NATİVE AOT

Claude Omosa

.NET CONF 2025
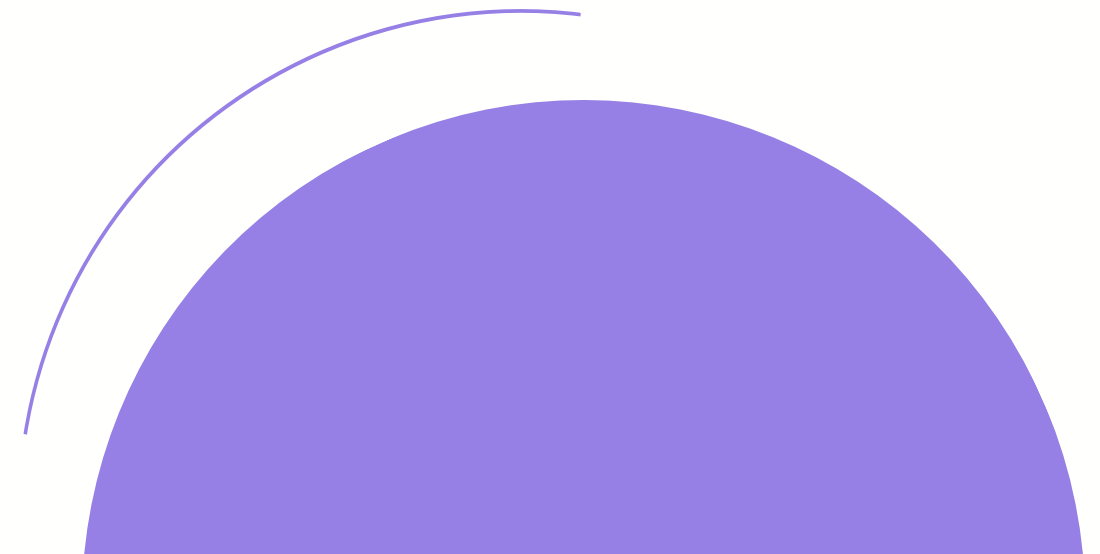
Hello,

I'm Claude Omosa

Software Developer,
Cognativ Technologies Kenya
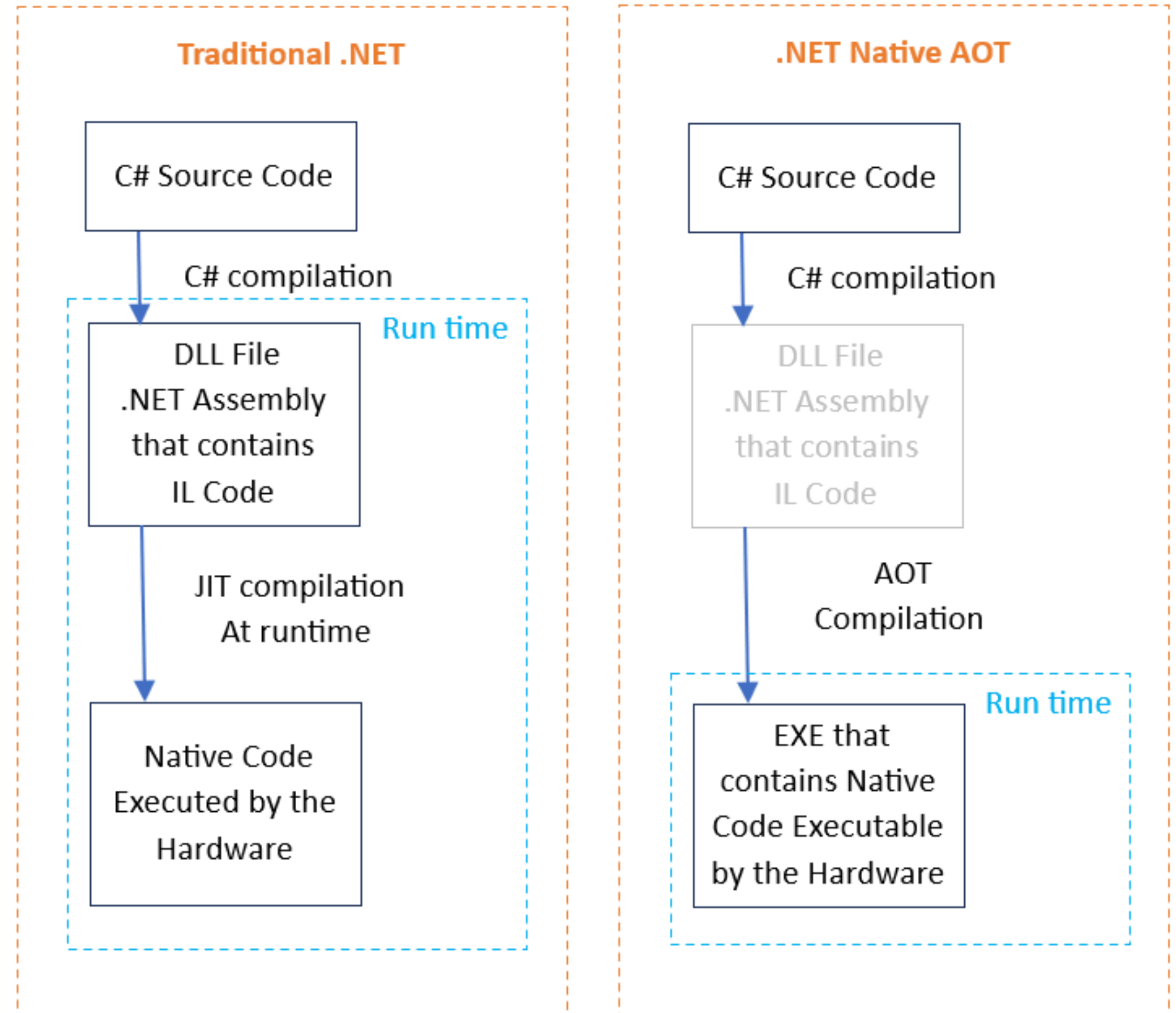
What is Native AOT?

Traditional .NET with JIT

C# Code
  → IL Code
    → [Deploy]
      → JIT at Runtime
        → Native Code

Native AOT

C# Code
  → [Compile to Native]
    → Native Executable
      → Run Immediately

Traditional .NET

C# Source Code

↓ C# compilation

Run time

DLL File
.NET Assembly
that contains
IL Code

↓ JIT compilation
At runtime

Native Code
Executed by the
Hardware

.NET Native AOT

C# Source Code

↓ C# compilation

DLL File
.NET Assembly
that contains
IL Code

↓ AOT
Compilation

Run time

EXE that
contains Native
Code Executable
by the Hardware

Benefits of .NET Native AOT

- Faster Cold Starts
- Smaller Deployments
- Reduced Memory Consumption

Demo

When to use Native AOT

- Microservices and containerized apps
- Serverless functions
- CLI tools
- APIs with Minimal APIs pattern
- Batch processing jobs
- High scale services where startup time matters

Avoid Native AOT:

- Legacy codebases with heavy reflection
- When build time is more crucial than runtime

What works with Native AOT

- ASP.NET Core Minimal APIs
- System.Text.Json with Source Generators
- EF Core
- Dependency Injection
- Configuration and Logging
- HttpClient
- gRPC
- Most modern NuGet packages

Limitations Native AOT

- Platform specific compilation
- Does not work well with Reflection
- Requires AOT compatible dependencies
- Longer Build Times

Thank you

Contact Details

Phone :   *+254-111822299*

Email :   *claudeomosa@gmail.com*