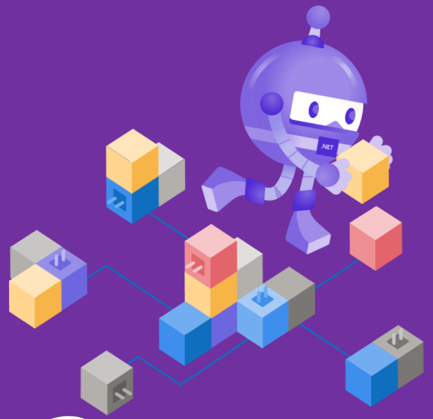


Why .NET ?
What is new .NET 8 and C# 12 ?

Build anything with a unified platform with .Net

.NET Momentum: Monthly active users 6.1+ Million

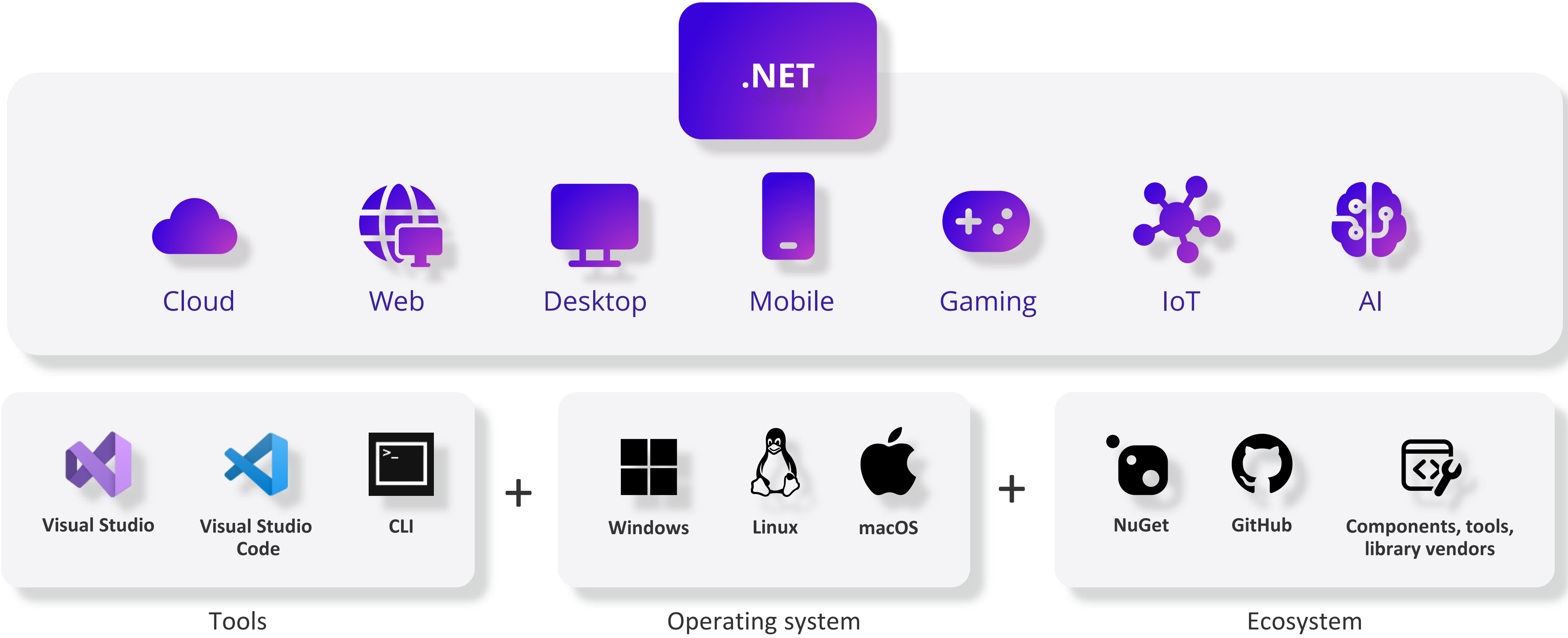
.Net 8 Most performant release

Cloud-Native Development with .Net

C# 12 Primary constructors, Collection expressions, Default lambda

Mehmet Ozkaya

Build Anything, Anywhere



.NET

Cloud · Web · Desktop · Mobile · Gaming · IoT · AI

Visual Studio · Visual Studio Code · CLI

Tools

+

Windows · Linux · macOS

Operating system

+

NuGet · GitHub · Components, tools, library vendors

Ecosystem

https://www.dotnetconf.net/

.NET Momentum

>21k
Contributions

>9k
Community members

6.1+ Million
Monthly active users

53,000+
Community members have contributed to .NET

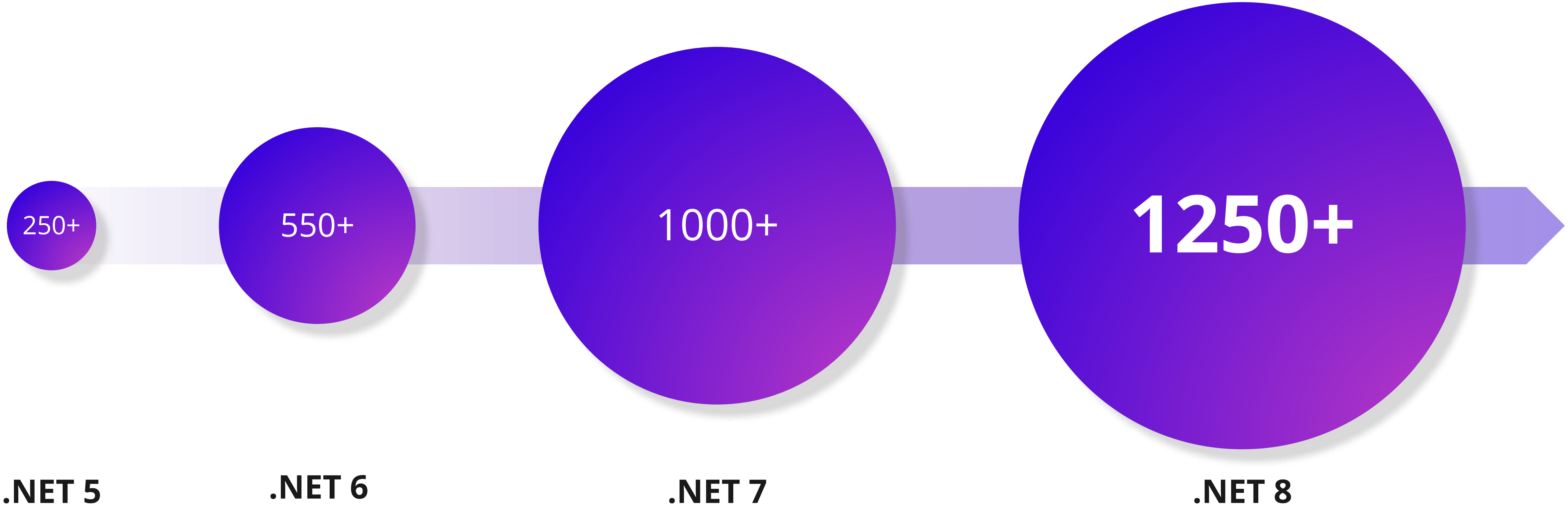
#1 2023 stackoverflow
Most Admired Developer Framework

Top 5 CLOUD NATIVE COMPUTING FOUNDATION
Highest velocity OSS project

Top 5
Top Programming Language on GitHub

https://www.dotnetconf.net/

Most performant release yet



250+

550+

1000+

1250+

.NET 5

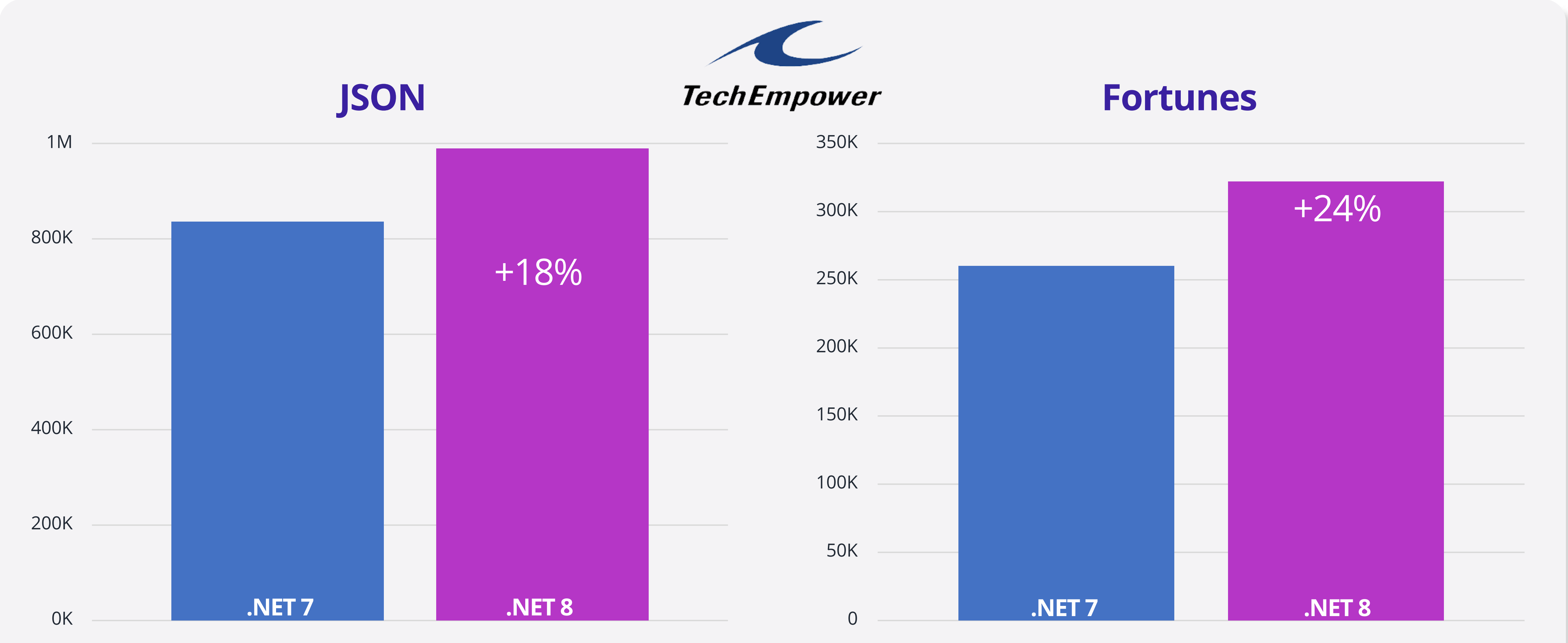
.NET 6

.NET 7

.NET 8

https://www.dotnetconf.net/

.NET 8 API Performance



Source: aka.ms/aspnet/benchmarks

Requests per Second (higher is better)

https://www.dotnetconf.net/

Mehmet Ozkaya

.NET is the best for modern apps

Modern Workloads

- Web
- Client
- Artificial Intelligence
- Cloud Native

Developer Productivity

- Language
- Productivity
- Modernization

Fundamentals

- Performance
- Supply Chain

https://www.dotnetconf.net/

Cloud-Native Development with .Net 8

Cloud Native

- Observability
- Resiliency
- Scalability
- Manageability

https://www.dotnetconf.net/

High-scale and high-availability services in .NET 8

Cloud Native

Resilience

Health Checks

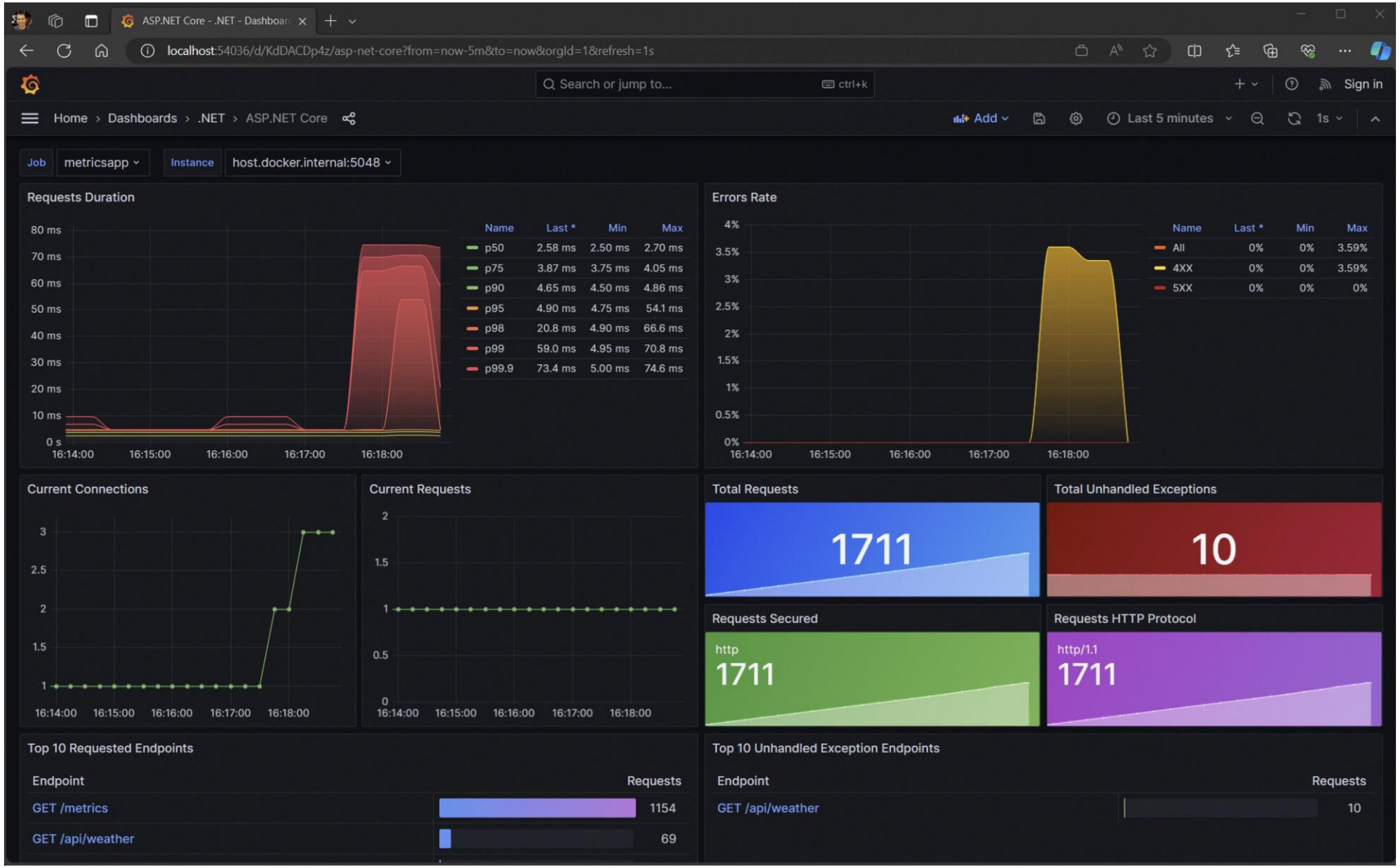
Testing/Fakes

Observability

- Extensions.Resilience
- Extensions.Http.Resilience
- Extensions.Diagnostics.HealthChecks.Commo
- Extensions.Diagnostics.Probes
- Extensions.Telemetry

- Extensions.Compliance.Redaction
- Extensions.Http.Telemetry
- AspNetCore.Testing
- Hosting.Testing
- Extensions.TimeProvider.Testing

Enhanced Open Telemetry Support



Requests Duration

Name	Last *	Min	Max
p50	2.58 ms	2.50 ms	2.70 ms
p75	3.87 ms	3.75 ms	4.05 ms
p90	4.65 ms	4.50 ms	4.86 ms
p95	4.90 ms	4.75 ms	54.1 ms
p98	20.8 ms	4.90 ms	66.6 ms
p99	59.0 ms	4.95 ms	70.8 ms
p99.9	73.4 ms	5.00 ms	74.6 ms

Errors Rate

Name	Last *	Min	Max
All	0%	0%	3.59%
4XX	0%	0%	3.59%
5XX	0%	0%	0%

Current Connections

Current Requests

Total Requests

1711

Total Unhandled Exceptions

10

Requests Secured

http

1711

Requests HTTP Protocol

http/1.1

1711

Top 10 Requested Endpoints

Endpoint	Requests
GET /metrics	1154
GET /api/weather	69

Top 10 Unhandled Exception Endpoints

Endpoint	Requests
GET /api/weather	10

Mehmet Ozkaya

https://www.dotnetconf.net/

Cloud-Native in .NET 8

Observability

- **Built in metrics with dimensions**
- **DI integration for metrics**
- **Better Logging support** (faster, can object serialization)
- **Enrichment**
- **Redaction**
- **Testing fakes for Logging & Metrics**

Resiliency

- **New Polly based resiliency packages**
- **SignalR Stateful Reconnect**

Scalability

- **AOT** (increased density)
- **Performance**
- **Chiseled Ubuntu**

Manageability

- **Certificate auto-rotation support in Kestrel**

https://www.dotnetconf.net/

.NET💜Containers

Hardened

- Non-Root Base Images
- USER "app"
- Default port - 8080
- Mariner distroless

Smaller

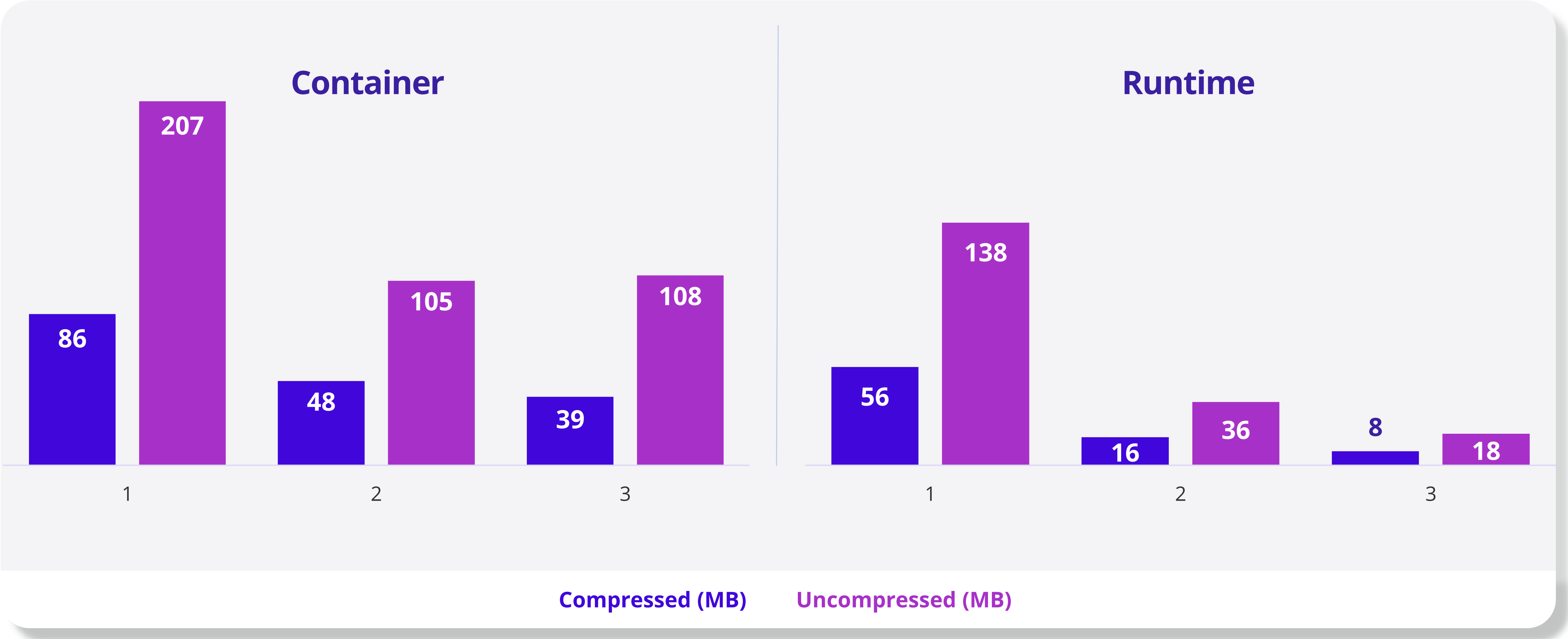
- AOT base images
- "Composite" base images
- "extra" base images
- Distroless / Chiseled

More Productive

- Publish with .NET SDK
- Cross compilation
- Non-root by default
- Supports all Azure auth

https://www.dotnetconf.net/

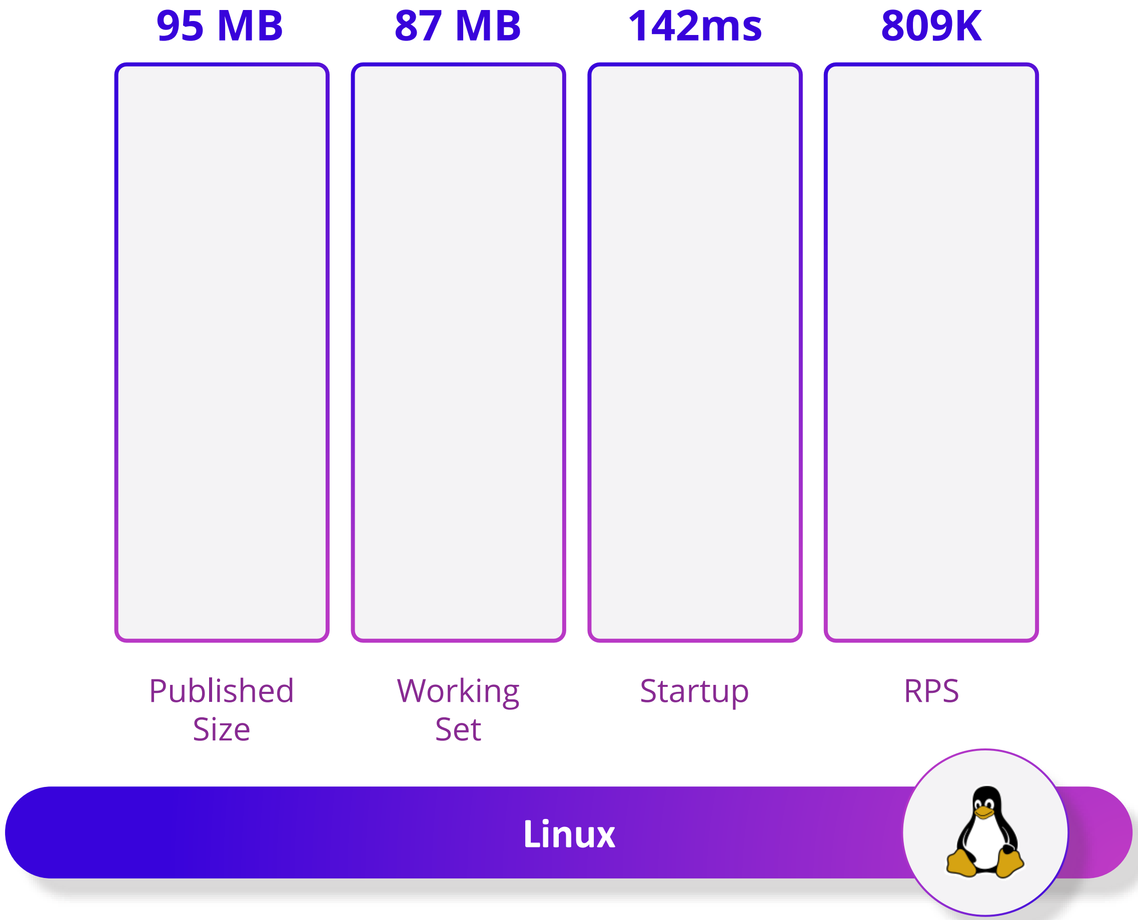
Container Size Improvements



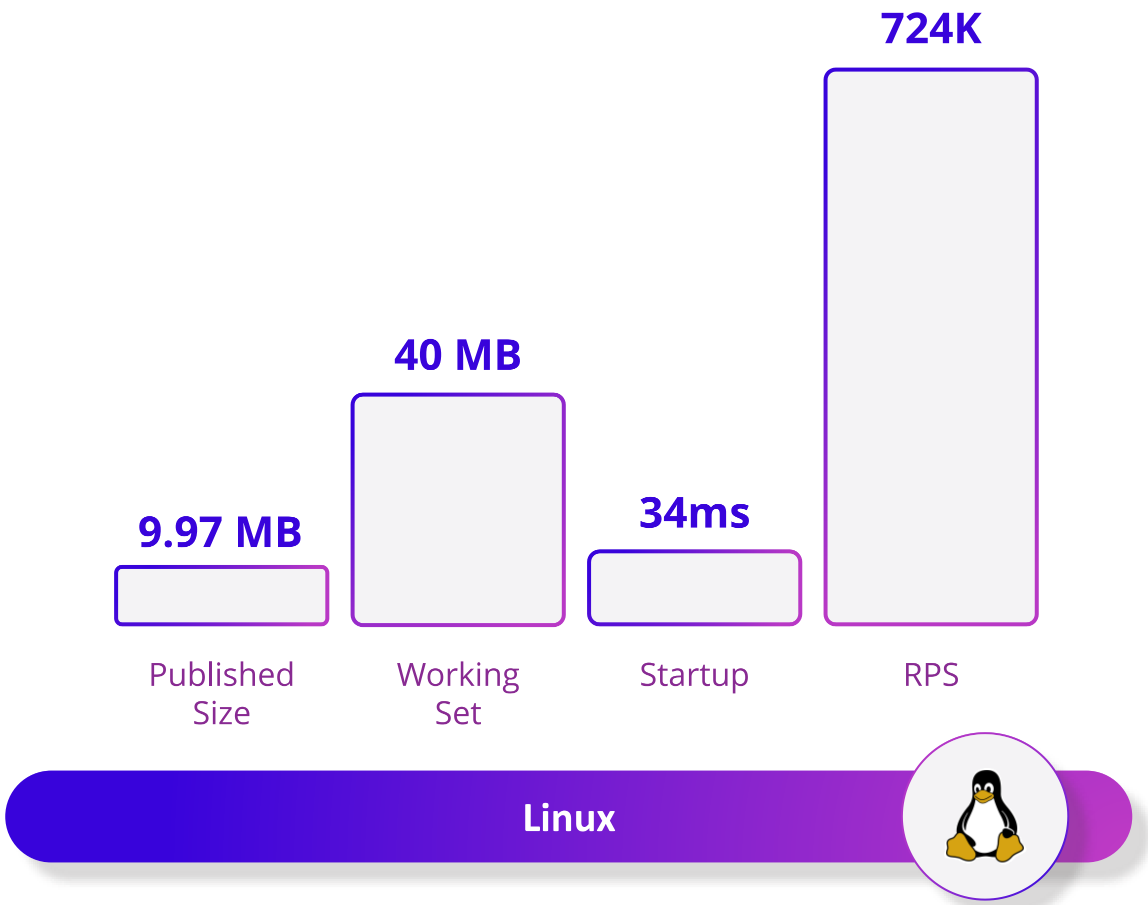
Mehmet Ozkaya

https://www.dotnetconf.net/

Native AOT - Container Size Improvements



Before Native AOT



With Native AOT

https://www.dotnetconf.net/

What is new .NET 8 ?

.NET Aspire

- Cloud-ready stack designed for building observable, production-ready distributed apps.
- NuGet packages targeting specific cloud-native concerns, now available in preview.

Core .NET Libraries

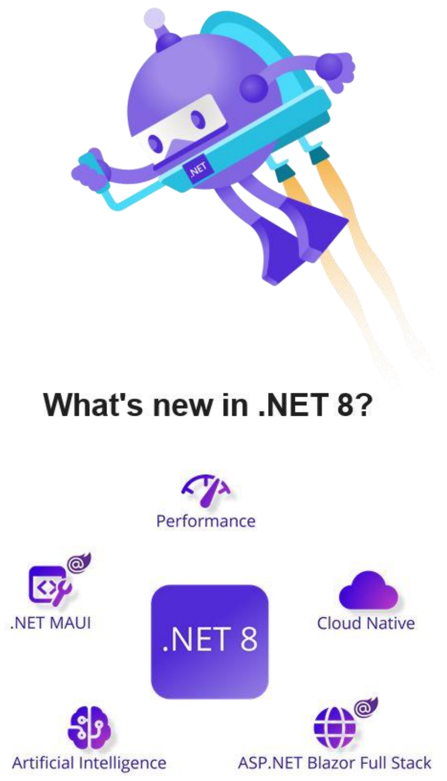
- Serialization enhancements, time abstraction, UTF8 improvements, methods for working with randomness, and performance-focused types like System.Numerics and System.Runtime.Intrinsics.

Metrics

- Attach key-value pair tags to Meter and Instrument objects, allowing for more nuanced differentiation in aggregated metric measurements.

Networking

- Support for HTTPS proxy, ensuring encrypted communication even in proxy scenarios, thus enhancing privacy and security.



What is new .NET 8 ? – Part 2

Extension Libraries

- Options validation, LoggerMessageAttribute constructors, extended metrics, hosted lifecycle services, keyed DI services.

Garbage Collection

- On-the-fly adjustment of memory limits, a crucial feature for cloud-service scenarios where dynamic scaling is mandatory.

Reflection Improvements

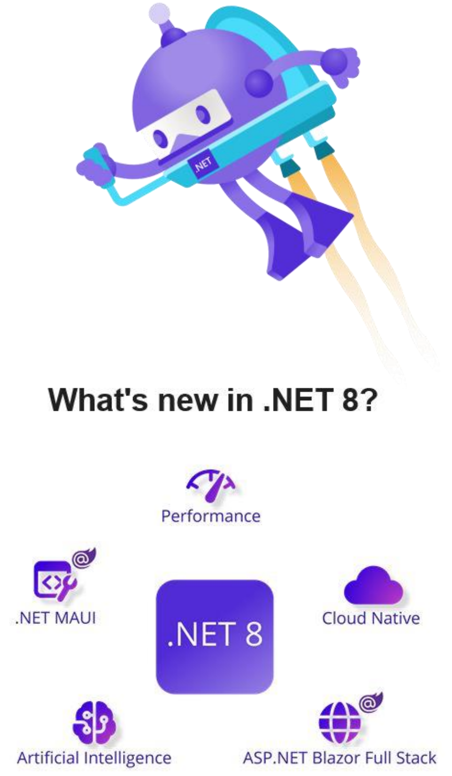
- Enhanced for better performance and more efficient memory usage. Function pointers also added reflection capabilities.

Native AOT Support

- Efficient compilation and execution, particularly beneficial for cloud-native and high-performance scenarios.

.NET SDK

- More robust and feature-rich, aligning with the evolving needs of modern .NET development. Enhanced dotnet publish and dotnet pack commands.



What is new C# 12 ?

Primary Constructors

- Primary constructors have been extended beyond record types. Parameters are now in scope for the entire class body.
- Should assigned, explicitly declared constructors must call the primary constructor using this() syntax.
  - public class Person(string name, int age)
  - {
  -     // Name and Age are in scope for the entire class body
  -     public string Name => name;
  -     public int Age => age;
  - }

Collection Expressions

- More concise syntax to create common collection values. Simplifies the way collections are initialized and manipulated.
  - var numbers = new List<int> { 1, 2, 3, ..otherNumbers };
  - var numbers = [1, 2, 3, .. otherNumbers];

What is new C# 12 ? – Part 2

Inline Arrays

- Enhance performance by enabling developers to create fixed-size arrays in struct types.
- Useful for optimizing memory layout and improving runtime performance.
  - public struct Buffer
  - {
  -     public Span<int> InlineArray => MemoryMarshal.CreateSpan(ref _array[0], 10);
  -     private int[] _array;
  - }

Optional Parameters in Lambda Expressions

- Default values for parameters in lambda expressions. This mirrors the syntax and rules for adding default values in methods, making lambda expressions more flexible.
  - Func<int, int, int> add = (x, y = 1) => x + y;
  - Console.WriteLine(add(5)); // Outputs 6

What is new C# 12 ? – Part 3

ref readonly parameters

- Enhances the way readonly references are passed in C#.
- Optimizing memory usage and performance in scenarios involving large data structures.
    - public void ProcessLargeData(in LargeData data)
    - {
    -     // Processing data without the risk of modifications
    - }

Alias Any Type

- Using alias directive to include any type, not just named types.
- Creation of semantic aliases for complex types like tuples, arrays, and pointer types.
    - using Coordinate = System.ValueTuple<int, int>;
    - Coordinate location = (10, 20);

C# Top-level statements, Global usings, Pattern Matching

Top-Level Statements

- Simplify the entry point of your applications. Instead of wrapping your main logic in a Main method, you can directly write the code at the top level of your file.
  - using System;
  - Console.WriteLine("Hello, World!");

Global Usings

- Make namespaces available across your entire project. Instead of repeating using statements in every file, you declare them globally in one place.
  - global using System;
  - global using System.Collections.Generic;

C# Pattern Matching

Pattern Matching

- More expressive syntax for checking and deconstructing values in your code.
    - public class Person
    - {
    -     public string Name { get; set; }
    -     public string Title { get; set; }
    - }
    - Person person = new Person { Name = "John", Title = "Manager" };
    - if (person is { Title: "Manager" })
    - {
    -     Console.WriteLine($"{person.Name} is a manager.");
    - }

C# Pattern Matching with Switch Expressions

Pattern Matching

- Pattern Matching - using switch expressions to compare discrete values.

```
public State PerformOperation(Operation command) =>
   command switch
   {
       Operation.SystemTest => RunDiagnostics(),
       Operation.Start => StartSystem(),
       Operation.Stop => StopSystem(),
       Operation.Reset => ResetToReady(),
       _ => throw new ArgumentException("Invalid enum value for command", nameof(command)),
   };
```