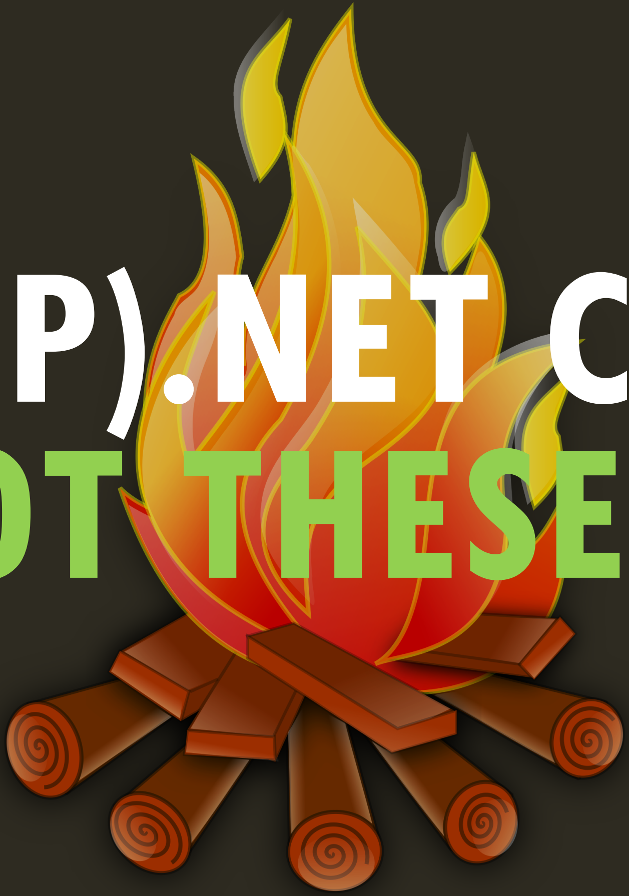
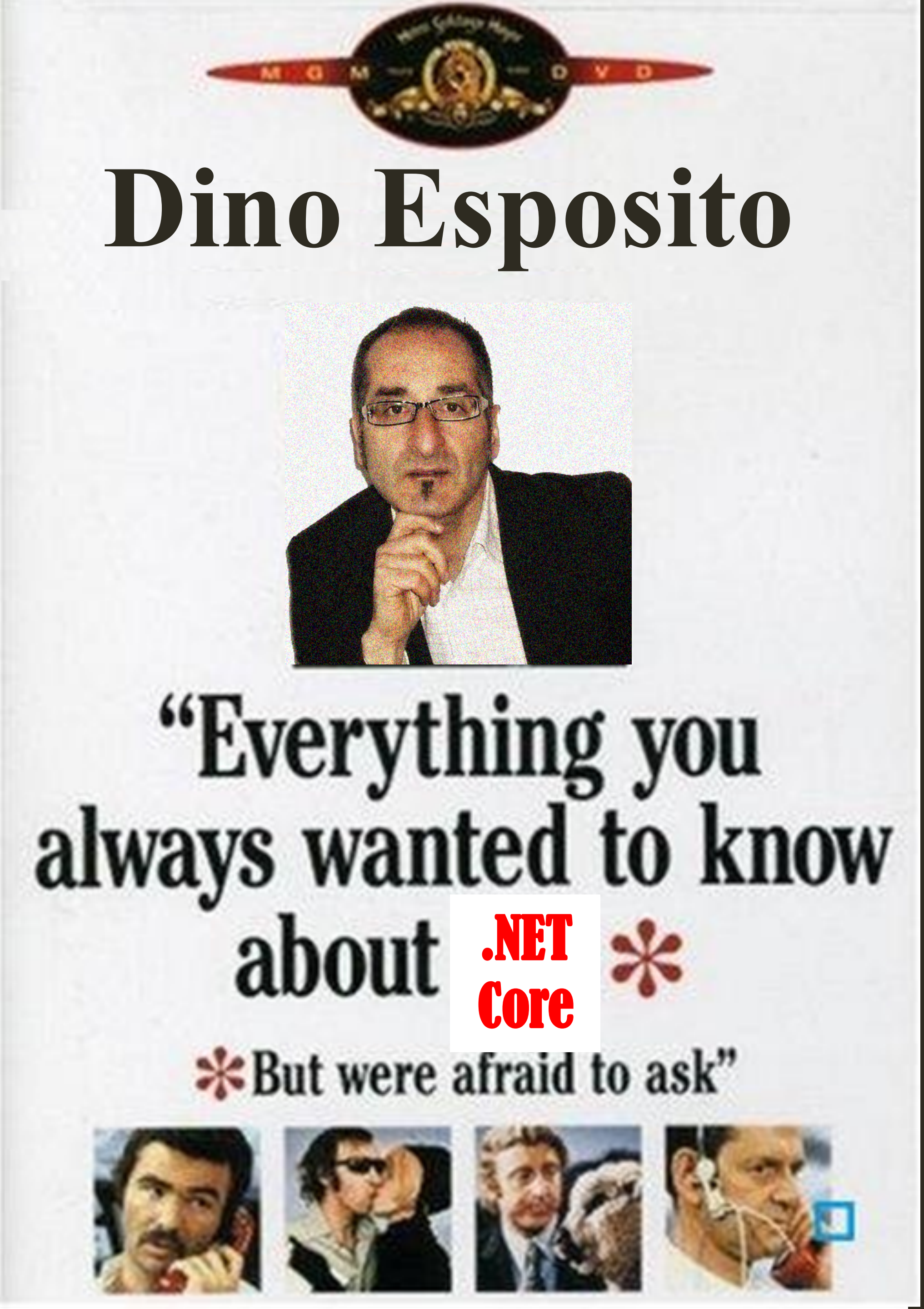
DOTNEXT

MIGRATING TO ASP.NET CORE:
SHOULD I STAY OR SHOULD I GO?

DINO ESPOSITO
@despos
dino.esposito@jetbrains.com

(ASP).NET CORE
IS HOT THESE DAYS





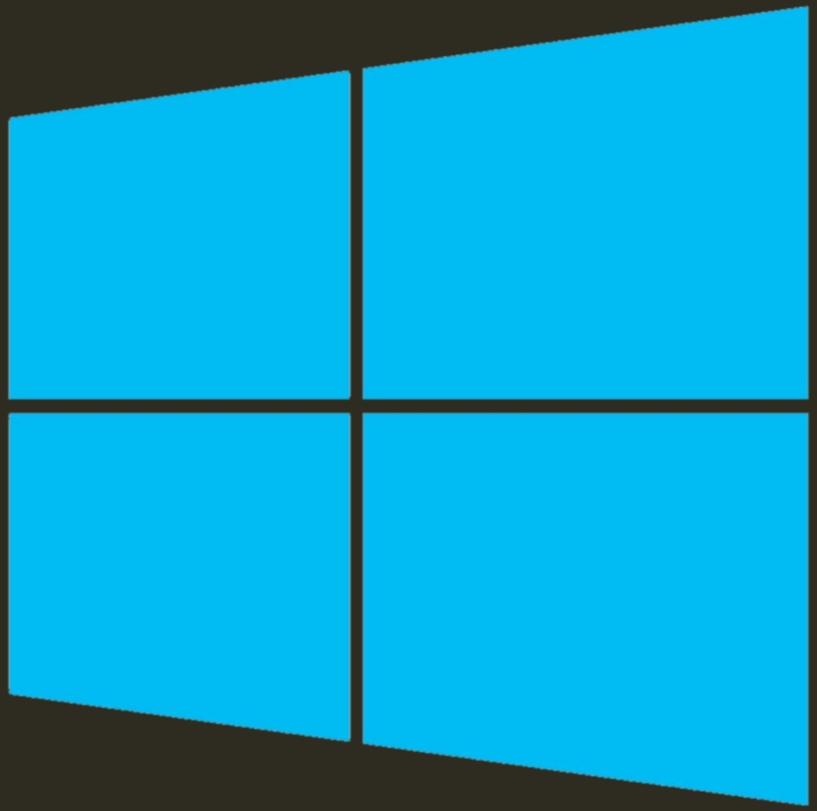
Dino Esposito

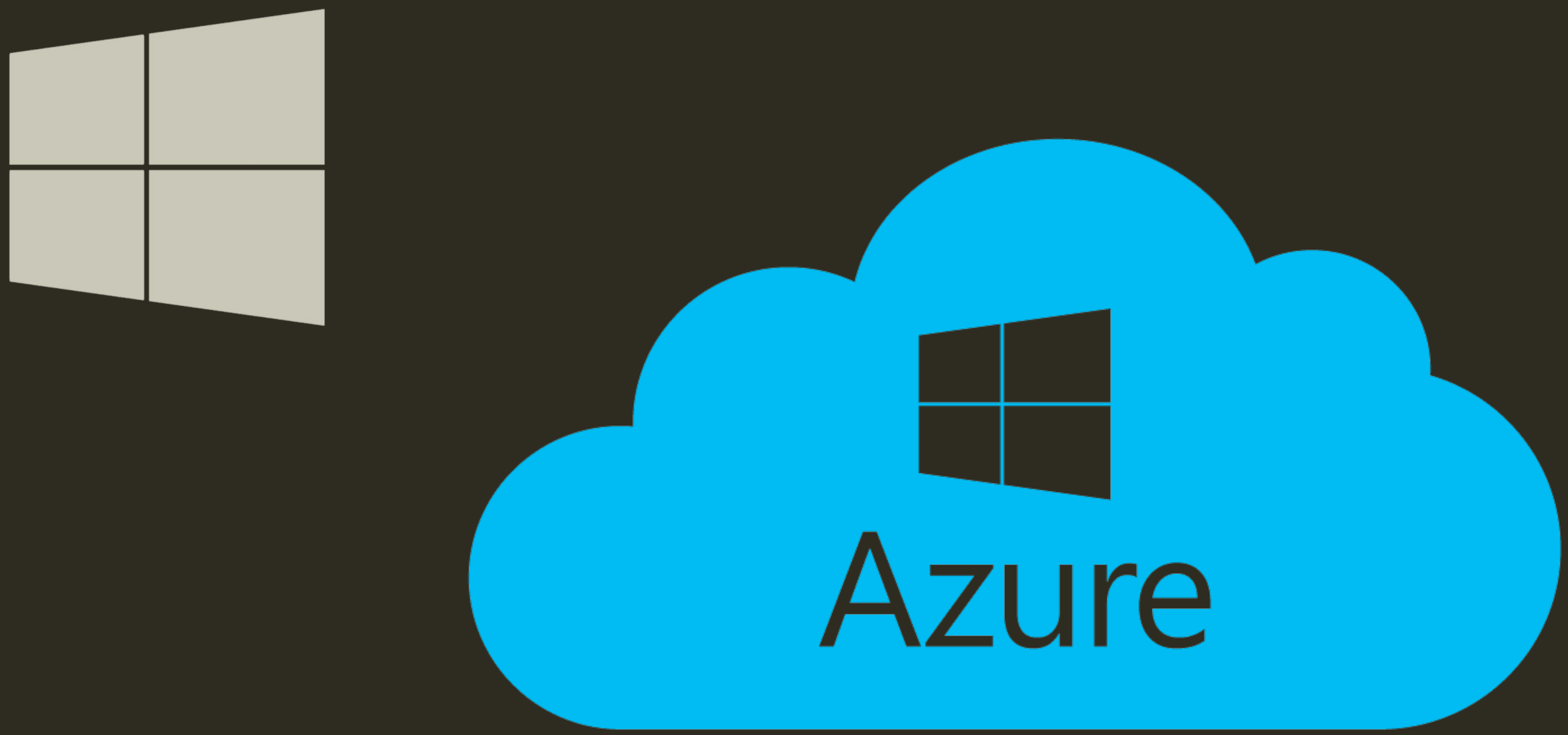
"Everything you always wanted to know about .NET Core *

*But were afraid to ask"

THE UNTOLD STORY OF (ASP).NET CORE

ONCE UPON A TIME...




Azure

[illegible] ♥ [illegible]

[illegible]

WHAT DOES IT MEAN?

MICROSOFT IS LOOKING FOR NEW/MORE CUSTOMERS
TO BRING TO AZURE

The future of Microsoft is all in bringing as many people as possible to the cloud.

Subsequently, everything must be redesigned to play well with (and in) the cloud.

PERSONAL THOUGHTS AND REFLECTIONS

Microsoft’s culture changed in past 10 years

- From “We know how to do things”
- To “Others do it always better than us”

WEB FORMS

ASP.NET MVC

THEN

ASP.NET CORE

ASP.NET CORE was initially supposed to be just the cool new version of ASP.NET but along the way it turned into just one face of a new .NET foundation.

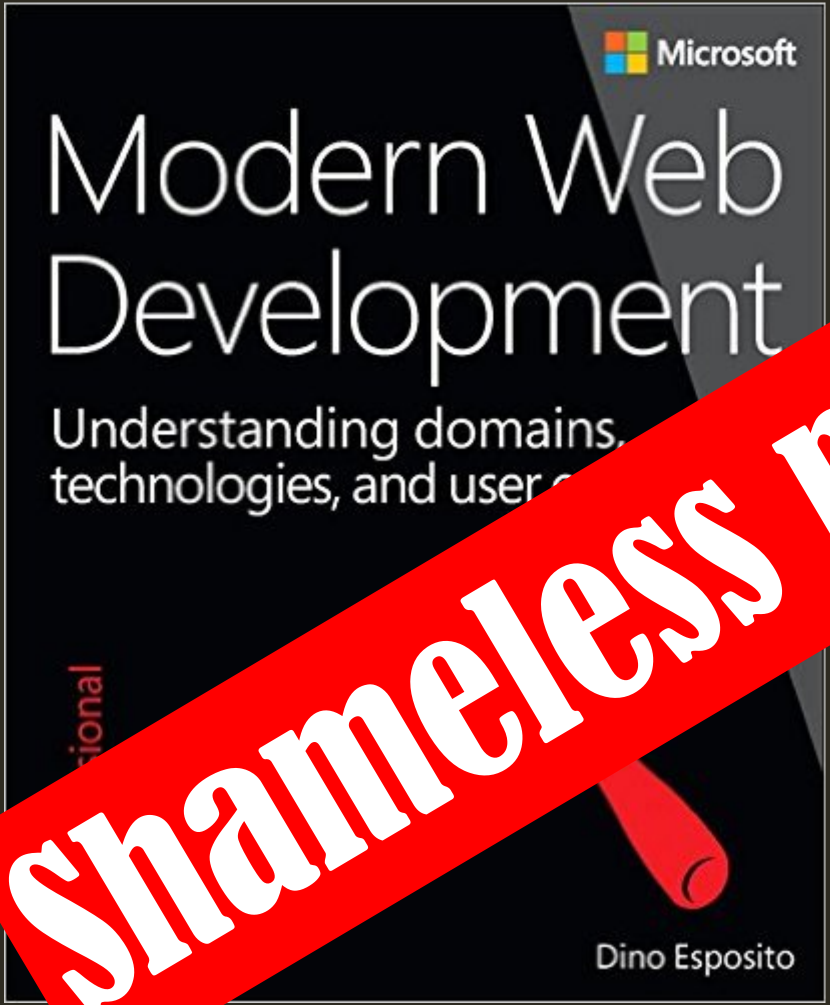
BETA
1...12345

RC
1...1234

RTM
1...123

AND STILL

NO VISUAL BASIC
NO F#
NO SIGNALR



Microsoft

Modern Web Development

Understanding domains, technologies, and user

sional

Dino Esposito

ORDER TODAY !

Shameless plug ☹

MODERN WEB

cript

Mobile stuff

Compact architecture

User experience

MODERN WEB

FACTS	ANSWERS
JavaScript	Use the frameworks you like
Mobile stuff	Use RWD
Compact architecture	Apply design you like
User experience	Apply principles you like
IIS / VS experience	**Here's an alternative**

CROSS-PC

CROSS-PLATFORM

CROSS-TOOLS

HENCE

.NET CORE

THE UNTOLD STORY OF (ASP).NET CORE

AT THE VERY END OF THE DAY

IS THE STORY OF A NEW CROSS-PLATFORM FRAMEWORK

THAT CRAZY LITTLE THING CALLED

LEGACY CODE

THINGS NOBODY **MENTIONS**

What about existing EF code?

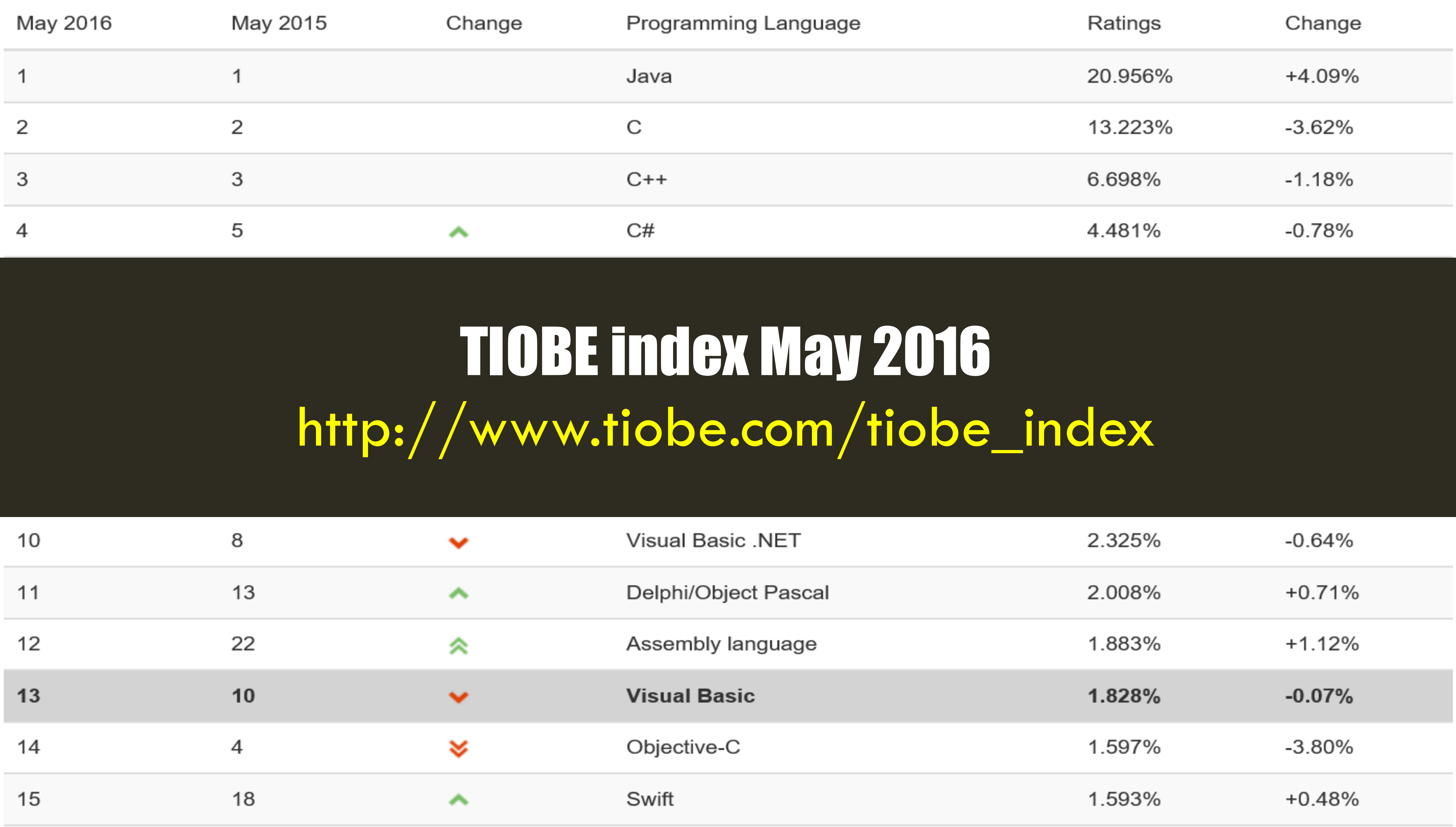
What about existing security layers?

What about existing custom libraries?

Every developer must be trained to ASP.NET Core

THAT CRAZY LITTLE THING CALLED

TRADE-OFF



TIOBE index May 2016

http://www.tiobe.com/tiobe_index

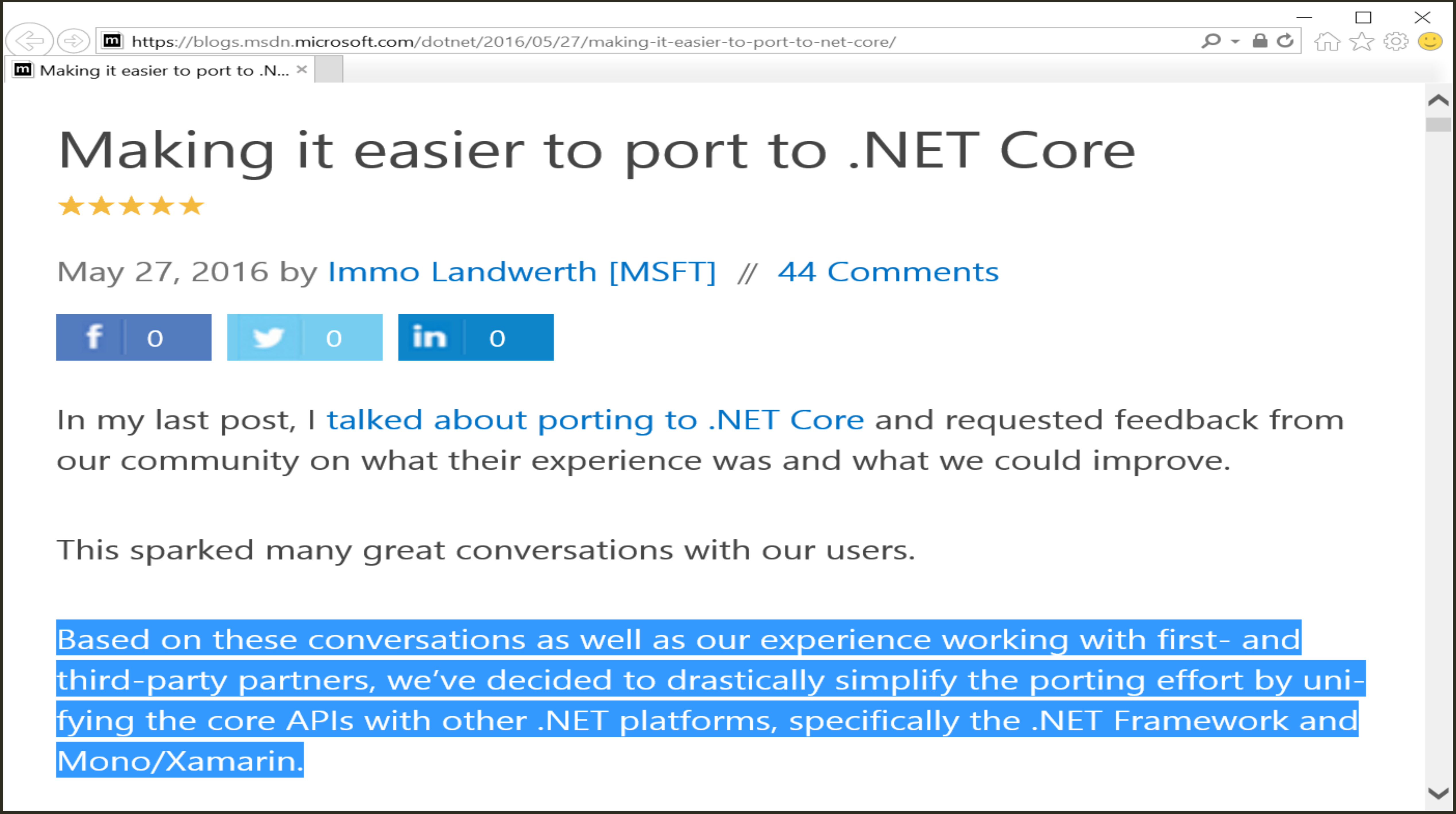
May 2016	May 2015	Change	Programming Language	Ratings	Change
1	1		Java	20.956%	+4.09%
2	2		C	13.223%	-3.62%
3	3		C++	6.698%	-1.18%
4	5	▲	C#	4.481%	-0.78%
10	8	▼	Visual Basic .NET	2.325%	-0.64%
11	13	▲	Delphi/Object Pascal	2.008%	+0.71%
12	22	▲▲	Assembly language	1.883%	+1.12%
13	**10**	▼	**Visual Basic**	**1.828%**	**-0.07%**
14	4	▼▼	Objective-C	1.597%	-3.80%
15	18	▲	Swift	1.593%	+0.48%

ENTERPRISE WORLD **FACTS**

1. We do our very own business and our business is NOT (primarily) helping others to write software
2. We build on stable platforms and we build software that lasts
3. If it works, we tend not to touch it (except technical debt grows)
4. Legacy code? What's NOT legacy?
5. Goal is to move slowly and constantly to never be too far behind (aka, technical debt)

.NET CORE IS HIGHLY INTRUSIVE

1. MUCH MORE THAN .NET WAS 15 YEARS AGO
2. THE WORLD IS DIFFERENT 15 YEARS LATER
3. CUSTOMERS ARE DIFFERENT FROM DEVELOPERS
4. DEVELOPERS ARE MICROSOFT’s PRIMARY CUSTOMERS
5. DEVELOPERS WORK / BUILD for CUSTOMERS


https://blogs.msdn.microsoft.com/dotnet/2016/05/27/making-it-easier-to-port-to-net-core/
Making it easier to port to .N...
f 0
0
in 0

Making it easier to port to .NET Core

★★★★★

May 27, 2016 by Immo Landwerth [MSFT] // 44 Comments

In my last post, I talked about porting to .NET Core and requested feedback from our community on what their experience was and what we could improve.

This sparked many great conversations with our users.

Based on these conversations as well as our experience working with first- and third-party partners, we've decided to drastically simplify the porting effort by unifying the core APIs with other .NET platforms, specifically the .NET Framework and Mono/Xamarin.

Less than **two weeks after** releasing RC2 and announcing RTM

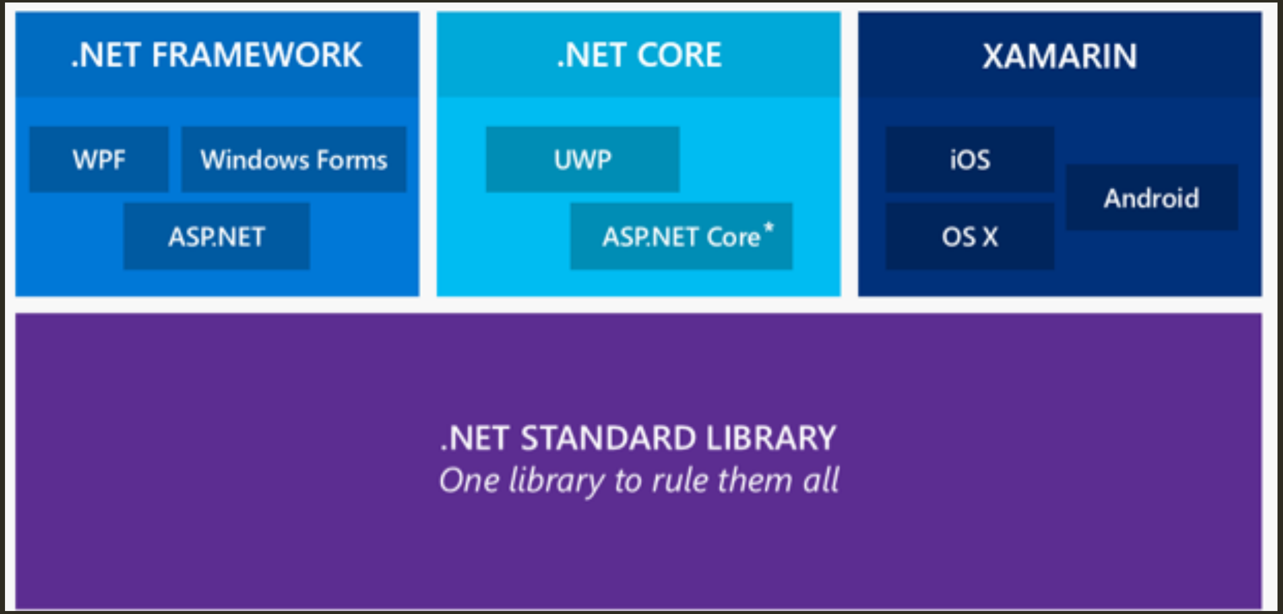
THE **OMG** SERIES -- PART 1

It became clear that adopting .NET Core would require existing .NET developers to spend a considerable amount of time to port to it.

While there is certainly some value in presenting new customers with a cleaner API, it disproportionately penalized our existing loyal customers who have invested over many years in using the APIs and technologies we advertised to them.

We want to extend the reach of the .NET platform and gain new customers, but we can’t do so at the expense of existing users.

THE **OMG** SERIES -- PART 2



The process to extend the API surface of .NET Core will come after we ship .NET Core 1.0 RTM.

THE **OMG** SERIES -- PART 3

Here is the promise we want to make to you:

Whether you need to build a desktop application, a mobile app, a web site, or a micro service: you can rely on .NET to get you there. Code sharing is as easy as possible because we provide a unified BCL. As a developer, you can focus on the features and technologies that are specific to the user experiences and platforms you're targeting.

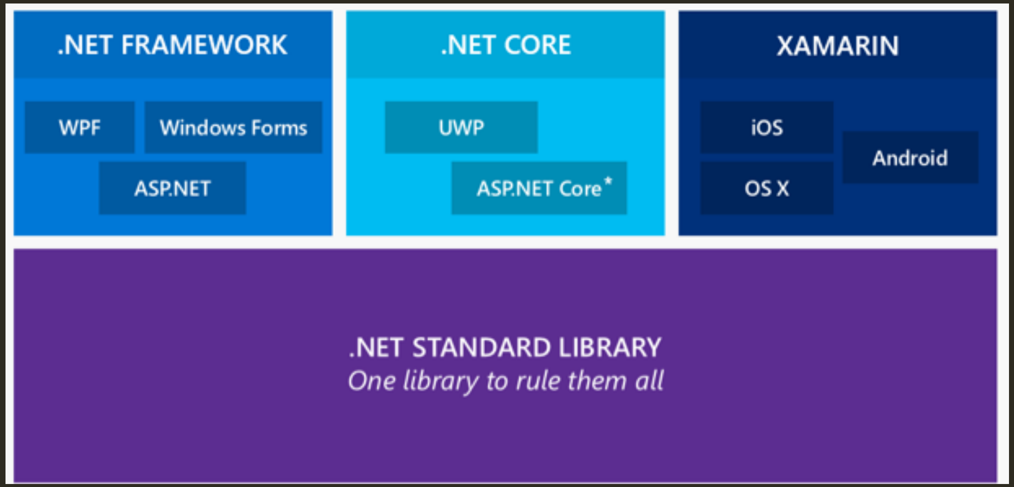
DICTATORSHIP OF THE “INSTANT”

The moment something is announced it is perceived as REAL and EXISTING and WORKING

THE **XAMARIN** PATTERN

Xamarin did not start with reimagining the .NET stack

Xamarin took Mono (nearly identical to .NET), removed the .NET specific application model components (Windows.Forms, ASP.NET) and added new models for iOS and Android



IN THE END?

Transitioning to a new platform occurs over 5 years

It all started in 2014 (discussions even earlier)

Realistically .NET CORE won’t be ready for primetime for 2+ more years

Exceptions are hopefully possible

.NET Core 2.0 is bare minimum (2017 H2 perhaps?)

IN THE END?

.NET Core works well for the scenarios that we set out to address, it provides access to fewer technologies than other .NET platforms, especially the .NET Framework.

If you fit NOW in those scenarios, then it's fine and safe.

If not, just wait until it stabilizes.

http://imgur.com/user/tougherguythanu

MUST READ

http://blogs.msdn.microsoft.com/dotnet/2016/05/27/making-it-easier-to-port-to-net-core/

Thx!

Q&A