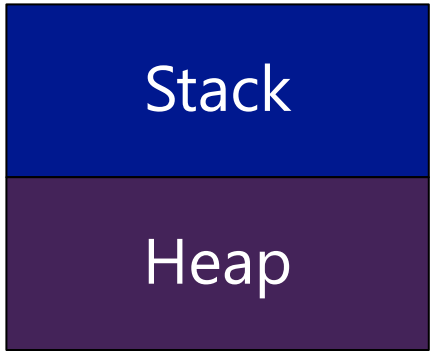
Not related to Concurrency and parallelism!

Asynchrony is used to present the impression of concurrent or parallel tasking.

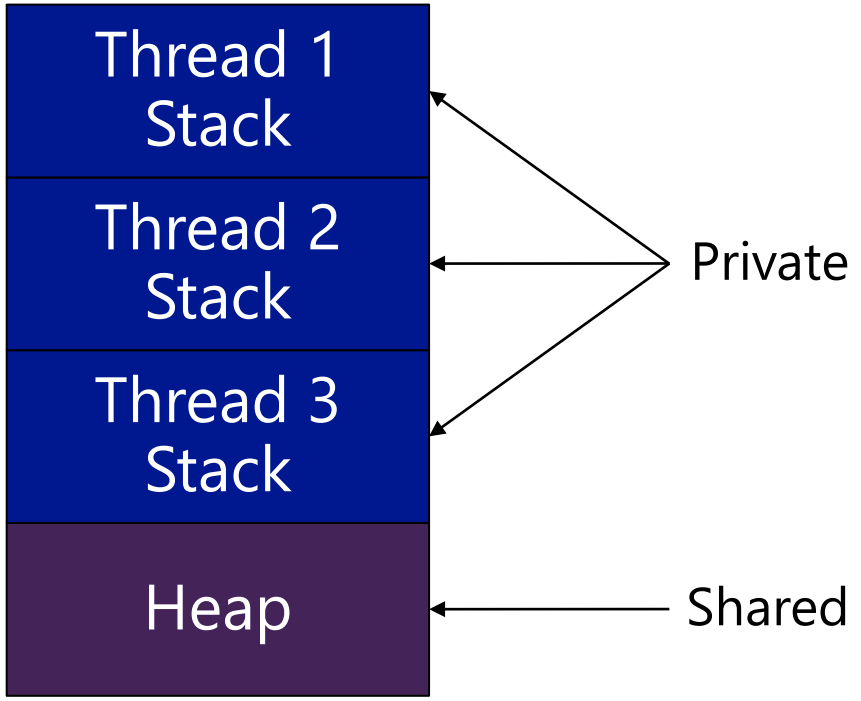
Normally used for a process that needs to do work away from the current application where we don't want to wait and block our application awaiting the response.

Threads

Threads

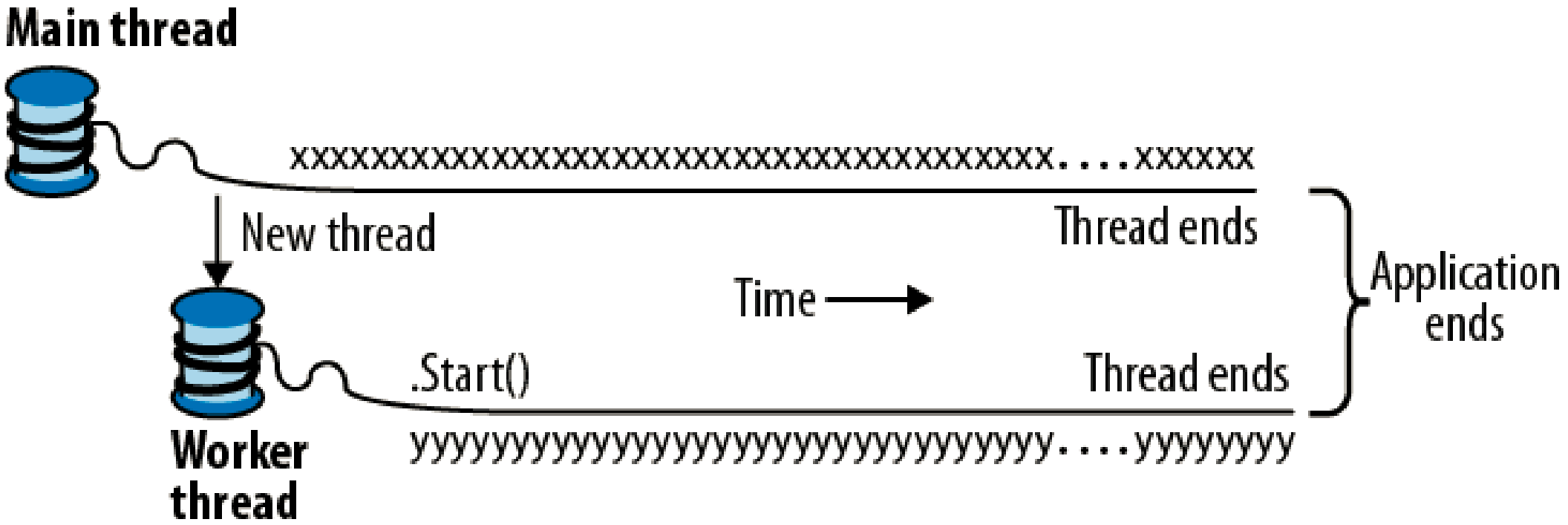


Single Threaded Program



Multithreaded Program

Threads Example



Threads

Demo

Race Condition

Ralf Schumacher, 3 Mar 2002, Reporter Images

Race Condition

Behavior of a program where the output is **dependent** on the **sequence** or **timing** of other **uncontrollable** events.

→ Bug, when events do not happen in the order the programmer intended.

Race Condition

Demo

Deadlock

Deadlock by David Mailtland

Deadlock

A situation in which two or more competing actions are each waiting for the other to finish, and thus neither ever does.

Deadlock

Demo

Task Parallel Library

Task.Run

Task.Factory…

Task.Delay

Parallel.For

Parallel.ForEach

Parallel.Invoke

Parallel Linq → .AsParallel()

Task Parallel Library

Demo

System.Collections.Concurrent

ConcurrentQueue<T>

ConcurrentStack<T>

BlockingCollection<T>

ConcurrentDictionary<TKey, TValue>

Concurrent Collections

Demo

Asynchronous Programming

Asynchronous Programming

Asynchronous programming is a means of parallel **programming** in which a unit of work runs separately from the main application thread and notifies the calling thread of its completion, failure or progress.

Asynchronous Programming

Demo

async/await

`async` →

Method must return `void`, `Task`, `Task<T>`, or a task-like type.

Specifically: a type, which satisfy the `async` pattern, meaning a `GetAwaiter` method must be accessible.

`await` → Await task(s)…

Note: `Main` and *test* methods must return `Task`

Speed
Multiprocessor
Parallel execution

Async ≠ Parallel ≠ Threads

Non-blocking UI,
background tasks,
asynchronous

Low-level building
block
Do not use directly!