Testing.Platform,

the new way to run .NET tests

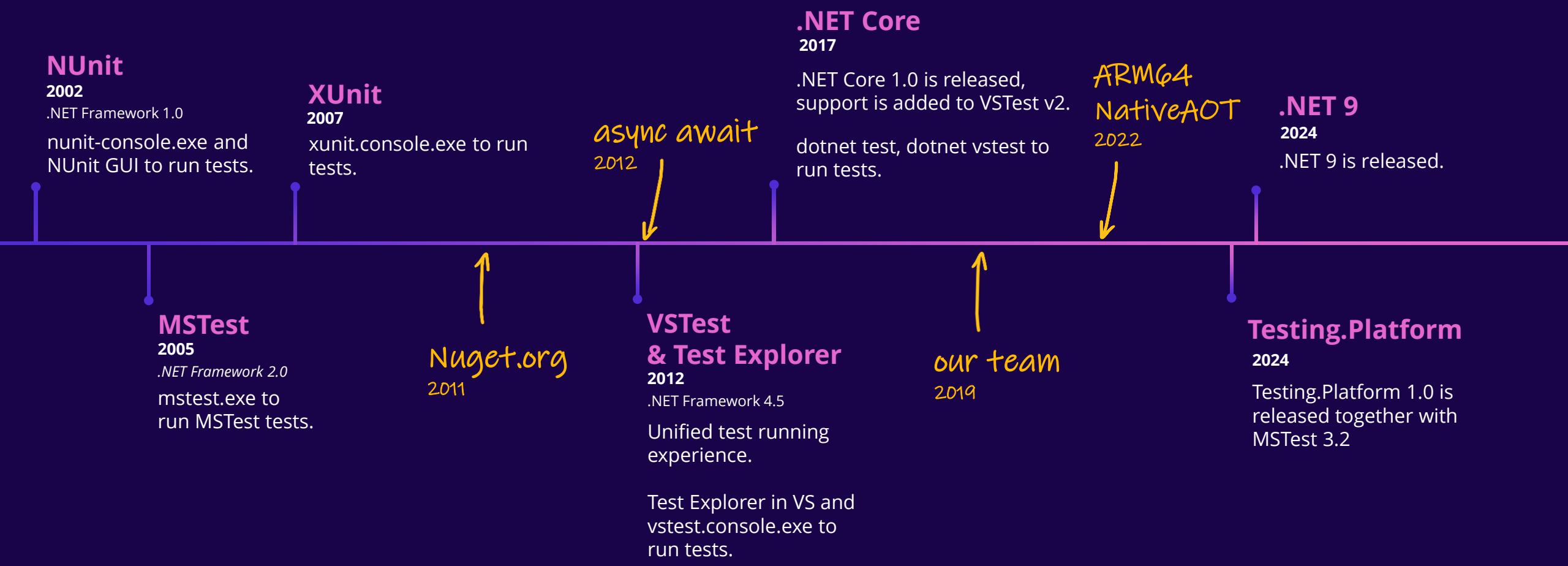
Jakub Jareš

01

Testing.Platform

New, lightweight platform to replace VSTest.

History


NUnit
2002
.NET Framework 1.0
nunit-console.exe and NUnit GUI to run tests.
XUnit
2007
xunit.console.exe to run tests.
async await
2012
.NET Core
2017
.NET Core 1.0 is released, support is added to VSTest v2.
dotnet test, dotnet vstest to run tests.
ARM64 NativeAOT
2022
.NET 9
2024
.NET 9 is released.
MSTest
2005
.NET Framework 2.0
mstest.exe to run MSTest tests.
Nuget.org
2011
VSTest & Test Explorer
2012
.NET Framework 4.5
Unified test running experience.
Test Explorer in VS and vstest.console.exe to run tests.
our team
2019
Testing.Platform
2024
Testing.Platform 1.0 is released together with MSTest 3.2

Deterministic

Portable

Future proof

Extensible

Performant

Secure

Deterministic

Running the same tests, with the same configuration produces the same results on your machine and on build server.

My tests are crashing, because they pick up wrong version of TestAdapter dll from NuGet cache.

Static extension registration.

No reflection lookup, or folder scanning.

No custom assembly resolver.

Portable

Tests are portable, and detached from their surrounding environment. Moving tests from one computer to another is easy.

My test are failing on server, but not locally. I cannot debug the problem.

No folder scanning of VisualStudio or NuGet cache.

No sharing of adapters between projects.

Future proof

Test projects are not special, they can be manipulated and handled as any other dotnet application.

New form factors and architectures should require minimal effort to adopt.

Running with NativeAOT is not possible with `độţŋêţ ţêşţ`.

`độţŋêţ ţsắçê` does not work with tests.

Whole platform compatible with NativeAOT. MSTest added NativeAOT support in preview.

Test projects are simple .exe executables what you can do with console, you can do with tests.

Extensible

The platform is built around extensibility.

Framework authors, or service providers should have easy time extending the platform.

My tests are flaky, how do I retry them?

VSTest relies on Newtonsoft.Json, how do I test Newtonsoft.Json?

We added extension for retrying tests, and other extensions.

The core platform has no dependencies. You can test any dll.

Performant

Overhead of testing platform is minimal. All computing power and memory is spent running your tests.

Test running is slow, vstest.console takes too much memory.

New platform reduced number of started process and serialization.

Teams see 30% increase in speed just by migrating to Testing.Platform.

Secure

Whole platform is shipped as nuget package, with minimal dependencies.

Dependencies of vstest.console are marked as vulnerable in our code compliance checks.

Whole platform is shipped as nugets, we have minimal dependencies.

Security fixes can be shipped next day.

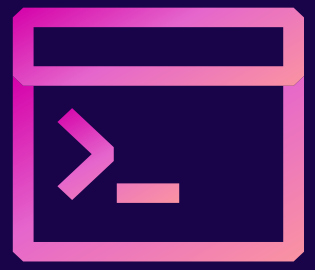
02

Running tests with Testing.Platform

Demo:
Running our first test

Demo summary

Testing.Platform test projects are built as executables.



HelloWorldTests.exe

03

Extension and MSTest project SDK.

Demo:
Using extensions

Demo summary

We have cool extensions.

Hot reload, Retry, Code coverage, Crash dump, Hang dump, Fakes, Output with progress, Trx

https://aka.ms/testingplatform/extensions

MSTest.SDK is a great way to manage your project features and dependencies.

https://aka.ms/mstest/sdk

See also TUnit, a new testing framework using Testing.Platform:

https://github.com/thomhurst/TUnit

04

Running with xUnit and NUnit

Demo:
Running with XUnit and NUnit

Demo summary

XUnit and NUnit both support Testing.Platform.

- Both require preview versions of packages to be used. Please use the latest preview available.

05

Visual Studio, dotnet test

Demo:
Running in VS

Demo summary

Testing.Platform enabled projects are still usable in Visual Studio, and dotnet test.

- MSTest, XUnit and NUnit are all keeping backwards compatibility.

- You can enable Testing.Platform today, and reap the benefits, while keeping your workflow.

06
Summary

We would love your feedback, ideas, improvements, and issues:

github.com/microsoft/testfx

Get .NET 9

Download .NET 9
aka.ms/get-dotnet-9

Resources

Testing.Platform overview

https://aka.ms/testingplatform

GitHub repository

https://github.com/microsoft/testfx

Blog: Introducing MSTest SDK

https://devblogs.microsoft.com/dotnet/introducing-mstest-sdk/

xUnit Testing.Platform support

https://xunit.net/docs/getting-started/v3/microsoft-testing-platform

Blog: Testing Your Native AOT Applications

https://devblogs.microsoft.com/dotnet/testing-your-native-aot-dotnet-apps/

NUnit Testing Platform support

https://learn.microsoft.com/en-us/dotnet/core/testing/unit-testing-nunit-runner-intro

Blog: Introducing the MSTest Runner

https://devblogs.microsoft.com/dotnet/introducing-ms-test-runner/

TUnit documentation

https://thomhurst.github.io/TUnit/

Thank you!