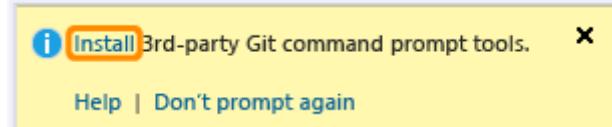


# Work from the Git command prompt

Visual Studio provides most of the fundamental capabilities to develop an app in a Git version-controlled codebase. You might have to use the command prompt for some manual tasks or to automate work using a script.



## Work from the command prompt

**Caution:** If you are not an experienced Git user, use the command prompt carefully. Make sure to research the command thoroughly before you use it.

I want to...

Can I do it in  
Visual Studio?

How do I do it from the command prompt?

Amend my last commit. Some typical cases:

Yes

[git-commit](#)

- Add a file you wanted to include in the commit. See [Undoing Things](#).
- Modify a comment. For example, maybe you want to change "Fix a bug" to "Fix bug #32" in order to associate the commit with a work item when you push your changes to TFS. See [How do I edit an incorrect commit message in Git?](#)

Apply a tag to a commit

Yes

You can use the command prompt to push, edit tags (see [Git-scm: Git Basics - Tagging](#)) from a repository if you have sufficient permissions.

Branch and merge

Yes (but some conflicts can be resolved only at command prompt).

[git-branch](#), [git-merge](#)

Commit my changes

Yes

[git-commit](#)

**Tip:** You can associate a work item with a commit by adding the ID in your comment. For example, you apply #35 Catch null exception to your commit. When you push the commit into TFS, the commit will be associated with work item #35.

Copy (clone) a remote repository to my dev machine

Yes

[git-clone](#)

Create (initialize) a local repository

Yes

[git-init](#)

Create or edit a note

No

You can use the command prompt to push, edit notes (see [Git-scm: Note to Self](#)) from a repository

have sufficient permissions.		
Get information about my local repository (such as the remotes I am tracking)	Yes	<a href="#">git-remote</a>
Preview (fetch) and then download (pull) changes from a remote repository	Yes (but some conflicts can be resolved only at the command prompt)	<a href="#">git-fetch</a> , <a href="#">git-pull</a>
Push changes to a remote repository	Yes	<a href="#">Git-scm: git-push</a>
Replay commits from one branch onto another	Yes	<a href="#">Git-scm: Git Branching - Rebasing</a> <a href="#">Git-scm: Rewriting History</a> <a href="#">git-rebase</a>
Re-order history, combine (squash) commits.	No	<a href="#">Git-scm: Git Branching - Rebasing</a> <a href="#">Git-scm: Rewriting History</a> <a href="#">git-rebase</a>
Revert a committed change by applying the inverse of the commit. See rolling back changes with revert.	Yes	<a href="#">git-revert(1) Manual Page</a>
Stash changes	No	<a href="#">Git-scm: Git Tools - Stashing</a>
Undo committed changes by returning my local repo to a prior commit and de-referencing the later commit. <b>Caution:</b> According to <a href="#">Undoing Things</a> , "...this is a dangerous command: Any changes you made to that file are gone - you just copied another file over it. Don't ever use this command unless you absolutely know that you don't want the file."	No	<a href="#">git-reset</a>
View and manage my changes since the last commit	Yes	<a href="#">Git-scm: Git Basics - Recording Changes to the File System</a> <a href="#">Git-scm: Git Basics - Undoing Things</a>
View history	Yes	<a href="#">Git-scm: Git Basics - Viewing the Commit History</a>

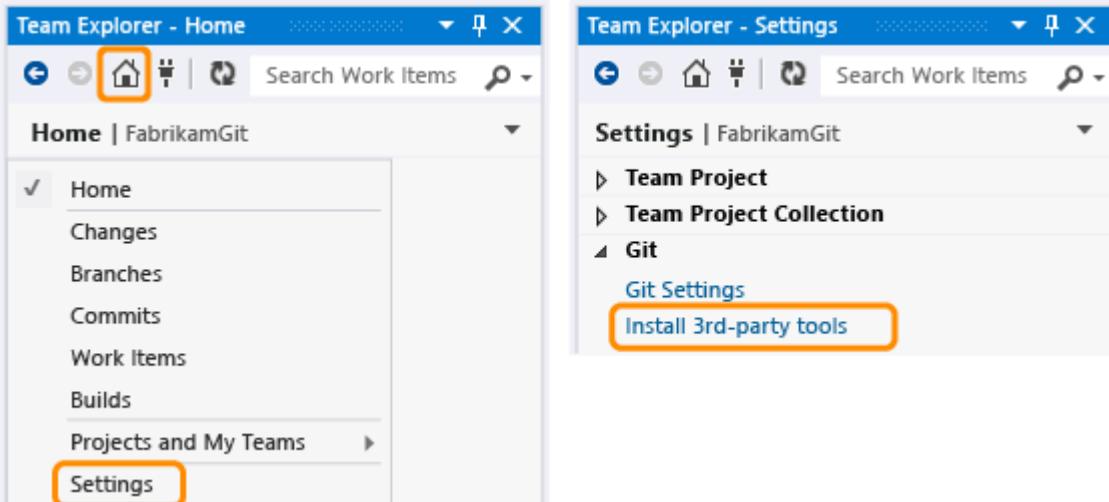
## Get set up to use the command prompt tools

If you already have Git installed and on your PATH, then you only need to [create a personal access token](#) or [alternate authentication credentials](#) for authentication.

If you do not have Git, you can install it from [here](#) or follow the instructions below to install Git and command prompt tools.

### Install the command prompt tools

If you have not already installed some command prompt tools, you can get some quickly from Visual Studio. (One way you can tell that you don't have the tools is if you try to enter a git command and get the `'git' is not recognized as an internal or external command...` message.)

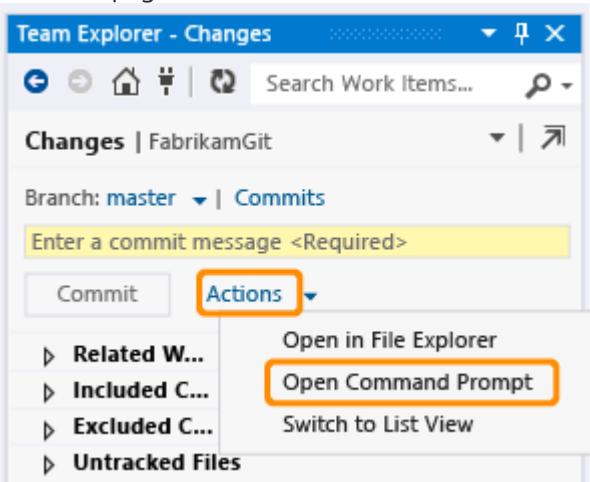


#### Tips:

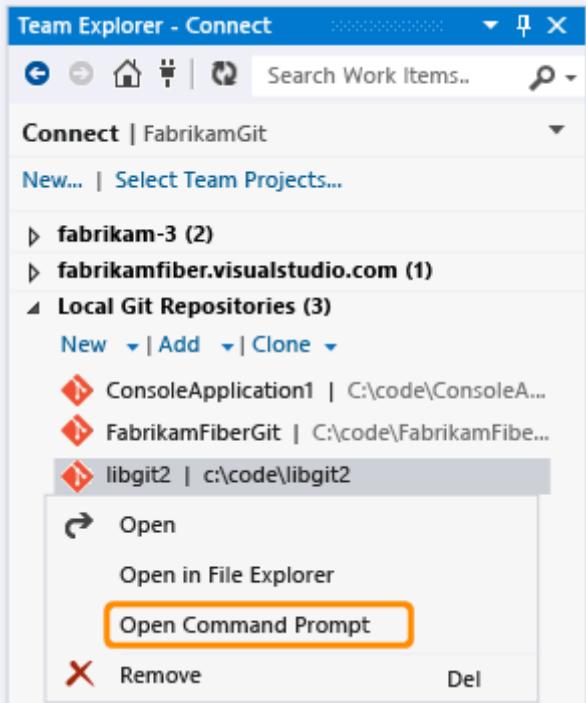
- The install process drops a Git Bash icon on your desktop. We recommend you delete this icon because we don't believe this entry point leads to the best experience. If for some reason you want to run Git Bash later, you can do so from Windows Start.
- To make using the command prompt less tedious (for example, to avoid having to enter your credentials every time you push), you might want to also install [Windows Credential Store for Git](#).
- If you want to run Git commands from PowerShell, install [Posh-Git \(a PowerShell environment for Git\)](#).

## Launch the Git command prompt

You can launch the Git command prompt from the Actions menu on the Changes (Keyboard: Ctrl + 0, G), Commits, and Branches pages.



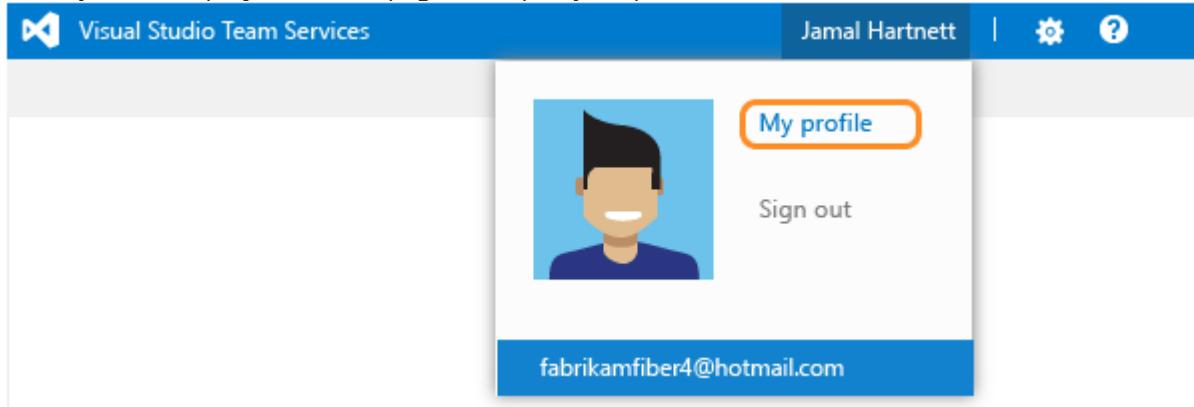
You can also launch the Git command prompt from repositories on the Connect (Keyboard: Ctrl + 0, C) page.



## Create a personal access token for your Visual Studio Team Services account

If your repository is hosted on Visual Studio Team Services, you must authenticate access to your account before you can use the command prompt to perform Git tasks. To authenticate access, create a personal access token.

1. Sign in to your Visual Studio Team Services account ([http://\[youraccount\].visualstudio.com](http://[youraccount].visualstudio.com)).
2. Go to your team project's home page and open your profile.



3. Create a new personal access token for this account.

Visual Studio Team Services

DETAILS

Profile Security

Personal Access Tokens

Alternate Authentication Credentials

OAuth Authorizations

Add Revoke All

You have not created any personal access tokens.

4. Enter the details for this token.

- Create a personal access token

Description	<input type="text"/>
Expires In	90 days <input type="button" value="▼"/>
Accounts	fabrikam-inc <input type="button" value="▼"/>

5. Select the scopes that this token authorizes.

Authorized Scopes

All authorized scopes

Selected scopes

- |   |   |
|---|---|
| <input type="checkbox"/> Build (read and execute)             | <input type="checkbox"/> Build (read)                     |
| <input checked="" type="checkbox"/> Code (read and write)     | <input type="checkbox"/> Code (read)                      |
| <input type="checkbox"/> Code (read, write, and manage)       | <input type="checkbox"/> Team rooms (read and write)      |
| <input type="checkbox"/> Team rooms (read, write, and manage) | <input type="checkbox"/> Test management (read and write) |
| <input type="checkbox"/> Test management (read)               | <input type="checkbox"/> User profile (read)              |
| <input type="checkbox"/> Work items (read and write)          | <input type="checkbox"/> Work items (read)                |

6. After you finish creating the token, make sure to copy the token. You'll use this token as your password for your Git tools or application.

### Actions

[Copy Token](#) [Revoke](#)

**Make sure you copy the token now. You won't be able to see it again!**

rueagz4fv4hognmodgzjjhpubxvmf15jpwr6efzsdl7ldosgegq

**Note:** Remember that these tokens are your identity. When used, the token is acting as you. Keep your tokens secret and treat them like your password. To help keep your token more secure, consider using the [Windows Credential Store for Git](#) so that you don't have to enter your credentials every time you push.

For example, if you use the Git command prompt to run a Git command, you'll be prompted for a username and password.

```
git clone https://[account].visualstudio.com/DefaultCollection/_git/[team project]
```

Enter a username that does not contain an @ character (for example, Jamal, not fabrikamfiber4@hotmail.com). Use the token that you created as your password.

```
Username for 'https://fabrikam-inc.visualstudio.com': Jamal
```

```
Password for 'https://fabrikam-inc.visualstudio.com': [COPY THE TOKEN HERE]
```

- When you don't need your token anymore, just revoke the token to remove its access from this account.

## Q & A

### Q: Where can I learn more commands?

A: <http://git-scm.com/docs>

### Q: Why should I use personal access tokens?

A: Personal access tokens are a more convenient and secure replacement for alternate authentication credentials. You can create a token and limit its use to:

- The token's lifetime
- A Visual Studio Team Services account
- [Scopes](#) of activities that this token authorizes

### Q: Can I still use alternate authentication credentials?

A: Yes.

- Sign in to your Visual Studio Team Services account ([http://\[youraccount\].visualstudio.com](http://[youraccount].visualstudio.com)).
- Go to your team project's home page and open your profile.

A screenshot of the Visual Studio Team Services profile page. At the top, there's a blue header bar with the "Visual Studio Team Services" logo and the user name "Jamal Hartnett". Below the header is a large white area containing a placeholder profile picture of a man with dark hair and a blue shirt. To the right of the picture is a blue button labeled "My profile". Below the picture is a "Sign out" link. At the bottom of this section is a blue bar with the email address "fabrikamfiber4@hotmail.com".

3. Enable alternate authentication credentials for this account. Then provide a secondary username and password.

A screenshot of the Visual Studio Online security settings page. The top navigation bar shows "Visual Studio Online" and has tabs for "Profile" and "Security", with "Security" being the active tab. On the left, there's a sidebar with links for "Personal Access Tokens", "Alternate Authentication Credentials" (which is highlighted in blue), and "OAuth Authorizations". The main content area is titled "Alternate Authentication Credentials". It contains a checked checkbox for "Enable alternate authentication credentials". Below it are four input fields: "User name (primary)" with the value "fabrikamfiber4@hotmail.com", "User name (secondary)" with the value "Jamal", "Password" with the value "\*\*\*\*\*", and "Confirm Password" with the value "\*\*\*\*\*". At the bottom are "Save" and "Cancel" buttons.