



I'll Get Back to You

Task, Await, and Asynchronous Programming

Jeremy Clark

www.jeremybytes.com

Asynchronous Patterns

- Asynchronous Programming Model (APM)
- Event Asynchronous Pattern (EAP)
- Task Asynchronous Pattern (TAP)

Asynchronous Programming Model (APM)

- Method-Based
- Methods
 - BeginGetData()
 - EndGetData()
- IAsyncResult

Event Asynchronous Pattern (EAP)

- Method/Event-Based
- Method
 - GetDataAsync()
- Event
 - GetDataCompleted
 - Results in EventArgs

Task Asynchronous Pattern (TAP)

- Task-Based
- Method Returns a Task
 - `Task<T> GetDataAsync()`
- Task
 - Represents a concurrent operation
 - May or may not operate on a separate thread
 - Can be chained and combined

async & await

- Syntactic Wrapper Around Task
 - “await” pauses the current method until Task is complete.
 - Looks like a blocking operation
 - Does not block current thread
- “async” is just a Hint
 - Does not make a method run asynchronously
 - Tells the compiler to treat “await” as noted above

Continuation Options

- TaskContinuationOptions
 - OnlyOnRanToCompletion
 - OnlyOnCanceled
 - OnlyOnFaulted
 - NotOnRanToCompletion
 - NotOnCanceled
 - NotOnFaulted

Task Properties

- Task Properties

- IsCanceled
- IsCompleted*
- IsFaulted
- Status

**Note: Means “no longer running” not “completed successfully”*

- TaskStatus

- **Canceled**
- Created
- **Faulted**
- **RanToCompletion**
- Running
- WaitingForActivation
- WaitingForChildrenToComplete
- WaitingToRun

Cancellation

- CancellationToken is ReadOnly
 - `new CancellationToken(true)`
 - `new CancellationToken(false)`
- CancellationTokenSource
 - `var cts = new CancellationTokenSource()`
 - `var token = cts.Token`
 - `cts.Cancel()`



Thank You!

Jeremy Clark

- <http://www.jeremybytes.com>
- jeremy@jeremybytes.com
- @jeremybytes