

.NET Core Interview Questions & Answers

.NET
Core



By Satyaprakash Samantaray

An Author and Blogger

 **DotNetTricks**

.NET Core Interview Questions & Answers

All rights reserved. No part of this book can be reproduced or stored in any retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, uploading on server and scanning without the prior written permission of the Dot Net Tricks Innovation Pvt. Ltd.

The author of this book has tried their best to ensure the accuracy of the information described in this book. However, the author cannot guarantee the accuracy of the information contained in this book. The author or Dot Net Tricks Innovation Pvt. Ltd. will not be liable for any damages, incidental or consequential caused directly or indirectly by this book.

Further, readers should be aware that the websites or reference links listed in this book may have changed or disappeared between when this book was written and when it is read.

All other trademarks referred to in this book are the property of their respective owners.

Release History

- Initial Release 1.0 - 16th Apr 2020
- Second Release 2.0 - 25th Jan 2022

About Dot Net Tricks

Dot Net Tricks is founded by Shailendra Chauhan (Microsoft MVP), in Jan 2010. Dot Net Tricks came into existence in the form of a blog post over various technologies including .NET, C#, SQL Server, ASP.NET, ASP.NET MVC, JavaScript, Angular, Node.js and Visual Studio, etc.

The company which is currently registered by a name of Dot Net Tricks Innovation Pvt. Ltd. came into shape in 2015. Dot Net Tricks website has an average footfall on the tune of 300k+ per month. The site has become a cornerstone when it comes to getting skilled-up on .NET technologies and we want to gain the same level of trust in other technologies. This is what we are striving for.

We have a very large number of trainees who have received training from our platforms and immediately got placement in some of the reputed firms testifying our claims of providing quality training. The website offers you a variety of free study material in the form of articles.

Unlimited Live Training Membership

Upgrade your skills set with hands-on real-time project-based training programs to build expertise on in-demand job skills and become industry competent. DotNetTricks Unlimited Live Training enables you to Become:

- **Full-stack JavaScript Developer** - JavaScript, Node.js, React, Angular
- **Full-stack .NET Developer** - .NET, MVC, ASP.NET Core, WebAPI
- **Cloud Engineer/Architect** - AWS, Microsoft Azure
- **Technical Architect** - Microservices, Design Patterns and Clean Architecture
- **DevOps Engineer** - DevOps, Docker and Kubernetes
- **Mobile Developer** - Xamarin, React Native

Learn more about Unlimited Live here: <https://www.dotnettricks.com/membership>

Self-Paced Course Membership

The most effective way to gain job-ready expertise for your career. Learn how to build highly scalable modern web applications & transform your coding skills. Learn to build projects and build expertise on .NET, JavaScript, Database, Cloud, DevOps, Docker, Front-end technologies and many more cutting-edge technologies which are in industry demand today.

- 5,00+ hours of unlimited access to our premium content
- Real Hands-on Labs with integrated IDE
- Full access to training, study mode quizzes, and assignments
- Step-by-step learning with exclusive Learning Paths
- Become job-ready with interview prep sessions

Learn more about Self-paced training membership here: <https://www.dotnettricks.com/plus-membership>

Interview Q&A eBooks

Dot Net Tricks offer a wide range of eBooks on technical interviews Q&A. industry experts and coaches write all eBooks. These eBooks will help you to prepare yourself for your next job within a short time. We offer the eBooks in the following categories:

- .NET Development
- Front-end Development
- Cloud
- DevOps
- Programming Languages
- Database - SQL and NoSQL
- Mobile Development
- ML/AI and many more...

For other eBooks, do refer to <https://www.dotnettricks.com/books>

Corporate Training

Dot Net Tricks has a pool of mentors who help the corporation enhance its employment skills by changing the technology landscape. Dot Net Tricks offers customized training programs for new hires and experienced employees through online and classroom modes. As a trusted and resourceful training partner, Dot Net Tricks helps the corporation achieve success with its industry-leading instructional design and customer training initiatives.

Apart from these, we also provide on-demand boot camps and personalized project consultation.

For more details about Corporate Training, do refer to <https://www.dotnettricks.com/corporate-training>

Technical Recruiting

We provide a full technical staffing service, which suits our client's needs. Our specialized recruiters search worldwide to find highly skilled professionals that will fit our client's needs. If you are looking for a job change, do share your resume at hr@dotnettricks.com. Dot Net Tricks will help you to find your dream job in MNCs.

Join us today, learn to code, prepare yourself for interviews, and get hired!

Dedication

To my parents, Mr Kedarnath Samantaray(Father) and Mrs Charulata Samantaray(Mother)for everything they did for me. I would like to thank my friends, Deb and Gourav, for their ideas and feedback on the technical aspects of this book.

- Satyaprakash Samantaray

Introduction

What Where Author Qualification to Write This Book

Satyaprakash Samantaray is a software developer currently working with an MNC company in Bangalore, India. He acquired a bachelor's degree in E&TC from Biju Patnaik University of Technology, Odisha. He has over six years of extensive experience in Microsoft technologies including C#, ASP.NET, MVC, SQL Server, Entity Framework as well as in other technologies such as jQuery, CSS, Angular and Bootstrap and also has experience in Business intelligence tools like MSBI tool.

He is a technical author and speaker who loves to contribute to the technical community. He writes articles for multiple platforms including Microsoft ASP.NET Community and TechNet Wiki. For his dedicated contribution to the developer community, he has been recognized as an MVP (Microsoft Most Valuable Professional), MVB (Most Valuable Blogger) and Monthly award winner in Microsoft TechNet wiki.

What This Book Is

.NET Core Interview Questions and Answers book aims to help you to understand the newly introduced .NET framework .Net Core so that you can use .NET Core for developing the modern .NET application with the help of .NET. In this book, each topic has been explained with suitable examples. This book has been written by referring to the .NET Core documents official website at www.docs.microsoft.com/en-us/dotnet/core.

What You'll Learn

This book will help you to learn Python basic and advanced topics. This book covers the following topics:

- Introduction to .NET 4.x
- .NET Core and Features
- .NET 5
- .NET 6
- .NET Core SDK
- .NET Core Compilation
- Core CLR and RyuJIT
- Garbage Collector
- .NET CLI
- .NET Standard
- VS Code with .NET Core
- Porting .NET to .NET Core

Hope you will enjoy this book and find it valuable for yourself.

Our best wishes are always with you for your learning and growth!

About the Author

Satyaprakash Samantaray - An Author and Blogger



He is working as a software developer in an MNC organization and has more than 6 years of experience. He is very passionate about Microsoft Technologies. He is an author, speaker, MVP and MVB.

He has the experience to develop enterprise applications using Microsoft technologies such as ASP.NET MVC, Entity Framework, ASP.NET Core, C#, SQL server, etc. and other technologies such as Angular, Bootstrap as well as in MSBI tool etc.

During his education, he has achieved awards for participating in Quiz, Speech at the state level. He received medals for participating in the General Knowledge competition at the State level.

He always tries to keep updated himself about new technologies, learning new skills and sharing with others in a better and easy way.

He hopes that this e-book will help you to crack your interview on .NET Core. This is the first edition of this book but not the last. Please provide your feedback that help us to improve this book quality.

How to Contact Us

Although the author of this book has tried to make this book as accurate as possible, if something strikes you as odd or finds an error in the book, please drop a line via e-mail.

The e-mail addresses are listed as follows:

- mentor@dotnettricks.com
- info@dotnettricks.com

We are always happy to hear from our readers. Please provide your valuable feedback and comments!

You can follow us on [YouTube](#), [Facebook](#), [Twitter](#), and [LinkedIn](#) or subscribe to our [RSS feed](#).

Table of Contents

.NET Core Interview Questions & Answers	2
Release History	2
About Dot Net Tricks	3
Unlimited Live Training Membership	3
Self-Paced Course Membership	3
Interview Q&A eBooks	4
Corporate Training.....	4
Technical Recruiting	4
Dedication.....	5
Introduction.....	6
About the Author.....	7
How to Contact Us.....	8
Introduction to .NET 4.x.....	17
Q1. Explain the version history of .NET?.....	17
Q2. What new features were added in .Net 4.0?	17
Q3. Where .NET Framework 4.0 can run?	17
Q4. What is the purpose behind Windows Server AppFabric in .Net 4.0?	18
Q5. Which CLR Version is used by .Net 4.5?	18
Q6. What is CustomReflectionContext class in .Net 4.5?	18
Q7. Which version of encoding does the console support in .Net 4.5?.....	18
Q8. What are .Net 4.5.1 features and when it was released?	18
Q9. What features .Net 4.5.2 provides for Asp.Net?	18
Q10. What are .Net 4.6 features and when it was released?	18
Q11. What are .Net 4.6.1 features and when it was released?	18
Q12. What are .Net 4.6.2 features and when it was released?	18
Q13. What are .Net 4.7 features and when it was released?	19
Q14. What are .Net 4.7.1 features and when it was released?	19
Q15. What are .Net 4.7.2 features and when it was released?	19
Q16. What are .Net 4.8 features and when it was released?	19

.NET Core/.NET Core 3.x	20
Q1. What is .NET Core?	20
Q2. Does RHEL support .NET Core?	20
Q3. What are the features of the .NET Core?	20
Q4. Which languages are used to create libraries for .NET Core?	21
Q5. What IDE can be used for .NET Core development?	21
Q6. What is about .NET Core compatibility with Mono APIs?	21
Q7. Explain about .Net core components?	21
Q8. What is the difference between .NET Core runtime and .NET Core SDK?	21
Q9. What are the versions of .NET Core/.NET?	21
Q10. How .NET Core does work with MVC and Web APIs?	22
Q11. How does Dependency Injection work in .NET Core?	22
Q12. What does .NET Core provide for Angular and React developers?	22
Q13. What does .NET Core provide to store static files?	22
Q14. What does .NET Core provide for deployment?	22
Q15. What does .NET Core provide for Mobile App Development?	22
Q16. What is new in .NET Core 3.0 version?	22
Q17. How .NET Core 3.0 app is now executable by default?	23
Q18. How .NET Core 3.0 will help you in IoT development?	23
Q19. Which operating systems are supported by .NET Core 3.0?	23
Q20. Which chips are supported by .NET Core 3.0?	23
Q21. What is the role of Nullable reference types of C# 8.0 in .NET Core 3.0?	23
Q22. What is HTTP/2 Support in .NET Core 3.0?	24
Q23. What are the benefits of using Docker in .NET Core 3.0?	24
Q24. What technologies .NET Core does not support?	24
Q25. Why .NET Remoting technology isn't available with .NET Core?	24
Q26. Why Code Access Security isn't available with .NET Core?	24
.NET 5	25
Q1. What is .NET 5?	25
Q2. What new features are available in .NET 5?	26
Q3. What is the Mono AOT compiler in .NET 5?	26

Q4.	How many types of AOT solutions for Mono AOT?.....	26
Q5.	What exceptions are found in Mono AOT?.....	26
Q6.	What is CoreFX framework in .NET 5?.....	26
Q7.	How does .NET 5 work with .NET CLI?.....	26
Q8.	What is the objective of .NET 5?	26
.NET 6		27
Q1.	What is .NET 6?.....	27
Q2.	What does SDK workload support in .NET 6?.....	27
Q3.	Which features are updated in SDK workload to add list and update verbs?	27
Q4.	What are the uses of the reflection mechanism and its drawbacks?	28
Q5.	What is Compile-Time Source Generator?	28
Q6.	What is OpenTelemetry Metrics and its benefits?.....	28
Q7.	What is OpenSSL in .NET 6?.....	28
Q8.	What are Roslyn Analyzers in .NET 6?	28
Q9.	What is WebSocket Compression and its advantages?.....	28
Q10.	Which framework version is not supported in .NET 6?.....	28
Q11.	What is NuGet package validation tool in .NET 6?.....	28
Q12.	What is the role of .NET Hot for ASP.NET Core & Blazor Projects?.....	29
Q13.	What are the benefits of using Interface Casting in .NET 6?.....	29
Q14.	What new collection is added in .NET 6?	29
Q15.	How does JsonSerializer work in .NET 6?	29
Q16.	What are two ignore options available in System.Text.Json in .NET 6?.....	29
Q17.	What are the Improvements of the client app development in .NET 6?	29
Q18.	How to check the available versions of .NET SDK and Runtime in .NET6?.....	29
Q19.	How to enable compression for Single-file bundles in .NET 6 and why it is needed?	29
Q20.	What is .Net MAUI and how does it associate with .NET 6?	30
Q21.	What is Blazor desktop apps in .NET 6?	30
Q22.	What is the new project template in .NET 6?	30
Q23.	What is the keyword called global in C# 10?.....	30
Q24.	What is implicit using the directives feature in .NET 6?.....	30
Q25.	What are the commands required for SDK workload in .NET 6?	30

.NET Core SDK and Compilation	31
Q1. What is .NET Core SDK?	31
Q2. What components does .NET Core SDK contain?	31
Q3. What are the different ways to install the SDK On Windows?	31
Q4. How to erase the .NET Core Runtime and SDK?	31
Q5. What is the NuGet fallback folder?	31
Q6. What is the path of the NuGet fallback folder on Windows and macOS?	32
Q7. In which case the NuGet fallback folder can be removed?	32
Q8. How does Docker work with .NET Core SDK?	32
Q9. What are the different ways to install the SDK On Linux?	32
Q10. What are the different ways to install the SDK on macOS?	32
Q11. What compilers are used in the .NET Core execution process?	32
Q12. What is the reimplement of the class libraries for .NET Core?	32
Q13. What is the status of tiered compilation in .NET Core 2.x and 3.x?	32
Q14. What are the setting names of Tiered compilation in .NET Core?	33
Q15. What is Roslyn and its primary features?	33
Q16. What are the different types of APIs for Roslyn?	33
Q17. What is the Feature APIs of Roslyn?	33
Q18. What is the Work-space APIs of Roslyn?	33
Q19. What are the Compiler APIs of Roslyn?	33
Core CLR and RyuJIT	34
Q1. What is CoreCLR?	34
Q2. How does CoreCLR work with Console App?	34
Q3. What is the difference between CoreCLR and Corefx?	35
Q4. What are the lines of code in CoreCLR?	35
Q5. How to find CoreCLR in the cross-platform system?	35
Q6. What are the types of CoreClrHost methods used for hosting .NET Core?	35
Q7. What is RyuJIT?	35
Q8. What is the difference between RyuJIT and Roslyn compiler?	35
Q9. What features are provided by RyuJIT?	35
Q10. What is SIMD does support via RyuJIT?	36

Q11. How does RyuJIT define in case of performance case?	36
Q12. What is the difference between RyuJIT and JIT32?.....	36
Garbage Collector	37
Q1. What is garbage collection in .NET Core?	37
Q2. What are the advantages of garbage collection in .NET Core?	37
Q3. What are the conditions for garbage collection in .NET Core?	37
Q4. What are the flavors of garbage collection in .NET Core?.....	37
Q5. What are the settings to select flavors of GC in .NET Core?.....	37
Q6. What are the settings to manage the garbage collector's memory and processor usage?	38
Q7. What are the values of System.GC.HeapCount and COMPlus_GCHeapCount settings?.....	38
Q8. What is Thread_UseAllCpuGroups element option?.....	38
Q9. How to enable Thread_UseAllCpuGroups element option for .NET Core apps?	38
Q10. What is Standalone GC in .NET Core?	38
.NET CLI.....	39
Q1. What is .NET Core command-line interface (CLI)?	39
Q2. How to get .NET Core command-line interface (CLI)?	39
Q3. How to verify the .NET Core command-line interface (CLI) is installed?	39
Q4. How does .NET Core CLI install by default?	39
Q5. What are the benefits of using .NET Core CLI commands?	40
Q6. What is the Command Structure of .NET Core CLI?.....	40
Q7. What is the CLI command to create the project in the C# language?	40
Q8. How to get the list of available .NET CLI commands?	40
Q9. What is the basic .NET Core CLI commands?	41
Q10. What is the .NET Core CLI project modification commands?	41
Q11. What are the .NET Core CLI project advanced commands?	42
Q12. Which command is used to add Newtonsoft.json package to our console project?	42
Q13. How to create new .NET Core project using template name argument?.....	42
Q14. How you can create and run a console application using .NET CLI?.....	42
Q15. Which command does create a new console application name to the project directory?	43
Q16. How to create a .NET Standard class library project in the specified directory?.....	43
Q17. How to Create a new ASP.NET Core C# MVC project in the current directory with no authentication?	43

Q18.	How to create a new xUnit project?.....	43
Q19.	How to get list all templates available for MVC?.....	44
Q20.	How to get List all templates matching the mentioned(“se”) substring?.....	44
Q21.	How to get List all templates matching the mentioned(“ng”) template?	44
Q22.	Which CLI command is used to install version 2.0 of the SPA templates for ASP.NET Core?	44
Q23.	Which CLI command is used to create a global.json in the current directory?	44
.NET Standard		45
Q1.	What is .NET Standard?	45
Q2.	What are the differences between .NET Standard and Portable Class Libraries?	45
Q3.	How to describe .NET Standard compatibility with PCL profiles?	46
Q4.	What does .NET standard Support by .NET implementations?.....	46
Q5.	What is new in the .NET Standard 2.0?.....	46
Q6.	What is not included in .Net Standard?	46
Q7.	What is the Portable Class Library?	46
Q8.	What are the disadvantages of Portable Class Library?	46
Q9.	How do I identify which .NET Standard version I should target?	46
Q10.	What are the benefits of targeting .NET Standard 2.0?.....	47
Q11.	How do PCL profiles map with the .NET standard?.....	47
VS Code With .NET Core.....		49
Q1.	What is VS Code?	49
Q2.	What does enable VS Code to be cross-platform?	49
Q3.	How many icons are in VS Code sidebar?	49
Q4.	What are the keyboard shortcuts for quick file picker and hiding the sidebar?	49
Q5.	What is the keyboard shortcut difference between mac, Windows and Linux?	49
Q6.	How does Git work in VS code?	50
Q7.	What is the Activity Bar in VS code?	50
Q8.	What is the Integrated Terminal in VS Code?	50
Q9.	What is the Command Palette in VS code?	50
Q10.	What common operations are performed using the Command Palette?	50
Q11.	How to open a new window in VS Code with the content of the current folder?	50
Q12.	How to show the difference between two files using VS Code?	50

Q13.	What are .NET CLI options in VS Code?.....	50
Q14.	How to get a list of command-line options in VS Code?	51
Q15.	How to configure for launching VS Code from the command line in different platforms?	51
Q16.	How to launch VS Code Insiders build?.....	51
Q17.	Do share important options used with VS Code?	51
Q18.	How many Core CLI options can use for Opening VS Code with URLs?	51
Q19.	How to solve 'code' is not recognized as an internal or external command?	52
Q20.	How to open a command line (terminal) from within VS Code?	52
Q21.	How to install and manage VS Code extensions?.....	52
Q22.	What various options are used for installing and managing VS Code extensions?.....	52
Q23.	How to find the path of the extensions is installed?	52
Q24.	How to change the path of the extensions by launching VS Code?	53
Q25.	How to control proxy settings using the command line?.....	53
Q26.	How to manage VS Code by providing extension recommendations?	53
Porting .NET to .NET Core		54
Q1.	How to port your code to .NET Core or .NET Standard?	54
Q2.	How to Migrate the NuGet packages to PackageReference?	54
Q3.	How to Upgrade the NuGet packages?	54
Q4.	What to do when NuGet package dependency failed to run on .NET Core?	54
Q5.	What is the .NET Portability Analyzer tool?	54
Q6.	How to analyze dependencies if they aren't NuGet packages?.....	54
Q7.	What is the Target Framework Monikers (TFMs)?.....	55
Q8.	How to analyze the converted and upgraded package dependencies work on .NET Core?	55
Q9.	What are Windows Compatibility Pack and its package contents?.....	55
Q10.	What is the .NET Framework compatibility mode?.....	55
Q11.	Why do you retarget to .NET Framework 4.7.2?	55
Q12.	What are the steps to retarget to .NET Framework 4.7.2?	55
Q13.	What is the testing framework that builds and runs tests for .NET Core.?	55
Q14.	How to port your code with the base of your library?	56
Q15.	What are those tools to help with porting to .NET Core?	56
Q16.	How to generate a portability report for analysis?.....	56

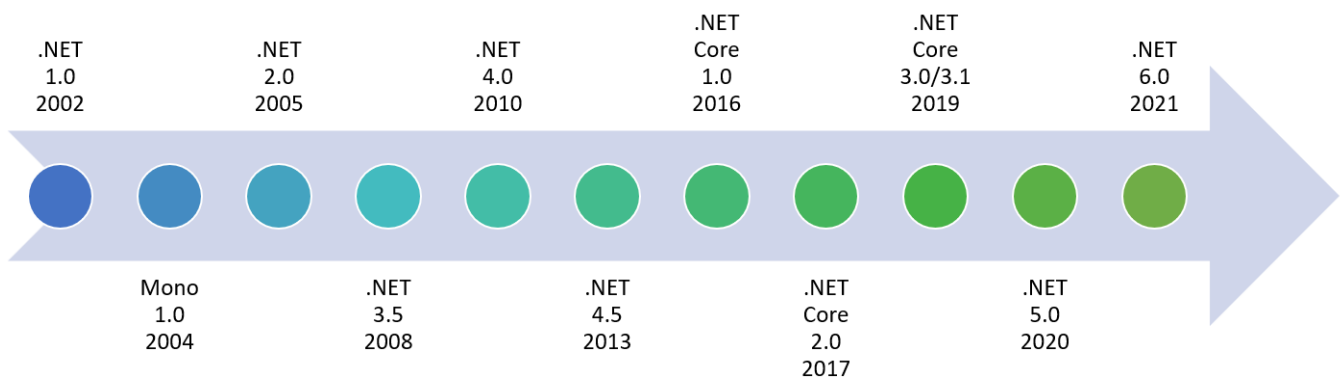
Q17.	What is the Target Framework Monikers (TFMs)?.....	56
Q18.	How to analyze the converted and upgraded package dependencies work on .NET Core?	56
Q19.	What are Windows Compatibility Pack and its package contents?.....	56
Q20.	What is .NET Framework compatibility mode?	56
Q21.	Why do you retarget to .NET Framework 4.7.2?	57
Q22.	What are the steps to retarget to .NET Framework 4.7.2?	57
Q23.	What is the testing framework that builds and runs tests for .NET Core.?.....	57
Q24.	How to port your code with the base of your library?	57
Q25.	What are those tools to help with porting to .NET Core?	57
Q26.	How to generate a portability report for analysis?.....	57
Q27.	What is the Target Framework Monikers (TFMs)?.....	58
Q28.	How to analyze the converted and upgraded package dependencies work on .NET Core?	58
Q29.	What are Windows Compatibility Pack and its package contents?.....	58
Q30.	What is the .NET Framework compatibility mode?.....	58
Q31.	Why do you retarget to .NET Framework 4.7.2?	58
Q32.	What are the steps to retarget to .NET Framework 4.7.2?	58
Q33.	What is the testing framework that builds and runs tests for .NET Core.?.....	58
Q34.	How to port your code with the base of your library?	59
Q35.	What are those tools to help with porting to .NET Core?	59
Q36.	How to generate a portability report for analysis?.....	59
References		60

1

Introduction to .NET 4.x

Q1. Explain the version history of .NET?

Ans. The version history of .NET is shown in a given pic.



Q2. What new features were added in .Net 4.0?

Ans. The following new features are added:

- Dynamic binding
- Named arguments
- Optional Parameters
- Generic Covariance and Contra-variance
- COM Interop

Q3. Where .NET Framework 4.0 can run?

Ans. .NET Framework 4.0 is supported on the following platforms:

- Windows XP & Vista
- Windows Server 2003 and higher
- Windows 7 and Higher

Q4. What is the purpose behind Windows Server AppFabric in .Net 4.0?

Ans. It is used for application server capabilities about AppFabric Hosting including in-memory distributed caching support.

Q5. Which CLR Version is used by .Net 4.5?

Ans. The .NET Framework 4.5 uses Common Language Runtime 4.0.

Q6. What is CustomReflectionContext class in .Net 4.5?

Ans. This class is used to customize a reflection context to override the default reflection property.

Q7. Which version of encoding does the console support in .Net 4.5?

Ans. It supports Unicode (UTF-16) encoding.

Q8. What are .Net 4.5.1 features and when it was released?

Ans. .NET Framework 4.5.1 was announced on 17 October 2013 with Visual Studio 2013. It's having the following features:

1. Debugger support for analysing .NET memory dumps and for watching managed return values.
2. Coding analysis for better User interface improvements.

Q9. What features .Net 4.5.2 provides for Asp.Net?

Ans. It was announced on 5 May 2014. For ASP.NET, it provides the below mentioned things:

1. HTTP header inspection.
2. HTTP header modification methods.

Q10. What are .Net 4.6 features and when it was released?

Ans. .NET Framework 4.6 was released on 20 July 2015. Its features are mentioned below:

1. It supports a new just-in-time compiler supports for 64-bit systems called RyuJIT.
2. TLS 1.1 and TLS 1.2 are added to WCF.

Q11. What are .Net 4.6.1 features and when it was released?

Ans. .NET Framework 4.6.1 was released on 30 November 2015. Its features are mentioned below:

1. Azure SQL Database supports distributed transactions.
2. SQL Connectivity for AlwaysOn, Always Encrypted is added.
3. Performance related issues are fixed in RyuJIT, WPF and WCF.

Q12. What are .Net 4.6.2 features and when it was released?

Ans. .NET Framework 4.6.2 was released on August 2, 2016. Its features are mentioned below:

1. It supports data annotations localization in ASP.NET.
2. It supports TLS 1.1/1.2 support.

Q13. What are .Net 4.7 features and when it was released?

Ans. .NET Framework 4.7 was released on 5 April 2017. Its features are mentioned below:

1. It supports elliptic curve cryptography.
2. It supports New APIs for WPF.
3. It supports TLS version 1.2.

Q14. What are .Net 4.7.1 features and when it was released?

Ans. .NET Framework 4.7.1 was released on 17 October 2017. Its features are mentioned below:

1. It provides the .NET Standard 2.0 compatibility.
2. The d3dcompiler dependency issue is resolved.

Q15. What are .Net 4.7.2 features and when it was released?

Ans. .NET Framework 4.7.2 was released on 30 April 2018. Its features are mentioned below:

1. This version supports Windows Server 2019.
2. Many new features are added to ASP.NET, BCL, CLR, ClickOnce, Networking, SQL, WCF, Windows Forms, Workflow and WPF.

Q16. What are .Net 4.8 features and when it was released?

Ans. .NET Framework 4.8 was released on 18 April 2019. Its features are mentioned below:

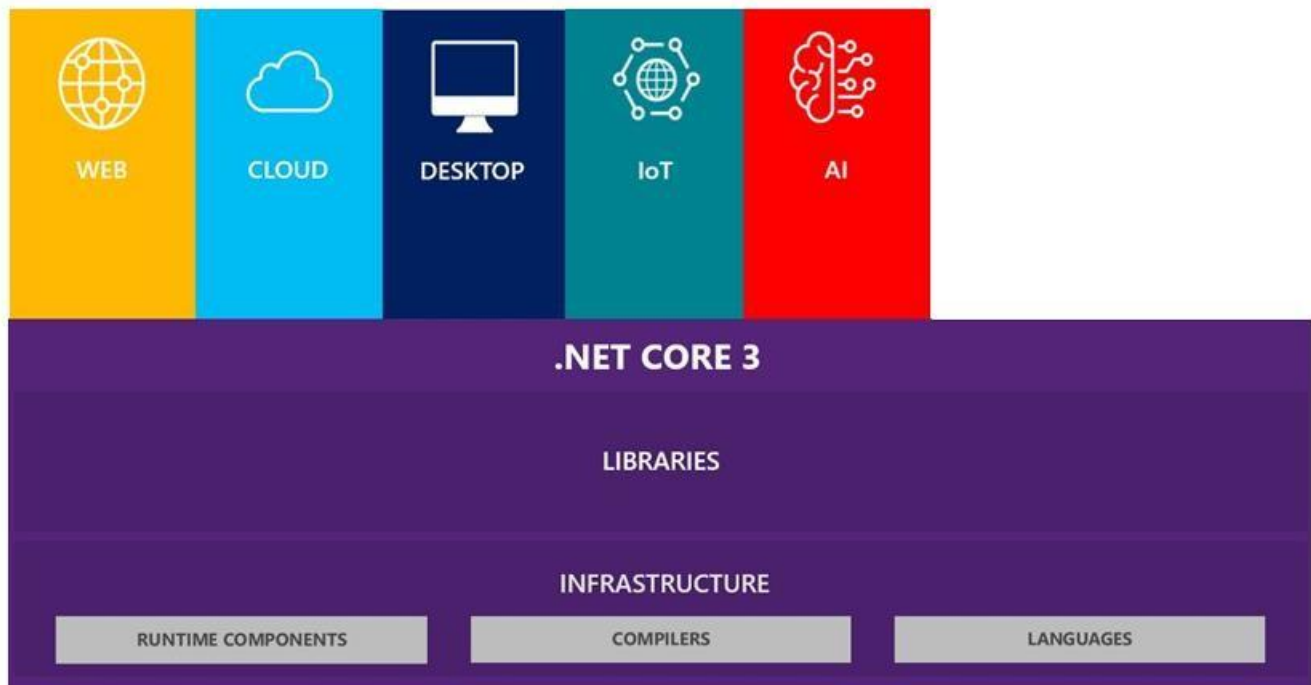
1. It added new changes for high-resolution displays, performance updates.
2. It has many improvements to security enhancements.

2

.NET Core/.NET Core 3.x

Q1. What is .NET Core?

Ans. It is a cross-platform framework that can run on Windows, macOS, and Linux. It is used to build Web, APIs, Cloud, Desktop, ML/AI and IoT based applications.



Q2. Does RHEL support .NET Core?

Ans. RHEL stands for Red Hat Enterprise Linux and has support for .NET Core. Hence, .NET Core is available for the RedHat Software Collections.

Q3. What are the features of the .NET Core?

Ans. The features are mentioned below:

1. It is compatible with .NET Framework, Xamarin, and Mono

2. It can be used with Docker containers
3. It can run on Windows, macOS, and various Linux distributions like Ubuntu, RHEL etc.
4. It is open-source and supports MIT and Apache licenses.

Q4. Which languages are used to create libraries for .NET Core?

Ans. You can create libraries for .NET Core using C#, Visual Basic, and F# languages.

Q5. What IDE can be used for .NET Core development?

Ans. The .NET Core development can be done using .NET CLI and one of the following IDE:

1. Visual Studio for Windows
2. Visual Studio for Mac
3. Visual Studio Code (For Linux, Windows and Mac)
4. Sublime Text
5. Vim

Q6. What is about .NET Core compatibility with Mono APIs?

Ans. .NET Core provides compatibility with Mono APIs by implementing the .NET Standard specification.

Q7. Explain about .Net core components?

Ans. .NET Core components are given below:

1. **.NET Core Runtime:** It contains the .NET Core runtime and framework libraries.
2. **ASP.NET Core Runtime:** It contains ASP.NET Core and .NET Core runtime and framework libraries.
3. **.NET Core SDK:** It contains the .NET CLI Tools, ASP.NET Core runtime, and .NET Core runtime and framework.

Q8. What is the difference between .NET Core runtime and .NET Core SDK?

Ans. The runtime plays a role to run apps and the SDK plays a role to create apps. The SDK contains the runtime. The .NET core runtime is required for a user to run apps and the .NET core SDK is required for a developer to create apps.

Q9. What are the versions of .NET Core/.NET?

Ans. There are following versions of .NET core till 2021:

1. .NET Core 1.0
2. .NET Core 1.1
3. .NET Core 2.0
4. .NET Core 2.1
5. .NET Core 2.2
6. .NET Core 3.0
7. .NET Core 3.1

8. .NET 5

9. .NET6

*Note: starting from .NET 5 **core** word is not included in the official release name.*

Q10. How .NET Core does work with MVC and Web APIs?

Ans. .NET Core provides a unified programming model for MVC and Web API. MVC and Web API have been merged with ASP.NET Core to build UIs and APIs.

Q11. How does Dependency Injection work in .NET Core?

Ans. The Dependency Injection is a Built-in feature of .Net Core. It is the chosen way for logging contexts, database contexts and any other dependencies into your ASP.NET core controllers.

Q12. What does .NET Core provide for Angular and React developers?

Ans. The features for Angular and React developers are mentioned below:

1. It supports Angular CLI and React CLI based SPA templates.
2. The packages required for Angular & React will be installed automatically using NPM while building the ASP.NET Core project.

Q13. What does .NET Core provide to store static files?

Ans. .Net Core provides a default folder named *wwwroot* for storing static files like CSS, Images, Scripts and any others.

Q14. What does .NET Core provide for deployment?

Ans. .NET Core provides features of Cloud Ready Configuration to publish the applications directly to the cloud. It supports environment-specific configuration files.

Q15. What does .NET Core provide for Mobile App Development?

Ans. .NET Core is compatible with Xamarin by using the .NET Standard Library. The mobile app developers can build cross-platform mobile apps in C# with a shared code base and the same set of APIs used with .NET Core.

Q16. What is new in .NET Core 3.0 version?

Ans. The new features in .NET Core 3.0 are mentioned below:

1. Support for C# 8.0.
2. Supports for Windows Forms and WPF.
3. Supports for F# 4.7.
4. High-performance JSON APIs support.
5. Raspberry Pi and ARM chips are now supported to enable IoT development
6. Azure App Service deployment of .NET Core 3.0 is currently available.

Q17. How .NET Core 3.0 app is now executable by default?

Ans. Based on the operating system, .NET Core 3.0 apps are launched with an app-specific executable like `myapp` or `./myapp`. But in the previous releases, .NET Core apps launched via the `dotnet` command like `dotnet myapp.dll`.

Q18. How .NET Core 3.0 will help you in IoT development?

Ans. Starting from .NET Core 3.0, IoT development has been simplified.

1. ASP.NET is used to send data as an API or as a site that enables configuring an IoT device.
2. Using GPIO APIs, we can deploy apps that work with sensors on IoT development.
3. Using remote Visual Studio debugger, Raspberry Pi and ARM chips are now supported in .NET Core for IoT development.

Q19. Which operating systems are supported by .NET Core 3.0?

Ans. .NET Core 3.0 is supported on the following operating systems:

- SLES: 12+
- macOS: 10.13+
- Windows Client: 7, 8.1, 10 (1607+)
- Windows Server: 2012 R2 SP1+
- Fedora: 26+
- Ubuntu: 16.04+
- RHEL: 6+
- Alpine: 3.9+
- Debian: 9+
- OpenSUSE: 42.3+

Q20. Which chips are supported by .NET Core 3.0?

Ans. .NET Core 3.0 supports the following chips:

- ARM32 on Windows and Linux
- ARM64 on Linux (kernel 4.14+)
- x64 on Windows, macOS, and Linux
- x86 on Windows

Q21. What is the role of Nullable reference types of C# 8.0 in .NET Core 3.0?

Ans. The nullable reference types provide significant benefits over the handling of reference variables and the compiler helps to do that correctly and discover bugs in the code.

Q22. What is HTTP/2 Support in .NET Core 3.0?

Ans. .NET Core 3.0 having supports for HTTP/2 in HttpClient. It is useful for APIs like gRPC and Apple push notification Service.

Q23. What are the benefits of using Docker in .NET Core 3.0?

Ans. With .NET Core developing container-based applications are easy and have the following benefits.

- Garbage Collector (GC) heap size is 75% of the memory limit on the container.
- The minimum reserved segment size reduces the number of heaps that are created on machines.
- .NET Core 3.0 on Linux in addition to Docker performs better with cgroup memory limits.

Q24. What technologies .NET Core does not support?

Ans. .NET Core does not support the following technologies:

- WebForms
- WCF
- LINQ to SQL
- WWF
- AppDomains
- Remoting
- Code Access Security (CAS)
- Security Transparency
- System.EnterpriseServices

Q25. Why .NET Remoting technology isn't available with .NET Core?

Ans. There are the following reasons why .NET Remoting isn't supported on .NET Core:

1. It's used for cross-AppDomain communication, which can no longer be supported.
2. Remoting requires runtime support, which is expensive to maintain.

Q26. Why Code Access Security isn't available with .NET Core?

Ans. The CAS is used to make the implementation more complicated and often has correctness performance implications for the applications that don't mean to use it. The use of security boundaries provided by the operating system is mentioned below:

1. Virtualization
2. Containers
3. User accounts for running the processes with the minimum set of privileges

3

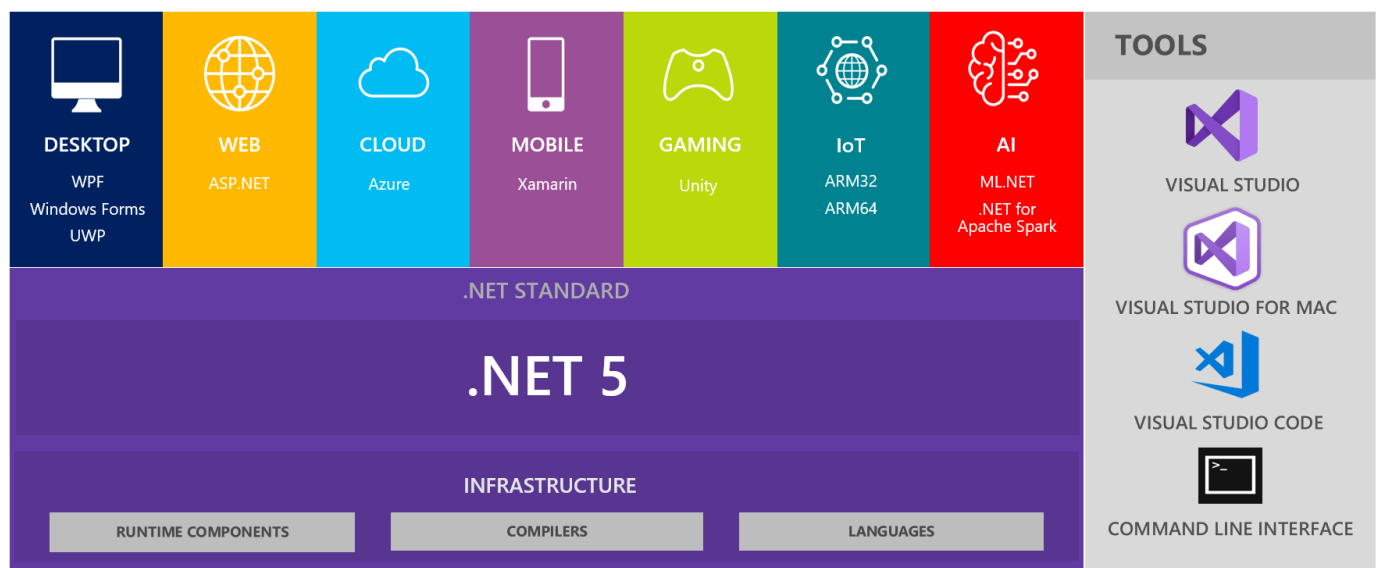
.NET 5

Q1. What is .NET 5?

Ans. .NET 5 is the new release name for the next .NET Core version. There is no .NET Core 4.0 version and Microsoft will not continue with Core word in the upcoming .NET Core release. So, you can think, .NET 5 is the release name for .NET Core 4.0. .NET5 was released in November 2020.

Using .NET 5, you can target the following multiple platforms:

1. Windows
2. Linux
3. macOS
4. iOS
5. Android
6. tvOS
7. watchOS
8. WebAssembly



Q2. What new features are available in .NET 5?

Ans. The following main features will be available in .NET 5:

1. There are much more choices on the runtime experiences.
2. Java interoperability feature will be seen on many platforms.
3. Objective C will be available on many platforms.
4. Swift interoperability will be available on many platforms.

Q3. What is the Mono AOT compiler in .NET 5?

Ans. The following main features are provided by the Mono AOT compiler:

1. This compiler converts .NET code to be constructed into a single native code executable.
2. It can run on a machine, like C++ code.
3. The AOT compiled apps can run efficiently.
4. The main aim of using Mono AOT in .NET 5 is to lower memory usage and faster start-up.

Q4. How many types of AOT solutions for Mono AOT?

Ans. There are 2 types of AOT solutions:

1. One requires 100% AOT compilation.
2. Another one is JIT or interpreter which is used for code patterns like generics.

Q5. What exceptions are found in Mono AOT?

Ans. There are two exceptions - iOS and client-side Blazor i.e. web assembly and some game consoles. These need ahead-of-time or AOT native compilation.

Q6. What is CoreFX framework in .NET 5?

Ans. The .NET 5 applications use the CoreFX framework. The CoreFX works well in the case of Xamarin and client-side Blazor workloads.

Q7. How does .NET 5 work with .NET CLI?

Ans. The .NET 5 applications will be integrated with the .NET CLI. That will be common across all projects.

Q8. What is the objective of .NET 5?

Ans. .NET 5 is introduced with the following objectives:

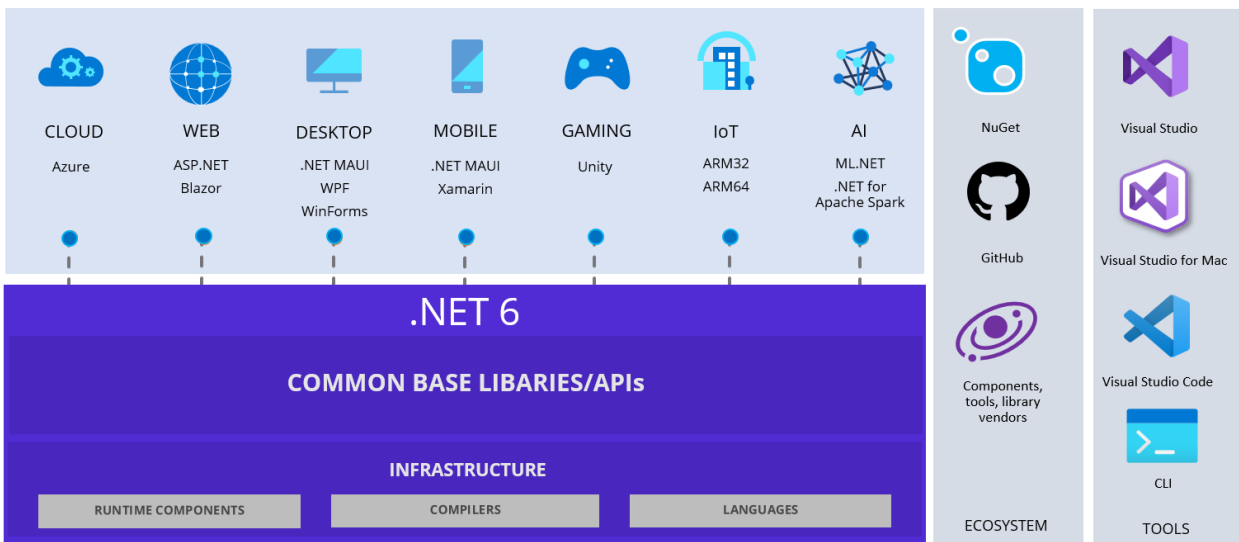
1. Optimize for every workload
2. Diagnostics need to be equal across .NET 5, for functional as well as performance diagnostics
3. It requires optimizations, accurate where many workloads have overlapping needs
4. Developers develop .NET 5 apps will be available to the new C# version

4

.NET 6

Q1. What is .NET 6?

Ans. .NET 6 is the latest release of .NET which is the fastest full-stack web framework and lowers the compute costs if you're running in the application workload in the cloud. .NET 6 supports the latest version of C# and F# i.e., C# 10 and F# 6 which deliver language improvements that make your code simpler and better.



Q2. What does SDK workload support in .NET 6?

Ans. SDK workload is introduced as a new .NET SDK feature by Microsoft. SDK workload supports Web Assembly and mobile application without affecting the size of the SDK.

Q3. Which features are updated in SDK workload to add list and update verbs?

Ans. There are 2 features as mentioned below:

- .NET workload list: It decides which workloads have been installed and informs application developers.
- .NET workload update: It adds/updates every installed workload to the new available version.

Q4. What are the uses of the reflection mechanism and its drawbacks?

Ans. It is used for the object serialization process by .NET serializers. It doesn't work for high-performance cloud-native applications. It is an impediment for startup, memory usage, and assembly trimming.

Q5. What is Compile-Time Source Generator?

Ans. It is a new source generator at compile time as part of System.Text.Json. It will create C# source files that can be compiled as part of the library or software build. It replaces the reflection mechanism because of the following reasons:

- Eliminate runtime use of System.Reflection and System.Reflection.Emit
- Decrease in start-up time
- Better serialization throughput
- Lesser use of private memory

Q6. What is OpenTelemetry Metrics and its benefits?

Ans. Microsoft provides support for OpenTelemetry to the .NET 6 in order to aim at observability. In .NET 6, Microsoft added support for the OpenTelemetry Metrics API. With the help of OpenTelemetry Metrics, applications can easily interoperate with other OpenTelemetry systems.

Q7. What is OpenSSL in .NET 6?

Ans. It is an open-source cryptography toolkit for TLS and SSL protocols.

Q8. What are Roslyn Analyzers in .NET 6?

Ans. Roslyn analyzers help in inspecting the code of a given application development project to find issues related to style, quality and maintainability, and design. There are 250 more analyzers as part of the .NET 6 release.

Q9. What is WebSocket Compression and its advantages?

Ans. Data compression helps in reducing storage space, transmission time, and communication bandwidth. In .NET 6, WebSocket compression is enabled for saving huge operational costs.

Q10. Which framework version is not supported in .NET 6?

Ans. In .NET 6, Microsoft will remove the support for any framework that is older than

- .NET Framework 4.6.1
- .NET Core 3.1
- .NET Standard 2.0

Q11. What is NuGet package validation tool in .NET 6?

Ans. In .NET 6, a Package Validation tool is provided to NuGet library developers to check that their packages are consistent. This tool is used to validate the following things:

- Find out any target-framework- or runtime- applicability gaps

- Validate whether the package has the same group of public APIs for all runtime-specific implementations
- Validate whether there are any breaking changes or not across versions

Q12. What is the role of .NET Hot for ASP.NET Core & Blazor Projects?

Ans. It allows developers to change the source code of an application in the running state, without pausing manually or using a breakpoint. In .NET 6, the .NET Hot Reload is available with the help of dotnet watch.

Q13. What are the benefits of using Interface Casting in .NET 6?

Ans. The performance of Interface casting is scaled up by 16% to 38%. So. It is helpful for C# pattern matching to and between interfaces.

Q14. What new collection is added in .NET 6?

Ans. A new collection `PriorityQueue<TElement, TPriority>` (System.Collections.Generic) is introduced in .NET 6. It enables adding new items with a value and a priority. On dequeue, this new collection returns an item with the least priority value.

Q15. How does JsonSerializer work in .NET 6?

Ans. In .NET 6, JsonSerializer (available in System.Text.Json namespace) supports the ability to avoid cycles when carrying out a serialization process on an object graph.

Q16. What are two ignore options available in System.Text.Json in .NET 6?

Ans. The `ReferenceHandler.IgnoreCycles` and `Newtonsoft.Json ReferenceLoopHandling.Ignore` options are available.

Q17. What are the Improvements of the client app development in .NET 6?

Ans. In .NET 6, iOS, Android, and MacOS app development will be integrated into the .NET SDK and leverage .NET libraries. It will offer a completely unified mobile product for .NET.

Before .NET 6, Mobile App Development is provided as a separate Xamarin product. So, Microsoft has decided to make Xamarin much like to mainline .NET.

Q18. How to check the available versions of .NET SDK and Runtime in .NET6?

Ans. A new command **dotnet sdk check**, is added to .NET 6. This new command will tell you about the latest available version of the .NET SDK and .NET Runtime.

Q19. How to enable compression for Single-file bundles in .NET 6 and why it is needed?

Ans. In .NET 6, Single-file bundles support compression. This can be enabled by changing the property `EnableCompressionInSingleFile` to true. It is required because compression can save large space. During runtime, files are in decompress mode.

Q20. What is .Net MAUI and how does it associate with .NET 6?

Ans. .NET MAUI is a cross-platform UI toolkit, offering a single stack that supports all platforms- Android, iOS, macOS, and Windows. The Xamarin.iOS and Xamarin.Android will be part of .NET 6.

Q21. What is Blazor desktop apps in .NET 6?

Ans. Earlier, Blazor was supported on the server and then on a browser with WebAssembly. Blazor desktop allows developers to build hybrid client apps, which blend together web and native UI in a native client application.

Q22. What is the new project template in .NET 6?

Ans. .NET 6 comes with C# 10. The templates are based on the new changes that C# 10 introduces. For .NET 6, The console application for the Hello world application is shown below:

```
Console.WriteLine("Hello, World!");
```

In .NET 6, We can set the Hello World application with one effective line of code.

Q23. What is the keyword called global in C# 10?

Ans. Using global keyword, we can define global **usings** for the whole project in a separate file which will contain these imports (e.g. usings.cs).

Q24. What is implicit using the directives feature in .NET 6?

Ans. It forces the compiler to automatically import a set of usings based on the project type. for each project type, an implicit set of global using is defined, so it doesn't need to be explicitly stated in each file as mentioned below:

```
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Net.Http;  
using System.Net.Http.Json;
```

Q25. What are the commands required for SDK workload in .NET 6?

Ans. The below commands are mentioned below:

- **dotnet workload install name_of_the_workload** : This command is to install any SDK workload.
- **dotnet workload list**: list all installed workloads.
- **dotnet workload update**: updating all installed SDK workloads to latest version.
- **dotnet workload search**: list workloads available to install.
- **dotnet workload uninstall**: uninstall workload.
- **dotnet workload repair**: something went wrong (like your internet connection broke) during workload installation, you can repair it with.
- **dotnet sdk check**: check whether new versions of the SDK and Runtimes are available.

.NET Core SDK and Compilation

Q1. What is .NET Core SDK?

Ans. The .NET Core SDK contains a set of libraries and tools. Using these tools, developers can build .NET Core applications and libraries.

Q2. What components does .NET Core SDK contain?

Ans. The following list of components are required to build and run applications:

1. .NET Core libraries and runtime
2. The .NET Core CLI tools
3. The dotnet driver

Q3. What are the different ways to install the SDK On Windows?

Ans. There are the following ways to install SDK:

1. Use the native installers
2. Install with Visual Studio
3. Install alongside with Visual Studio Code
4. Install with PowerShell automation

Q4. How to erase the .NET Core Runtime and SDK?

Ans. By using the following methods, you can remove .NET Core runtime and SDK:

1. Using the Windows Add/Remove Programs dialog to remove versions of the .NET Core runtime and SDK
2. Using the .NET Core Uninstall Tool
3. Removing the NuGet fallback folder

Q5. What is the NuGet fallback folder?

Ans. The .NET Core SDK installers used the NuGetFallbackFolder to store a cache of NuGet packages.

Q6. What is the path of the NuGet fallback folder on Windows and macOS?

Ans. The NuGetFallbackFolder is located on Windows at C:\Program Files\dotnet\sdk and on macOS at /usr/local/share/dotnet/sdk.

Q7. In which case the NuGet fallback folder can be removed?

Ans. If a developer is developing using .NET Core 3.0 SDK or later versions, then this folder can be removed. To remove the NuGet fallback folder, admin privileges are required.

Q8. How does Docker work with .NET Core SDK?

Ans. .NET Core based applications can run as a Docker container. .NET Core Docker images can be published to the Microsoft Container Registry (MCR) or Docker Hub repository. These published images can be pulled where you want to run your packaged .NET Core application.

Q9. What are the different ways to install the SDK On Linux?

Ans. There are the following ways to install SDK:

1. Install with a package manager
2. Install alongside with Visual Studio Code
3. Install with bash automation

Q10. What are the different ways to install the SDK on macOS?

Ans. There are the following ways to install SDK:

1. Install with an installer
2. Install with Visual Studio for Mac
3. Install alongside with Visual Studio Code
4. Install with bash automation

Q11. What compilers are used in the .NET Core execution process?

Ans. The compilers are used for .NET Core are mentioned below:

1. For C# and VB: Roslyn Compiler
2. For F#: new F# 4.1 compiler

Q12. What is the reimplement of the class libraries for .NET Core?

Ans. CoreFx is the reimplement of the class libraries for .NET Core. This one is the replacement of framework class libraries (FCL) for a different set of libraries that are used.

Q13. What is the status of tiered compilation in .NET Core 2.x and 3.x?

Ans. The status is mentioned below:

1. For .NET Core 3.0 and later >> tiered compilation is enabled by default.

2. For .NET Core 2.1 and 2.2 >> tiered compilation is disabled by default.

Q14. What are the setting names of Tiered compilation in .NET Core?

Ans. There are 2 setting names are mentioned below:

1. System.Runtime.TieredCompilation
2. COMPlus_TieredCompilation

Q15. What is Roslyn and its primary features?

Ans. Roslyn is a group of open-source compilers. Its features are mentioned below:

1. Compilers for the C# and Visual Basic .NET act as services through APIs
2. The APIs are used for code analysis.
3. The APIs are used for refactoring

Q16. What are the different types of APIs for Roslyn?

Ans. There are three types of APIs as mentioned below:

1. Feature APIs
2. Work-space APIs
3. Compiler APIs

Q17. What is the Feature APIs of Roslyn?

Ans. The Feature APIs allows source code tool developers to do below mentioned things:

1. Code refactoring
2. Code fixes

Q18. What is the Work-space APIs of Roslyn?

Ans. The Work-space APIs allows plugin developers to take actions specifically needed in integrated development environments (IDEs) like Visual Studio is mentioned below:

1. Finding references for a variable
2. Code formatting

Q19. What are the Compiler APIs of Roslyn?

Ans. The Compiler APIs approve the analysis of source code for performing syntax tree and the binding flow analysis.

6

Core CLR and RyuJIT

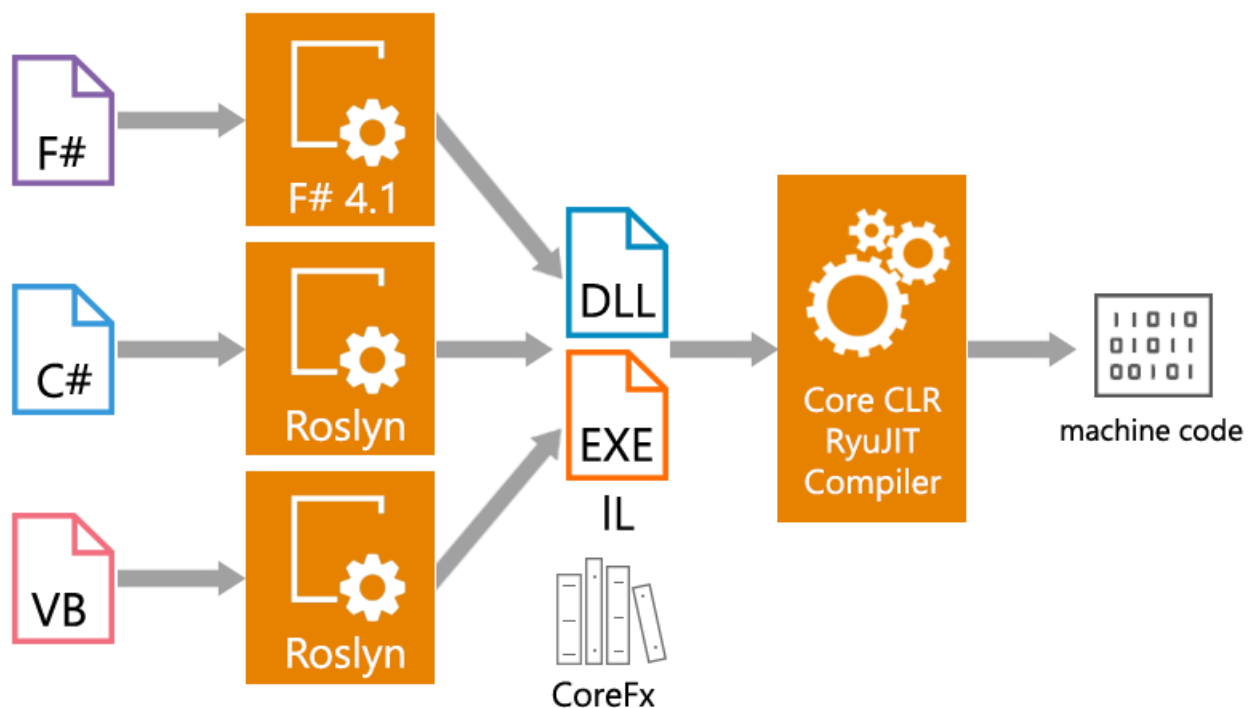
Q1. What is CoreCLR?

Ans. CoreCLR is the runtime for .NET Core. It includes:

1. Garbage collector
2. JIT compiler
3. Primitive data types
4. Low-level classes

Q2. How does CoreCLR work with Console App?

Ans. The console app type is the best way to kick the CoreCLR tiers. You can build your own custom CoreCLR and run console apps on top of it.



Q3. What is the difference between CoreCLR and Corefx?

Ans. corefx includes C# and coreclr includes large collections of both C# and C++ code.

Q4. What are the lines of code in CoreCLR?

Ans. If we look from a size perspective, the coreclr repo has 2.6M lines of code. Within that count, the JIT is about 320K lines and the GC about 55k.

Q5. How to find CoreCLR in the cross-platform system?

Ans. The .NET Core runtime APIs are in:

- coreclr.dll on Windows
- libcoreclr.so on Linux
- libcoreclr.dylib on macOS

Q6. What are the types of CoreClrHost methods used for hosting .NET Core?

Ans. CoreClrHost has several important methods useful for hosting .NET Core are mentioned below:

1. coreclr_initialize
2. coreclr_execute_assembly
3. coreclr_create_delegate
4. coreclr_shutdown
5. coreclr_shutdown_2

Q7. What is RyuJIT?

Ans. It is one of the foundational components of the .NET runtime. The RyuJIT compiler compiles IL byte code to machine code for multiple processors.

Q8. What is the difference between RyuJIT and Roslyn compiler?

Ans. The Roslyn C# compiler compiles C# code to IL byte code. The RyuJIT compiler compiles IL byte code to machine code for multiple processors.

Q9. What features are provided by RyuJIT?

Ans. It is focused on evolving the code base for enabling better support as mentioned below:

1. Multiple code generation targets
2. Improved optimizations
3. Better and more flexible code generation
4. Open, flexible, and robust design and implementation
5. Fast Span<T> support
6. Inlining improvements

7. Devirtualization
8. Significantly better floating-point performance
9. Better performance with value types (structs)

Q10. What is SIMD does support via RyuJIT?

Ans. Microsoft introduced the SIMD that is Single Instruction, Multiple Data support via RyuJIT. The usage goes from physics engines in games and scientific research, to financial applications by performing complex calculations of stocks and bonds.

Q11. How does RyuJIT define in case of performance case?

Ans. RyuJIT generally takes longer to compile than JIT32, but its optimizer can generate faster code.

Q12. What is the difference between RyuJIT and JIT32?

Ans. RyuJIT is open source. JIT32 is not open source. The RyuJIT JIT compiler will be used for all platform combinations for .NET Core 2.0 and beyond. JIT32 will be removed from the product. For .NET Core 1.x, RyuJIT is used for x64 and JIT32 is used for x86 chip support.

Garbage Collector

Q1. What is garbage collection in .NET Core?

Ans. The garbage collector manages the allocation and release of memory. The garbage collector serves as an automatic memory manager.

Q2. What are the advantages of garbage collection in .NET Core?

Ans. The list of advantages is given below:

1. The objects which are no longer used then it will take action on those objects by managing their memory, and making the memory available for the next allocations.
2. It also allocates objects on the managed heap efficiently.

Q3. What are the conditions for garbage collection in .NET Core?

Ans. Due to the following condition, the garbage collection will occur:

1. When the system has low physical memory
2. The GC.Collect method is primarily used for unique situations and testing. you do not have to call this method, because the garbage collector runs continuously.

Q4. What are the flavors of garbage collection in .NET Core?

Ans. There are four flavors of garbage collection as mentioned below:

1. Workstation GC
2. Server GC
3. Background
4. Non-concurrent

Q5. What are the settings to select flavors of GC in .NET Core?

Ans. There are 2 settings to select workstation garbage collection and server garbage collection flavors of garbage collection as mentioned below:

1. System.GC.Server
2. COMPlus_gcServer

Also, there are 2 settings to select the background and non-concurrent flavors of garbage collection as mentioned below:

1. `System.GC.Concurrent`
2. `COMPlus_gcConcurrent`

Q6. What are the settings to manage the garbage collector's memory and processor usage?

Ans. The settings are mentioned below:

1. `System.GC.HeapCount`
2. `COMPlus_GCHeapCout`

Q7. What are the values of `System.GC.HeapCount` and `COMPlus_GCHeapCount` settings?

Ans. To minimize the number of heaps to 16, the values would be 16 for the JSON file and 10 for the environment variable.

1. The value of `System.GC.HeapCount` is a decimal value.
2. The value of `COMPlus_GCHeapCount` is hexadecimal.

Q8. What is `Thread_UseAllCpuGroups` element option?

Ans. To set up the common language runtime for distributing threads from the thread pool across all the CPU groups, it enables the `Thread_UseAllCpuGroups` element option.

Q9. How to enable `Thread_UseAllCpuGroups` element option for .NET Core apps?

Ans. You can enable this option by setting the value of the `COMPlus_Thread_UseAllCpuGroups` environment variable to 1.

Q10. What is Standalone GC in .NET Core?

Ans. It specifies a way to the library containing the garbage collector that the runtime intends to put by implementing `COMPlus_GCName` setting. The value of `COMPlus_GCName` setting is `string_path`.

8

.NET CLI

Q1. What is .NET Core command-line interface (CLI)?

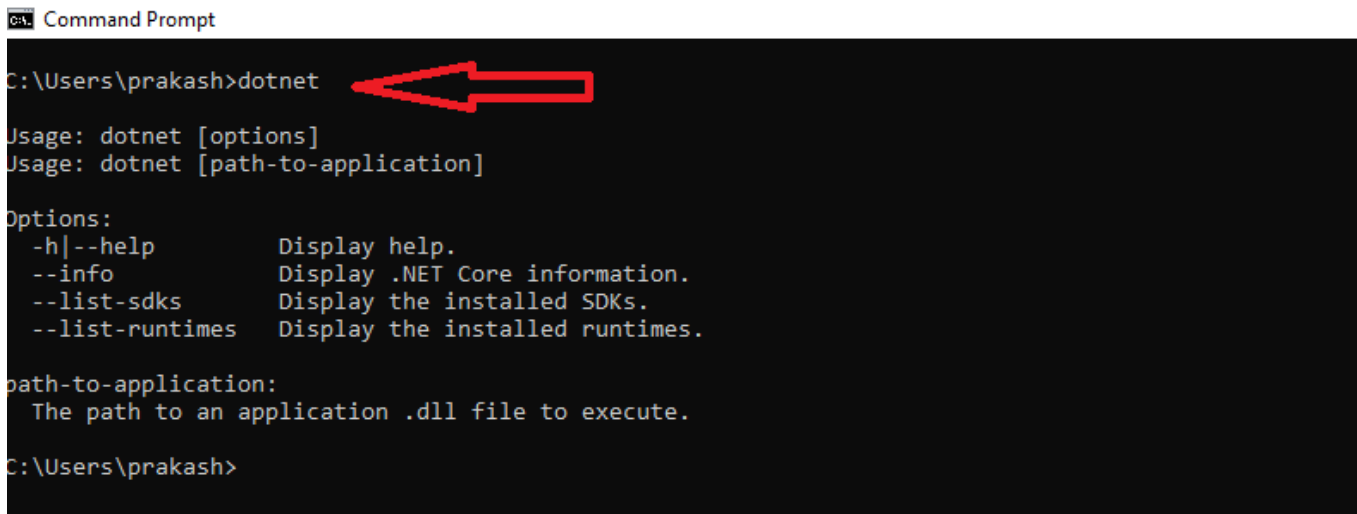
Ans. The .NET Core command-line interface (CLI) is a new cross-platform tool for creating, restoring packages, building, running and publishing .NET applications.

Q2. How to get .NET Core command-line interface (CLI)?

Ans. CLI tools are available in DEB packages for Ubuntu and MSI bundles for Windows. This installer will install the tool and set up the environment required to run the CLI tool.

Q3. How to verify the .NET Core command-line interface (CLI) is installed?

Ans. We verify whether the CLI is installed properly by command prompt in Windows and writing dotnet and pressing Enter. The image as shown below:



```
Command Prompt
C:\Users\prakash>dotnet
Usage: dotnet [options]
Usage: dotnet [path-to-application]

Options:
  -h|--help           Display help.
  --info              Display .NET Core information.
  --list-sdks          Display the installed SDKs.
  --list-runtimes      Display the installed runtimes.

path-to-application:
  The path to an application .dll file to execute.

C:\Users\prakash>
```

Q4. How does .NET Core CLI install by default?

Ans. By default, the CLI installs in a side-by-side manner, so multiple versions of the CLI tools can be available on a single machine.

Q5. What are the benefits of using .NET Core CLI commands?

Ans. CLI tools are really powerful and easy to use. we should know how to use these commands. These commands are very useful if you are looking to create Continuous Integration (CI) and/or Continuous Deployment (CD).

Q6. What is the Command Structure of .NET Core CLI?

Ans. The following is a command structure.

dotnet <command> <argument> <option>

All the commands start with the driver name as *dotnet*. The driver starts the execution of the specified command. After *dotnet*, we can supply a command (also known as a verb) to perform a specific action. Each command can be followed by arguments and options.

- | | |
|--------------------------|-----------------------------------|
| • dotnet new | Scaffolds a minimal app |
| • dotnet restore | Restores packages for the project |
| • dotnet run | Compiles and runs the app |
| • dotnet build | Compiles to IL |
| • dotnet build -native | Compiles to a single executable |
| • dotnet dosomethingcool | Build your own commands |

Q7. What is the CLI command to create the project in the C# language?

Ans. The following is a command structure.

dotnet new console -lang C#

```
C:\Users\prakash>dotnet new console -lang C#

Welcome to .NET Core!
-----
Learn more about .NET Core: https://aka.ms/dotnet-docs
Use 'dotnet --help' to see available commands or visit: https://aka.ms/dotnet-cli-docs
```

Q8. How to get the list of available .NET CLI commands?

Ans. You can write a help command to list the details.

Command Prompt

```
C:\Users\prakash>dotnet help
.NET Command Line Tools (2.1.401)
Usage: dotnet [runtime-options] [path-to-application] [arguments]

Execute a .NET Core application.
```

Q9. What is the basic .NET Core CLI commands?

Ans. The list of basic commands is shown below:

Command	Description
new	Creates a new project, configuration file, or solution based on the specified template.
restore	Restores the dependencies and tools of a project.
Build	Builds a project and all of its dependencies.
Run	Runs source code without any explicit compile or launch commands.
publish	Packs the application and its dependencies into the folder for deployment to a hosting system.
test	Executes unit tests.
vtest	Runs tests from the specified files.
pack	Packs the code into a NuGet package.
clean	Cleans the output of a project.
sln	Modifies a .NET Core solution file.
help	Display the list of available various options and commands

Q10. What is the .NET Core CLI project modification commands?

Ans. The list of command is shown below:

Command	Description
add package	Adds a package reference to a project.
add reference	Adds project-to-project (P2P) references.
remove package	Removes package reference from the project.
remove reference	Removes project reference
list reference	Lists all project-to-project references

Q11. What are the .NET Core CLI project advanced commands?

Ans. The list of command is shown below:

Command	Description
nuget delete	Deletes or unlists a package from the server.
nuget locals	Clears or lists local NuGet resources.
nuget push	Pushes a package to the server and publishes it.
msbuild	Builds a project and all of its dependencies.
dotnet install script	This script is used to install the .NET Core CLI tools and the shared runtime.

Q12. Which command is used to add Newtonsoft.json package to our console project?

Ans. The following command is used to include Newtonsoft.json package in our project. We can verify it by opening .csproj file.

dotnet add package Newtonsoft.json

Q13. How to create new .NET Core project using template name argument?

Ans. For creating a new .NET Core project, we have to use new command followed by template name argument. We can create console, class library, web, mvc, webapi, razor, angular, react etc. projects using CLI.

Q14. How you can create and run a console application using .NET CLI?

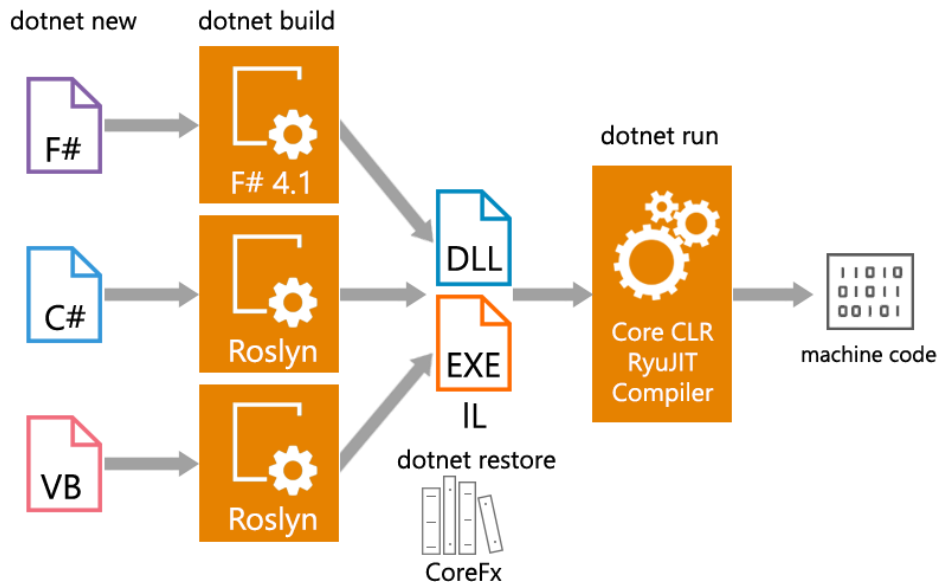
Ans. The following creates new console project in the current directory with the same name as the current directory.

dotnet new console

The following command creates a new console project named myApp. The -n or --name option species the name of a project.

dotnet new console -n myApp

Further, you can run *dotnet restore* and *dotnet run* command to run your application as shown in the fig.



Q15. Which command does create a new console application name to the project directory?

Ans. The following command creates a new console application named SatyaConsoleApp to SatyaProjects directory. The -o or --output option is used to specify an output directory where the project should be generated.

dotnet new console -n SatyaConsoleApp -o C:\SatyaProjects

After creating a project, navigate to the project directories in the command prompt to apply project-specific commands as shown: **C:\SatyaConsoleApp**

Q16. How to create a .NET Standard class library project in the specified directory?

Ans. Refer below command:

dotnet new classlib -lang C# -o SatyaLibrary

Q17. How to Create a new ASP.NET Core C# MVC project in the current directory with no authentication?

Ans. Refer below command:

dotnet new mvc -au None

Q18. How to create a new xUnit project?

Ans. Refer below command:

dotnet new xunit

Q19. How to get list all templates available for MVC?

Ans. Refer below command:

dotnet new mvc -l

Q20. How to get List all templates matching the mentioned("se") substring?

Ans. Refer below command:

dotnet new se -l

If there is no match is found, substring matching runs against both the short name and name columns.

Q21. How to get List all templates matching the mentioned("ng") template?

Ans. Refer below command to attempt to invoke the template matching ng:

dotnet new ng

If there is no match is found, list the templates that are partial matches.

Q22. Which CLI command is used to install version 2.0 of the SPA templates for ASP.NET Core?

Ans. Refer below command:

dotnet new -i Microsoft.DotNet.Web.Spa.ProjectTemplates::2.0.0

This command option is available for .NET Core SDK 1.1 and later versions only.

Q23. Which CLI command is used to create a global.json in the current directory?

Ans. Refer below command:

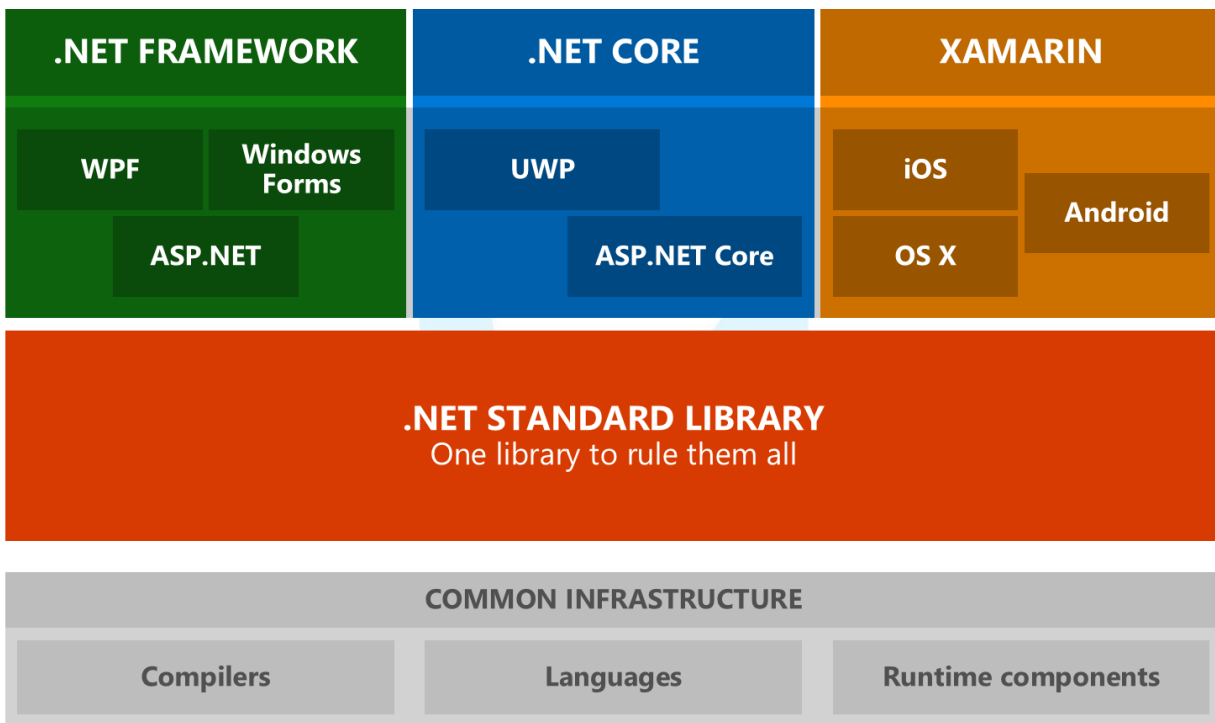
dotnet new globaljson

This command option is available only with .NET Core SDK 2.0 or later versions.

.NET Standard

Q1. What is .NET Standard?

Ans. The .NET Standard is a specification of .NET APIs that are intended to be available on all .NET implementations. It enables developers to produce portable libraries that are usable across .NET implementations, using this same set of APIs.



Code Sharing among different .NET Implementation

Q2. What are the differences between .NET Standard and Portable Class Libraries?

Ans. The .NET Standard is the replacement for Portable Class Libraries (PCL). There are the following differences between .NET Standard and PCL:

1. .NET Standard is a curated set of APIs, while PCL profiles are defined by intersections of existing platforms
2. .NET Standard linearly versions, while PCL profiles do not

3. PCL profiles represent Microsoft platforms while the .NET Standard is platform-agnostic

Q3. How to describe .NET Standard compatibility with PCL profiles?

Ans. .NET Standard is compatible with a subset of PCL profiles. Profile-based PCL compatibility is provided by **Microsoft.NETCore.Portable.Compatibility** NuGet package. Profile-based PCLs packaged as NetStandard are easier to consume and it is compatible with existing users.

Q4. What does .NET standard Support by .NET implementations?

Ans. The .NET Standard 2.0 is supported by the following .NET implementations:

1. .NET Core 2.0 or later
2. .NET Framework 4.6.1 or later
3. Mono 5.4 or later
4. Xamarin.iOS 10.14 or later
5. Xamarin.Mac 3.8 or later
6. Xamarin.Android 8.0 or later
7. Universal Windows Platform 10.0.16299 or later

Q5. What is new in the .NET Standard 2.0?

Ans. The .NET Standard 2.0 includes the following new features:

1. The .NET Standard 2.0 added over 20,000 more APIs.
2. you can access .NET Framework libraries from a .NET Standard library by using a compatibility shim.
3. You can now develop .NET Standard libraries in Visual Basic.
4. Visual Studio 2017 and the .NET Core Command Line Interface (CLI) include tooling support for creating .NET Standard libraries.

Q6. What is not included in .Net Standard?

Ans. The Application model API's are commonly called Framework Class Libraries (FCL) like WebForms, Windows Forms, WPF, WCF, etc. and OS-specific API's like Registry, AppDomain, etc. are out of the NetStandard specification.

Q7. What is the Portable Class Library?

Ans. It can be used as a re-usable library project where we mention the .Net Platform to target during the creation of the library. The portable libraries allowed code sharing between multiple platforms.

Q8. What are the disadvantages of Portable Class Library?

Ans. Introducing the latest platform support requires recompiling against the new set of platforms and erasing the API references that are not part of the new intersection of base class libraries. Portable Class Libraries does not have controls on the target platforms.

Q9. How do I identify which .NET Standard version I should target?

Ans. When choosing a .NET Standard version, you should follow the below guidelines:

1. The higher the version, the more APIs are available to you.
2. The lower the version, the more platforms implement it.

Q10. What are the benefits of targeting .NET Standard 2.0?

Ans. The Consumers of the package that is running on .NET Standard 2.0 compatible frameworks need to fetch fewer package dependencies as a result. You can use the below table to understand versions:

.NET Standard	1.0	1.1	1.2	1.3	1.4	1.5	1.6	2.0	2.1
.NET Core	1.0	1.0	1.0	1.0	1.0	1.0	1.0	2.0	3.0
.NET Framework	4.5	4.5	4.5.1	4.6	4.6.1	4.6.1	4.6.1	4.6.1	N/A
Mono	4.6	4.6	4.6	4.6	4.6	4.6	4.6	5.4	6.4
Xamarin.iOS	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.14	12.16
Xamarin.Mac	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.8	5.16
Xamarin.Android	7.0	7.0	7.0	7.0	7.0	7.0	7.0	8.0	10.0
Unity	2018.1	2018.1	2018.1	2018.1	2018.1	2018.1	2018.1	2018.1	TBD
Universal Windows Platform	8.0	8.0	8.1	10.0	10.0	10.0.16299	10.0.16299	10.0.16299	TBD

.NET Standard is that you can target, based on which .NET platforms you intend to run on. For example, if you want to run on .NET Framework 4.5 and .NET Core 1.0. You can target .NET Standard 1.1.

Q11. How do PCL profiles map with the .NET standard?

Ans. .NET Standard is also compatible with Portable Class Libraries (PCLs). The mapping from PCL profiles to .NET Standard versions are listed below:

PCL Profile	.NET Standard	PCL Platforms
7	1.1	.NET Framework 4.5, Windows 8
31	1.0	Windows 8.1, Windows Phone Silverlight 8.1
32	1.2	Windows 8.1, Windows Phone 8.1
44	1.2	.NET Framework 4.5.1, Windows 8.1
49	1.0	.NET Framework 4.5, Windows Phone Silverlight 8

78	1.0	.NET Framework 4.5, Windows 8, Windows Phone Silverlight 8
84	1.0	Windows Phone 8.1, Windows Phone Silverlight 8.1
111	1.1	.NET Framework 4.5, Windows 8, Windows Phone 8.1
151	1.2	.NET Framework 4.5.1, Windows 8.1, Windows Phone 8.1
157	1.0	Windows 8.1, Windows Phone 8.1, Windows Phone Silverlight 8.1
259	1.0	.NET Framework 4.5, Windows 8, Windows Phone 8.1, Windows Phone Silverlight 8

if your PCL is configured to target .NET Framework 4.5.1 and Windows 8.1, it uses profile 44. So, you can convert your PCL to .NET Standard 1.2



VS Code With .NET Core

Q1. What is VS Code?

Ans. Visual Studio Code is a very powerful and easy-to-use code editor. It is the best code editor for creating applications that run on Windows, Linux, and macOS. Use Visual Studio Code with the C# extension to get full support for C# IntelliSense and debugging.

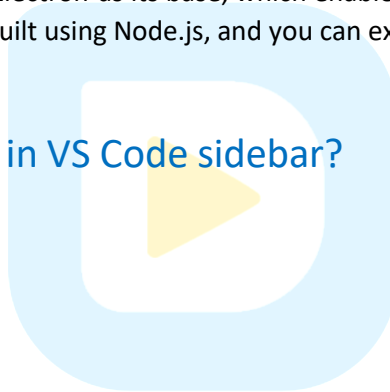
Q2. What does enable VS Code to be cross-platform?

Ans. It is open source. It uses Electron as its base, which enables it to be cross-platform and work on Mac, Windows and Linux. It is built using Node.js, and you can extend it using JavaScript.

Q3. How many icons are in VS Code sidebar?

Ans. There are 5 icons:

1. The File Explorer
2. Search
3. Source Control
4. The Debugger
5. The Extensions



Q4. What are the keyboard shortcuts for quick file picker and hiding the sidebar?

Ans. Using the shortcut CTRL+P will show you a quick file picker. Using the shortcut CTRL+B You can hide the sidebar that hosts the file.

For Keyboard Shortcuts Reference, Visit below URL:

<https://code.visualstudio.com/docs/getstarted/keybindings#keyboard-shortcuts-reference>

Q5. What is the keyboard shortcut difference between mac, Windows and Linux?

Ans. For Mac, the keyboard shortcut will start with CMD. Most of the time, on Windows and Linux you just change CMD to CTRL and it works.

Q6. How does Git work in VS code?

Ans. VS Code comes with Git support out of the box. In this case, the folder we opened does not have source control initialized. Clicking the icon with the Git logo, allows us to initialize the Git repository.

Q7. What is the Activity Bar in VS code?

Ans. The Activity Bar allows you to access the features of VS Code as mentioned below:

1. The Explorer contains your project data
2. Search allows you to replace specific code in your project files
3. Source Control Management provides support for integrated source control tools like Git
4. VS Code also comes with integrated Debugging tools
5. Extensions allow you to add additional features to VS Code

Q8. What is the Integrated Terminal in VS Code?

Ans. The terminal automatically navigates to your current project folder. You can also run multiple terminals simultaneously.

Q9. What is the Command Palette in VS code?

Ans. The Command Palette lets you search for any kind of information and navigate to any menu in VS Code. This includes navigating to the settings menu.

Q10. What common operations are performed using the Command Palette?

Ans. Common operations are performed as shown below:

1. Extensions: We can Install Extensions
2. Preferences: Color Theme to change the colour theme
3. Format Document, which formats code automatically
4. Run Code, which is supplied by the Code Runner and it executes the highlighted lines of JavaScript

Q11. How to open a new window in VS Code with the content of the current folder?

Ans. When you install VS Code, the code command is available in your command line globally. open a new window with the content of the current folder, with code.

code -n will create a new window.

Q12. How to show the difference between two files using VS Code?

Ans. VS Code can show easily the difference between two files, with **code --diff file1.js file2.js**.

Q13. What are .NET CLI options in VS Code?

Ans. You can open files, install extensions, change the display language, and the output diagnostics through command-line options (switches).

Q14. How to get a list of command-line options in VS Code?

Ans. Open a terminal or command prompt and type **code --help**.

Q15. How to configure for launching VS Code from the command line in different platforms?

Ans. For macOS: Users must first run a command (Shell Command: Install 'code' command in the PATH) to add VS Code executable to the PATH environment variable.

For Windows and Linux: The Installations should add the VS Code binaries location to your system path or you can manually add the location to the Path environment variable (\$PATH on Linux). For example, on Windows, VS Code is installed under AppData\Local\Programs\Microsoft VS Code\bin.

Q16. How to launch VS Code Insiders build?

Ans. If someone is using the VS Code Insiders preview, you launch your Insiders build with **code-insiders**.

Q17. Do share important options used with VS Code?

Ans. You can use the following options with VS Code:

➤ code <option>

Option	Description
-h or --help	Display all the commands/options to be used with VS Code
-v or --version	Print VS Code version.
-n or --new-window	Opens a new session of VS Code.
-d or --diff	Open a file difference editor. Requires two file paths as arguments.
--locale <locale>	Set the display language (locale) for the VS Code session. (for eg, en-US or en-GB)

Q18. How many Core CLI options can use for Opening VS Code with URLs?

Ans. You can open projects and files using the platform's URL handling mechanism. You can use the URL in applications such as browsers or file explorers that can parse and redirect the URL. For example, on Windows, you can pass a **vscode://** URL directly to the Windows Explorer or the command line as start **vscode://{full path to file}**.

The Core CLI options are mentioned below:

Open a project

vscode://file/{full path to project}/

E.g. > vscode://file/c:/myProject/

Open a file

vscode://file/{full path to file}

E.g. > `vscode://file/c:/myProject/package.json`

Open a file to line and column

`vscode://file/{full path to file}:line:column`

E.g. > `vscode://file/c:/myProject/package.json:5:10`

Q19. How to solve 'code' is not recognized as an internal or external command?

Ans. For Windows, Linux and macOS, this issue can be resolved using the below steps:

For Windows and Linux: The VS Code Windows and Linux installations should have installed VS Code on your path. Try uninstalling and reinstalling VS Code.

For macOS: On macOS, you need to manually run the Shell Command: Install 'code' command in PATH command available through the Command Palette *Ctrl+Shift+P*.

Note: If code is still not found, consult the platform-specific setup topics for Windows, Linux and macOS.

Q20. How to open a command line (terminal) from within VS Code?

Ans. VS Code contains a Terminal option in the menu. From there you can open the terminal and can run the command line tools within the VS Code.

Q21. How to install and manage VS Code extensions?

Ans. You can install and manage VS Code extensions from the command line or VS Code extensions option.

Q22. What various options are used for installing and managing VS Code extensions?

Ans. You can use the following options with VS Code:

➤ `code <option>`

Options	Description
<code>--install-extension <ext></code>	Install an extension by passing the extension name as an argument. Use <code>--force</code> argument to avoid prompts.
<code>--uninstall-extension <ext></code>	Uninstall an extension by passing the extension name as an argument.
<code>--disable-extensions</code>	Disable all installed extensions. Extensions will still be visible in the disabled section of the Extensions view but they are not activated.
<code>--list-extensions</code>	List all the installed extensions.
<code>--show-versions</code>	Show version of installed extensions, when using with <code>--list-extensions</code>

Q23. How to find the path of the extensions is installed?

Ans. Extensions are installed in a per-user extensions folder. Depending on your platform, the location is in the following folder:

1. **Windows:** %USERPROFILE%\vscode\extensions
2. **macOS:** ~/.vscode/extensions
3. **Linux:** ~/.vscode/extensions

Q24. How to change the path of the extensions by launching VS Code?

Ans. You can change the location by launching VS Code with the **--extensions-dir <dir>** command-line option.

Q25. How to control proxy settings using the command line?

Ans. If your machine is going through a proxy server to access the Internet. The following command-line arguments to control your proxy settings:

Disable proxy

--no-proxy-server

Manual proxy address

--proxy-server=<scheme>=<uri>[:<port>][;...] | <uri>[:<port>] | "direct://"

Manual PAC address

--proxy-pac-url=<pac-file-url>

Disable proxy per host

--proxy-bypass-list=(<trailing_domain>|<ip-address>)[:<port>][;...]

Q26. How to manage VS Code by providing extension recommendations?

Ans. You can modify the following settings for making VS Code display extension recommendations in the Extensions view or you can do it through notifications:

- extensions.showRecommendationsOnlyOnDemand - Set to true to remove the RECOMMENDED section.
- extensions.ignoreRecommendations - Set to true to silence extension recommendation notifications.
- The Show Recommended Extensions command is always available for checking recommendations.

Porting .NET to .NET Core

Q1. How to port your code to .NET Core or .NET Standard?

Ans. We need to understand the dependencies. The External dependencies are the NuGet packages for reference in the project. For more details refer here: <https://docs.microsoft.com/en-us/dotnet/core/porting/>

Q2. How to Migrate the NuGet packages to PackageReference?

Ans. .NET Core refers PackageReference to specify package dependencies. If you're using the packages.config to specify packages in the project, convert it to the PackageReference format since packages.config file isn't supported in .NET Core.

Q3. How to Upgrade the NuGet packages?

Ans. Initially, upgrade your packages to the latest version. Using NuGet Package Manager UI in Visual Studio, it can be done. It is compatible with .NET Core.

Q4. What to do when NuGet package dependency failed to run on .NET Core?

Ans. The simple steps are mentioned below:

1. Search for another package that runs on the .NET Core that accomplishes a similar task as the package you used.
2. Attempt to write the code the package was doing yourself.
3. Discard the dependency on the package by changing the functionality of the app until a compatible edition of the package is available.

Q5. What is the .NET Portability Analyzer tool?

Ans. The tool is used to analyze the assemblies that aim the .NET Framework and identifies APIs that aren't portable to different .NET platforms such as .NET Core. One can run the tool as a console application and Visual Studio extension.

Q6. How to analyze dependencies if they aren't NuGet packages?

Ans. The only way to determine the portability of that dependency is to run the .NET Portability Analyzer tool.

Q7. What is the Target Framework Monikers (TFMs)?

Ans. The Target Framework Monikers that maps to versions of the .NET Standard, .NET Core, and traditional Portable Class Library profiles that are compatible with .NET Core.

Q8. How to analyze the converted and upgraded package dependencies work on .NET Core?

Ans. There are a few ways that you can achieve that:

1. Analyze NuGet packages using nuget.org
2. Analyze NuGet packages using NuGet Package Explorer

Q9. What are Windows Compatibility Pack and its package contents?

Ans. It provides technologies, so it's easier to build .NET Core applications and .NET Standard libraries. This package is a logical extension of .NET Standard 2.0. The Windows Compatibility Pack is provided via the Microsoft.Windows.Compatibility NuGet package and can be referenced from projects that target .NET Core or

.NET Standard.

Q10. What is the .NET Framework compatibility mode?

Ans. This compatibility mode is used to introduce the .NET Standard and .NET Core projects to reference .NET Framework libraries. The .NET Framework compatibility mode was coming with .NET Standard 2.0.

Q11. Why do you retarget to .NET Framework 4.7.2?

Ans. If the code doesn't aim for the .NET Framework 4.7.2, You must retarget to .NET Framework 4.7.2. It is the availability of the latest API alternatives for cases where the .NET Standard doesn't support existing APIs.

Q12. What are the steps to retarget to .NET Framework 4.7.2?

Ans. For every project, you must port. You need to follow the steps mentioned below:

1. Right-click on the project and select Properties.
2. In the Target Framework dropdown, select .NET Framework 4.7.2.
3. Recompile the project.

Q13. What is the testing framework that builds and runs tests for .NET Core.?

Ans. You need to use a testing framework that builds and runs tests for .NET Core. These are mentioned below:

1. xUnit
2. NUnit
3. MSTest

Q14. How to port your code with the base of your library?

Ans. This is the foundational components of your code. This might be data models or some other foundational classes and methods. Few steps are mentioned below:

1. Port the test project that tests the layer of a library that you're currently porting.
2. Copy over the base of your library into a new .NET Core project and select the version of the .NET Standard you wish to support.
3. Make any changes needed to get the code to compile by adding NuGet package dependencies to your .csproj file.
4. Run the tests and make any needed adjustments.
5. Catch the next layer of code to port over and repeat the prior steps.

Q15. What are those tools to help with porting to .NET Core?

Ans. The list of tools is mentioned below:

1. The .NET Portability Analyzer - It generates a report of how portable your code is between .NET Framework and .NET Core as a command-line tool Or as a Visual Studio extension.
2. The .NET API Analyser (Roslyn Analyzer) - It discovers potential compatibility risks for C# APIs on many platforms.

Q16. How to generate a portability report for analysis?

Ans. Run the API Portability Analyzer (ApiPort) is used to produce the portability report for the analysis.

Q17. What is the Target Framework Monikers (TFMs)?

Ans. The Target Framework Monikers that maps to versions of the .NET Standard, .NET Core, and traditional Portable Class Library profiles that are compatible with .NET Core.

Q18. How to analyze the converted and upgraded package dependencies work on .NET Core?

Ans. There are a few ways that you can achieve that:

1. Analyze NuGet packages using nuget.org
2. Analyze NuGet packages using NuGet Package Explorer

Q19. What are Windows Compatibility Pack and its package contents?

Ans. It provides technologies, so it's easier to build .NET Core applications and .NET Standard libraries. This package is a logical extension of .NET Standard 2.0. The Windows Compatibility Pack is provided via the Microsoft.Windows.Compatibility NuGet package and can be referenced from projects that target .NET Core or .NET Standard.

Q20. What is .NET Framework compatibility mode?

Ans. This compatibility mode is used to introduce the .NET Standard and .NET Core projects to reference .NET Framework libraries. The .NET Framework compatibility mode was coming with .NET Standard 2.0.

Q21. Why do you retarget to .NET Framework 4.7.2?

Ans. If the code doesn't aim the .NET Framework 4.7.2, You must retarget to .NET Framework 4.7.2. It is the availability of the latest API alternatives for cases where the .NET Standard doesn't support existing APIs.

Q22. What are the steps to retarget to .NET Framework 4.7.2?

Ans. For every project, you must port. You need to follow the steps mentioned below:

1. Right-click on the project and select Properties.
2. In the Target Framework dropdown, select .NET Framework 4.7.2.
3. Recompile the project.

Q23. What is the testing framework that builds and runs tests for .NET Core.?

Ans. You need to use a testing framework that builds and runs tests for .NET Core. These are mentioned below:

1. xUnit
2. NUnit
3. MSTest

Q24. How to port your code with the base of your library?

Ans. This is the foundational components of your code. This might be data models or some other foundational classes and methods. Few steps are mentioned below:

1. Port the test project that tests the layer of a library that you're currently porting.
2. Copy over the base of your library into a new .NET Core project and select the version of the .NET Standard you wish to support.
3. Make any changes needed to get the code to compile by adding NuGet package dependencies to your .csproj file.
4. Run the tests and make any needed adjustments.
5. Catch the next layer of code to port over and repeat the prior steps.

Q25. What are those tools to help with porting to .NET Core?

Ans. The list of tools is mentioned below:

1. The .NET Portability Analyzer - It generates a report of how portable your code is between .NET Framework and .NET Core as a command-line tool Or as a Visual Studio extension.
2. The .NET API Analyser (Roslyn Analyzer) - It discovers potential compatibility risks for C# APIs on many platforms.

Q26. How to generate a portability report for analysis?

Ans. Run the API Portability Analyzer (ApiPort) is used to produce the portability report for the analysis.

Q27. What is the Target Framework Monikers (TFMs)?

Ans. The Target Framework Monikers that maps to versions of the .NET Standard, .NET Core, and traditional Portable Class Library profiles that are compatible with .NET Core.

Q28. How to analyze the converted and upgraded package dependencies work on .NET Core?

Ans. There are a few ways that you can achieve that:

1. Analyze NuGet packages using nuget.org
2. Analyze NuGet packages using NuGet Package Explorer

Q29. What are Windows Compatibility Pack and its package contents?

Ans. It provides technologies, so it's easier to build .NET Core applications and .NET Standard libraries. This package is a logical extension of .NET Standard 2.0. The Windows Compatibility Pack is provided via the Microsoft.Windows.Compatibility NuGet package and can be referenced from projects that target .NET Core or .NET Standard.

Q30. What is the .NET Framework compatibility mode?

Ans. This compatibility mode is used to introduce the .NET Standard and .NET Core projects to reference .NET Framework libraries. The .NET Framework compatibility mode was coming with .NET Standard 2.0.

Q31. Why do you retarget to .NET Framework 4.7.2?

Ans. If the code doesn't aim for the .NET Framework 4.7.2, You must retarget to .NET Framework 4.7.2. It is the availability of the latest API alternatives for cases where the .NET Standard doesn't support existing APIs.

Q32. What are the steps to retarget to .NET Framework 4.7.2?

Ans. For every project, you must port. You need to follow the steps mentioned below:

1. Right-click on the project and select Properties.
2. In the Target Framework dropdown, select .NET Framework 4.7.2.
3. Recompile the project.

Q33. What is the testing framework that builds and runs tests for .NET Core.?

Ans. You need to use a testing framework that builds and runs tests for .NET Core. These are mentioned below:

1. xUnit
2. NUnit
3. MSTest

Q34. How to port your code with the base of your library?

Ans. This is the foundational components of your code. This might be data models or some other foundational classes and methods. A few steps are mentioned below:

1. Port the test project that tests the layer of a library that you're currently porting.
2. Copy over the base of your library into a new .NET Core project and select the version of the .NET Standard you wish to support.
3. Make any changes needed to get the code to compile by adding NuGet package dependencies to your .csproj file.
4. Run the tests and make any needed adjustments.
5. Catch the next layer of code to port over and repeat the prior steps.

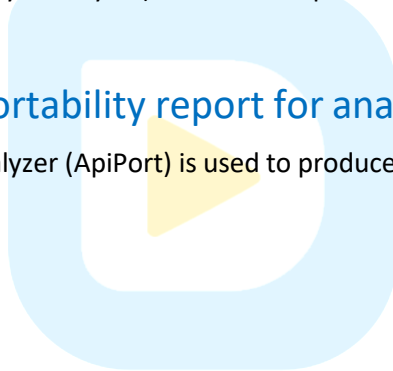
Q35. What are those tools to help with porting to .NET Core?

Ans. The list of tools is mentioned below:

1. The .NET Portability Analyzer - It generates a report of how portable your code is between .NET Framework and .NET Core as a command-line tool Or as a Visual Studio extension.
2. The .NET API Analyser (Roslyn Analyzer) - It discovers potential compatibility risks for C# APIs on many platforms.

Q36. How to generate a portability report for analysis?

Ans. Run the API Portability Analyzer (ApiPort) is used to produce the portability report for the analysis.



References

This book has been written by referring to the following sites:

1. <https://docs.microsoft.com/en-us/dotnet/core> - Microsoft Docs - .NET Core
2. <https://www.dotnettricks.com/learn> - Dot Net Tricks
3. <https://stackoverflow.com/questions/tagged/.net-core> - Stack Overflow - .Net

