Team Foundation Server

This page contains descriptions of the fields and options available when setting up a VCS root to connect to Microsoft Team Foundation Server Version Control.
Common VCS Root properties are described here.

When connecting to a Azure DevOps Git repository, select Git as Type of VCS.

> TeamCity can also automatically configure the project / VCS root from a repository URL.
>
> If you have a TFVC root configured, TeamCity will suggest configuring the Team Foundation Work Items as well.

On this page:

Cross-Platform TFS Integration

TeamCity features the cross-platform TFS integration, which works on Linux, macOS, and Windows platforms. Without installing additional software, TeamCity servers and build agents can interact with Team Foundation Servers 2010 - 2018 and Visual Studio Team Services.

The built-in TFS plugin can work in two modes: the default and cross-platform. The working mode is based on the availability of Team Explorer (default mode): if it is not present, the plugin falls back from the default to cross-platform mode.

When detecting the Team Explorer version, TeamCity checks .NET GAC and the following paths:
- `Windows x86: %CommonProgramFiles%\Microsoft Shared\Team Foundation Server\%version_number%`
- `Windows x64: %CommonProgramFiles(x86)%\Microsoft Shared\Team Foundation Server\%version_number%`

To enforce the cross-platform mode on TeamCity, set the `teamcity.tfs.mode=java` internal property or build configuration parameter.

TFS Settings

Option	Description
URL	Team Foundation Server URL in the following format: TFS 2010+: `http[s]://<host>:<port>/tfs/<collection>` TFS 2005/2008: `http[s]://<host>:<port>` Azure DevOps: `https://dev.azure.com/<organization>` VSTS: `https://<accountname>.visualstudio.com`
Root	Specify the root using the following format: `$<project name><project catalogue>`
Username	Specify a user to access Team Foundation Server. This can be a user name or `DOMAIN\UserName` string. Use blank to let TFS select a user account that is used to run the TeamCity Server (or Agent for the agent-side checkout).
Password	Enter the password of the user entered above

Learn more about authentication in Azure DevOps.

Agent-Side Checkout

The agent-side checkout is supported on Windows, as well as Linux and Mac agent machines.

TeamCity automatically creates a TFS workspace for each checkout directory used. The workspace is created on behalf of the user account specified in the VCS root settings.

By default, the created TFS workspace uses the location defined in the TFS server settings. You can force TeamCity to use a specific workspace location via the build configuration parameter `teamcity.tfs.workspace.location` set to `local` or `server`.

The created TFS workspaces are automatically removed based on the timeout configured via the `teamcity.tfs.workspace.idleTime` build agent property , set to the default value of `1209600` sec (2 weeks).

Option	Description
Enforce overwrite all files	When the option is enabled, TeamCity will call TFS to update workspace rewriting all files.

Normally, there is no need to do a forced update for every build. But, if you suspect that TeamCity is not getting the latest version from the repository, you can use this option.

TFS does not allow several workspaces on a machine mapped to the same directory. If it happens, the TeamCity TFS agent-side checkout will attempt to remove intersecting workspaces to create a new workspace that matches the specified VCS root and checkout rules.

Note that it can fail to remove workspaces created by another user, and in this case you need to remove such workspaces manually.

It is recommended to use checkout rules of the format below to differentiate local mappings:

```
$/root1 => /root1
$/root2 => /root2
```

Authentication Notes

The following authentication options are available in Azure DevOps.

Personal Access Tokens

To use access tokens, you need to create a personal access token in your Azure DevOps account, where you have to set some Code access scope in your repositories and use it when configuring a VCS root.

Option	Description
Username	Leave blank for TFVC, any value for Git, e.g. username
Password	Enter your personal access token created earlier

Required Access Scope

TFS subsystem	Scopes
TFVC	All scopes
Git	Code (read) / Code (read and write) for versioned settings
Work Items	Work items (read)
Commit Status	Code (status)

Alternate Authentication Credentials

To use the login/password pair authentication, you have to enable alternate credentials in your Azure DevOps account, where you can set a secondary username and password to use when configuring a VCS root.

NTLM/Kerberos on Linux and macOS

To use this authentication method, check that your machine includes Kerberos libraries and that the authentication is properly configured. If you encounter any issues, please check the steps described in the Microsoft documentation.

Team Foundation Proxy Configuration

To enable usage of Team Foundation Proxy, define the `TFSPROXY` environment variable for the user account which runs the TeamCity server or agent and restart them to apply changes.

Example

```
TFSPROXY=https://tfs-proxy:8081
```

HTTP Proxy Server Configuration

Default .NET Working Mode

To interact with the TFS server, the proxy server settings specified for the user account which runs the TeamCity server or agent will be used.

Cross-Platform Working Mode

The default Java proxy server settings specified for the TeamCity server or agent will be used in the TFS integration. On the TeamCity server, internal properties or Java options can be used. On the TeamCity agent, build agent configuration or Java options can be used. The TeamCity-TFS integration supports the following options:

```
http.proxyHost
http.proxyPort
http.nonProxyHosts
https.proxyHost
https.proxyPort
http.proxyUser
http.proxyPassword
```