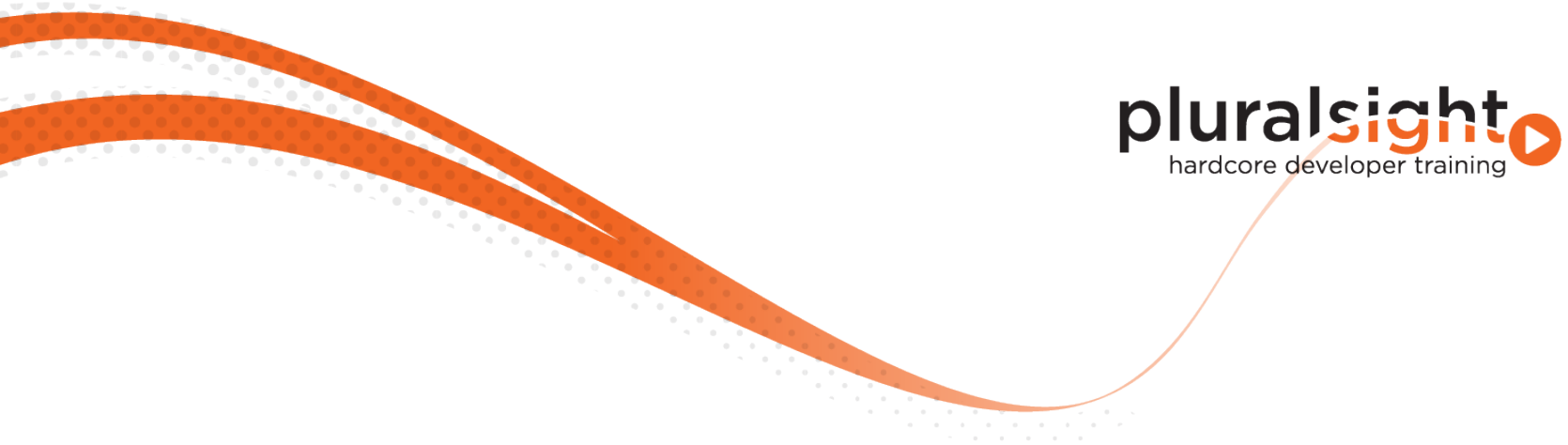
Using Tasks with Other Asynchronous Patterns

Ian Griffiths
http://www.interact-sw.co.uk/iangblog/
ian@interact-sw.co.uk



pluralsight
hardcore developer training

The APM

- **Asynchronous Programming Model**
  - aka IAsyncResult design pattern

```
IAsyncResult BeginSomething(
    string whatever, int args, bool are, int necessary,
    AsyncCallback onComplete, object asyncState);
```

```
IEnumerable<DateTime> EndSomething(IAsyncResult iar);
```

Mapping the APM to a Task

- **TaskCompletionSource?**
  - Not necessary

```
public static class StreamAsyncExtensions
{
  public static Task<int> ReadAsync(
     this Stream s, byte[] data, int offset, int count)
  {
    return Task<int>.Factory.FromAsync(
      s.BeginRead, s.EndRead, data, offset, count, null);
```

```
public Task<TResult> FromAsync<TArg1, TArg2, TArg3>(
  Func<TArg1, TArg2, TArg3, AsyncCallback, object, IAsyncResult> beginMethod,
  Func<IAsyncResult, TResult> endMethod,
  TArg1 arg1, TArg2 arg2, TArg3 arg3,
  object state)
```

Presenting a Task through the APM

- **Some frameworks consume the APM**
- **Full APM implementation relatively complex**
  - IAsyncResult: 2 completion mechanisms
  - Begin/End: 2 more completion mechanisms
- **Use Task**
  - Implements IAsyncResult
  - Continuation for callback
  - Wait or Result for End

The EAP

- The Event-based Asynchronous Pattern

```
public void DownloadStringAsync(Uri address)

public event DownloadStringCompletedEventHandler
                    DownloadStringCompleted;
```

Summary

- APM
- FromAsync
- Task and IAsyncResult
- EAP