Hands-on Azure Repos

Understanding Centralized and Distributed Version Control in Azure DevOps Services

Chaminda Chandrasekara
Pushpa Herath

Apress®

Hands-on Azure Repos

Understanding Centralized and Distributed Version Control in Azure DevOps Services

Chaminda Chandrasekara
Pushpa Herath

Apress®

Hands-on Azure Repos: Understanding Centralized and Distributed Version Control in Azure DevOps Services

Chaminda Chandrasekara
Colombo, Sri Lanka

Pushpa Herath
Hanguranketha, Sri Lanka

ISBN-13 (pbk): 978-1-4842-5424-0
ISBN-13 (electronic): 978-1-4842-5425-7
https://doi.org/10.1007/978-1-4842-5425-7

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Smriti Srivastava
Development Editor: Siddhi Chavan
Coordinating Editor: Shrikant Vishwakarma

Cover designed by eStudioCalamar

Cover image designed by Freepik (www.freepik.com)

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail rights@apress.com, or visit www.apress.com/rights-permissions.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/978-1-4842-5424-0. For more detailed information, please visit www.apress.com/source-code.

Printed on acid-free paper

Let this book be a daily reference guide for all the developers who use Azure Repos.

Table of Contents

About the Authors

Chaminda Chandrasekara is a Microsoft Most Valuable Professional (MVP) for Visual Studio ALM and a Scrum Alliance Certified ScrumMaster, who focuses on continuous improvement of the software development lifecycle. He works as a lead engineer in DevOps at Xameriners (Pvt) Ltd, Singapore. Chaminda is an active Microsoft Community Contributor (MCC) who is well recognized for his contributions in Microsoft forums, TechNet galleries, wikis, and Stack Overflow, and he contributes extensions to Azure DevOps Server and Services (formerly VSTS/TFS) in the Microsoft Visual Studio Marketplace. He also contributes to other open source projects in GitHub. Chaminda has published four books with Apress, and he blogs at https://chamindac.blogspot.com/.

Pushpa Herath is an author, blogger, and speaker at technical community events and works as a DevOps engineer at Xamariners (Pvt) Ltd.

She has years of experience in DevOps with Azure DevOps, Octopus, JIRA, and many other DevOps tools. She is an expert on functional test automation using Selenium and BDD. Pushpa blogs about technology at https://devopsadventure.blogspot.com/. She has published two books with Apress.

About the Technical Reviewer

Mittal Mehta has 15 years of IT experience and currently is working as a configuration manager. He also has eight years of experience working in TFS, C#, Navision, build-release, Azure DevOps, automation, and configuration in Microsoft technologies.

Acknowledgments

We are thankful for all the mentors who have encouraged and helped us during our careers and who have provided us with so many opportunities to gain the maturity and the courage we needed to write this book.

We would also like to thank our friends and colleagues who have helped and encouraged us in so many ways.

Last, but in no way least, we owe a huge debt to our families, not only because they have put up with late-night typing, research, and our permanent air of distraction, but also because they have had the grace to read what we have written. Our heartfelt gratitude is offered to them for helping us make this dream come true.

Introduction

Collaboration among developers is a vital aspect in software development. Sharing code while working in teams to achieve software delivery goals increases end-user satisfaction. Hence, source code control tools are essential for software development teams.

Azure Repos offers you both a centralized version control system and a distributed version control system.

- Team Foundation Version Control (TFVC) is the centralized version control system that comes with Azure Repos.
- Azure Git repos provide you with distributed version control and support all the popular Git repo concepts.

Hands-on Azure Repos gives you step-by-step guidance on working with TFVC and Git, while exploring best practices in each step. You will discover branching and merging techniques to resolve conflicts while sharing code with teams as well as how to track the changes you make to the code using repos. You will explore the essential command-line options, REST API usage, and security options with hands-on lessons to give you the ability to manage TFVC and Git effectively to support your teams. Additionally, code review procedures for repos and integration of a repo with other Azure DevOps features such as boards, pipelines, etc., are discussed in detail.

The hands-on steps in the book will provide you with a comprehensive understanding, from the basics to advanced topics, as you go through each chapter. Lessons comprise secrets to getting started quickly with Azure Repos in the right way and integrating it with popular development tools such as Visual Studio, VS Code, etc. The tips and tricks in the book will make you a productive developer and prevent you from taking the wrong steps while using Azure Repos.

We hope *Hands-on Azure Repos* will be your go-to resource for delivering value to your end users with software, using any platform and any language you prefer to use.

CHAPTER 1

Getting Started with Azure Team Foundation Version Control

Team Foundation Version Control (TFVC) is a centralized version control system for your source code management. Generally, a team member will have one version of each source code file on their machine while using TFVC. Branches of source code are based on paths and get created on the server. A history of version control is maintained on the source control server, not on the local developer machine. In TFVC you can apply permissions at a granular level, and restrictions can be applied at the file level, which we will discuss in Chapter 6.

In this chapter, we will explore the steps required to get started with TFVC in Azure DevOps using Visual Studio. You will be able to understand how to set up a team project to use TFVC or add a TFVC repo to an existing team project that is currently using Git repos. The steps required to set up your machine to develop with TFVC using Visual Studio will be described, and you will learn how to add new or existing solutions to TFVC in Visual Studio. Further, this chapter will discuss the different workspaces available in TFVC and the options to define and use code check-in (commit) policies. In addition, using TFVC with other developer tools such as VS Code and Eclipse will be described for you to get started even faster.

C. Chandrasekara and P. Herath, *Hands-on Azure Repos*, https://doi.org/10.1007/978-1-4842-5425-7_1

Lesson 1-1: Creating a Team Project with TFVC and Adding a TFVC Repo to the Existing Team Project

We discussed how to create a new team project in Chapter 1 of the first book, *Hands-on Azure Boards*, of this book series. However, let's take a quick look at creating a team project with TFVC in this lesson to keep your experience seamless.

Prerequisites: You have an Azure DevOps organization created, and you have some experience working with Azure DevOps to create team projects, or you have followed the lessons in Chapter 1 of the book *Hands-on Azure Boards*.

Navigate to your Azure DevOps organization's home page and click the "Create project" button in the top-right corner. See Figure 1-1.

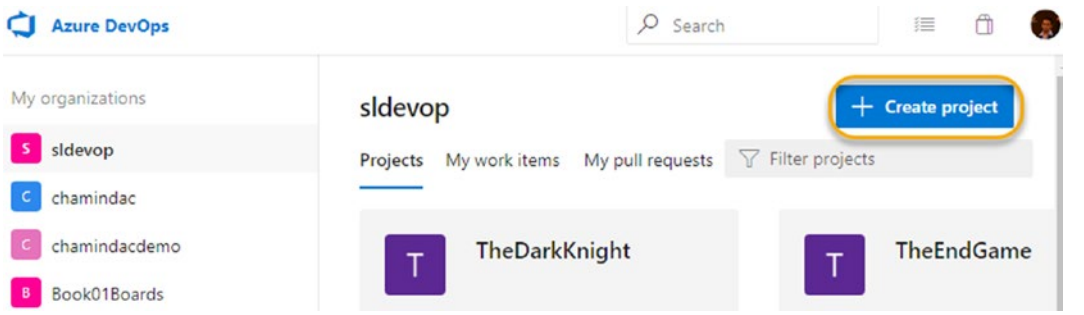

Figure 1-1. *Creating project*

In the team project creation pane, provide a name and select TFVC as the version control system. You can set "Work item process" to whatever you'd like, as discussed in detail in the *Hands-On Azure Boards* book. See Figure 1-2.

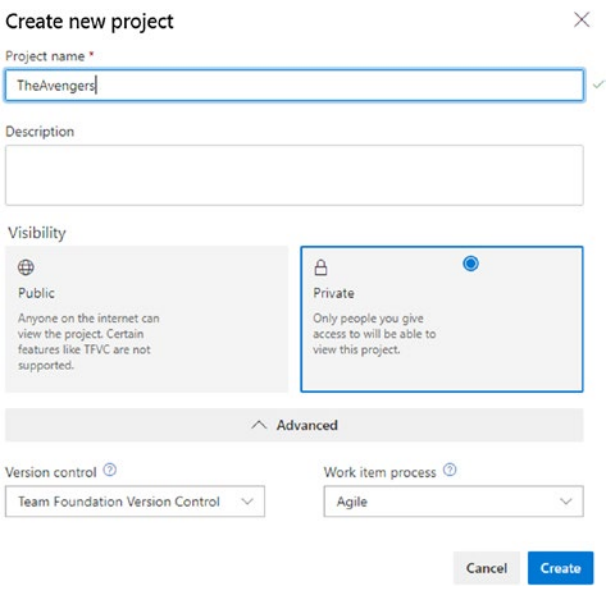

Figure 1-2. *Creating a team project with TFVC*

The created team project has TFVC set as the repo by default, and you can view it by clicking the Repos menu option in the left menu. See Figure 1-3.

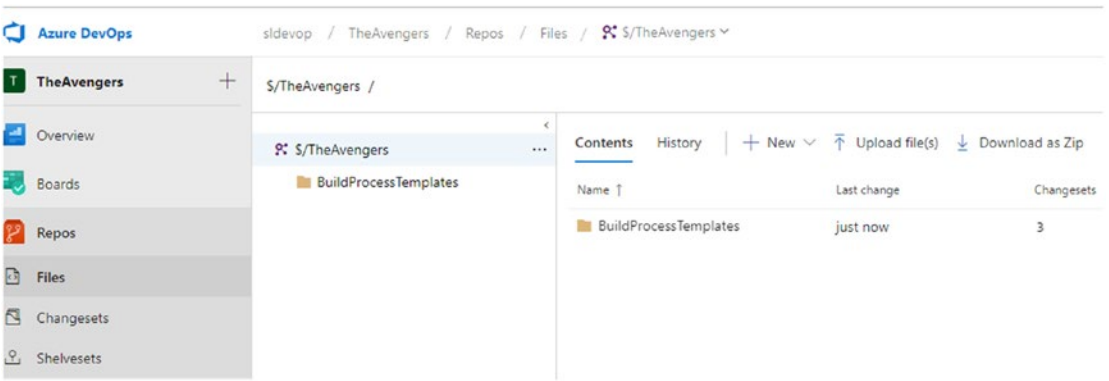

Figure 1-3. *TFVC repo*

Now, let's look at how we can add a TFVC repo to an existing team project. Unlike Git where multiple Git repos can be added to a single team project, you can have only one TFVC repo per team project in Azure DevOps. Create a new team project with Git as the version control system. Then navigate to Repos in the left menu. On the Repos tab, click the drop-down next to the Git repo name and click "New repository." See Figure 1-4.

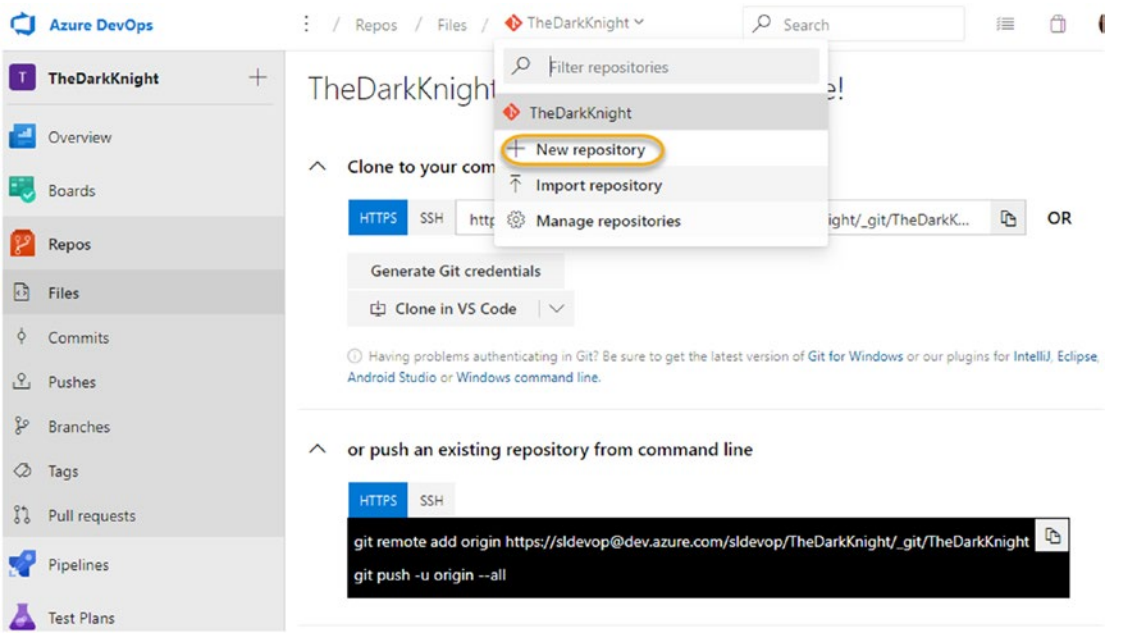

Figure 1-4. *Creating a new repo in an existing project*

In the dialog that appears, select TFVC as the repo type and click Create to create a new TFVC repo in a team project that already has Git repos. See Figure 1-5.

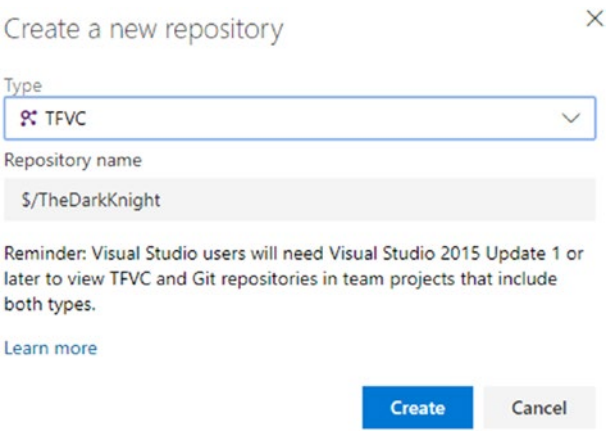

Figure 1-5. *Creating a TFVC repo*

In this lesson, we explored how to create a TFVC repo in an existing team project or create a new team project with TFVC as the source control system.

Lesson 1-2: Using Visual Studio Team Explorer

We created a new team project in the previous lesson with TFVC as the source control system. As a next step, we need to connect it to Visual Studio to get started with the source code development. In this lesson, let's discuss the steps required to get Visual Studio connected to your newly created team project in the TFVC repository, using the Team Explorer window in Visual Studio.

Prerequisites: You have Visual Studio 2019 installed on your machine and are familiar with working with Visual Studio. You have followed the steps in Lesson 1-1 of this chapter and have a team project created with TFVC as the source control system.

In Visual Studio, to open Team Explorer, you can click View ➤ Team Explorer in the menu or press Ctrl+\ and then Ctrl+M. The Team Explorer window lets you connect to Azure DevOps. You can click the Manage Connection toolbar icon to go to the Manage Connections page of the Team Explorer window. See Figure 1-6.

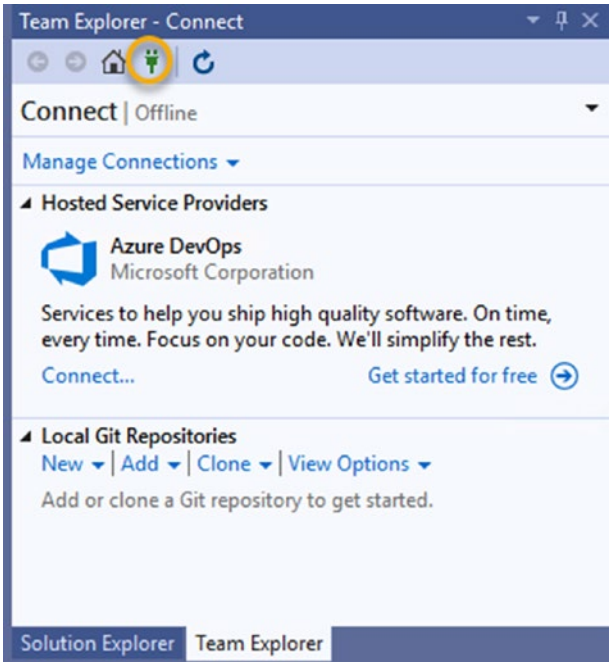

Figure 1-6. *Managing connections in Team Explorer*

Click the Connect link or click the drop-down next to the Manage Connections link (see Figure 1-6) and then click "Connect to a project." In Team Explorer, the Manage Connection page will open as a pop-up window. In the Connect to a Project dialog, you can see the Azure DevOps organizations you have access to if you have already logged in to Visual Studio with a Microsoft account or your organization's account. You can click "Add an account" or select a different account if you have connected more than one account. See Figure 1-7. If you click "Add an account," you will be prompted for your credentials, and you can provide them to connect your Microsoft account or your organization's account to Visual Studio.

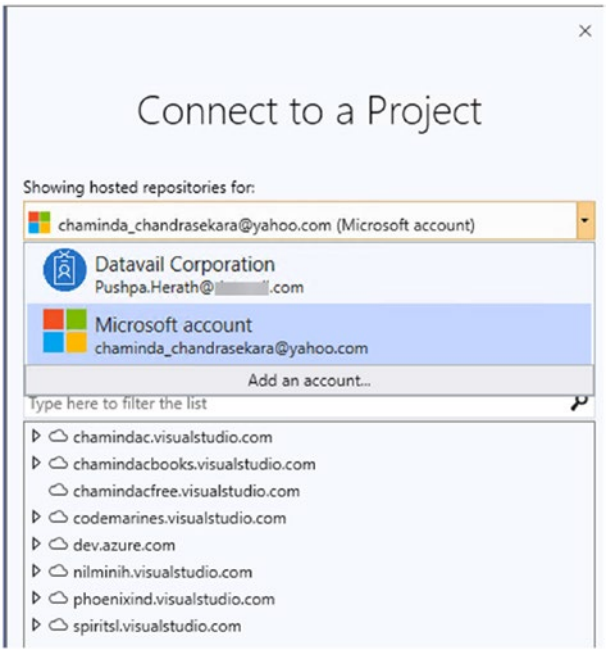

Figure 1-7. *Adding or selecting an account*

Log on from the account you used to create a team project in the previous lesson and expand the Azure DevOps services organization to view your team projects and repositories. If you are using an on-premises Azure DevOps server, you have the option to provide an Azure DevOps server URL and connect it. In the expanded view, select the TFVC repo you want to connect and click the Connect button. See Figure 1-8.

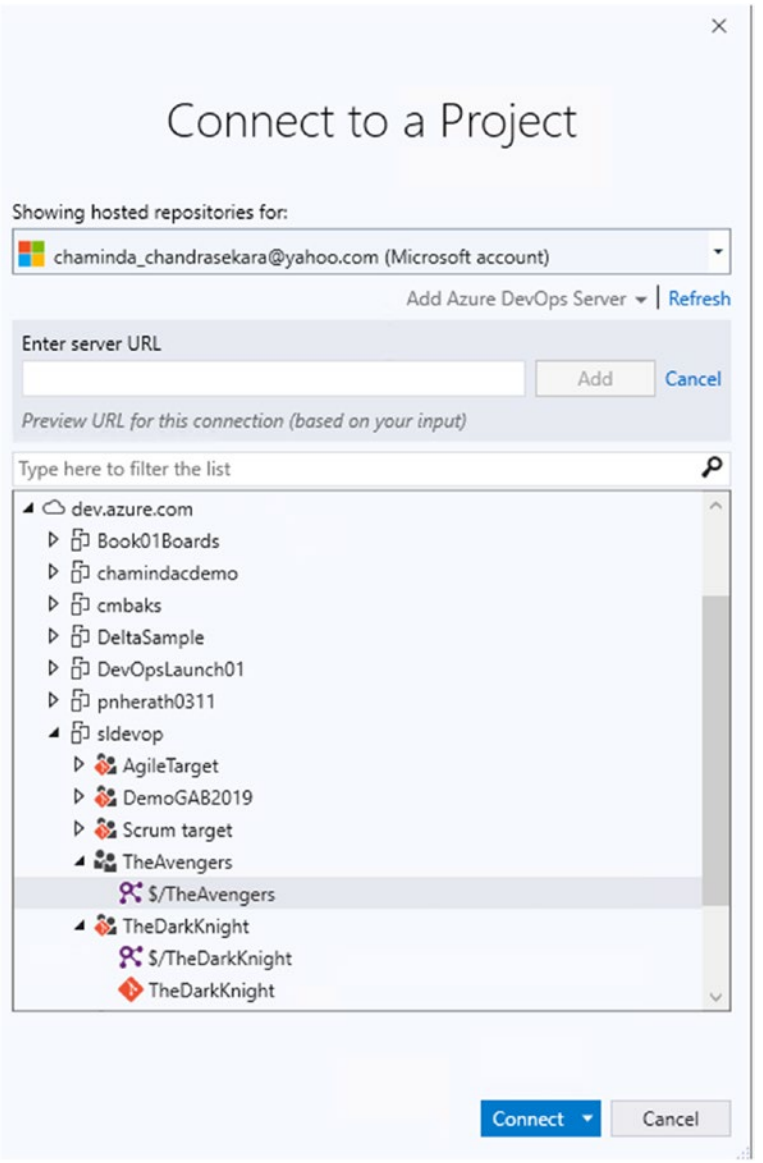

Figure 1-8. *Connecting a TFVC repo*

Once the team project TFVC repo is connected, Team Explorer will allow you to map the TFVC path of the project to a local folder and create a workspace. There are two types of workspaces, and we will discuss them in Lesson 1-4. Provide a desired local path and click the Map & Get button. See Figure 1-9.

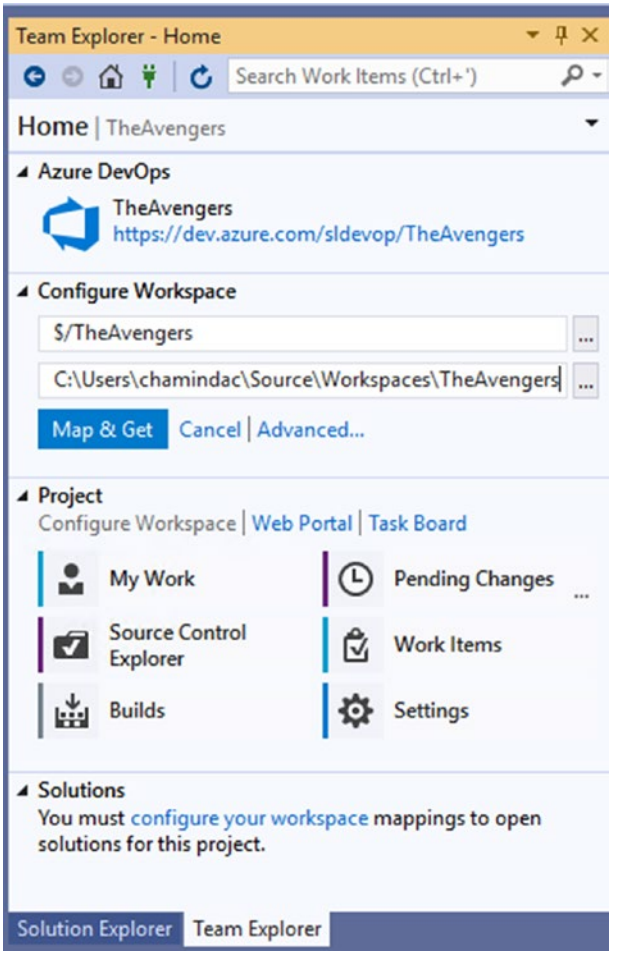

Figure 1-9. *Map & Get button*

In this lesson, we discussed the steps required to connect and map a local path for a TFVC repository using Visual Studio.

Lesson 1-3: Adding New/Existing Solutions to TFVC

Once we map the TFVC repository in Visual Studio, we are allowed to add new solutions to version control using the Solution Explorer window of Visual Studio. Let's look at the steps to add a new solution to TFVC and how you can add an existing solution to TFVC using the Solution Explorer window of Visual Studio.

Prerequisites: You performed the steps described in the first two lessons in this chapter.

In Team Explorer, you will see the New link, which allows you to create a new solution. Click it. See Figure 1-10.

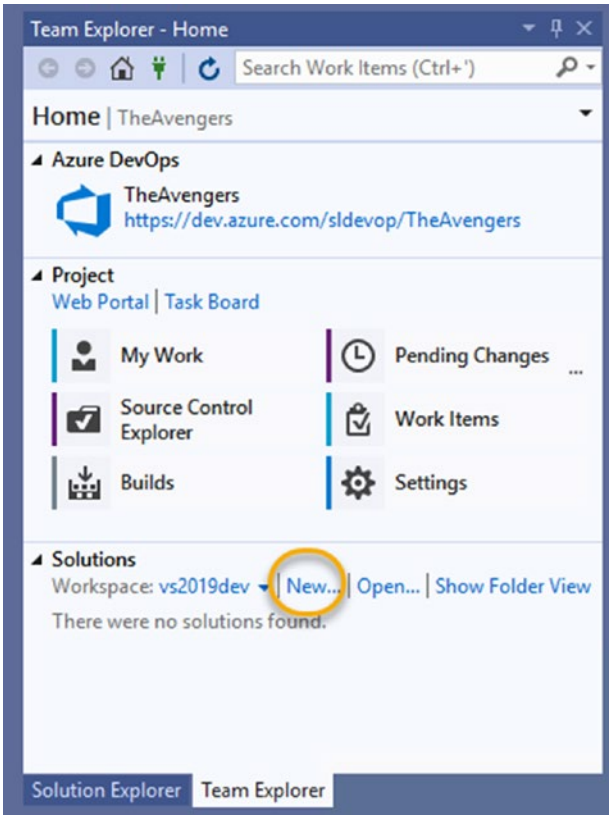

Figure 1-10. *Creating a new solution*

Then in the "Create a new project" dialog, search for *console application*, select the .NET Framework console application, and click Next. See Figure 1-11.

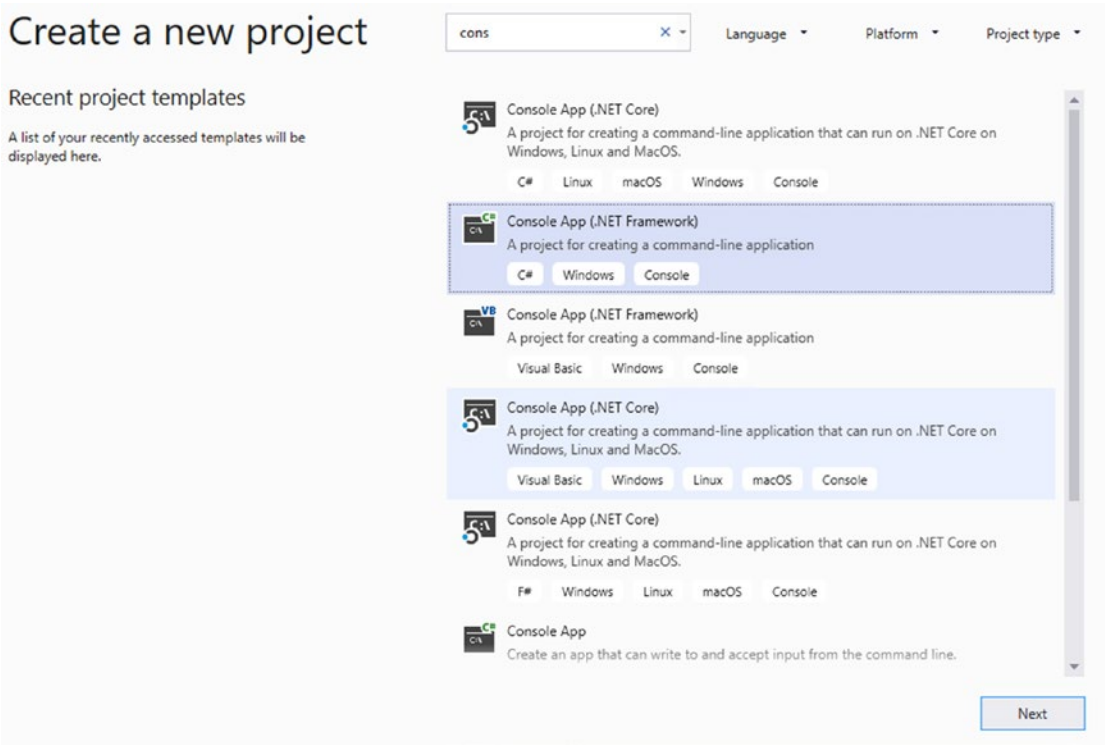

Figure 1-11. *Creating a console application*

In the next step, provide a name for the project. Do not change the Location path as the new solution is already being created in the mapped path of TFVC. Click Create to create the new application. See Figure 1-12.

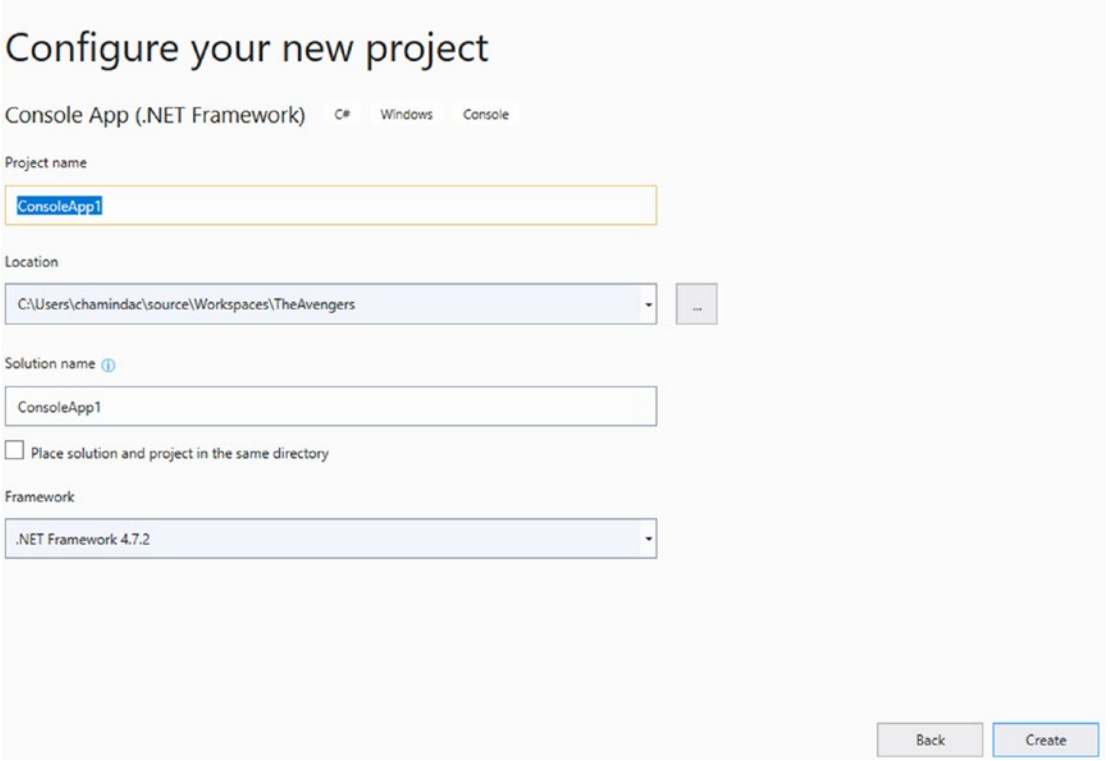

Figure 1-12. *Creating the console application in the mapped source path*

Once the new solution is created, view it in the Solution Explorer window of Visual Studio. You can open Solution Explorer by pressing Ctrl+Alt+L or by clicking View ➤ Solution Explorer in the Visual Studio menu. If you have an existing solution that you need to add to TFVC, copy all the content of the solution to the mapped local drive path of TFVC and then open that solution in Visual Studio. Once you create/open a solution in Visual Studio, go to the Solution Explorer window and right-click the solution you want to add to TFVC. Then click Add Solution to Source Control in the context menu. See Figure 1-13.

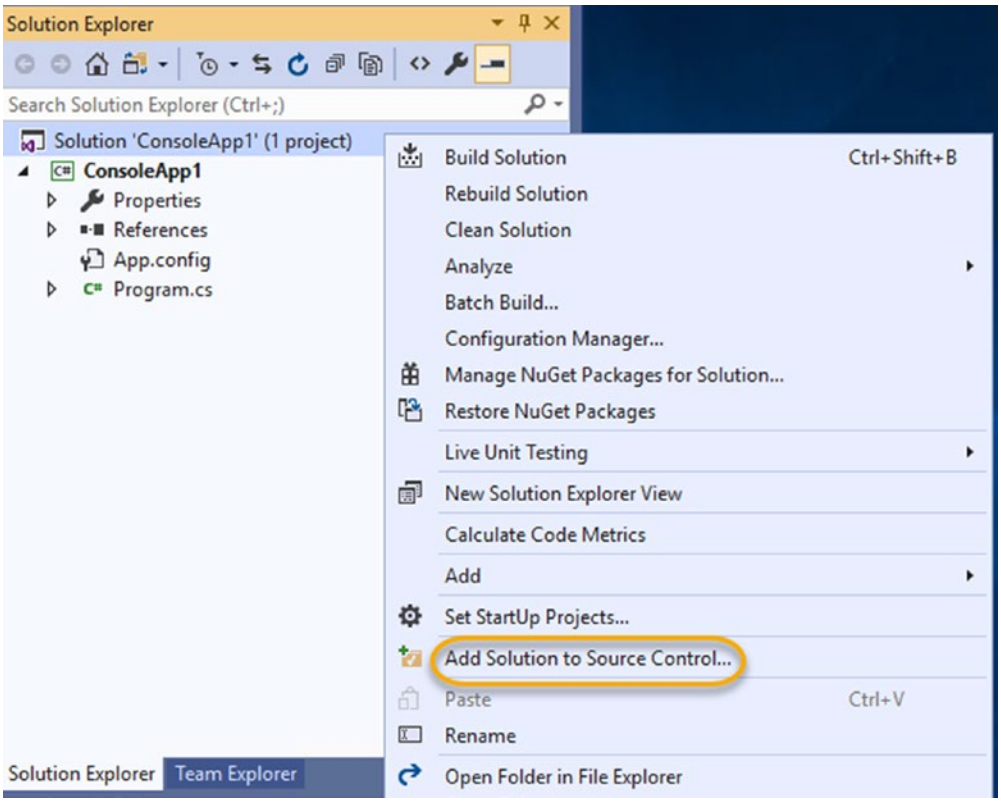

Figure 1-13. *Adding the solution to source control*

You will notice that all the files in the solution are marked with a +, indicating that they are ready to be checked in (committed) to TFVC. See Figure 1-14.

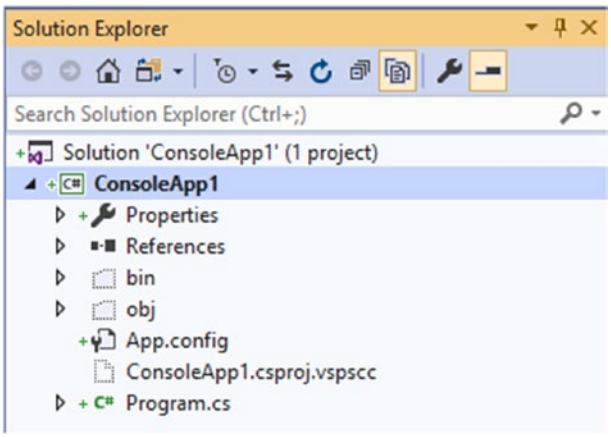

Figure 1-14. *Solution ready to be checked in*

Next open the Team Explorer window and click Pending Changes. In the Pending Changes window of Team Explorer, you will be able to see the new solution files are ready to be checked in. You can provide a comment and check in your code to TFVC. Further, you can see the Related Work Items options allowing you to add a work item, which we will discuss in Chapter 2. Note that there are some local file changes detected (this is because of the default local workspace; we will discuss the difference between the server and local workspaces that are available for TFVC in Lesson 1-4). See Figure 1-15.

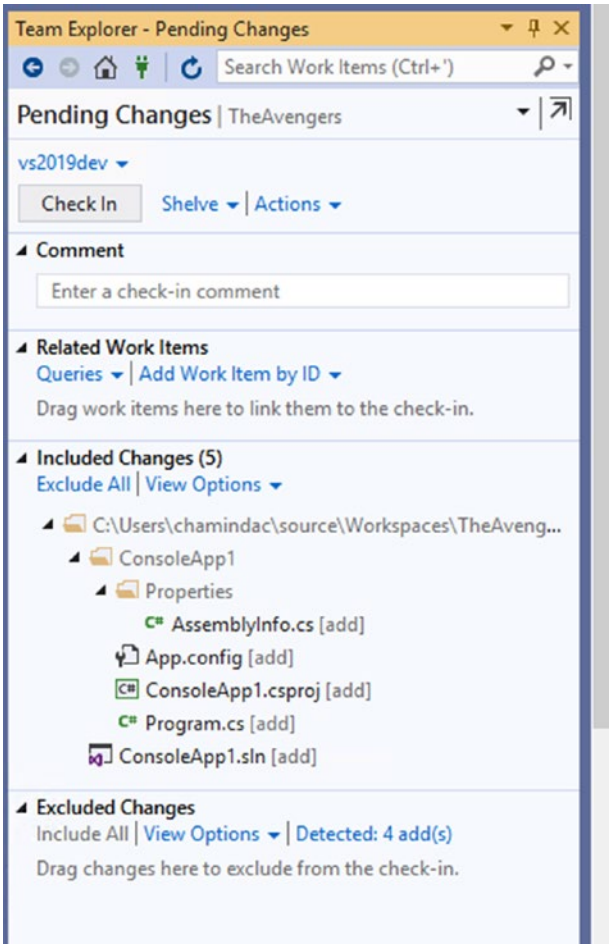

Figure 1-15. *Pending changes*

Click the detected changes in the Excluded Changes section. A dialog will appear, and you will be able to see the local files that should be ignored by source control. Select all the files and right-click to open a context menu. In this window, you are allowed to

promote files, which will be included as changes. Or you can ignore local-only files. Since the files detected in this instance are local files, click "Ignore these local items." Note that several ignore options are available when you have selected a single file, the same file extension, etc. See Figure 1-16.

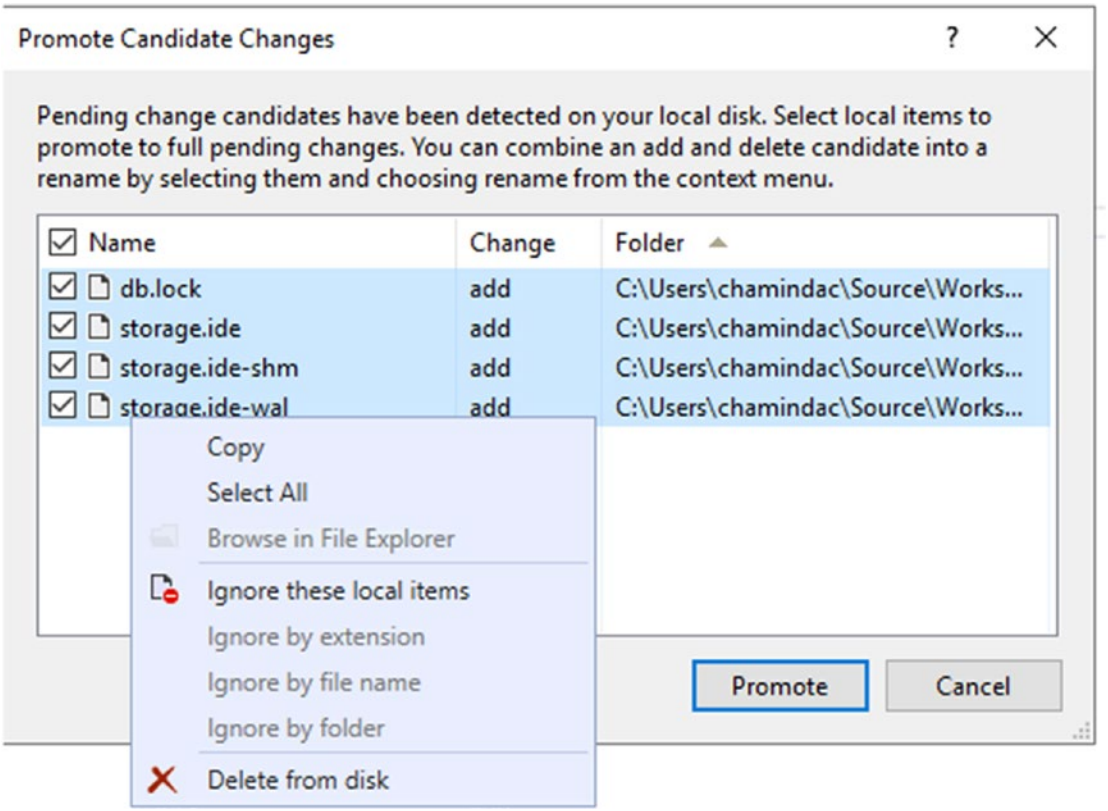

Figure 1-16. *Local changes*

You will notice a new file is added to the included changes named `.tfignore` in the pending changes. See Figure 1-17.

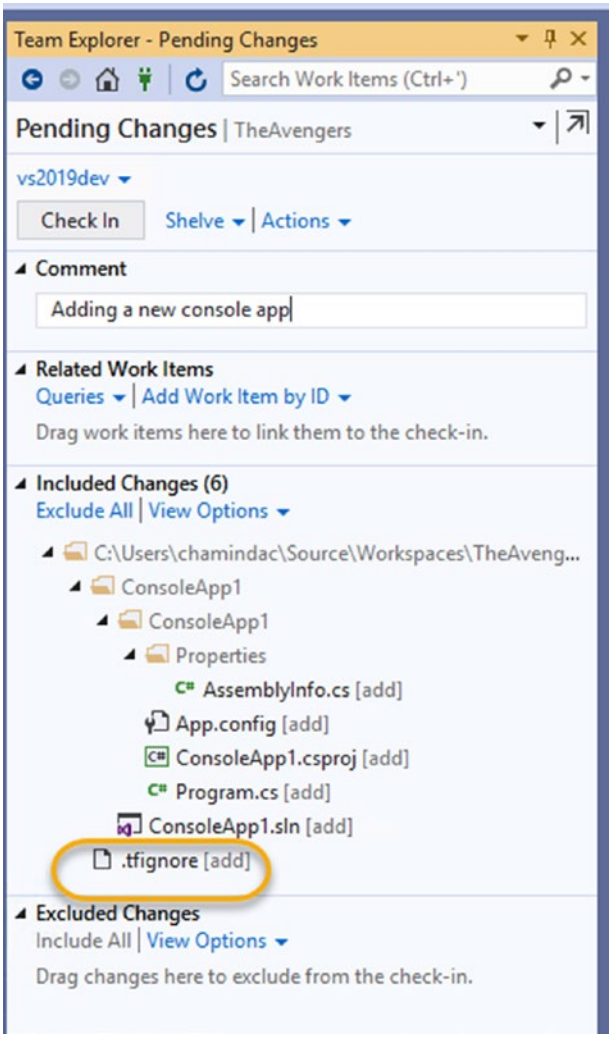

Figure 1-17. *The .tfignore file added*

Double-click the `.tfignore` file and inspect its content. The `.tfignore` file is used to specify which files/paths should be ignored from TFVC. You can define the ignore file patterns using wildcards. The `.tfignore` file contains a documentation header that itself is a good explanation of how to use the file. Provide a comment in the Pending Changes window and click the "Check in" button to commit the code to TFVC. In the Solution Explorer, the files are now marked with a lock icon indicating they are checked in to TFVC.

In this lesson, we discussed how to get a solution added to TFVC using Visual Studio's Solution Explorer. Further, we looked at how to ignore local files from getting checked in to TFVC using a `.tfignore` file.

Lesson 1-4: Changing Settings for TFVC in Visual Studio

There are a couple of settings you can set in Visual Studio to manage the behavior of TFVC. They are divided into two levels: project collection settings that are applicable to an Azure DevOps organization or a project collection in Azure DevOps Server and team project settings that applicable in a team project scope.

You can access the project collection TFVC settings by clicking Team ➤ Team Project Collection Settings ➤ Source Control. See Figure 1-18.

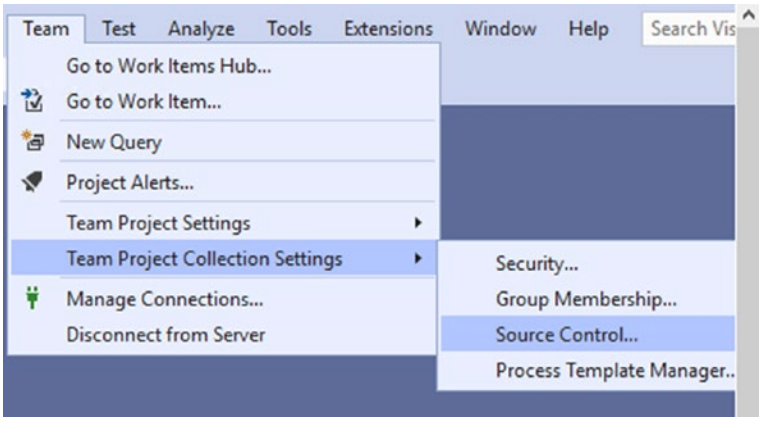

Figure 1-18. *TFVC team project collection settings*

File Types

The File Types settings let you define the enabled file types that can be added to source control and the types that are prevented from being added to source control in TFVC. See Figure 1-19.