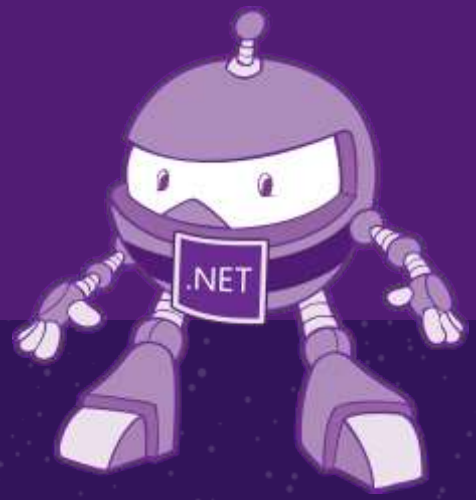
.NET Core 3, C#, EF, Desktop, IoT

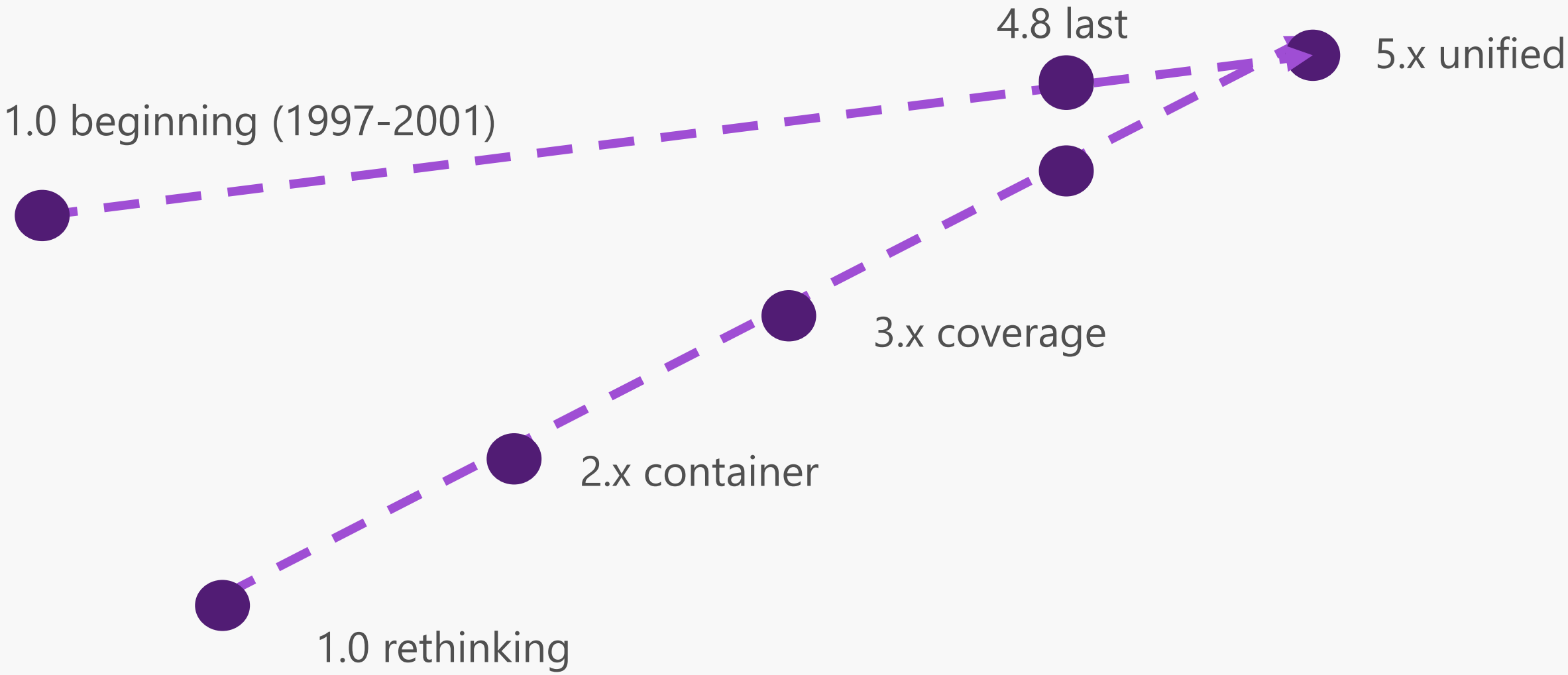
Marco Parenzan – Microsoft MVP (Azure)



.NET Core 3.0

.NET

The .NET and .NET Core Path



What's new in .NET Core 3.0

C# 8.0

.NET Standard 2.1

Improved .NET Core Version APIs

.NET Platform-Dependent Intrinsics

Default executables

Single-file executables

Assembly linking

Tiered compilation

ReadyToRun images

Build copies dependencies

Local tools

Major-version Roll Forward

Windows desktop

COM-callable components - Windows Desktop

MSIX Deployment - Windows Desktop

WinForms high DPI

Ranges and indices

Async streams

IEEE Floating-point improvements

Fast built-in JSON support

Interop improvements

HTTP/2 support

TLS 1.3 & OpenSSL 1.1.1 on Linux

Cryptography ciphers

Cryptographic Key Import/Export

SerialPort for Linux

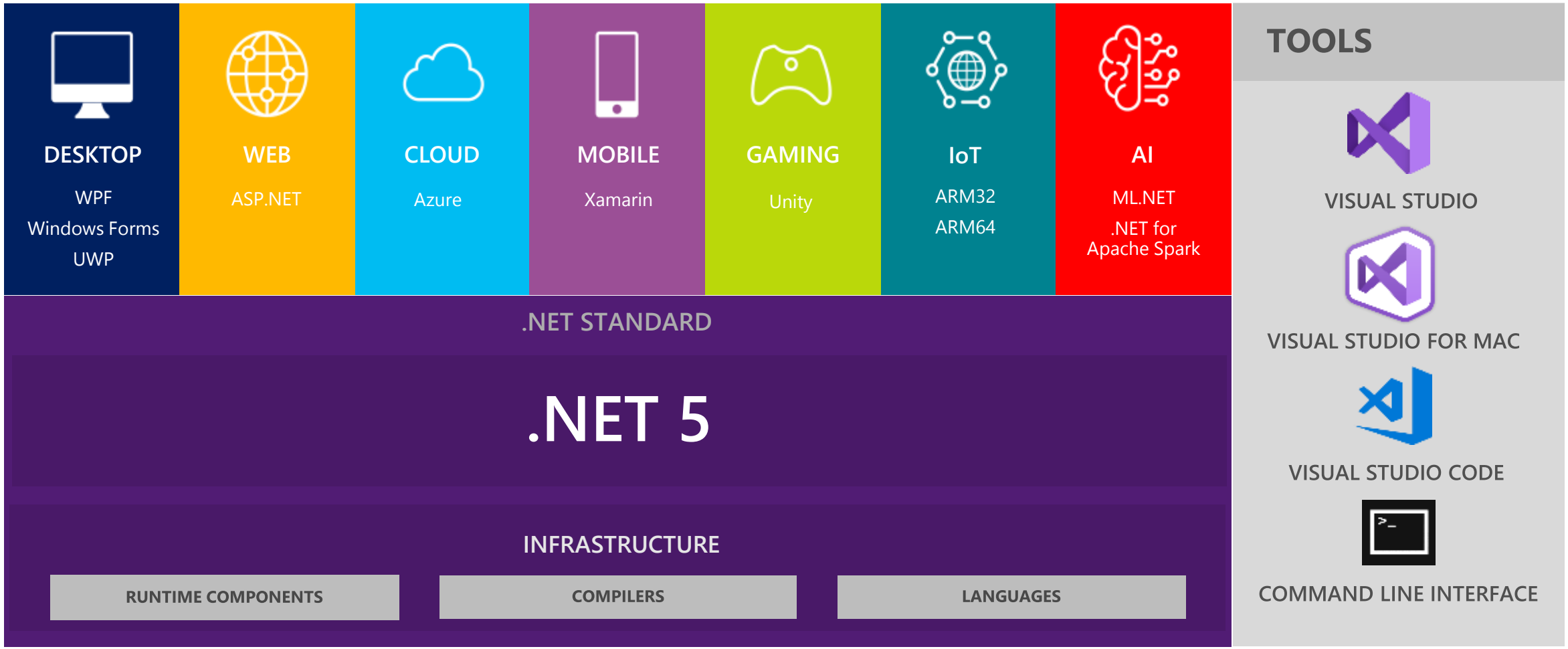
Docker and cgroup memory Limits

Smaller Garbage Collection heap sizes

Garbage Collection Large Page support

GPIO Support for Raspberry Pi

.NET – A unified platform



.NET Schedule

- .NET Core 3.0 released today!
- .NET Core 3.1 = Long Term Support (LTS)
- .NET 5.0 release in November 2020
- Major releases every year, LTS for even numbered releases
- Predictable schedule, minor releases if needed

.NET Core SDK Windows Installer

- The MSI installer for Windows has changed starting with .NET Core 3.0. The SDK installers will now upgrade SDK feature-band releases in place. Feature bands are defined in the *hundreds* groups in the *patch* section of the version number. For example, 3.0.*101* and 3.0.*201* are versions in two different feature bands while 3.0.*101* and 3.0.*199* are in the same feature band. And, when .NET Core SDK 3.0.*101* is installed, .NET Core SDK 3.0.*100* will be removed from the machine if it exists. When .NET Core SDK 3.0.*200* is installed on the same machine, .NET Core SDK 3.0.*101* won't be removed.

.NET Standard 2.1

- Span<T>
- Index and Range
- IAsyncEnumerable<T>
- Reflection emit and capability APIs
- Hardware Intrinsics API [post]
- DbProviderFactories
- No support in full .NET Framework

	.NET Framework 4.8	.NET Core 3.0
Install	Global	Side-by-Side or In-app
.NET Native Compilation	No	Yes
Async Streams	No	Yes
Indexers	No	Yes
Ranges	No	Yes
.NET Standard 2.1	No	Yes
Default Interface Implementations	No	Yes
Nullable Reference Types	Yes	Yes
Recursive Patterns	Yes	Yes
Switch Expressions	Yes	Yes
Target-typed new-expressions	Yes	Yes

JSON

- System.Text.Json
  - Utf8JsonReader
  - Utf8JsonWriter
  - JsonDocument
  - JsonSerializer

Microsoft.Data.SqlClient package

C# 8.0

.NET

C# language evolution

- C# 1.0 was a new managed language
- C# 2.0 introduced generics
- C# 3.0 enabled LINQ
- C# 4.0 was all about interoperability with dynamic non-strongly typed languages.
- C# 5.0 simplified asynchronous programming with the async and await keywords.
- C# 6.0 the language has been increasingly shaped by conversation with the community, now to the point of taking language features as contributions from outside Microsoft
- C# 7.x will be no exception, with tuples and pattern matching as the biggest features, transforming and streamlining the flow of data and control in code. Point releases
- C# 7.1, 7.2, 7.3 Safe Efficient Code, More Freedom, Less Code
- C# 8.x (“unicorn”) running in the function path

C# 7 objectives

- Working with data
  - Pattern Matching
  - Tuples
  - Deconstruction
  - Discards
- Code Simplification
  - Expression Bodied Properties, Constructor Finalizer
  - Throw expression
  - Out Variables
  - Binary Literals and Digit separator
- Improved Performances
  - Local Functions
  - Return values and local variables by reference

What's new in C# 8

- Default interface methods
- Pattern matching enhancements:
  - Switch expressions
  - Property patterns
  - Tuple patterns
  - Positional patterns
- Using declarations
- Static local functions
- Indices and ranges
- Null-coalescing assignment
- stackalloc in nested expressions

Two more C# 8 enhancement

- Better to see with EF3.0

Entity Framework Core 3.0

.NET

Two more C# 8 enhancement

- Nullable reference types
  - I call it my billion-dollar mistake (Tony Hoare)
  - It's a breaking change for the semantics of C#
- Asynchronous streams

Entity Framework Core

- C# 8 support
  - **like async streams and nullable reference types**
- EF 6.3 ported to .NET Core
- Lots of breaking changes
- LINQ improvements
- Cosmos DB support
- Microsoft.Data.SqlClient
- Database views reverse engineering
  - **using ~~Query types~~ Entities without keys**
- EF Core no longer part of ASP.NET Core shared framework
- Targets .NET Standard 2.1

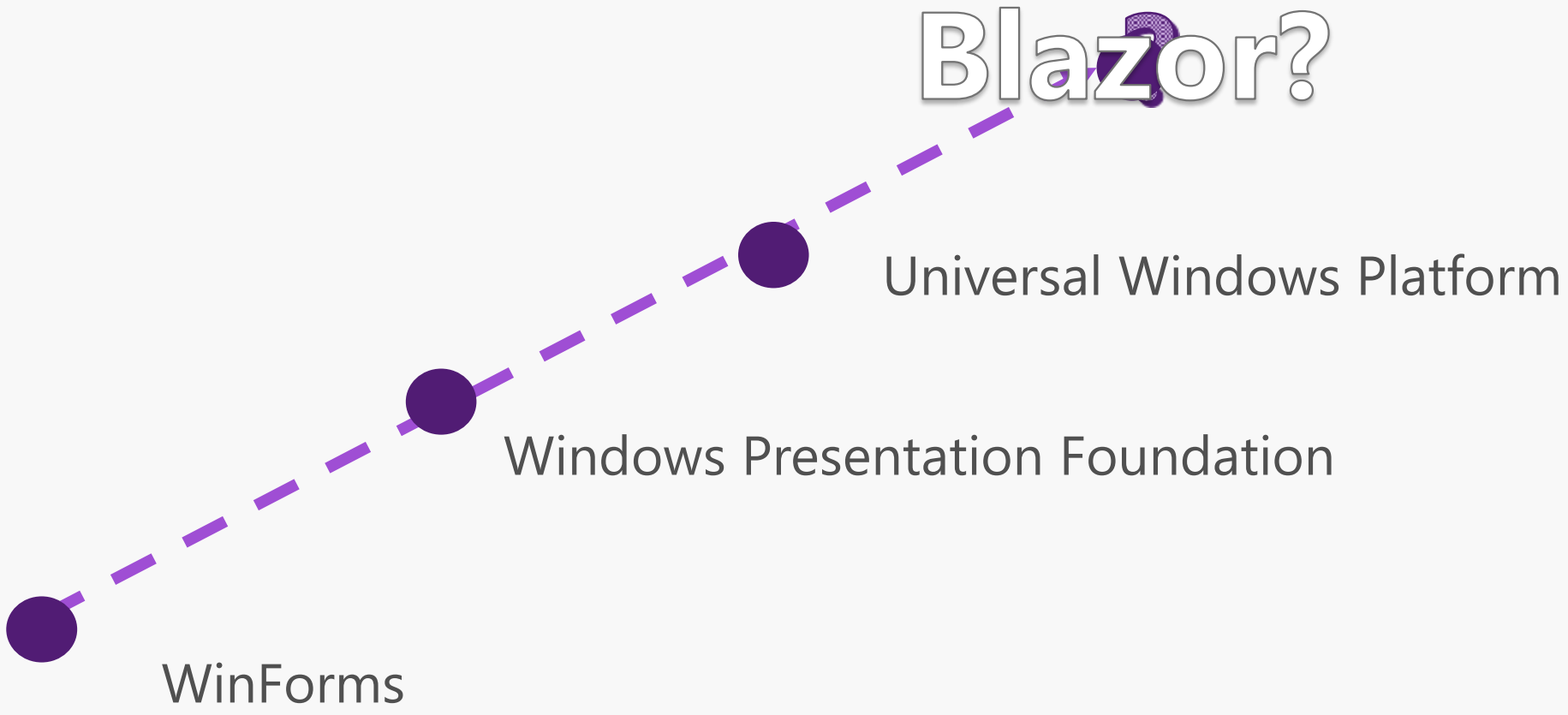
What's new in EF Core 3.0

- Relational SQL
  - Nullable reference types
  - LINQ overhaul
    - New architecture
    - Constrained client evaluation
    - Single SQL query per LINQ query
  - Reverse engineering of views
  - Microsoft.Data.SqlClient
- Cosmos DB provider
  - Async streams

Desktop Applications in .NET Core 3.0

.NET

The .NET Desktop Development evolution



Windows Desktop

- Windows Forms
- WPF
- UWP
- Open-source

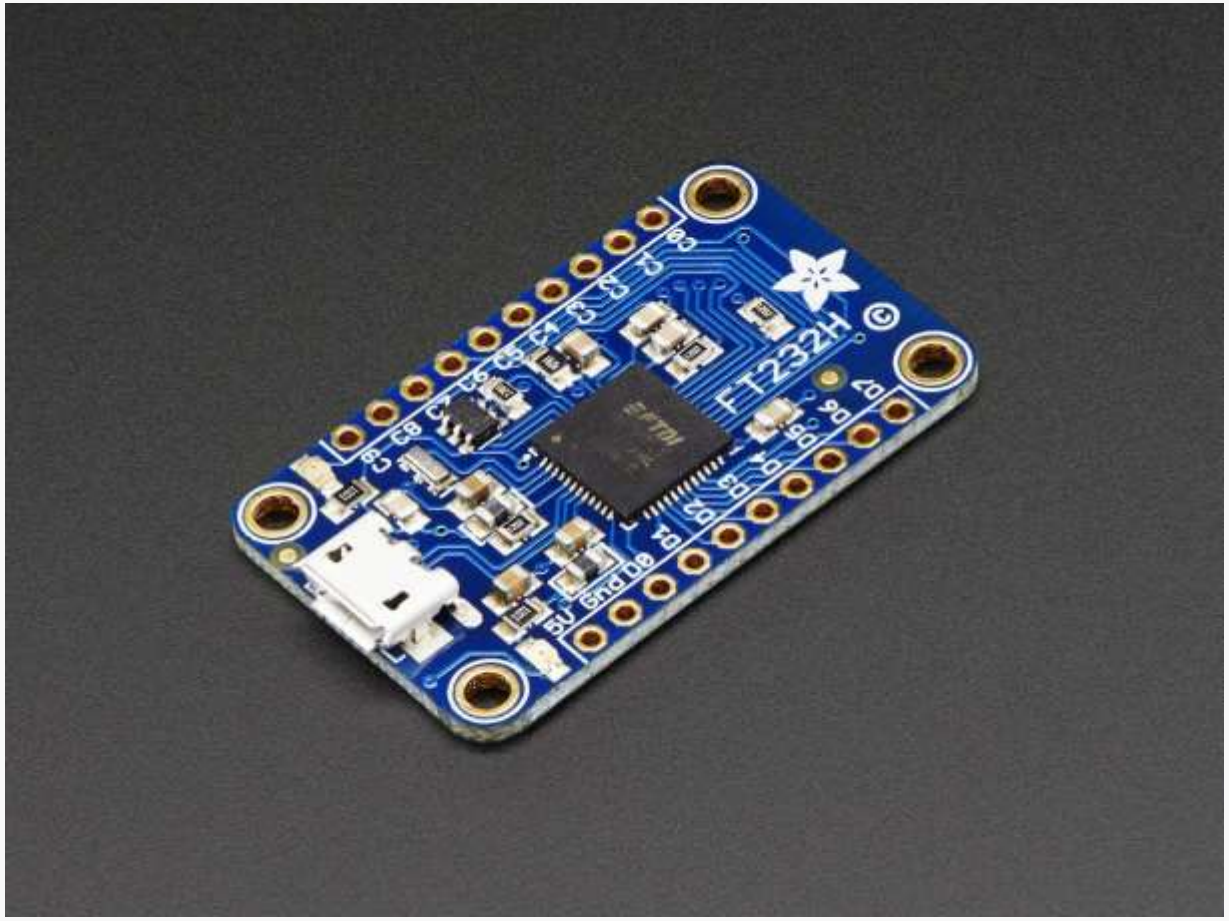
IoT in .NET Core 3.0

.NET

IoT in .NET Core 3

Raspberry PI

Windows 10 IoT Core



Questions and Answers

.NET