

SYNERGY DEVPARTNER CONFERENCE 2026

Synergy DBL Meets .NET 10: New Language Features in Action

Marty Lewis, Software Development Manager

At a glance



- New paradigms
  - Span<T> all the things
  - Implement interfaces on ref struct
  - Extension members
- Code gen friendly
  - Partial members
- Syntax Sugar
  - Null conditional assignment
  - .NET inline arrays
- More!

SYNERGY 2026 DEVPARTNER CONFERENCE



Compatibility is the #1 priority

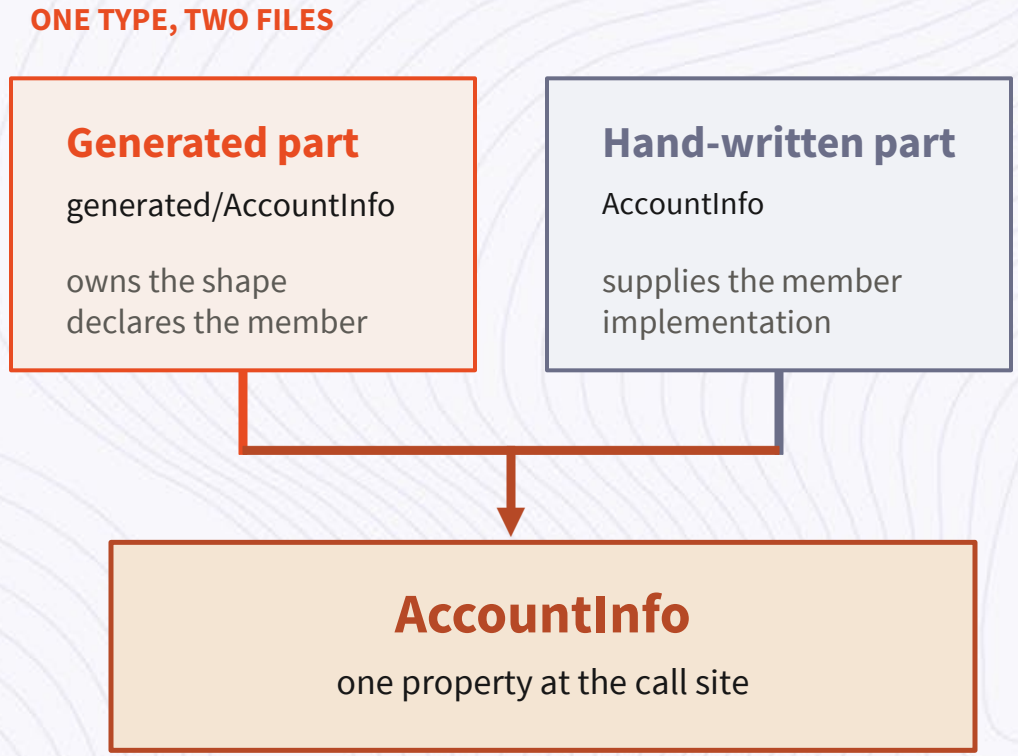
.NET Feature Compatibility

- Extension members
  - Full support
- Partial members
  - Partial properties and indexers!
  - In .NET as well as Traditional
- Span<T> & ReadOnlySpan<T>
  - Implicit conversion
    - Works with foreach and other collections
- Inline arrays
  - Fully functional
- Interfaces on ref structs
  - Fully functional
  - ref struct type constraints too
- So much more
  - OverloadResolutionPriorityAttribute
  - ^nameof improvements
  - Compound assignment operators


Striking the right balance

Partial Members

- Important for extending generated types
- Properties and Indexers
  - Declare on one side
  - Implement on the other
- Auto property declarations are restricted

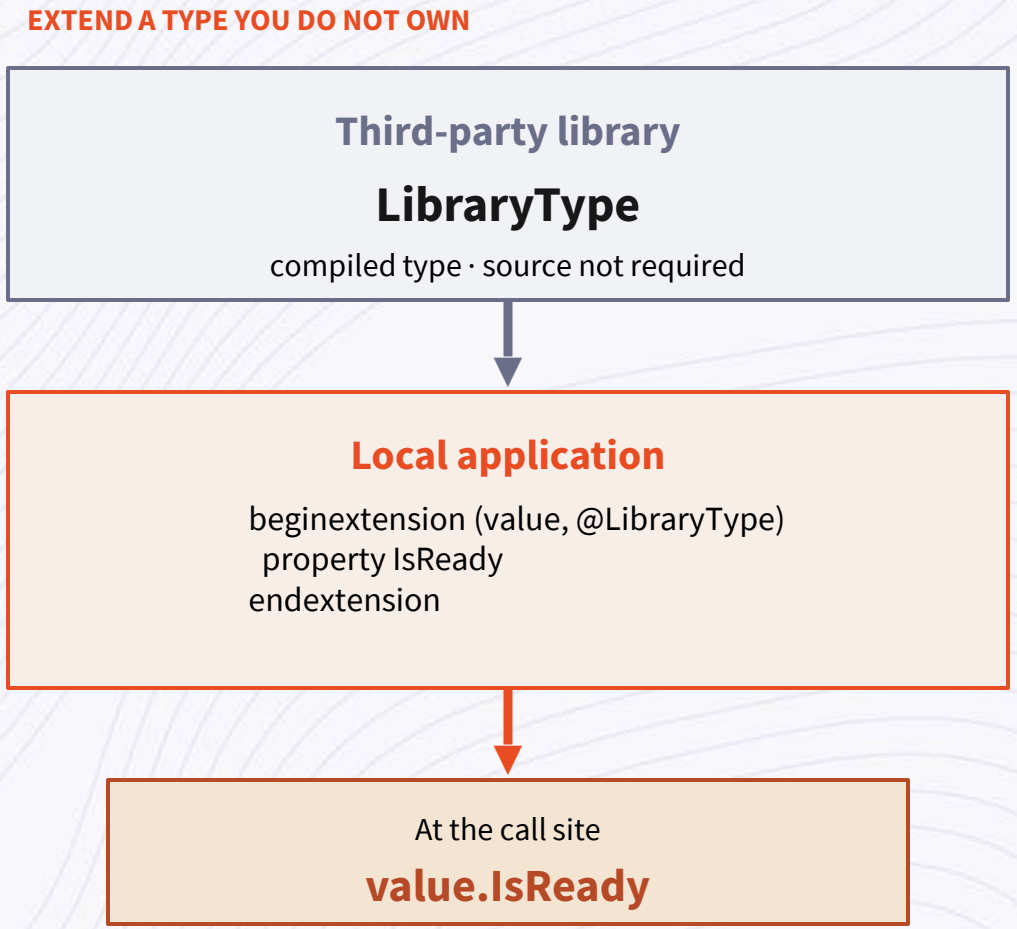


Partial Members

```
partial class OrderRecord
  public CustomerId, a12
  public partial property DisplayName, string
    endparams
    method get
    endmethod
  endproperty
endclass
```

```
partial class OrderRecord
  public partial property DisplayName, string
    endparams
    method get
    proc
      mreturn %atrim(CustomerId)
    endmethod
  endproperty
endclass
```

Extension Members



- Introducing
  - beginextension (name, @type)
  - endextension
- Still must be in a static class
- Support declaring:
  - Methods
  - Properties
  - Operators

Extension Members

```
public static class OrderRecordExtensions
  beginextension (order, @OrderRecord)
    public property IsHighValue, int
      method get
      proc
        if (order.Total .GE. 5000)
          mreturn 1
        mreturn 0
      endmethod
    endproperty

  public method ToReviewLine, string
    proc
      mreturn "Order " + %atrim(order.OrderId) + "  " +
      & order.DisplayName + "  " + %atrim(order.Status)
    endmethod
  endextension
endclass
```

- Introducing
  - beginextension (name, @type)
  - endextension
- Still must be in a static class
- Support declaring:
  - Methods
  - Properties
  - Operators

All about cls structure

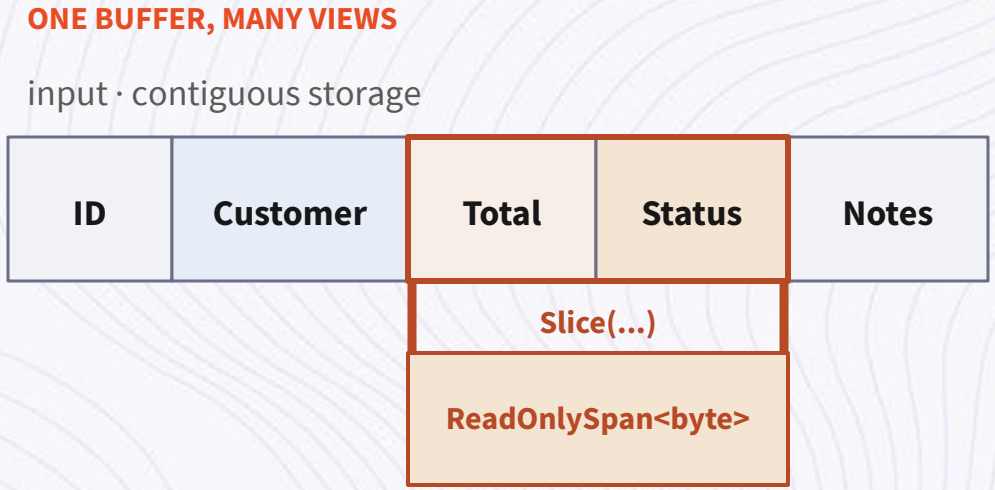
- cls structure != structure
  - Still value types
  - Does not support inheritance
  - Non-nullable
  - Boxable
- byref cls structure
  - Stack-only, no boxing
  - Limited use
- byref cls structure mystruct implements IInterface
- Also supported on type constraints now!

All about cls structure

- cls structure != structure
  - Still value types
  - Does not support inheritance
  - Non-nullable
  - Boxable
- byref cls structure
  - Stack-only, no boxing
  - Limited use
- byref cls structure mystruct implements IInterface
- Also supported on type constraints now!

Performance Features

- Native support
  - Span<T>
  - ReadOnlySpan<T>
- Conversion to/from*
  - Implicit conversions where it makes sense
  - Foreach support
- Inline array



Performance Features

- Native support
  - Span<T>
  - ReadOnlySpan<T>
- Conversion to/from*
  - Implicit conversions where it makes sense
  - Foreach support
- Inline array

```
public cls structure struct1
    fld1, int
endstructure

public cls structure struct2
    fld1, [3]int, 5, 6
endstructure

public static method printspan, void
    inputSpan, ReadOnlySpan<struct1>
proc
    foreach data svar in inputSpan
    begin
        Console.WriteLine(svar.fld1)
    end
end
```

Nice to have

- Null conditional assignment
  - myclass?.field = value
- Compound assignment
  - Externally defined operators
  - +=, *=, -=, etc
- ^nameof on unbound generics:
  - ^nameof(List<>) == List

Nice to have

- Null conditional assignment
  - myclass?.field = value
- Compound assignment
  - Externally defined operators
  - +=, *=, -=, etc
- ^nameof on unbound generics:
  - ^nameof(List<>) == List



SYNERGY 2026
DEVPARTNER CONFERENCE

Questions?