Get started with Azure DevOps documentation

Start using Azure DevOps to collaborate on code, build and deploy apps, or plan and track work.

About Azure DevOps

📖 OVERVIEW

What is Azure DevOps?

What is Azure Boards?

What is Azure Repos?

What is Azure Pipelines?

Get started as a Stakeholder

Sign up for free

🚀 QUICKSTART

Azure DevOps (includes all services)

Azure Boards

Azure Repos

Azure Pipelines

📖 HOW-TO GUIDE

Connect to a project

Manage your project

🚀 QUICKSTART

Manage your project

Add users to a team or project

Add a project administrator

Add a project collection administrator

{≡} CONCEPT

Default permissions and access

Permission lookup guide

Plan and track work

🚀 QUICKSTART

Plan and track work

Create your backlog

Collaborate on code

🚀 QUICKSTART

Clone a repository

Create a repository

Authenticate with SSH

Search code across projects

Build and deploy your apps

🚀 QUICKSTART

Create your first pipeline

📷 REFERENCE

YAML Schema

Set your preferences

🚀 QUICKSTART

[Change profile preferences](#)

[Manage your notifications](#)

[Preview new features](#)

[Set favorites](#)

Navigate

📖 HOW-TO GUIDE

[Web portal navigation](#)

[Team Explorer navigation](#)

Migrate & import

📑 CONCEPT

[Migrate from TFS to Azure DevOps Services](#)

[Migration options](#)

[Import](#)

What is Azure DevOps?

Article • 01/04/2024

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Azure DevOps supports a collaborative culture and set of processes that bring together developers, project managers, and contributors to develop software. It allows organizations to create and improve products at a faster pace than they can with traditional software development approaches.

You can work in the cloud with Azure DevOps Services or on-premises with Azure DevOps Server. For more information, see Differences between Azure DevOps Services and Azure DevOps Server.

Azure DevOps provides integrated features that you can access through your web browser or IDE client. You can use all the services included with Azure DevOps or choose just what you need to complement your existing workflows.

⟦ ⟧ Expand table

Standalone service	Description
Azure Boards ⧉	Delivers a suite of Agile tools to support planning and tracking work, code defects, and issues using Kanban and Scrum methods. For more information about Azure Boards, see What is Azure Boards?.
Azure Repos ⧉	Provides Git repositories or Team Foundation Version Control (TFVC) for source control of your code. For more information about Azure Repos, see What is Azure Repos?.
Azure Pipelines ⧉	Provides build and release services to support continuous integration and delivery of your applications. For more information about Azure Pipelines, see What is Azure Pipelines?.
Azure Test Plans ⧉	Provides several tools to test your applications, including manual/exploratory testing and continuous testing. For more information about Azure Test Plans, see Overview of Azure Test Plans.
Azure Artifacts ⧉	Allows teams to share packages such as Maven, npm, NuGet, and more from public and private sources and integrate package sharing into your pipelines. For more information about Azure Artifacts, see Overview of Azure Artifacts.

Azure DevOps supports adding extensions and integrating with other popular services, such as: Campfire, Slack, Trello, UserVoice, and more, and developing your own custom

extensions.

Choose Azure DevOps Services

Azure DevOps *Services* supports integration with GitHub.com and GitHub Enterprise Server repositories. Choose Azure DevOps Services when you want the following outcomes:

- Quick set-up
- Maintenance-free operations
- Easy collaboration across domains
- Elastic scale
- Rock-solid security

Azure DevOps Services also gives you access to cloud build and deployment servers, and application insights. Start for free and create an organization. Then, either upload your code to share or source control. Begin tracking your work using Scrum, Kanban, or a combination of methods.

For more information, see the Azure DevOps and GitHub integration overview.

Choose Azure DevOps Server

Azure DevOps *Server* supports integration with GitHub Enterprise Server repositories. Choose on-premises Azure DevOps Server when you need your data to stay within your network. or your work tracking customization requirements are better met with the on-premises XML process model over the inheritance process model. The on-premises model supports modification of XML definition files.

When you deploy Azure DevOps Server, you can also configure the following servers or integration points:

- **Build server** supports on-premises and cloud-hosted builds.
- **SQL Server and SQL Analysis Server** support SQL Server Reports and the ability to create Excel pivot charts based on the cube.

Start for free by downloading Azure DevOps Server Express . Then, either upload your code to share or source control. Or, begin tracking your work using Scrum, Kanban, or a combination of methods.

For more information about managing Azure DevOps Server, see the Administrative tasks quick reference.

Next steps

Sign up for Azure DevOps Services or **Install Azure DevOps Server**

Related articles

- A tour of services
- Data protection overview
- Client-server tools
- Software development roles
- Azure DevOps pricing 🗗
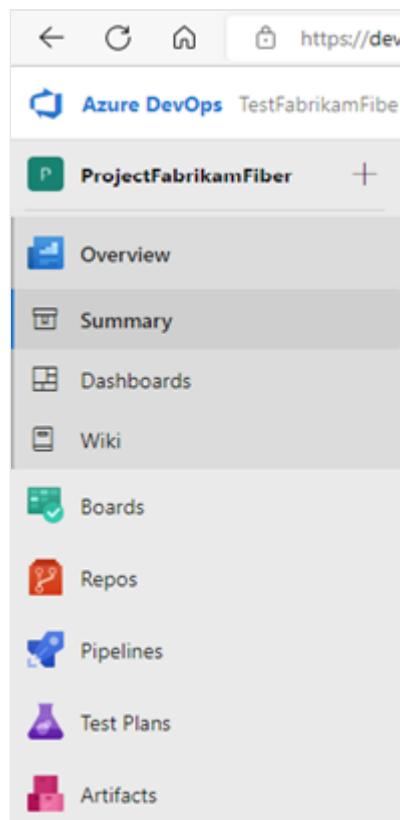- Azure DevOps and GitHub integration overview

Overview of services

Article • 11/11/2022

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Azure DevOps provides an integrated set of services and tools to manage your software projects, from planning and development through testing and deployment.

Azure DevOps delivers services through a client/server model. You can use most of the services via the web interface, which you can access from all major browsers. Some services, such as source control, build pipelines, and work tracking, can also be managed through a client.

Access Azure DevOps through the left navigational bar, as shown in the following image. For more information, see the following associated articles.

- Dashboards
- Wiki
- Boards
- Repos
- Pipelines
- Test Plans
- Artifacts

Many of our services are either free for small teams or available through a subscription model or per-use model. You can do a hybrid approach where you use an on-premises deployment to manage your code and work. Then, purchase cloud build or testing services on an as-needed basis.

For more information about client tools, see Tools and clients that connect to Azure DevOps.

Dashboards

Gain access to user-configurable dashboards from **Dashboards**.

You can do the following tasks in **Dashboards**:

- Add, configure, and manage dashboards
- Configure widgets that you add to dashboards
- Go to different areas of your project quickly

For more information, see Dashboards.

Repos

Source or version control systems allow developers to collaborate on code and track changes made to the code base. Source control is an essential tool for multi-developer projects.

Our systems support two types of source control: Git or Team Foundation Version Control (TFVC). You can check in files and organize files within folders, branches, and repositories in both systems.

Git repos

With Git, each developer has a copy on their dev machine of the source repository, including all branch and history information. Each developer works directly with their own local repository and changes are shared between repositories as a separate step.

Developers commit each set of changes and do version control operations like history and compare without a network connection. When developers need to switch contexts, they create a private local branch, and can switch from one branch to another to pivot among different variations of the codebase. Later, they merge, publish, or dispose of the branch.

> ⊘ **Note**
>
> Git in Azure DevOps is standard Git. You can use Visual Studio with third-party Git services. You can also use third-party Git clients with Azure DevOps Server.

TFVC

With TFVC, developers have only one version of each file on their dev machines. Historical data is maintained only on the server. Branches are path-based and created on the server.

Access Git and TFVC

From **Repos**, you gain access to your source control Git-based or Team Foundation Version Control (TFVC) repositories to support version control of your software projects. These repositories are private.

From Azure Repos for Git, you can do the following tasks:

- Review, download, and edit files, and review the change history for a file
- Review and manage commits that have been pushed
- Review, create, approve, comment on, and complete pull requests
- Add and manage Git tags

Boards

Software development projects require ways to easily share information and track the status of work, tasks, issues, or code defects. In the past, you might have used Microsoft Excel, Microsoft Project, a bug tracking system, or a combination of tools. Now, many teams have adopted Agile methods and practices to support planning and development.

From **Boards**, you gain access to Agile tools to support planning and tracking work.

You can do the following tasks with boards.

- Add and update work items
- Define work item queries, and create status and trend charts based on those queries
- Manage your product backlog
- Plan sprints by using sprint backlogs
- Review sprint tasks and update tasks through the task boards
- Visualize the workflow and update the status by using Kanban boards
- Manage portfolios by grouping stories under features and grouping features under epics
- Use task boards during daily Scrum meetings to review work that's completed, remaining, or blocked

Our systems provide several types of work items that you use to track features, requirements, user stories, tasks, bugs, and issues. Each work item is associated with a work item type and a set of fields that can be updated, as progress is made.

For planning purposes, you have access to several types of backlogs and boards to support the main Agile methods—Scrum, Kanban, or Scrumban.

Project managers and developers share information by tracking work items on the backlogs and boards. Useful charts and dashboards complete the picture and help teams monitor progress and trends.
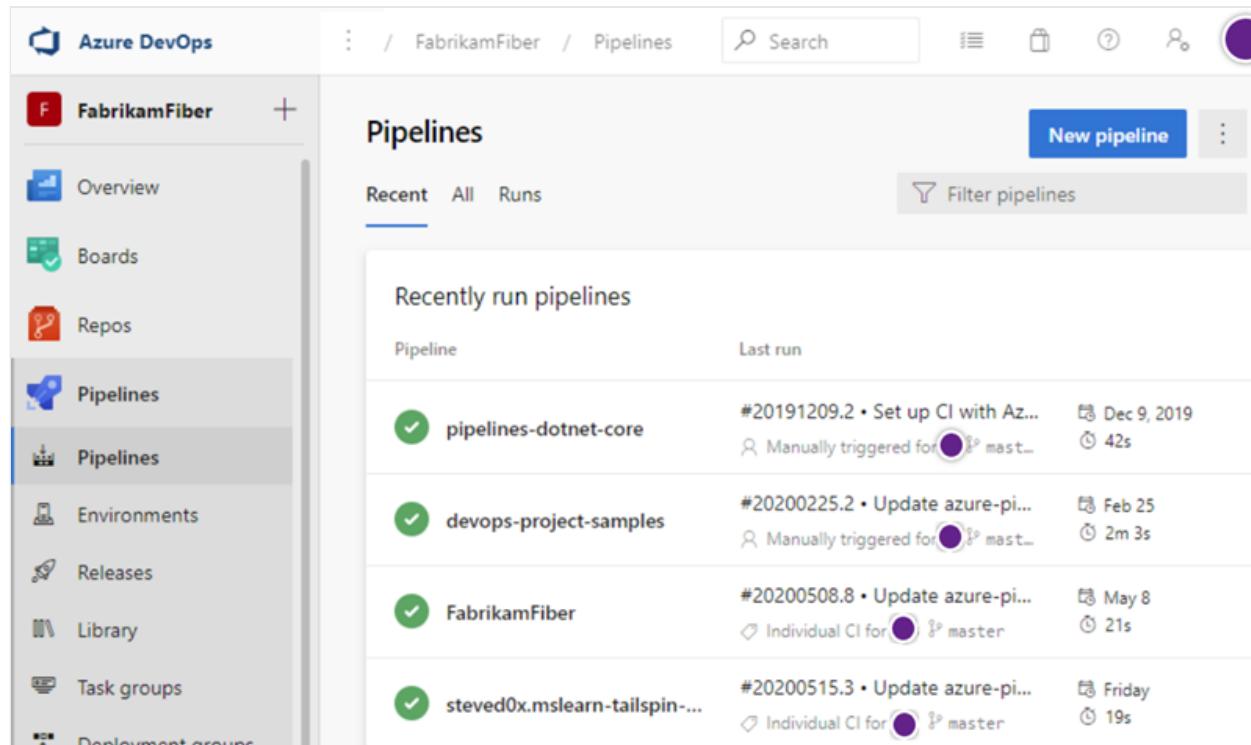
See Backlogs, boards, and plans for an overview of each.

Pipelines

The rapid and reliable release of software comes from automating as many processes as possible. Our systems support build, test, and release automation.

- You can define builds to automatically run whenever a team member checks in code changes.
- Your build pipelines can include instructions to run tests after the build runs.
- Release pipelines support managing deployment of your software builds to staging or production environments.

Azure Pipelines provides an integrated set of features to support building and deploying your applications.

Use pipelines to implement continuous integration and continuous delivery.

- **Build automation**: Define the steps to take during build and the triggers that start a build.
- **Release management**: Manage simultaneous releases. You can also do the following tasks:

- Configure release pipelines that represent your environments from development to production
- Run automation to deploy your app to each environment
- Add approvers to confirm that the app has been successfully deployed in an environment
- Create your release manually or automatically from a build
- Track your releases as they're deployed to various environments

For more information, see Continuous integration on any platform.

Test Plans

Test Plans supports creating and managing manual, exploratory, and continuous tests.

With test features, you gain access to the following features:

- Customization of workflows with test plan, test suite, and test case work items
- End-to-end traceability from requirements to test cases and bugs with requirement-based test suites
- Criteria-based test selection with query-based test suites
- Excel-like interface with the grid for easy creation of test cases
- Reusable test steps and test data with shared steps and shared parameters
- Sharable test plans, test suites, and test cases for reviewing with Stakeholders
- Browser-based test execution on any platform
- Real-time charts for tracking test activity

For more information, see Azure Test Plans documentation.

Collaboration services

Azure DevOps also provides the following collaboration services.

- [Team dashboards](#)
- [Project wiki](#)
- [Discussion within work item forms](#)
- Linking of [work items](#), [commits](#), [pull requests](#), and other artifacts to support traceability
- [Alerts and change notifications](#) managed per user, team, project, or organization
- Ability to [request](#), [provide](#), and manage feedback
- [Analytics service](#), [analytics views](#), and [Power BI reporting](#)

Service hooks

With service hooks, you can complete tasks on other services when events happen within your project hosted on Azure DevOps. For example, you can send a push notification to your team's mobile devices when a build fails. You can also use service hooks in custom apps and services as a more efficient way to drive activities in your projects.

The following services are available as the target of service hooks. For more information about other apps and services that integrate with Azure DevOps, visit the [Visual Studio Marketplace](#) ⧉ .

For the latest set of supported services, see [Integrate with service hooks](#).

Azure cloud-hosted services

Azure provides cloud-hosted services to support application development and deployment. You can make use of these services solely or in combination with Azure DevOps.

To browse the directory of integrated services, features, and bundled suites, see [Azure products](#) ⧉ .

For continuous delivery to Azure from Azure DevOps, see [Automatically build and deploy to Azure web apps or cloud services](#).

Administrative services

There are features and tasks associated with administering a collaborative software development environment. You can complete most of these tasks through the web portal. For more information, see [About user, team, project, and organization-level settings](#).

Related articles

- Azure DevOps Services **vs.** Azure DevOps Server
- Client-server tools
- Software development roles
- Azure DevOps pricing ⧉

Compare Azure DevOps Services with Azure DevOps Server

Article • 10/10/2022

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

The **cloud offering**, Azure DevOps Services, provides a scalable, reliable, and globally available hosted service. It's backed by a 99.9% SLA, monitored by our 24/7 operations team, and available in local data centers around the world.

The **on-premises offering**, Azure DevOps Server, is built on a SQL Server back end. Customers usually choose the on-premises version when they need their data to stay within their network. Or, when they want access to SQL Server reporting services that integrate with Azure DevOps Server data and tools.

Both offerings provide the same essential features and services, but Azure DevOps Services offers the following added benefits:

- Simplified server management
- Immediate access to the latest and greatest features
- Improved connectivity with remote sites
- A transition from capital expenditures (servers and the like) to operational expenditures (subscriptions)

To determine which offering—cloud or on-premises—meets your needs, consider the following key differences.

Key differences between Azure DevOps Services and Azure DevOps Server

When you're choosing which platform you want, or if you're considering a move from on-premises to the cloud, consider the following areas:

- Data scope and scalability
- Authentication
- Users and groups
- User access management
- Security and data protection

Differences in specific feature areas

Although Azure DevOps Services is a hosted version of Azure DevOps Server, there are some differences between features. Some Azure DevOps Server features aren't supported in Azure DevOps Services. For example, Azure DevOps Services doesn't support integration with SQL Server Analysis Services to support reporting.

Differences in support

- Process customization
- Reporting

If you're using Azure DevOps Server and considering a move to Azure DevOps Services, understand your migration options.

Scope and scale data

As your business grows, you may need to scale up your Azure DevOps instance.

Azure DevOps Services

Azure DevOps Services offer two options for scoping and scaling data: organizations and projects. Organizations in Azure DevOps Services get their own URLs (for example, `https://dev.azure.com/fabrikamfiber`), and they always have exactly one project collection. Organizations can have many projects within a collection.

We recommend that you create organizations in Azure DevOps Services wherever you would create collections in Azure DevOps Server. The following scenarios apply:

- You can purchase Azure DevOps Services users per organization - Paid users can access only the organization in which the payment is made. If you have users who need access to many organizations, Visual Studio subscriptions can be an attractive option. Visual Studio subscribers can be added to any number of organizations at no charge. We're also considering other ways to make access available to many organizations that are grouped into a single organization.
- You currently have to administer organizations one at a time. This process can be cumbersome when you have many organizations.

For more information, see Plan your organizational structure in Azure DevOps.

Azure DevOps Server

Azure DevOps Server offers the following three options for scoping and scaling data: deployments, project collections, and projects. In the simplest case, deployments are just servers.

Deployments can be more complicated, however, which could include:

- Two-server deployment where SQL is split out on a separate machine
- High-availability farms with lots of servers

Project collections serve as containers for security and administration, and physical database boundaries. They're also used to group related projects.

Finally, projects are used to encapsulate the assets of individual software projects, including source code, work items, and so on. For more information, see Plan your organizational structure in Azure DevOps.

Authentication

Azure DevOps Services

With Azure DevOps Services, you connect over the public internet (for example, `https://contoso.visualstudio.com`). You either authenticate with Microsoft account ⧉ credentials or with Azure AD credentials, depending on your organization setup. You can also set up Azure AD to require features such as multi-factor-authentication, IP address restrictions, and so on.

We recommend that you configure your organizations to use Azure AD rather than Microsoft accounts. This method provides a better experience in many scenarios and more options for enhanced security.

For more information, see About accessing Azure DevOps Services with Azure AD.

Azure DevOps Server

With Azure DevOps Server, you connect to an intranet server (for example, `https://tfs.corp.contoso.com:8080/tfs`). You authenticate with Windows Authentication and your Active Directory (AD) domain credentials. This process is transparent and you never see any kind of sign-in experience.

Manage users and groups

Azure DevOps Services

In Azure DevOps Services, you can use a similar mechanism to provide access to groups of users. You can add Azure AD groups to Azure DevOps Services groups. If you use Microsoft Accounts instead of Azure AD, you have to add users one at a time.

Azure DevOps Server

In Azure DevOps Server, you provide users access to deployments by adding Active Directory (AD) groups to various Azure DevOps groups (for example, the Contributors group for an individual project). The AD group memberships are kept in sync. As users are added and removed in AD, they also gain and lose access to Azure DevOps Server.

Manage user access

In both Azure DevOps Services and Azure DevOps Server, you manage access to features by assigning users to an access level. All users must be assigned to a single access level. In both the cloud and on-premises offerings, you can give free access to work item features to an unlimited number of Stakeholders. Also, an unlimited number of Visual Studio subscribers can have access to all Basic features at no extra charge. You pay only for other users who need access.

Azure DevOps Services

In Azure DevOps Services, you must assign an access level to each user in your organization. Azure DevOps Services validates Visual Studio subscribers as they sign in. You can assign Basic access for free to five users without Visual Studio subscriptions.

To give Basic access or higher to more users, set up billing for your organization and pay for more users. Otherwise, all other users get Stakeholder access.

Azure AD groups give access to groups of users. Access levels are automatically assigned at first sign-in. For organizations that are configured to use Microsoft accounts for signing in, you must assign access levels to each user explicitly.

Azure DevOps Server

In Azure DevOps Server, all use is on the honor system. To set access levels for users based on their licenses, specify their access levels on the administration page. For example, assign unlicensed users Stakeholder access only.

Users with an Azure DevOps Server Client Access License (CAL) can have Basic access. Visual Studio subscribers can have either Basic or Advanced access, depending on their subscriptions. Azure DevOps Server doesn't attempt to verify these licenses or enforce compliance.

Security and data protection

Many entities want to know more about data protection when they consider moving to the cloud. We're committed to ensuring that Azure DevOps Services projects stay safe and secure. We have technical features and business processes in place to deliver on this commitment. You can also take steps to secure your data. Learn more in our Data Protection overview.

Process customization

You can customize the work-tracking experience in different ways, depending on the supported process model:

Azure DevOps Services

Azure DevOps Services uses the **Inheritance** process model, which supports WYSIWYG customization.

Azure DevOps Server

With Azure DevOps Server, you can choose the **Inheritance** process model or the **On-premises XML** process model, which supports customization through import or export of XML definition files for work-tracking objects. Azure DevOps Server 2018 and earlier versions only has access to the **On-premises XML** process model. Although the **On-premises XML** process model option is powerful, it can cause various issues. The main issue is that processes for existing projects aren't automatically updated.

To help you avoid these issues in Azure DevOps Services, custom process templates and the **witadmin.exe** tool have always been disabled. This approach has enabled us to automatically update all projects with each Azure DevOps Services upgrade. Meanwhile, the product team is working hard to make customizing processes possible in ways that we can support easily and continuously. We recently introduced the first of these changes and more changes are on the way.

With the new process-customization capability, you can make changes directly within the web user interface (UI). If you want to customize your processes programmatically, you can do so through REST endpoints. When you customize projects this way, they're automatically updated when we release new versions of their base processes with Azure DevOps Services upgrades.

For more information, see Customize your work-tracking experience.

Analytics and reporting

Azure DevOps Services and Azure DevOps Server offer the following tools that give you insight into the progress and quality of your software projects:

Azure DevOps Server 2019 to Azure DevOps Services

- Dashboards and lightweight charts that are available in both the cloud and on-premises platforms. These tools are easy to set up and use.
- The Analytics service and Analytics widgets. The Analytics service is optimized for fast read-access and server-based aggregations.
- Microsoft Power BI integration, which supports getting Analytics data into Power BI reports and provides a combination of simplicity and power.
- OData support, which allows you to directly query the Analytics service from a supported browser, and then use the returned JSON data as you want. You can generate queries that span many projects or your entire organization. To learn more about the Analytics service, see our Reporting roadmap.

Azure DevOps Server 2018

- Dashboards and lightweight charts that are available in both the cloud and on-premises platforms. These tools are easy to set up and use.
- SQL Server Reporting Services (SSRS) reports are available when Azure DevOps Server is configured with SQL Server Analysis Services.

Visual Studio Team Services is now Azure DevOps Services

Many of the featured services in VSTS are now offered as standalone services in both Azure DevOps Services and Azure DevOps Server 2019 and up. You can get services separately or all together as Azure DevOps Services. If you're an Azure DevOps subscriber, you have access to all of the services already.

VSTS feature name	Azure DevOps service name	Description
Build & release	Azure Pipelines	Continuous integration and continuous delivery (CI/CD) that works with any language, platform, and cloud.
Code	Azure Repos	Unlimited cloud-hosted private Git and Team Foundation Version Control (TFVC) repositories for your project.
Work	Azure Boards	Work tracking with Kanban boards, backlogs, team dashboards, and custom reporting.

VSTS feature name	Azure DevOps service name	Description
Test	Azure Test Plans	All-in-one planned and exploratory testing solution.
Packages (extension)	Azure Artifacts	Maven, npm, Python, Universal Package, and NuGet package feeds from public and private sources.

Azure DevOps Services and Azure DevOps Server 2019 and up use the new navigation user interface, with a vertical sidebar to go to the main service areas: **Boards**, **Repos**, **Pipelines**, **Artifacts**, **Test Plans**, and more. For more information, see Web portal navigation in Azure DevOps.

> ⓘ **Note**
>
> You can disable select services from the user interface. For more information, see **Turn a service on or off**.

You can still use `visualstudio.com` to access Azure DevOps Services. We've moved to the new `dev.azure.com` domain name as the primary URL for new organizations. That URL is `https://dev.azure.com/{your organization}/{your project}`. If you want to change your URL to be based on `dev.azure.com` as the primary, an organization administrator can do so from the organization settings page.

Related articles

- Essential services
- Client-server tools
- Software development roles
- Pricing for Azure DevOps Services⤤
- Pricing for Azure DevOps Server ⤤

Connect to a project

Article • 12/05/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019

Learn how to connect to a project, from a client, to share code, build apps, track work, and collaborate with team members. You can connect to a project from any of the following clients:

- Web portal
- Visual Studio or Team Explorer
- Android Studio with the Azure DevOps Services Plugin for Android Studio
- IntelliJ with the Azure DevOps Services Plugin for IntelliJ
- Visual Studio Code

A project defines a process and data storage in which you manage your software projects from planning to deployment. When you connect to a project, you connect to an organization or project collection. For more information, see About projects and scaling your organization.

Prerequisites

- You must have a project in your organization. If you don't have access to the project, get invited to the team.
- From each client, you can switch context to a different project and connect as a different user. If you work remotely, configure your client to connect to an Azure DevOps Proxy Server.
- To get started with a code base, set up Git or set up TFVC.
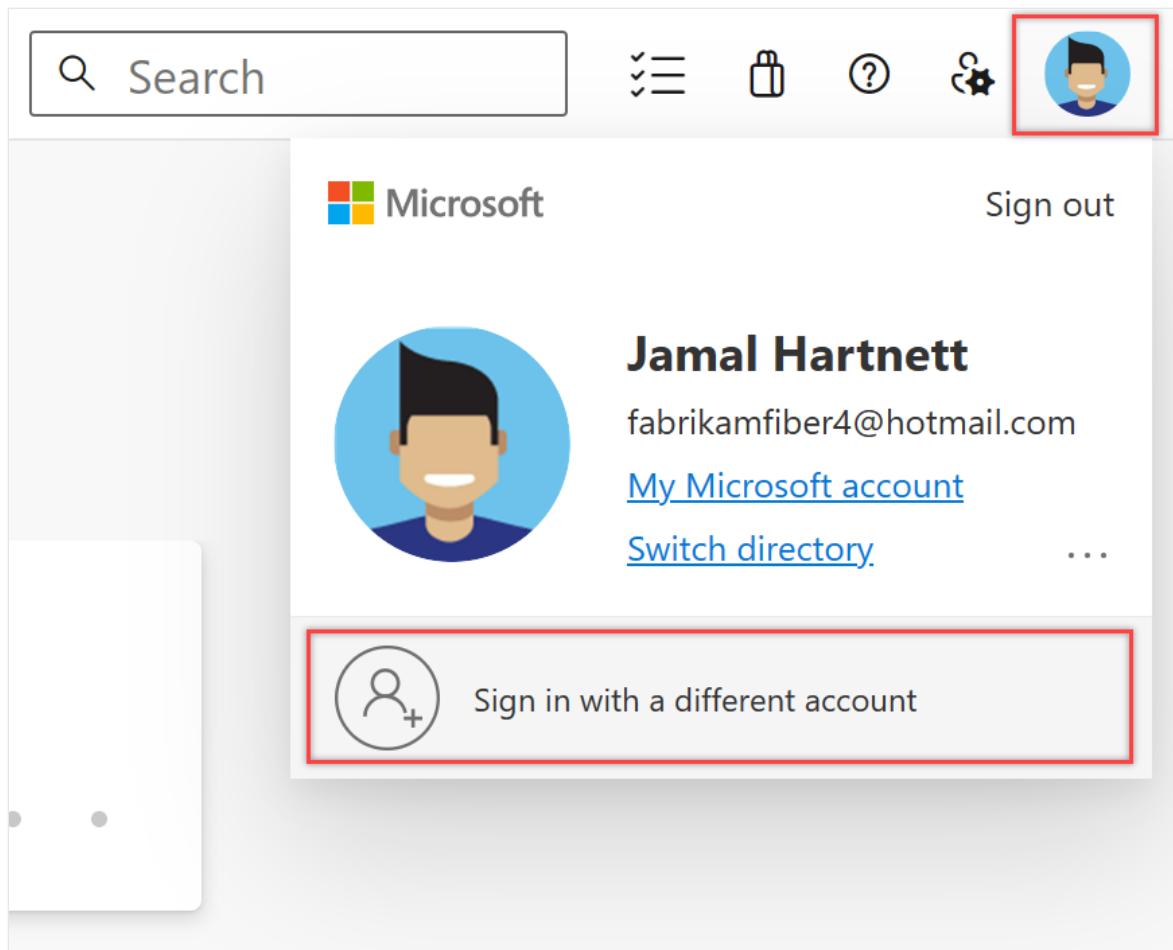
Connect from the web portal

- Sign in to your project (`https://dev.azure.com/{yourorganization}/{yourproject}`).

For more information, see Web portal navigation.

Sign in with different credentials
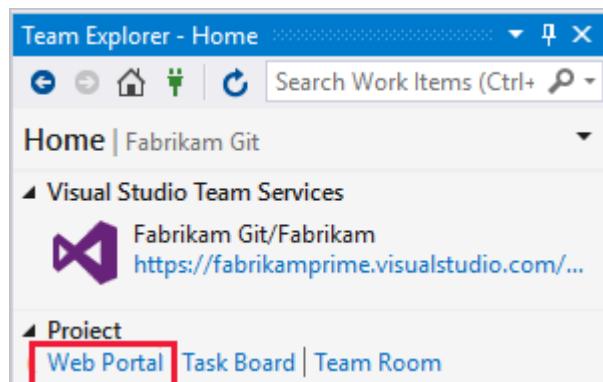
1. Open your profile menu and select **Sign in with a different account**.

2. Choose **Sign in** and enter your credentials.

Open the web portal from Team Explorer

Open the web portal from the home page.
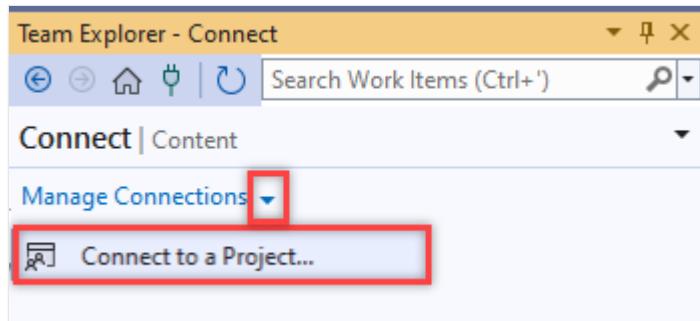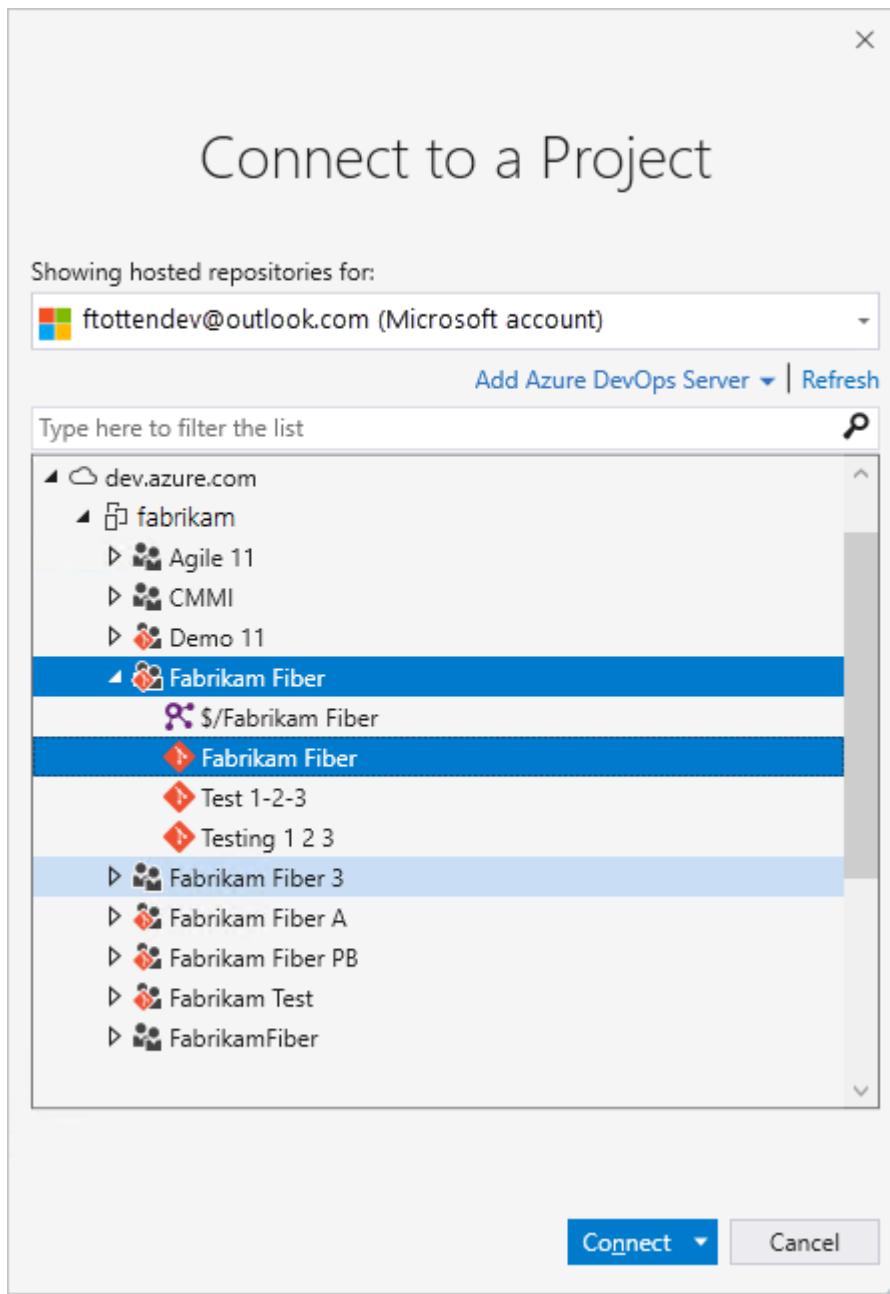

Connect from Visual Studio or Team Explorer

If you haven't already, download and install a version of Visual Studio ⧉.

If you're not a member of an Azure DevOps security group, get added to one. Check with a team member. You need the names of the server, project collection, and project to connect to.

1. Select the **Manage Connections** icon in Team Explorer, and then **Connect to a Project**.

All the projects that you can connect to are displayed, along with the repos in those projects.

2. Select **Add Azure DevOps Server** to connect to a project in Azure DevOps Server. Enter the URL to your server and select **Add**.
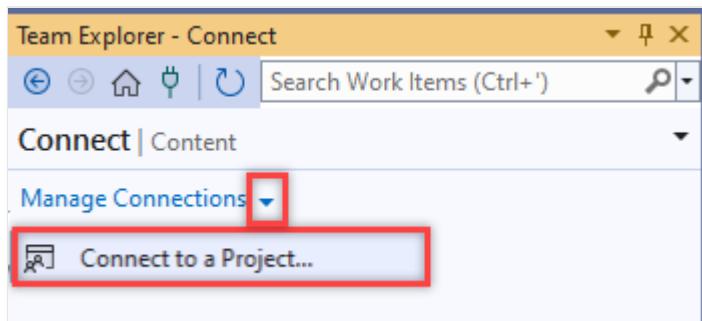
3. Select a project from the list and then select **Connect**.

Change sign-in credentials
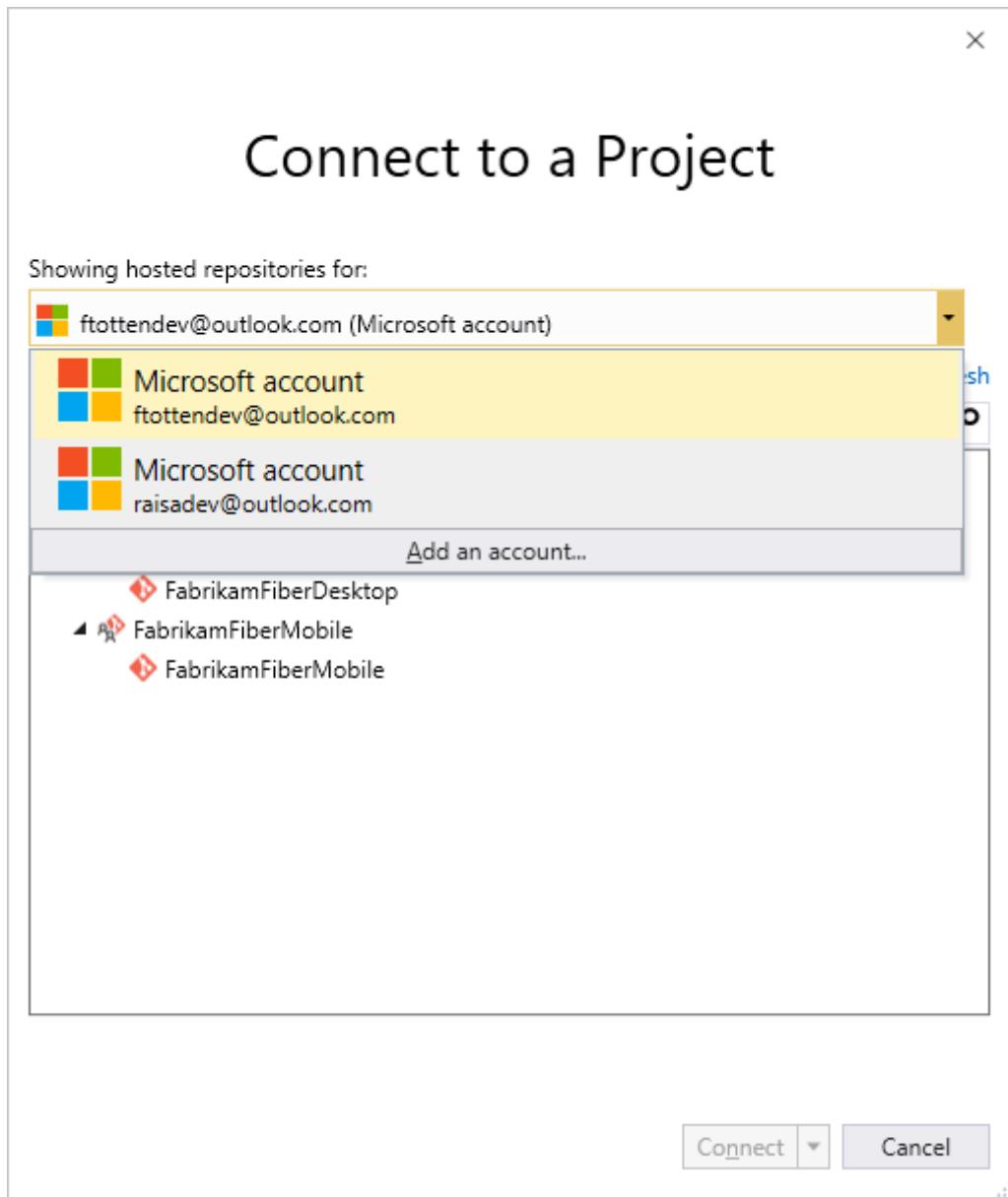
Visual Studio 2022

Visual Studio 2022

1. Select the **Manage Connections** icon in Team Explorer, and then **Connect to a Project**.

2. Select a different user or select **Add an account** to access a project using different credentials.
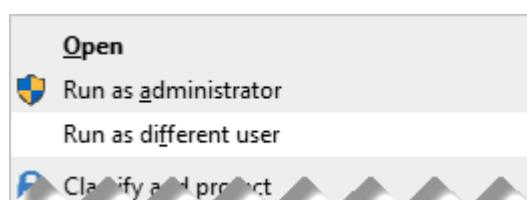
3. Sign in with a Microsoft or GitHub account associated with an Azure DevOps project.

Use different Visual Studio credentials

You can run Visual Studio with credentials different from your current Windows user account. Find *devenv.exe* under the *Program Files (86)* folder for your version of Visual Studio.

Select Shift and right-click *devenv.exe*, then select **Run as different user**.

User accounts and licensing for Visual Studio

To connect to a project, you need your user account added to the project. The
Organization owner for Azure DevOps or a member of the **Project Administrators**
group usually adds user accounts. For more information, see Add organization users and
manage access or Add or remove users or groups, manage security groups.
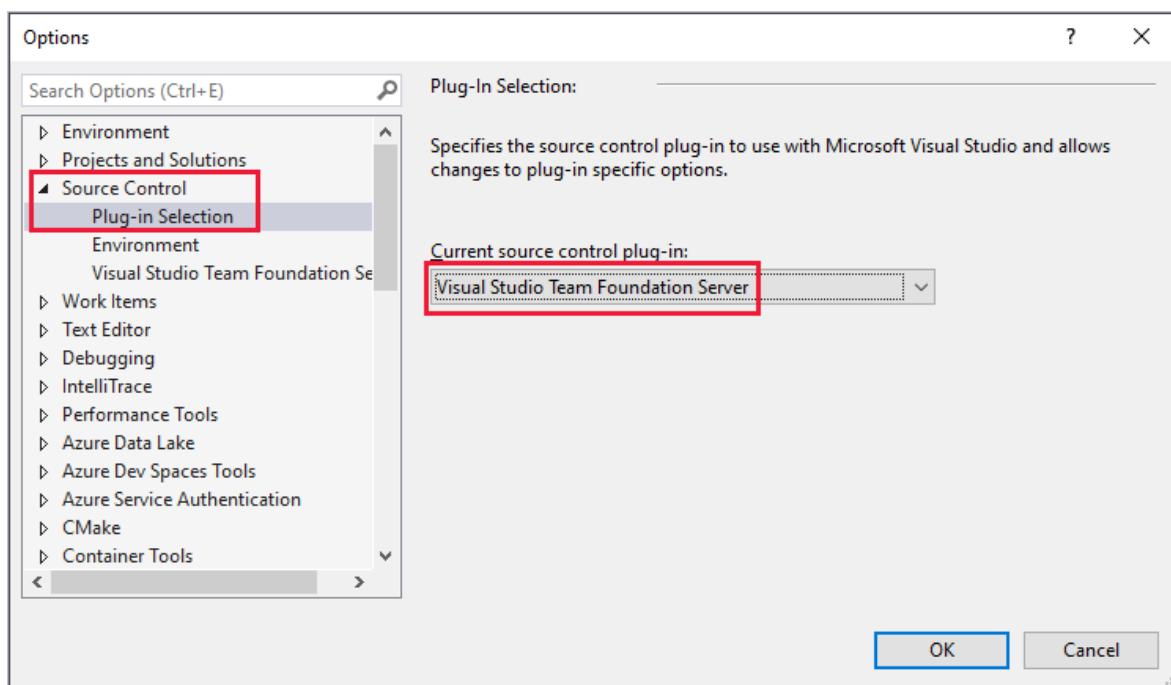
Azure DevOps Services provides access to the first five account users free. After that, you
need to pay for more users.

You can also provide access to Stakeholders in your organization with limited access to
select features as described in Work as a Stakeholder.
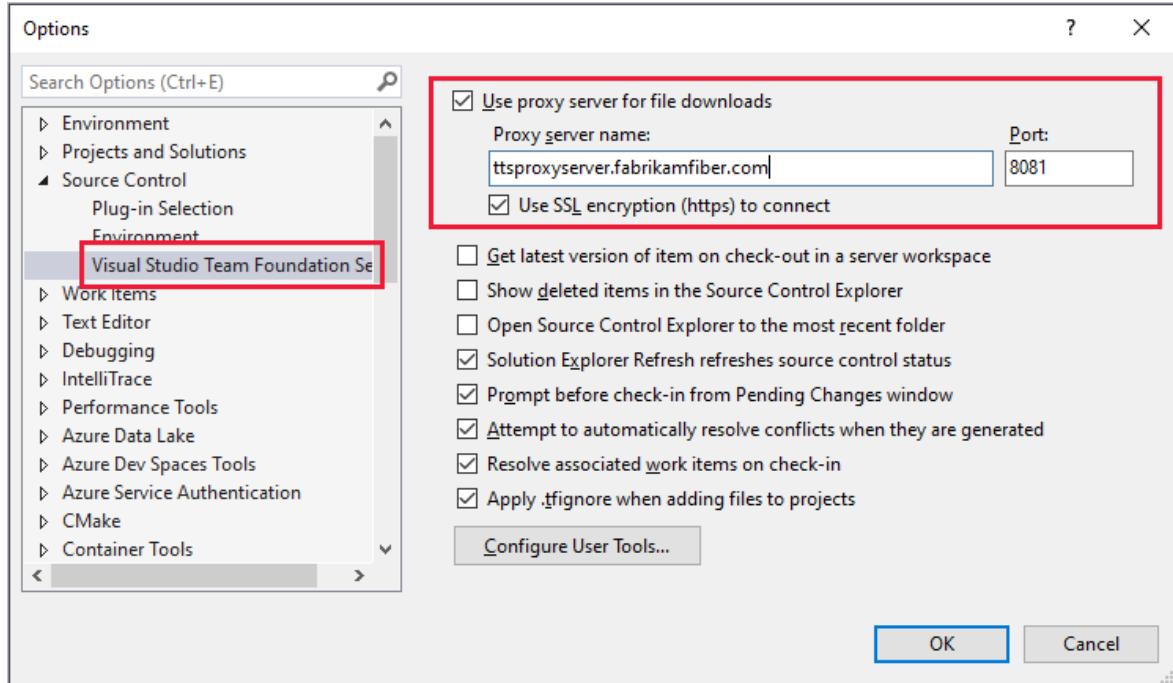
Configure Visual Studio to connect to Azure
DevOps Proxy Server

If your remote team uses an Azure DevOps Proxy Server to cache files, you can
configure Visual Studio to connect through that proxy server and download files under
Team Foundation version control.

1. Make sure you're connected to Azure DevOps, as described in the previous
   section.

2. From the Visual Studio **Tools** menu, select **Options**, and then select **Source Control**
   > **Plug-in Selection**. Select **Visual Studio Team Foundation Server**.

3. For **Visual Studio Team Foundation Server**, enter the name and port number for the Azure DevOps Proxy Server. Select **Use SSL encryption (https) to connect**.

Make sure you specify the port number that your administrator assigned to Azure DevOps Proxy.

To associate a file type with a compare or merge tool, see Associate a file type with a file-comparison tool or Associate a file type with a merge tool.

Requirements and client compatibility

Some tasks or features aren't available when you connect to a later version of Azure DevOps than your client supports. For more information, see client compatibility.

Determine your platform version

See Look up your Azure DevOps platform and version.

Next steps

Get started with Agile tools to plan and track work ·

Related articles

- Work in web portal
- Work in Team Explorer

- Work in Office Excel or Project
- Troubleshoot connection

Share your code with Git

Article • 10/20/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Share your code with others in Azure DevOps when you use a Git repository.

Prerequisites

You must have an organization and project in Azure DevOps. When you create a project, Azure DevOps automatically creates an empty repository in Repos.

1. Install Git command-line tools

Install one of the following Git command-line tools:

- Git for Windows and Git Credential Manager.
- To install on macOS or Linux, check out the Installing Git ⧉ chapter in the open-source *Pro Git* book. For macOS and Linux, we recommend that you configure SSH authentication.

2. Clone the repo to your computer

To work with a Git repo, clone it to your computer, which creates a complete local copy of the repo. Your code might be in one of several places.

1. Complete the following step that's applicable to your scenario:

   - If **You don't have any code yet**, first Create a new Git repo in your project, and then complete the next step.
   - If **the code is in another Git repo**, such as a GitHub repo or a different Azure Repo instance, import it into a new or existing empty Git repo, and then complete the next step.
   - If **the code is on your local computer and not yet in version control**, either create a new Git repo in your project or add your code to an existing repository.

2. From your web browser, open the team project for your organization and select **Repos** > **Files**.

3. Select **Clone** in the upper-right corner of the **Code** window and copy the URL.

4. Open the Git command window (Git Bash on Git for Windows). Go to the folder where you want the code from the repo stored on your computer, and run `git clone`, followed by the path copied from **Clone URL** in the previous step. See the following example:

```
git clone
https://FabrikamFiber01@dev.azure.com/FabrikamFiber01/FabrikamFiber01-
01/_git/FabrikamFiber01-01
```

Git downloads a copy of the code, including all commits, and branches from the repo, into a new folder for you to work with.

5. Switch your directory to the repository that you cloned.

```
cd fabrikam-web
```

Keep this command window open to work in a branch.

3. Work in a branch

Git branches isolate your changes from other work being done in the project. We recommend using the Git workflow, which uses a new branch for every feature or fix that you work on. For our examples, we use the branch, `users/jamal/feature1`.

1. Create a branch with the `branch` command.

```
git branch users/jamal/feature1
```

This command creates a reference in Git for the new branch. It also creates a pointer back to the parent commit so Git can keep a history of changes as you add commits to the branch.

If you're working with a previously cloned repository, ensure that you've checked out the right branch (`git checkout main`) and that it's up to date (`git pull origin main`) before you create your new branch.

2. Use `checkout` to switch to that branch.

```
git checkout users/jamal/feature1
```

Git changes the files on your computer to match the latest commit on the checked-out branch.

4. Work with the code

In the following steps, we make a change to the files on your computer, commit the changes locally, and then push the commit to the repo stored on the server.

1. Browse to the folder on your computer where you cloned the repo, open the `README.md` file in your editor of choice, and make some changes. Then, **Save** and

close the file.

2. In the Git command window, go to the `contoso-demo` directory by entering the following command:

```
cd contoso-demo
```

3. Commit your changes by entering the following commands in the Git command window:

```
git add .
git commit -m "My first commit"
```

The `git add .` command stages any new or changed files, and `git commit -m` creates a commit with the specified commit message.

Check which branch you're working on before you commit, so that you don't commit changes to the wrong branch. Git always adds new commits to the current local branch.

4. Push your changes to the Git repo on the server. Enter the following command into the Git command window:

```
git push origin users/jamal/feature1
```

Your code is now shared to the remote repository, in a branch named `users/jamal/feature1`. To merge the code from your working branch into the `main` branch, use a pull request.

5. Merge your changes with a pull request

Pull requests combine the review and merge of your code into a single collaborative process. After you're done fixing a bug or new feature in a branch, create a new pull request. Add the members of the team to the pull request so they can review and vote on your changes. Use pull requests to review works in progress and get early feedback

on changes. There's no commitment to merge the changes because you can abandon the pull request at any time.

The following example shows the basic steps of creating and completing a pull request.

1. Open the team project for your organization in your web browser and select **Repos** > **Files**. If you kept your browser open after getting the clone URL, you can just switch back to it.

2. Select **Create a pull request** in the upper-right corner of the **Files** window. If you don't see a message like **You updated users/jamal/feature1 just now**, refresh your browser.

New pull requests are configured to merge your branch into the default branch, which in this example is `main`. The title and description are prepopulated with your commit message.

You can add reviewers and link work items to your pull request.

You can review the files included in the pull request at the bottom of the **New Pull Request** window.
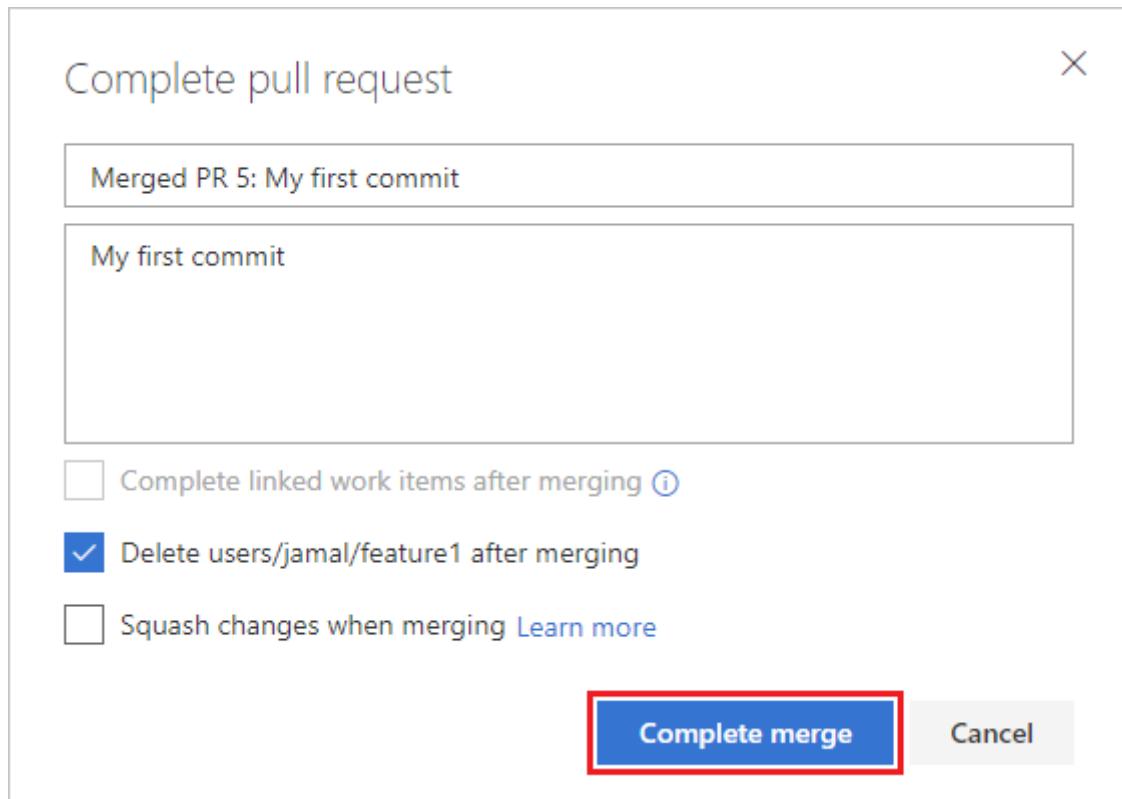
3. Select **Create**.

   View the details of your pull request from the **Overview** tab. You can also view the changed files, updates, and commits in your pull request from the other tabs.

4. Select **Complete** to begin the process of completing the pull request.

5. Select **Complete merge** to complete the pull request and merge your code into the `main` branch.

Your changes are now merged into the `main` branch, and your `users/jamal/feature1`
branch is deleted on the remote repository.

View history

1. Switch back to the web portal and select **History** from the **Code** page to view your
   new commit.

2. Switch to the **Files** tab, and select the README file to view your changes.

Clean up

Switch back to your Git Bash command prompt and run the following command to delete your local copy of the branch.

```
git checkout main
git pull origin main
git branch -d users/jamal/feature1
```

This action completes the following tasks:

- The `git checkout main` command switches you to the `main` branch.
- The `git pull origin main` command pulls down the latest version of the code in the main branch, including your changes and the fact that `users/jamal/feature1` was merged.
- The `git branch -d users/jamal/feature1` command deletes your local copy of that branch.

Next steps

Set up continuous integration & delivery

Related articles

- Key concepts for new users to Azure Pipelines
- What is Azure Repos?
- Learn more about working with a Git repo
- What is source control?

Create your first pipeline

Article • 08/18/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

This is a step-by-step guide to using Azure Pipelines to build a sample application from a Git repository. This guide uses YAML pipelines configured with the YAML pipeline editor. If you'd like to use Classic pipelines instead, see Define your Classic pipeline. For guidance on using TFVC, see Build TFVC repositories.

Prerequisites - Azure DevOps

Make sure you have the following items:

- A GitHub account where you can create a repository. Create one for free ↗.

- An Azure DevOps organization. Create one for free. If your team already has one, then make sure you're an administrator of the Azure DevOps project that you want to use.

- An ability to run pipelines on Microsoft-hosted agents. To use Microsoft-hosted agents, your Azure DevOps organization must have access to Microsoft-hosted parallel jobs. You can either purchase a parallel job or you can request a free grant.

Create your first pipeline

Java

Get the Java sample code

To get started, fork the following repository into your GitHub account.

```
https://github.com/MicrosoftDocs/pipelines-java
```

Create your first Java pipeline

1. Sign-in to your Azure DevOps organization and go to your project.

2. Go to **Pipelines**, and then select **New pipeline**.

3. Do the steps of the wizard by first selecting **GitHub** as the location of your source code.

4. You might be redirected to GitHub to sign in. If so, enter your GitHub credentials.

5. When you see the list of repositories, select your repository.

6. You might be redirected to GitHub to install the Azure Pipelines app. If so, select **Approve & install**.

7. Azure Pipelines will analyze your repository and recommend the **Maven** pipeline template.

8. When your new pipeline appears, take a look at the YAML to see what it does. When you're ready, select **Save and run**.

9. You're prompted to commit a new `azure-pipelines.yml` file to your repository. After you're happy with the message, select **Save and run** again.

   If you want to watch your pipeline in action, select the build job.

   You just created and ran a pipeline that we automatically created for you, because your code appeared to be a good match for the Maven⧉ template.

   You now have a working YAML pipeline (`azure-pipelines.yml`) in your repository that's ready for you to customize!

10. When you're ready to make changes to your pipeline, select it in the **Pipelines** page, and then **Edit** the `azure-pipelines.yml` file.

Learn more about working with Java in your pipeline.

View and manage your pipelines

You can view and manage your pipelines by choosing **Pipelines** from the left-hand menu to go to the pipelines landing page.

From the pipelines landing page you can view pipelines and pipeline runs, create and import pipelines, manage security, and drill down into pipeline and run details.

Choose **Recent** to view recently run pipelines (the default view), or choose **All** to view all pipelines.

Select a pipeline to manage that pipeline and view the runs. Select the build number for the last run to view the results of that build, select the branch name to view the branch for that run, or select the context menu to run the pipeline and perform other management actions.

Select **Runs** to view all pipeline runs. You can optionally filter the displayed runs.

Select a pipeline run to view information about that run.

You can choose to **Retain** or **Delete** a run from the context menu. For more information on run retention, see Build and release retention policies.

View pipeline details

The details page for a pipeline allows you to view and manage that pipeline.

Choose **Edit** to edit your pipeline. For more information, see YAML pipeline editor. You can also edit your pipeline by modifying the **azure-pipelines.yml** file directly in the repository that hosts the pipeline.

View pipeline run details

From the pipeline run summary you can view the status of your run, both while it is running and when it is complete.

From the summary pane you can view job and stage details, download artifacts, and navigate to linked commits, test results, and work items.

Jobs and stages

The jobs pane displays an overview of the status of your stages and jobs. This pane may have multiple tabs depending on whether your pipeline has stages and jobs, or just jobs. In this example, the pipeline has two stages named **Build** and **Deploy**. You can drill down into the pipeline steps by choosing the job from either the **Stages** or **Jobs** pane.

Choose a job to see the steps for that job.

From the steps view, you can review the status and details of each step. From the **More actions** ⋮ you can toggle timestamps or view a raw log of all steps in the pipeline.
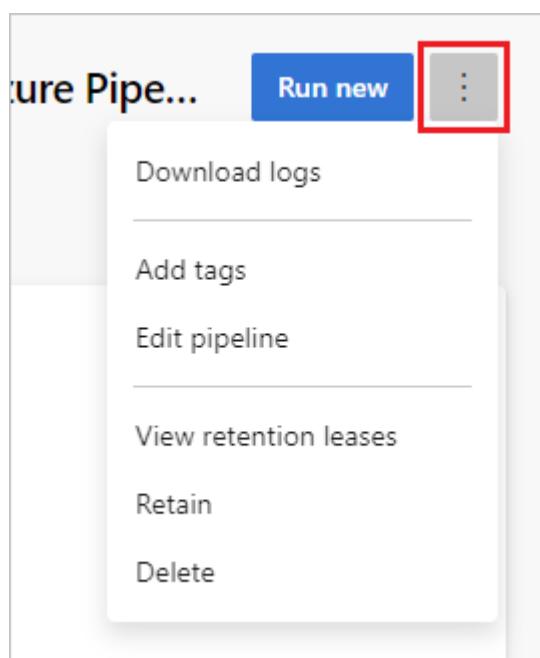
Cancel and re-run a pipeline

If the pipeline is running, you can cancel it by choosing **Cancel**. If the run has completed, you can re-run the pipeline by choosing **Run new**.

Pipeline run more actions menu

From the **More actions** ⋮ menu you can download logs, add tags, edit the pipeline, delete the run, and configure retention for the run.

> ⓘ **Note**
>
> You can't delete a run if the run is retained. If you don't see **Delete**, choose **Stop retaining run**, and then delete the run. If you see both **Delete** and **View retention releases**, one or more configured retention policies still apply to your run. Choose

Add a status badge to your repository

Many developers like to show that they're keeping their code quality high by displaying a status badge in their repo.

To copy the status badge to your clipboard:

1. In Azure Pipelines, go to the **Pipelines** page to view the list of pipelines. Select the pipeline you created in the previous section.

2. Select , and then select **Status badge**.

3. Select **Status badge**.

4. Copy the sample Markdown from the Sample markdown section.

Now with the badge Markdown in your clipboard, take the following steps in GitHub:

1. Go to the list of files and select `Readme.md`. Select the pencil icon to edit.

2. Paste the status badge Markdown at the beginning of the file.

3. Commit the change to the `main` branch.

4. Notice that the status badge appears in the description of your repository.

To configure anonymous access to badges for private projects:

1. Navigate to **Project Settings** in the bottom left corner of the page

2. Open the **Settings** tab under **Pipelines**

3. Toggle the **Disable anonymous access to badges** slider under **General**

> ⓘ **Note**
>
> Even in a private project, anonymous badge access is enabled by default. With anonymous badge access enabled, users outside your organization might be able

Because you just changed the `Readme.md` file in this repository, Azure Pipelines automatically builds your code, according to the configuration in the `azure-pipelines.yml` file at the root of your repository. Back in Azure Pipelines, observe that a new run appears. Each time you make an edit, Azure Pipelines starts a new run.

Next steps

You've just learned how to create your first pipeline in Azure. Learn more about configuring pipelines in the language of your choice:

- .NET Core
- Go
- Java
- Node.js
- Python
- Containers

Or, you can proceed to customize the pipeline you just created.

To run your pipeline in a container, see Container jobs.

For details about building GitHub repositories, see Build GitHub repositories.

To learn how to publish your Pipeline Artifacts, see Publish Pipeline Artifacts.
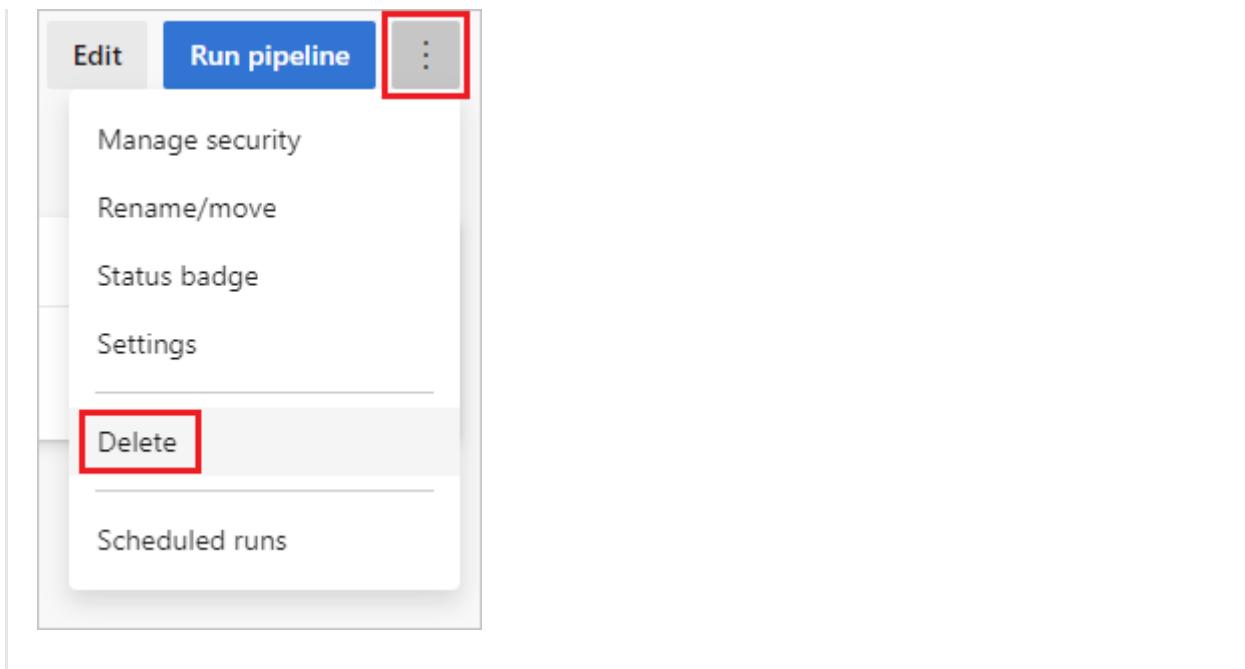
To find out what else you can do in YAML pipelines, see YAML schema reference.

Clean up

If you created any test pipelines, they are easy to delete when you are done with them.

Browser

To delete a pipeline, navigate to the summary page for that pipeline, and choose **Delete** from the **...** menu at the top-right of the page. Type the name of the pipeline to confirm, and choose **Delete**.

FAQ

Where can I read articles about DevOps and CI/CD?

[What is Continuous Integration?](#)

[What is Continuous Delivery?](#)

[What is DevOps?](#) ⧉

What version control system can I use?

When you're ready to get going with CI/CD for your app, you can use the version control system of your choice:

- Clients
  - [Visual Studio Code for Windows, macOS, and Linux](#) ⧉
  - [Visual Studio with Git for Windows](#) or [Visual Studio for Mac](#) ⧉
  - [Eclipse](#)
  - [Xcode](#)
  - [IntelliJ](#)
  - [Command line](#)

- Services
  - [Azure Pipelines](#) ⧉
  - Git service providers such as Azure Repos Git, GitHub, and Bitbucket Cloud
  - Subversion

How can I delete a pipeline?

To delete a pipeline, navigate to the summary page for that pipeline, and choose **Delete** from the **...** menu in the top-right of the page. Type the name of the pipeline to confirm, and choose **Delete**.

What else can I do when I queue a build?

You can queue builds automatically or manually.

When you manually queue a build, you can, for a single run of the build:

- Specify the pool into which the build goes.

- Add and modify some variables.

- Add demands.

- In a Git repository

  - Build a branch or a tag ⧉.

  - Build a commit.

Where can I learn more about pipeline settings?

To learn more about pipeline settings, see:

- Getting sources
- Tasks
- Variables
- Triggers
- Retention
- History

How do I programmatically create a build pipeline?

REST API Reference: Create a build pipeline

> ⓘ **Note**
>
> You can also manage builds and build pipelines from the command line or scripts using the **Azure Pipelines CLI**.

Plan and track work in Azure Boards
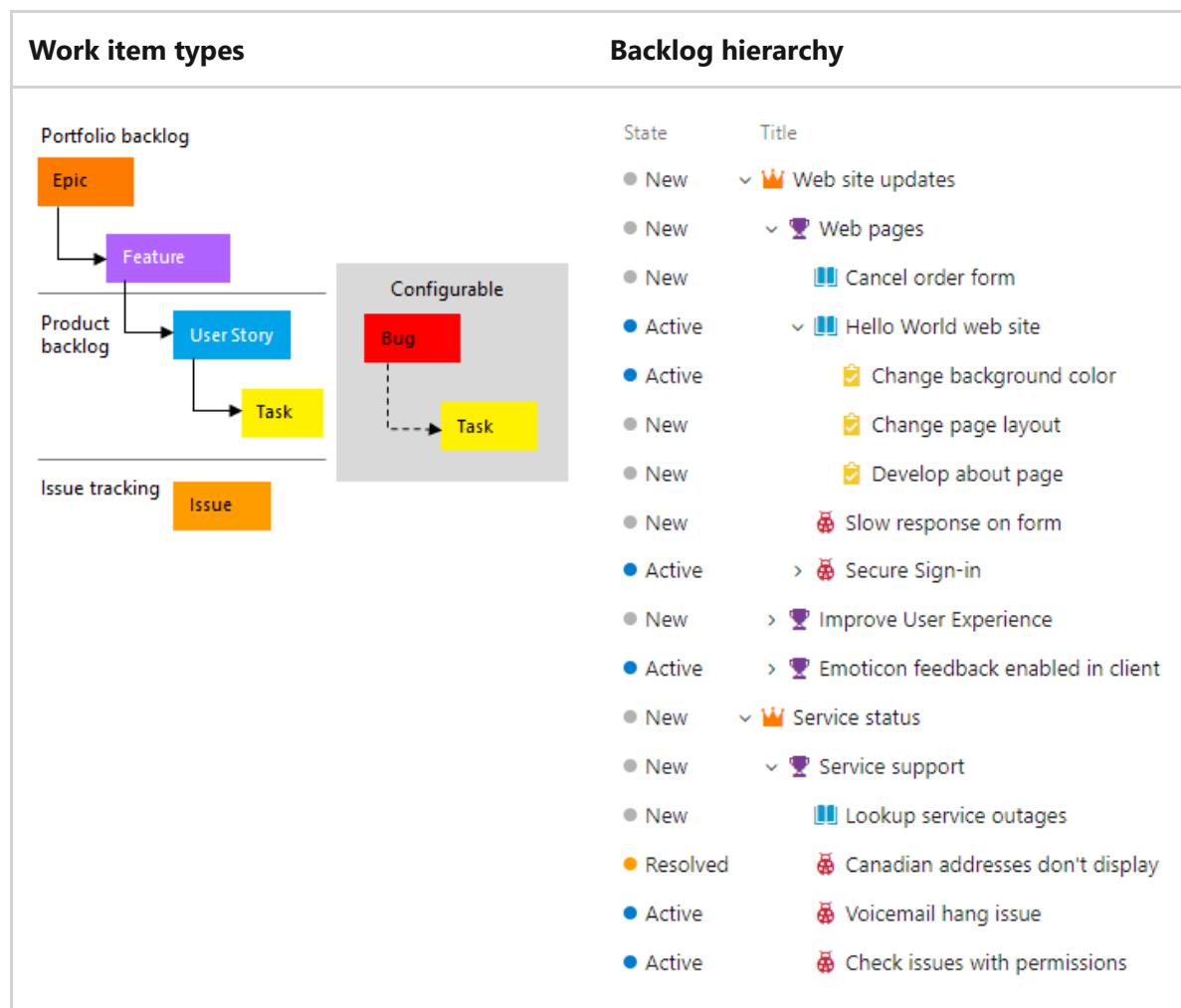
Article • 03/22/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

In this article, learn how to use Azure Boards to plan and track your work using an **Agile**, **Basic**, **Scrum**, or **Capability Maturity Model Integration (CMMI)** process. For more information, see About processes and process templates.

Agile process

The Agile process uses various work item types such as user stories, tasks, bugs, features, and epics to plan and track work. Begin by adding user stories and grouping them into features if needed. You can add tasks to a user story to track more details.

Work item types	Backlog hierarchy

Within each work item form, you can describe the work to be done, assign work to project contributors, track status, and collaborate with others through the Discussion section.

We show you how to add user stories and child tasks from the web portal and add details to those work items.

Prerequisites
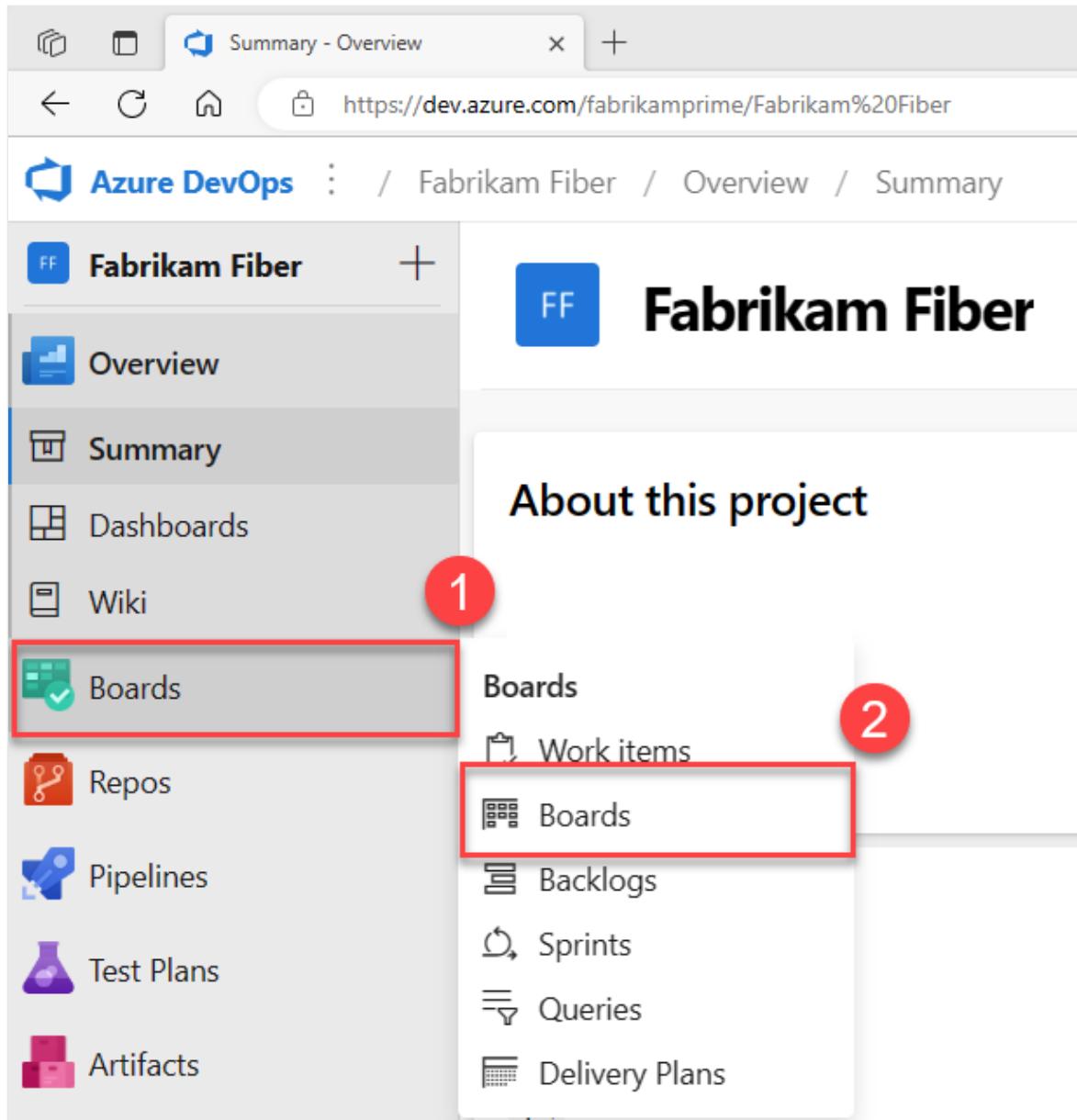
- You must have **Basic** access and be a member of the Contributors or Project Administrators group to add work items to a board and use all other board features.
- You must have **Stakeholder** access for a *private* project and be a member of the Contributors or Project Administrators group to view boards, open and modify work items, and add child tasks to a checklist. But, you can't reorder or reparent a backlog item using drag-and-drop, nor update a field on a card.
- You must have **Stakeholder** access for a *public* project and be a member of the Contributors or Project Administrators group to have full access to all Boards features.

For more information, see Default permissions and access for Azure Boards.
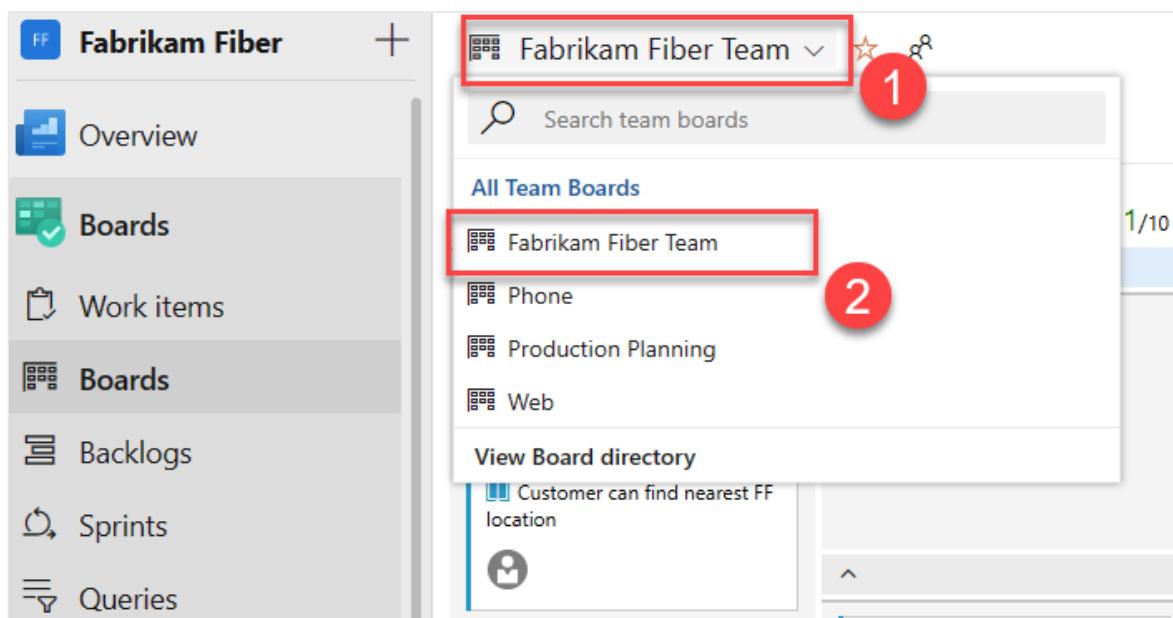
Open your Kanban board

A Kanban board is provisioned with the addition of each project and each team. You can only create or add Kanban boards to a project by adding another team. For more information, see About teams and Agile tools.

1. Sign in to your organization (`https://dev.azure.com/{your_organization}`) and go to your project.

2. Select **Boards** > **Boards**.

3. Select a board from the **All Team Boards** dropdown menu.
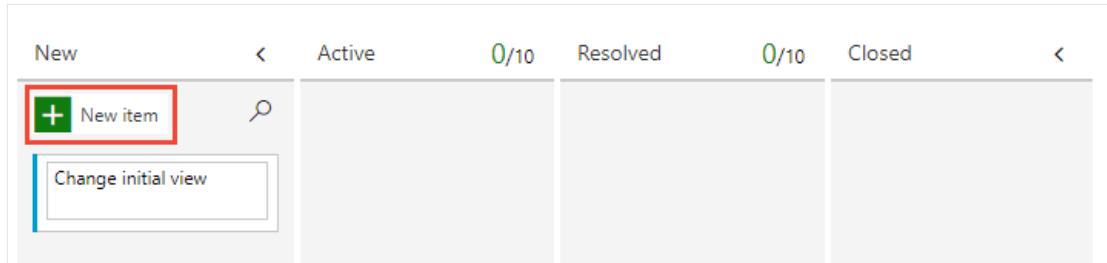
Add work items to your board
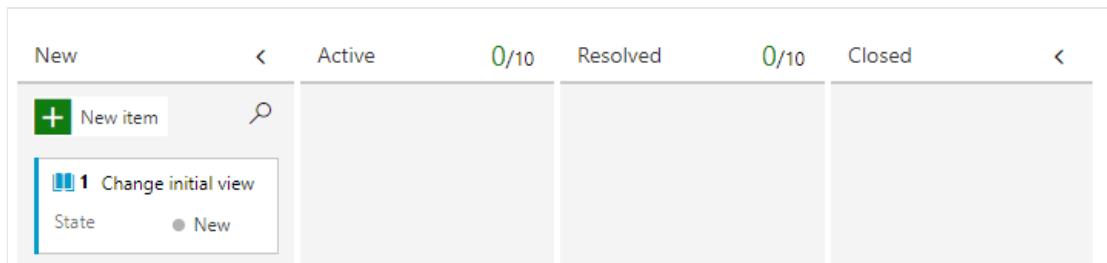
Work items on your board are automatically assigned the default **Area Path** and **Iteration Path** assigned to the team. For more information, see Configure team settings.

Agile process

1. From the Stories board, choose **New item** and the stories you want to track.
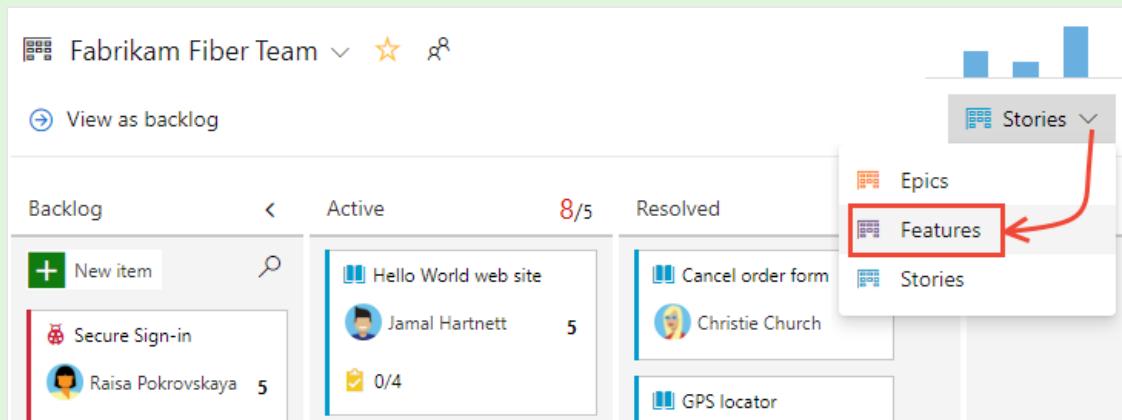
2. Enter return and the system assigns a work item ID to the user story.

3. Add as many user stories as you need.

> 💡 **Tip**
>
> To quickly add features and child user stories, choose **Features** from the board selector.
>
>
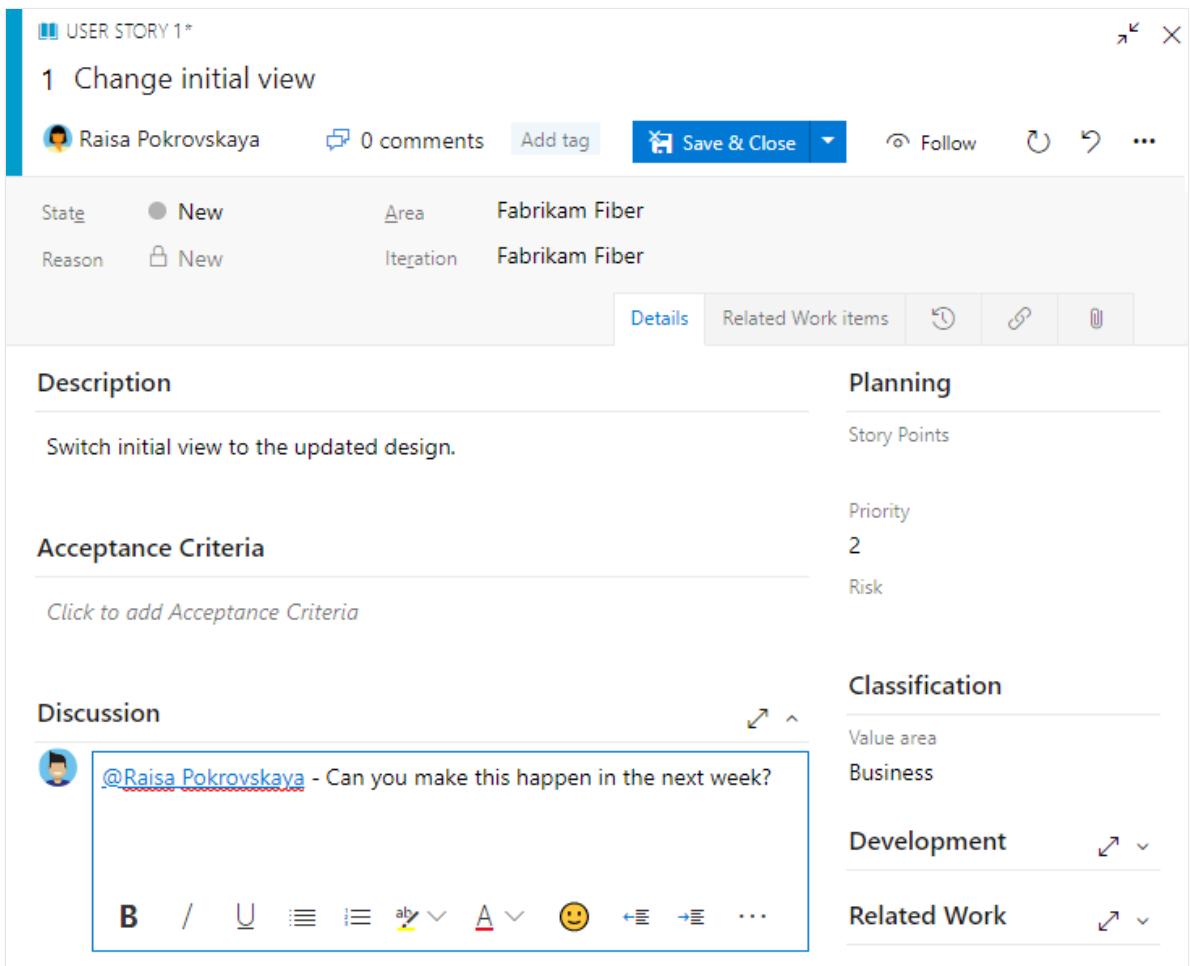
Add details to a board item

Select the issue or user story title to open it. Change one or more field values, add a description, or make a note in the **Discussion** section. You can also choose the 📎 **Attachments** tab and drag-and-drop a file to share the file with others.

Agile process

For example, here we assign the story to Raisa Pokrovskaya and add a discussion note, at-mentioning Raisa.

Choose **Save & Close** when you're done.

Field descriptions

Field

Usage

Title

Enter a description of 255 characters or less. You can always modify the title later.

Assigned To

Assign the work item to the team member responsible for performing the work. Depending on the context you are working in, the drop-down menu lists only team members or contributors to the project.

> ⓘ **Note**
>
> You can only assign work to a single user. If you need to assign work to more than one user, add a work item for each user and distinguish the work to be done by title and description. The Assigned To field only accepts user accounts that have been **added to a project or team**.

State

When the work item is created, the State defaults to the first state in the workflow. As work progresses, update it to reflect the current status.

Reason

Use the default first. Update it when you change state as need. Each State is associated with a default reason.

Area (Path)

Choose the area path associated with the product or team, or leave blank until assigned during a planning meeting. To change the dropdown list of areas, see Define area paths and assign to a team.

Iteration (Path)

Choose the sprint or iteration in which the work is to be completed, or leave it blank and assign it later during a planning meeting. To change the drop-down list of iterations, see Define iteration paths and configure team iterations.

Description

Provide enough detail to create shared understanding of scope and support estimation

efforts. Focus on the user, what they want to accomplish, and why. Don't describe how to develop the product. Do provide sufficient details so that your team can write tasks and test cases to implement the item.

Acceptance Criteria

Provide the criteria to be met before the work item can be closed. Define what "Done" means by describing the criteria for the team to use to verify whether the backlog item or bug fix is fully implemented. Before work begins, describe the criteria for customer acceptance as clearly as possible. Have conversations between the team and customers to determine the acceptance criteria. These criteria help ensure a common understanding within the team to meet customers' expectations. Also, this information provides the basis for acceptance testing.

Priority

A subjective rating of the issue or task it relates to the business. You can specify the following values:
- **1**: Product cannot ship without the successful resolution of the work item, and it should be addressed as soon as possible.
- **2**: Product cannot ship without the successful resolution of the work item, but it does not need to be addressed immediately.
- **3**: Resolution of the work item is optional based on resources, time, and risk.
- **4**: Resolution of the work item is not required.

Value Area

A subjective rating of the issue or task it relates to the business. You can specify the following values:
- **Architectural**: Technical services to implement business features that deliver solution .
- **Business**: Services that fulfill customers or stakeholder needs that directly deliver customer value to support the business (Default).

Effort, Story Points, Size

Provide a relative estimate of the amount of work required to complete an issue. Most Agile methods recommend that you set estimates for backlog items based on relative size of work. Such methods include powers of 2 (1, 2, 4, 8) and the Fibonacci sequence (1, 2, 3, 5, 8, etc.). Use any numeric unit of measurement your team prefers.
The estimates you set are used to calculate team velocity and forecast sprints.

Update work status

The State field tracks the status of a work item. With the Kanban board, you can quickly update the status of backlog items by dragging and dropping them to a different column.

Agile process

As work begins, drag the user story card from the **Backlog** column to the **Active** column. Once work is ready for review, move it to the **Resolved** column. After it's reviewed and accepted, move it to the **Closed** column.

> 💡 **Tip**
>
> To add or rename columns as needed, see **Customize your board**.

Add tasks

Task checklists provide a quick and easy way to track elements of work that are important to support completing a backlog item. Also, you can assign individual tasks to different team members.
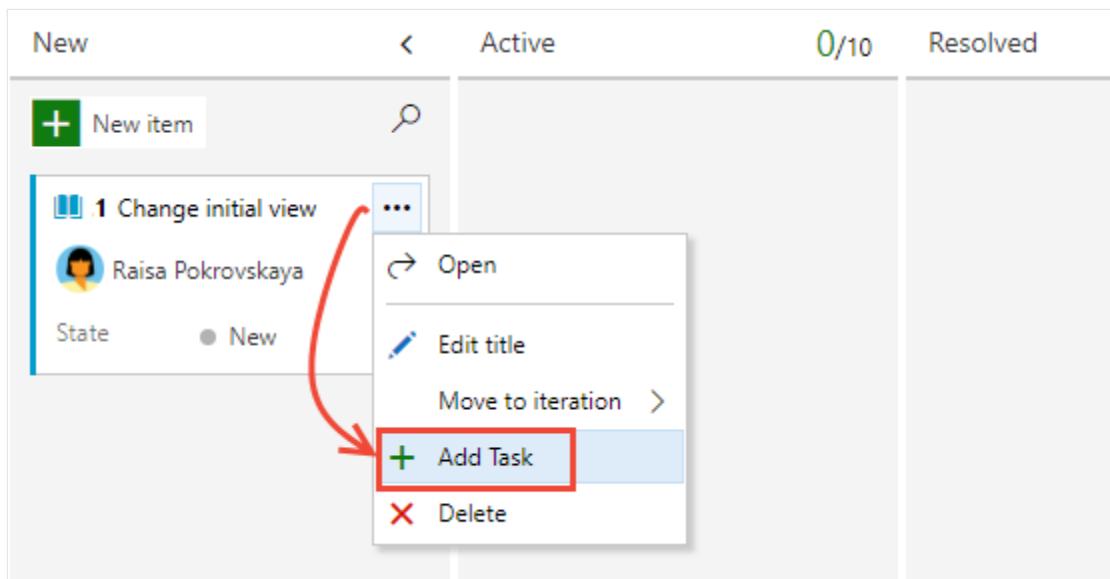
> 💡 **Tip**
>
> Tasks that you create from the Kanban board are automatically assigned the **Area Path** and **Iteration Path** of their parent work item and show up on your sprint taskboard.
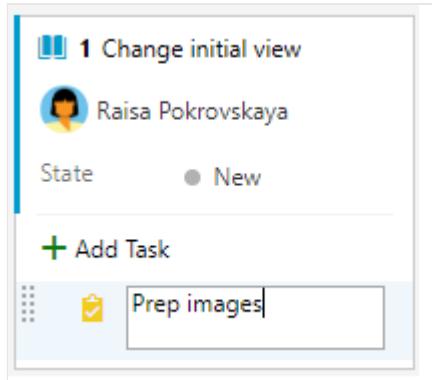
Tasks that you create from the sprint backlog or taskboard show up within tasks checklists on the Kanban board.
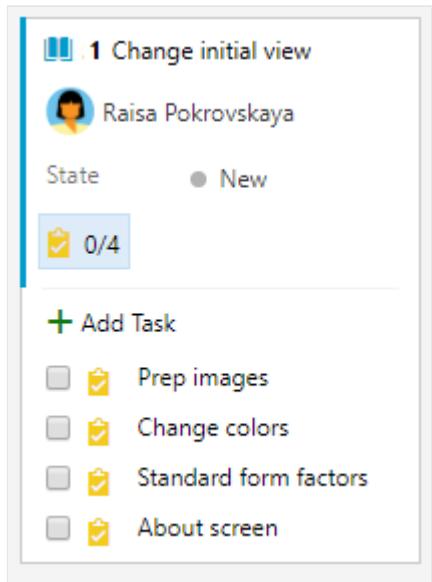
1. Select the ⋯ actions icon for the story and select **+** **Add Task**.
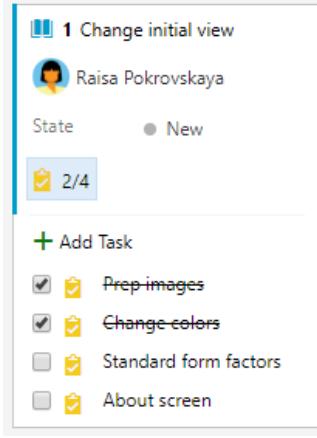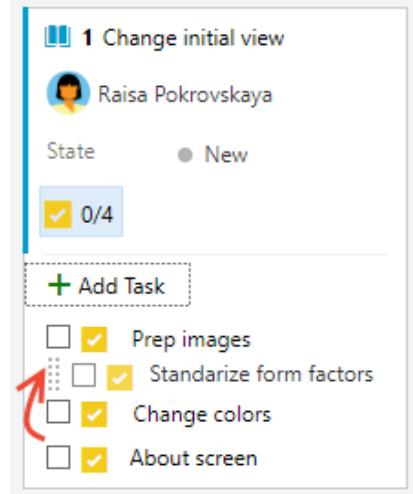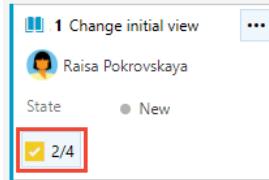
Enter a title for the task and select **Enter** when you're done.

2. If you have many tasks to add, keep typing your task titles and type Enter.

3. You can mark a task as done, expand or collapse the task checklist, or reorder and reparent tasks.

Mark a task as done	Reorder and reparent tasks	Expand or collapse the checklist
To mark a task as complete, check the task checkbox. The task State changes to **Done**.	To reorder a task, drag it within the checklist. To reparent a the task, drag it to another issue on the board.	To expand or collapse a task checklist, simply choose the task annotation.

Add details to a task

If you have details you want to add about a task, choose the title, to open it. Change one or more field values, add a description, or make a note in the **Discussion** section. Choose **Save & Close** when you're done.

Agile process

Here we assign the task to Christie Church.

Field descriptions

In addition to the fields you can define for a backlog item—user story, issue, product backlog item, or requirement—you can specify the following fields for a task to support capacity and time tracking.

> ⓘ **Note**
>
> There are no inherent time units associated with this field even though the taskboard always shows "h" for hours in relationship to Remaining Work. You can specify work in any unit of measurement your team chooses.

Field

Usage

Activity

The type of activity that's required to do a task.For more information about how this field is used, see Capacity planning. Allowed values are:

- **Deployment**
- **Design**
- **Development**
- **Documentation**
- **Requirements**
- **Testing**

Discipline (CMMI process)

The type of activity that's required to do a task.For more information about how this field is used, see Capacity planning. Allowed values are:

- **Analysis**
- **Development**
- **Test**
- **User Education**
- **User Experience**

Original Estimate

The amount of estimated work required to complete a task. Typically, this field doesn't change after it's assigned.

Remaining Work

The amount of work that remains to finish a task. You can specify work in hours or in days. As work progresses, update this field. It's used to calculate capacity charts and the sprint burndown chart.
If you divide a task into subtasks, specify Remaining Work for the subtasks only.

Completed Work

The amount of work spent implementing a task. Enter a value for this field when you complete the task.

Task Type (CMMI only)

Select the kind of task to implement from the allowed values:

- **Corrective Action**
- **Mitigation Action**
- **Planned**

Capture comments in the Discussion section

Use the **Discussion** section to add and review comments made about the work being performed.

The rich text editor tool bar displays below the text entry area. It appears when you select each text box that supports text formatting.

> ⓘ **Note**
>
> There isn't a **Discussion** work item field. To query work items with comments entered in the Discussion area, you filter on the **History field**. The full content of the text entered into the Discussion text box is added to the History field.

Mention someone, a group, work item, or pull request

To open a menu of recent entries you've made to mention someone, link to a work item, or link to a pull request, select # or ⚙ , or enter @, #, or ! .

Enter a name or number and the menu list filters to match your entry. Choose the entry you want to add. To bring a group into the discussion, enter @ and the group name, such as a team or security group.

Edit or delete a comment

To edit or delete any of your discussion comments, choose 🖉 **Edit** or choose the ⋯ actions icon, and then choose **Delete**.

After updating the comment, choose **Update**. To delete the comment, confirm that you want to delete it.

A full audit trail of all edited and deleted comments is maintained in the **History** tab on the work item form.

Add a reaction to a comment

Add one or more reactions to a comment by choosing a smiley icon at the upper-right corner of any comment. Or, choose from the icons at the bottom of a comment next to any existing reactions. To remove your reaction, choose the reaction on the bottom of your comment. The following image shows an example of the experience of adding a reaction and the display of reactions on a comment.

Save a comment without saving the work item

If you only have permissions to add to the **Discussion** of a work item, then you can do so by saving comments. This permission is controlled by Area Path nodes and the **Edit work item comments in this node** permission. For more information, see Set work tracking permissions, Create child nodes, modify work items under an area or iteration path.

Once you save the comments, you don't need to save the work item.

> ⓘ **Note**
>
> When you save changes made to the **Discussion** control, only the comment is saved. No **work item rules** defined for the work item type execute.

Next steps

Customize your board

Related articles

- Azure Boards FAQs
- Add tags to issues or tasks

Add, run, update inline tests

Article • 10/04/2022

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Learn how to add, run, update, and expand and collapse inline tests in Azure DevOps.

To start manual testing, add the test to the user story or bug that you want to test. From the Kanban board, you can define inline tests or a set of manual tests for a backlog item. You also can run these tests and update their status. If you're new to working with the Kanban board, see the Kanban quickstart.

Tests you create from the Kanban board are automatically linked to the user story or backlog item.

Open your Kanban board

1. From your web browser, open the project for your organization and select **Azure Boards**. If you don't have a project, create one now. If you haven't been added as a team member, get invited now.

   The URL follows this pattern: `https://dev.azure.com/fabrikamfiber/_boards/board`

   If you don't see the team or project you want, select ⬡ **Azure DevOps** to browse all projects and teams.

2. Select **Boards** to open the Kanban board.

Add tests

1. To add tests, open the menu for a work item.
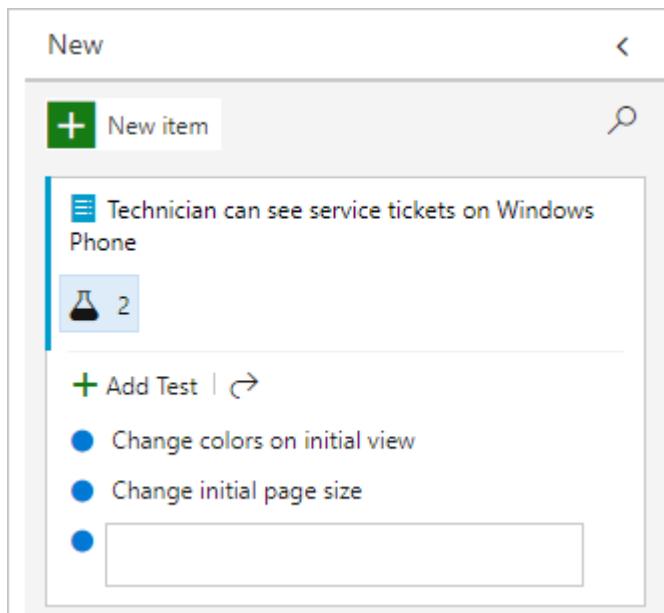
Inline tests are the same as test cases in a test suite. A default test plan and test suite automatically get created under which the manual test cases are grouped.

For example, a test suite is created for the following user story, and inline tests are added to that suite. User story 314 is highlighted. It has two manual tests defined with the IDs 337 and 341.

2. If you have a number of tests to add, enter each title and select **Enter**.

To add details to the test case, open it. You can select the title, double-select the inline item, or open the context menu and choose **Open**.
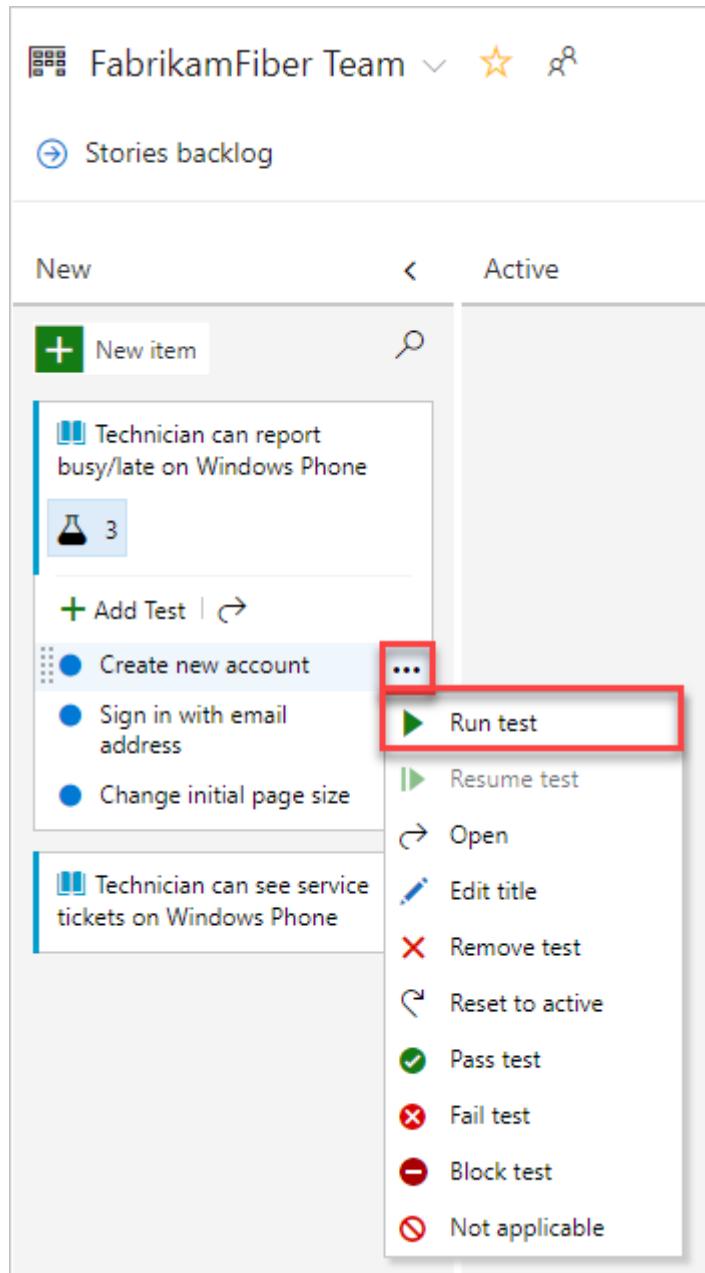
To learn more about how to define tests, see Create manual tests.

Before you run the test, you must add details.

Run a test

Run the test by selecting ▶ **Run test** from the ⋯ actions menu for the inline test.

Microsoft Test Runner starts in a new browser instance. For information on how to run a test, see Run manual tests.

Update the status of a test

You can update the status of the test from the ⋯ actions menu.

When you update the status of tests, you can track test results.

Expand or collapse inline tests

When you first open the Kanban board, you'll see an unexpanded view of checklists and tests.

Select the inline test summary to expand a collapsed set of tests. Select the same summary to collapse an expanded list.

Next steps

Kanban quickstart

Related articles

- Learn more about test case management
- Exploratory test your web app directly in your browser
- Essential services
- Client-server tools
- Software development roles

Tutorial: Follow changes made to a user story, bug, or other work item or pull request

Article • 06/27/2023
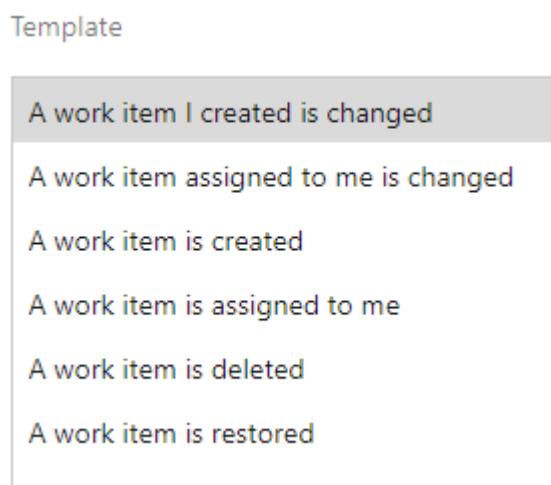
Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

To get notified of changes made to a specific work item or a pull request, you can choose to follow them. The Follow feature provides an improvised way of getting notified on a case-by-case basis.

If you want to subscribe to receive notifications automatically based on changes that occur based on your targeted set of criteria, see Manage personal notifications. For example, you can create a subscription to automatically get notified whenever a work item that you created or that was assigned to you is modified.

> ⓘ **Note**
>
> Notification subscriptions allow you to personalize the notifications you receive automatically based on additional criteria you specify for **yourself**, a team, or a project. For example, you can create a subscription and add field criteria to receive changes based on one or more of the following templates.
>
> | Template |
> | --- |
> | A work item I created is changed |
> | A work item assigned to me is changed |
> | A work item is created |
> | A work item is assigned to me |
> | A work item is deleted |
> | A work item is restored |

This article shows you how to:

- ✔ Follow a work item
- ✔ Follow a pull request
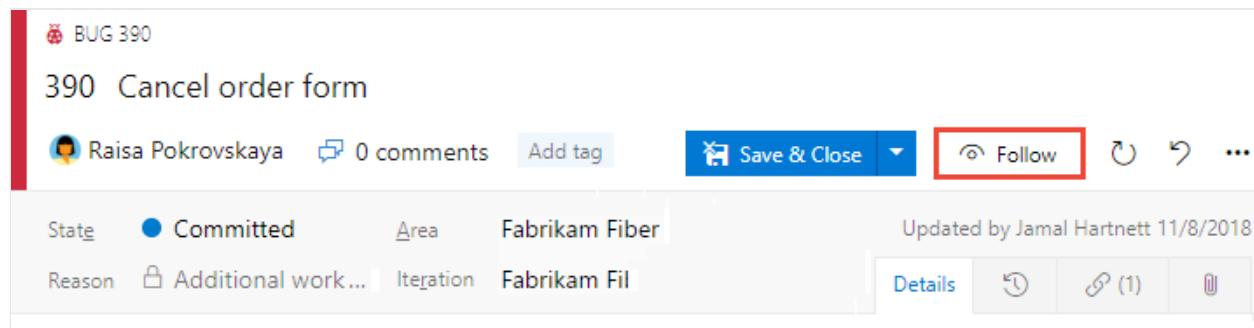
✔ Manage work items that you're following

Prerequisites

- Connect to a project. If you don't have a project yet, create one.
- You must be added to a project as a member of the **Contributors** or **Project Administrators** security group. To get added, Add users to a project or team.
- To view or follow work items, you must be granted **Stakeholder** access or higher. For more information, see About access levels. Also, you must have your **View work items in this node** and **Edit work items in this node** permissions set to **Allow**. By default, the **Contributors** group has this permission set. For more information, see Set permissions and access for work tracking.
- To view or follow pull requests, you must have **Basic** access or higher.
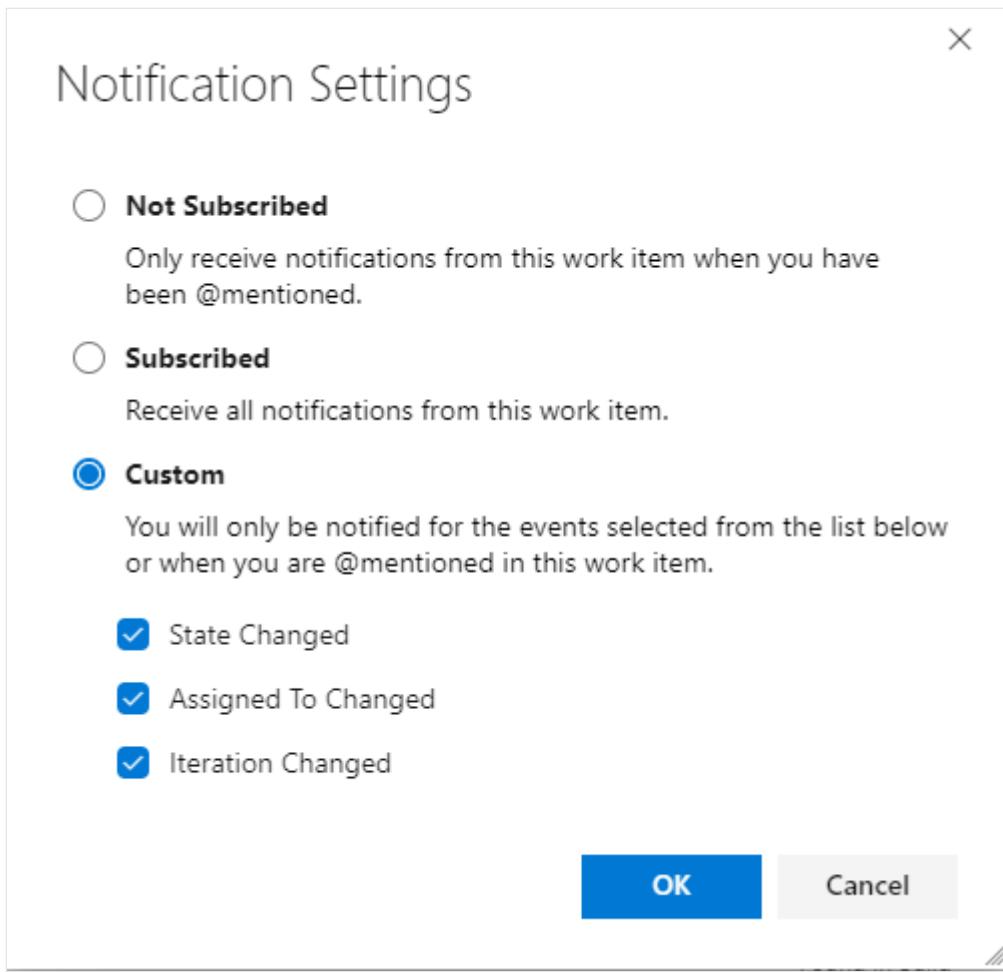
Follow a work item

When you want to track the progress of a single work item, choose the 👁 Follow follow icon. This signals the system to notify you when changes are made to the work item.

If you want to specify conditions on when you'll get notified of changes, choose the ⚙ gear icon and choose from the options provided.

By default, you're **Subscribed** to receive a notification when any change is made to the work item. Choose **Not Subscribed** to receive notification only when you're @mentioned. Or choose **Custom** to receive notifications when one of the checked fields changes, **State**, **Assigned To**, or **Iteration Path**.
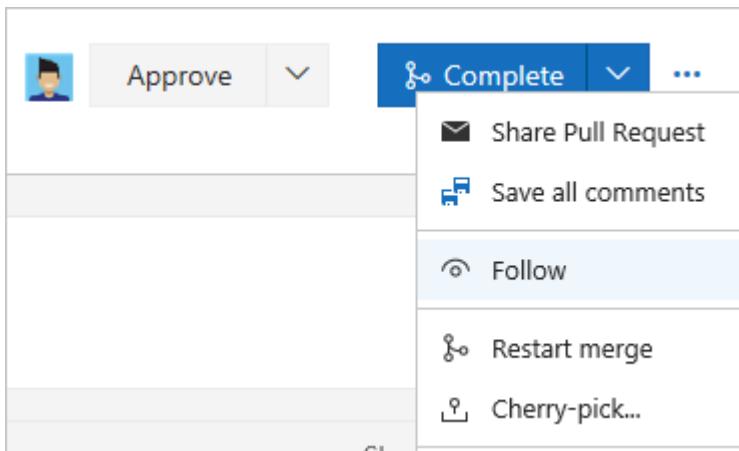
You'll only receive notifications when other members of your team modify the work item, such as adding to the discussion, changing a field value, or adding an attachment.

Notifications are sent to your preferred email address, which you can change from your user profile

To stop following changes, choose the ⊙ Following following icon.

Follow a pull request

To track the progress of a single pull request, choose the ⋯ actions icon for the pull request, and select the ⊙ Follow **Follow** option. This signals the system to notify you when changes are made to the PR.

You'll only receive notifications when other members of your team modify the PR, such as adding to the discussion or adding an attachment.

Notifications are sent to your preferred email address, which you can change from your user profile.

To stop following changes, open the PR context menu and choose the ⊙ Following Following icon.

Manage work items that you're following

You can review and manage all the work items you've selected to follow.

Open **Boards**>**Queries**, choose **All**, and under **My Queries**, choose **Followed work items**.

From this view, you can view all items you're following across all projects. Also, you can complete similar actions supported with a query results view, such as:

- Refresh the view
- Add or remove visible columns
- Sort the order of specific columns
- Filter results by text or tags
- Set work item pane
- Enter full screen mode.

You can also view and manage work that you're following from **Boards>Work Items** and pivot to **Following**.

Query work items that you're following

You can use the **@Follows** macro in a work item query to filter a list based on work items you're following along with other query filters.

For example, the following query shows how to query across all projects for active work items that you're following. You use the **ID** field and the **In** operator with the **@Follows** macro.

Next steps

Add, update, and follow a work item

Related articles

- Manage personal notifications
- View and update work items via the mobile work item form

Q: Can I add someone else to follow a work item or PR?

A: No, you can't add another team member to follow a work item or pull request at this time. You can subscribe them to get notified based on select criteria, such as when a work item is create or modified, or a pull request is created. For more information, see Manage team notifications.

Get started as a Stakeholder

Article • 04/25/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Stakeholders are users with free but limited access to Azure DevOps features and functions. With Stakeholder access, you can add and modify work items, manage build and release pipelines, and view dashboards. You can check project status and provide direction, feedback, feature ideas, and business alignment to a team. For information about working with pipelines, see Build your GitHub repository and Build OSS repositories.

For more information, see the Stakeholder access quick reference. To compare Stakeholder versus Basic access, see the feature matrix ☑.

In this tutorial, learn how to:

- ✓ Understand Stakeholder access and available features
- ✓ Sign in to a project
- ✓ Understand work items and types
- ✓ Open your Kanban board
- ✓ Add work items
- ✓ Update work items
- ✓ View the backlog
- ✓ Find work items

Prerequisites

- You must have Stakeholder access for a *private project* and be a member of the Contributors or Project Administrators group. You can view boards, open and modify work items, and add child tasks to a checklist. But, you can't reorder or reparent a backlog item using drag-and-drop, nor update a field on a card.
- You must have Stakeholder access for a *public project* and be a member of the Contributors or Project Administrators group to have full access to all Boards features. For more information, see Default roles and access for public projects

To get access as a Stakeholder, ask your organization owner or Project Collection Administrator to add you to a project with Stakeholder access.

Sign in to a project

1. Select the link provided in your email invitation or open a browser window and enter the URL for the web portal.

   `https://dev.azure.com/OrganizationName/ProjectName`

2. Enter your credentials. If you can't sign in, ask the organization owner or Project Administrator to add you as a member of the project with Stakeholder access.
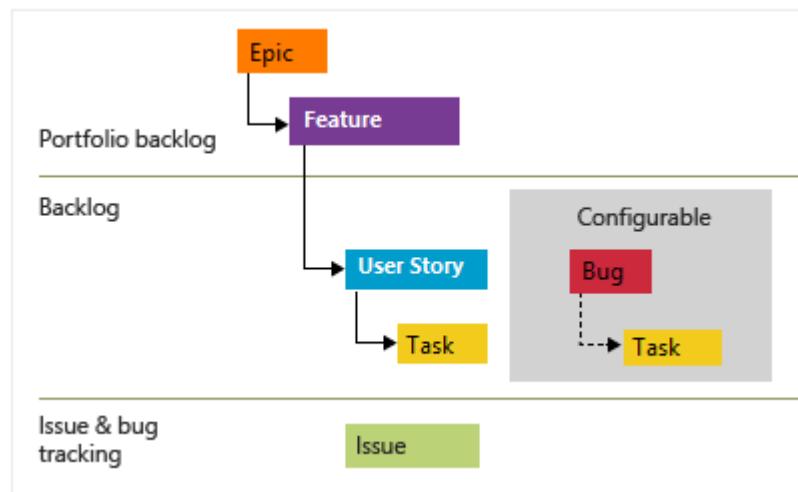
Understand work items and types

Work items support planning and tracking work. Each work item is based on a work item type and is assigned an identifier, which is unique within an organization or project collection.

Different work items are used to track different types of work, as described in About work items. The work item types available are based on the process used when your project was created—Agile, Basic, Scrum, or CMMI—as illustrated in the following images.

Agile process

The following image shows the Agile process backlog work item hierarchy. User Stories and Tasks are used to track work, Bugs track code defects, and Epics and Features are used to group work under larger scenarios.
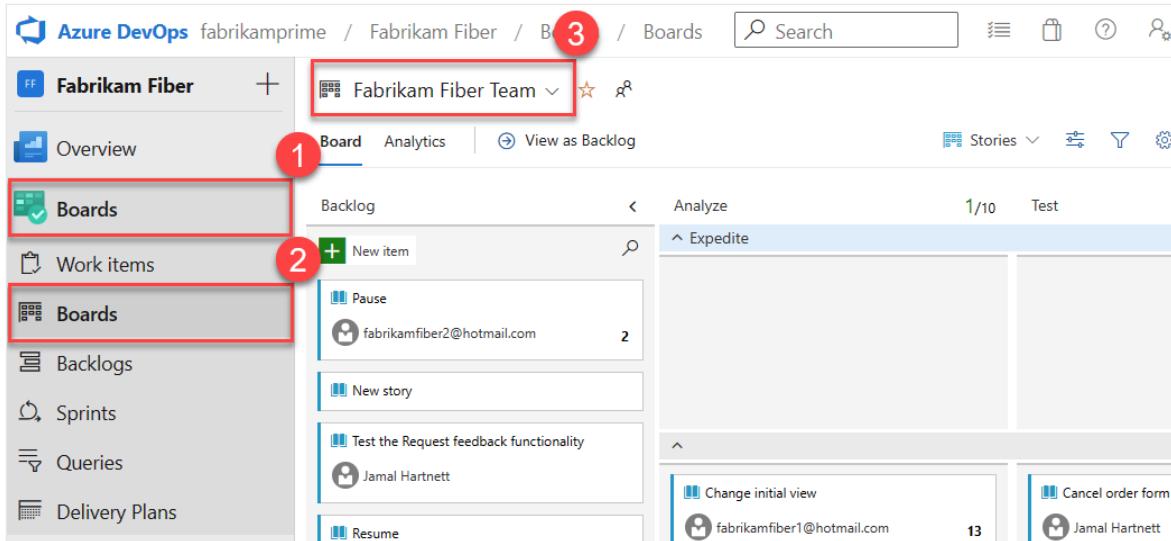
Each team can configure how they manage Bugs—at the same level as User Stories or Tasks—by configuring the Working with bugs setting. For more information about using these work item types, see Agile process.
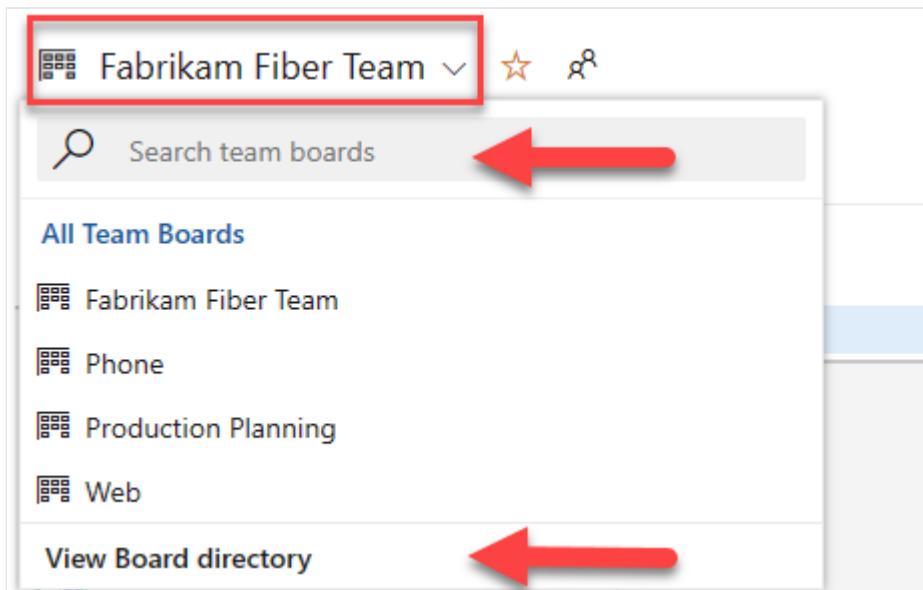
Open your Kanban board

You can view work items once you connect to a project.

1. In your project, and select **Boards** > **Boards**, and then select a **team board** from the dropdown menu.

You can also enter a keyword in the search box or select **View Board directory** to see a list of available team boards.

> 💡 **Tip**
>
> Select the ⭐ star icon to make a team board a favorite. Favorite artifacts (⭐ favorite icon) appear at the top of the team selector list.

2. Check that you selected **Stories** for Agile, **Issues** for Basic, **Backlog items** for Scrum, or **Requirements** for CMMI as the backlog level.

Add work items

From your board, select the plus sign, enter a title, and then select **Enter**.

For more information, see [View and add work items from the Work Items page](#).

Update work items

The work item forms you see may differ from the following images. The basic functionality is the same, however, changes have been made with different versions of Azure DevOps.

Change status

Drag and drop a work item to move it downstream as you complete work.

Add details

To open a work item, double-click the title or highlight it, and then select **Enter**. Here we show how to assign work. You can only assign work to a user who is added to the project.

For example, here we assign the story to Raisa Pokrovskaya and we add a discussion note, at-mentioning Raisa. When you're done, select **Save & Close**.

Add more details by changing field values, adding a description or tags, and adding comments. For more information, see the following articles:

- Update fields: Descriptions and usage
- Add tags to work items. As a Stakeholder, you can add existing tags to a work item, but you can't add new tags.
- Capture comments in the Discussion section

View as Backlog

Check the product backlog to see how the team prioritized their work. Backlog items appear in priority order. Work item types may include bugs depending on the process used when your project was created.

From the Kanban board, choose **View as Backlog**.

You should see the list of backlog items listed in priority order. You can add a backlog item, which goes to the bottom of the list. With Stakeholder access, you can't reprioritize work.

Find work items

Choose **Boards** > **Work Items** > and then select an option from the dropdown menu. Here we chose **Assigned to me**.

For more information, see the following articles:

- View, run, or email a work item query
- View and add work items using the Work Items page

Next steps

Create your product backlog

Related articles

- Add work items
- Kanban quickstart
- Access levels
- Change access levels

View permissions for yourself or others

Article • 02/24/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Learn how to view your permissions or the permissions that are set for others in Azure DevOps. If you don't have a permission to access a feature or function, you can request it from the right resource.

Permissions are set at the collection, project, and object level as described in Get started with permissions, access, and security groups. So to view the permissions you have, you need to open the permissions at the object, project, or collection level.

> ⓘ **Note**
>
> This article shows how to view permissions assigned to a user at the project-level or collection-level. However, the steps are similar when you work from the Security dialog of an object.

Prerequisites

- You must have a project to connect to. If you don't have a project yet, create one.
- You must be a member of the Project Valid Users Group or Project Collection Valid Users Group to view permissions.

View project-level permissions

> ⓘ **Note**
>
> To enable the preview feature, for the new user interface for the **Project Permissions Settings Page**, see Enable preview features.

Preview page

1. Choose **Project Settings** and then **Permissions**.

2. Choose **Users**. To filter the list, enter a name into the *Search groups or users* box.

3. Choose the name you want. The project-level permissions for that user displays. These permissions are based on the groups the user belongs to or the permissions set specifically for the user's account.

4. Choose **Member of** to see which security groups and teams that the user
   belongs to.

   Here we see that *Jamal Hartnett* belongs to several teams and the Project
   Collection Administrators group for several projects.

View organization or collection-level permissions

Open admin settings for the organization or a project collection.

1. Choose the ⬡ Azure DevOps logo to open **Projects**. Then choose **Organization settings**.

2. Choose **Permissions**, the **Project Collection Administrators** group, and then **Members**.

3. Follow steps 2 through 4 in the procedure outlined previously for view project-level permissions.

View object-level permissions

You can define the security or permissions for a number of objects. You access them from the context menu of the object.

From the web portal, open the Security dialog for the object whose permissions you want to set. For specific instructions, see the following articles:

Area

Task

Wiki & Dashboard permissions
- README & Wiki
- Dashboards

Azure Repos, Azure Pipelines/DevOps (code, build, test, release) permissions
- Git branch
- Git repository
- TFVC
- Builds
- Release pipeline security
- Approvals and approvers

Azure Boards/Work tracking permissions
- [Area and iteration paths](#)
- [Work item query and folder](#)
- Plan permissions

Next steps

[Look up a member of the Project Administrators group](#)

Related articles

- [Troubleshoot permissions](#)
- [Permissions and role lookup guide](#)

Sign up for Azure DevOps

Article • 10/24/2023

Azure DevOps Services

Sign up for Azure DevOps and get the [free tier of services ↗](#). For more information, see [What is Azure DevOps?](#)

Sign up

Sign up for Azure DevOps with either a Microsoft account or GitHub account.

Microsoft account

1. If you don't have one, [create a Microsoft account ↗](#).

2. Go to [Azure DevOps ↗](#) and select **Start free**.

3. Enter your account credentials and go through the sign-up process.

    Azure DevOps creates an organization.

- Azure DevOps creates a project named after your *newly created* Microsoft account.
- If you signed up with an existing Microsoft account, you need to [create a project](#) next.

Sign in to your organization at any time
`https://dev.azure.com/{Your_Organization}`.

Next steps

Create a project

Related articles

- [Plan your organizational structure in Azure DevOps](#)
- [Change the location of your organization](#)
- [Add users to your organization](#)

- Add users or groups to a team or project
- GitHub authentication FAQs

Create an organization

Article • 10/16/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Use an organization to connect groups of related projects, and help to scale up your enterprise. You can use a personal Microsoft account, GitHub account, or a work or school account. Use your work or school account to *automatically connect* your organization to your Microsoft Entra ID.

> ⓘ **Note**
>
> All organizations must be manually created via the web portal. We don't support automated creation of organizations. We do support automated organization configuration, project creation, and resource provisioning via **REST API**.

Prerequisites

- Understand how to plan your organizational structure.
- Determine whether you want to use only Microsoft accounts or authenticate users with Microsoft Entra ID. For more information, see Choosing your organization administrator account type.

> ⓘ **Important**
>
> Currently, you can only use letters from the English alphabet in your organization name. Start organization names with a letter or number, followed by letters, numbers, or hyphens. Organization names must not contain more than 50 Unicode characters and must end with a letter or number.
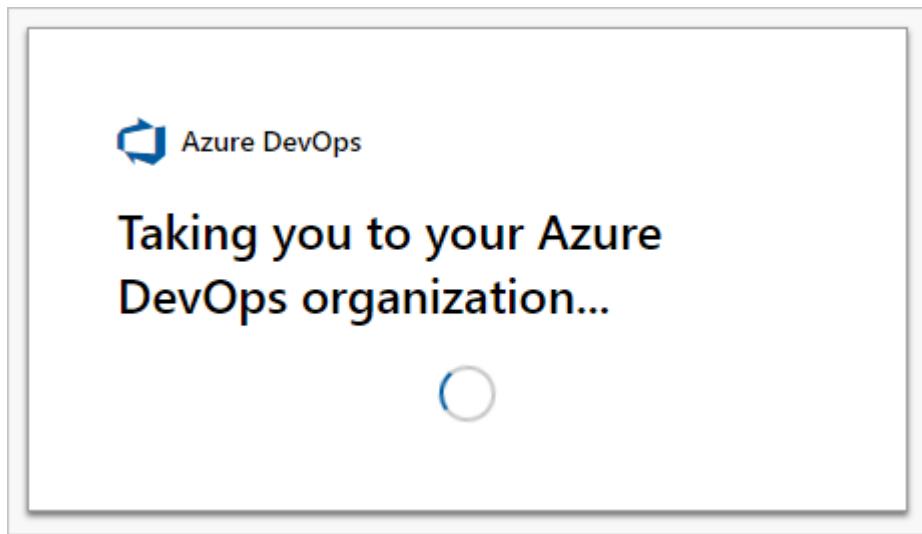
Create an organization

1. Sign in to Azure DevOps ↗.

2. Select **New organization**.

3. Confirm information, and then select **Continue**.

Congratulations, you're an organization owner!

Sign in to your organization at any time, `https://dev.azure.com/{yourorganization}`.

With your organization, the following aspects are included in the free tier:

- First five users free (Basic license)
- **Azure Pipelines:**
  - One [Microsoft-hosted CI/CD](#) (one concurrent job, up to 30 hours per month)
  - One self-hosted CI/CD concurrent job
- **Azure Boards:** Work item tracking and Kanban boards
- **Azure Repos:** Unlimited private Git repos
- **Azure Artifacts:** Two GiB free per organization

Next steps

[Create a project]

Related articles

- [Get started with Azure Repos and Visual Studio](#)
- [Rename your organization](#)
- [Change organization time-zone](#)
- [Change organization owner](#)
- [Delete your organization](#)
- [Resolve orphaned organization](#)

Get started managing your project

Article • 03/23/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

With most Azure DevOps Services, you can start using the service and configure resources as you go. No up-front work is required. Most settings define defaults.

If you created a project or you're added to the **Project Administrators** group, get familiar with the administrative tasks you're charged with. There are a few tasks you might want to do to ensure a smooth operational experience.

> ⓘ **Note**
>
> This article provides an overview of tasks a member of the **Project Administrators** group should review and attend to. For information on tasks to be performed by members of the **Project Collection Administrators** group, see **Manage your organization or project collection**.

Add users to your project

You add users to a team or project so they can contribute to the team and project. Users can be added to multiple teams and projects.

Users that have been added to an organization, can easily be added to a project by adding them to a team or inviting them to contribute to a project.

Team administrators can add users to their team which automatically adds them to the project. By adding users to a team, you make team-specific tools aware of them, such as the team security group, Team Members widget, and sprint capacity planning tools. To learn more about teams, see About teams and Agile tools.

Members of the **Project Administrators** group can add users to a project. Adding users to a team or project automatically adds them to the project's **Contributors** group. Members of this group have permissions to most features needed to contribute to work items, code, builds, and releases. For an overview of default permissions, see Default permissions quick reference.

Once users have been added to a project or organization, you can browse for their display name or user name (email alias) from any people-picker tool. Users can connect

to a project and access features available through a supported client or the web portal.

To learn more, see the following articles:

- [Add users or groups to a team or project](#)
- [Manage your organization or project collection, Add users to your organization](#)
- [Connect to a project](#)

Share your project vision, set up a project wiki

Each project has a summary page that's useful for sharing information through **README** files. Or, redirect users to a project Wiki. For users who are new to your project, we recommend that you [set up your project summary page](#). Or, you can [provision a Wiki](#). Use these features to share established processes and procedures for your project.

Remove unused services

To simplify the web portal user interface, you can disable select services. For example, if you use a project only to log bugs, then disable all services except for **Boards**. To learn more, see [Turn a service on or off](#).

This example shows that **Test Plans** is disabled:

Manage security and permissions

Permissions and security groups control access to select tasks. To quickly understand the defaults configured for your project, see Default permissions and access.

The following table lists the permissions assigned at the project-level. All of these permissions are granted to members of the **Project Administrators** group, except for the **Delete shared Analytics views** and **Edit shared Analytics views** permissions which are not set. For a description of each permission, see Permissions and groups reference, Groups.

General
- Delete team project
- Edit project-level information
- Manage project properties
- Rename team project
- Suppress notifications for work item updates
- Update project visibility
- View project-level information

Boards
- Bypass rules on work item updates
- Change process of team project
- Create tag definition
- Delete and restore work items
- Move work items out of this project
- Permanently delete work items

Analytics
- Delete shared Analytics views
- Edit shared Analytics views
- View analytics

Test Plans
- Create test runs
- Delete test runs
- Manage test configurations
- Manage test environments
- View test runs

For more information about security and setting permissions at the project-level, review the following articles:

- [Get started with permissions, access, and security groups](#)
- [Change permissions at the project-level](#)

Add members to the Project Administrators group

The person who creates a project is automatically added as a member to the **Project Administrators** group. Members of this group have permissions to manage project configuration, repositories, pipeline resources, teams, and all project-level permissions.

It's always a good idea to have more than one person who has administrative privileges. To add a user to this group, see [Change permissions at the project level, Add members to the Project Administrators group](#).

Grant or restrict permissions

Permissions are managed at the following three levels and through role-based assignments.

- object
- project
- organization or collection

As a member of the **Project Administrators** group, you can grant or restrict permissions for all objects and at the project-level. To delegate specific tasks to others, we recommend that you add them to a built-in or custom security group, or add them to a specific role. For more information, see the following articles.

- [Role-based permissions](#)
- [Add or remove users or groups, manage security groups](#)
- [Grant or restrict access to select features and functions](#)
- [Set object-level permissions](#)

Review and update notifications

A number of notifications are predefined for each project you add. Notifications are based on subscription rules. Subscriptions arise from the following areas:

- [Out-of-the-box or default subscriptions.](#)
- [Team, project, and organization or collection subscriptions](#) defined by a team administrator or member of the **Project Administrators** or **Project Collection Administrators** groups.

If users believe they're getting too many notifications, you can direct them to opt out of a subscription.

Determine traceability requirements

If you're using most of Azure DevOps Services—Boards, Repos, Pipelines, and Test Plans — you'll want to alert your teams to those features that support end-to-end traceability. To get started, we recommend that you review the following articles:

- Cross-service integration and collaboration overview
- End-to-end traceability

Set DevOps policies

Set policies to support collaboration across your teams and automatically remove obsolete files. To set policies that govern Azure Repos, Azure Pipelines, and Azure Test Plans, review the following articles:

- Manage branch policies
- Add Team Foundation Version Control (TFVC) check-in policies
- Set build and release pipeline retention policies
- Set test retention policies

Configure and customize Azure Boards

You can configure and customize Azure Boards to support many business requirements for planning and tracking work. At a minimum, you should configure the following

elements:

- Area paths to group work items by team, product, or feature area
- Iteration paths to group work into sprints, milestones, or other event-specific or time-related periods

If you're new to Azure Boards and want an in-depth overview of what you can configure and customize, see Configure and customize Azure Boards.

Define area and iteration paths to track work

If you support several products, you can assign work items by feature area by defining area paths. To assign work items to specific time intervals, also known as sprints, you configure iteration paths. To use the Scrum tools—sprint backlogs, taskboards, and team capacity—you need to configure several sprints. For an overview, see About areas and iteration paths.

Customize work-tracking processes

You and your team can start using all work-tracking tools immediately after you create a project. But often, one or more users want to customize the experience to meet one or more business needs. You can customize the process easily through the user interface. As such, you'll want to establish a methodology for who will manage the updates and evaluate requests.

> ⓘ **Note**
>
> By default, organization owners and users added to the **Project Collection Administrators** security group are granted permission to create, edit, and manage processes used to customize the work-tracking experience. If you want to lock down who is able to perform these tasks, you can set permissions at the organization-level to **Deny**.

To learn more, see these articles:

- About process customization and inherited processes
- Customize a project
- Add and manage processes
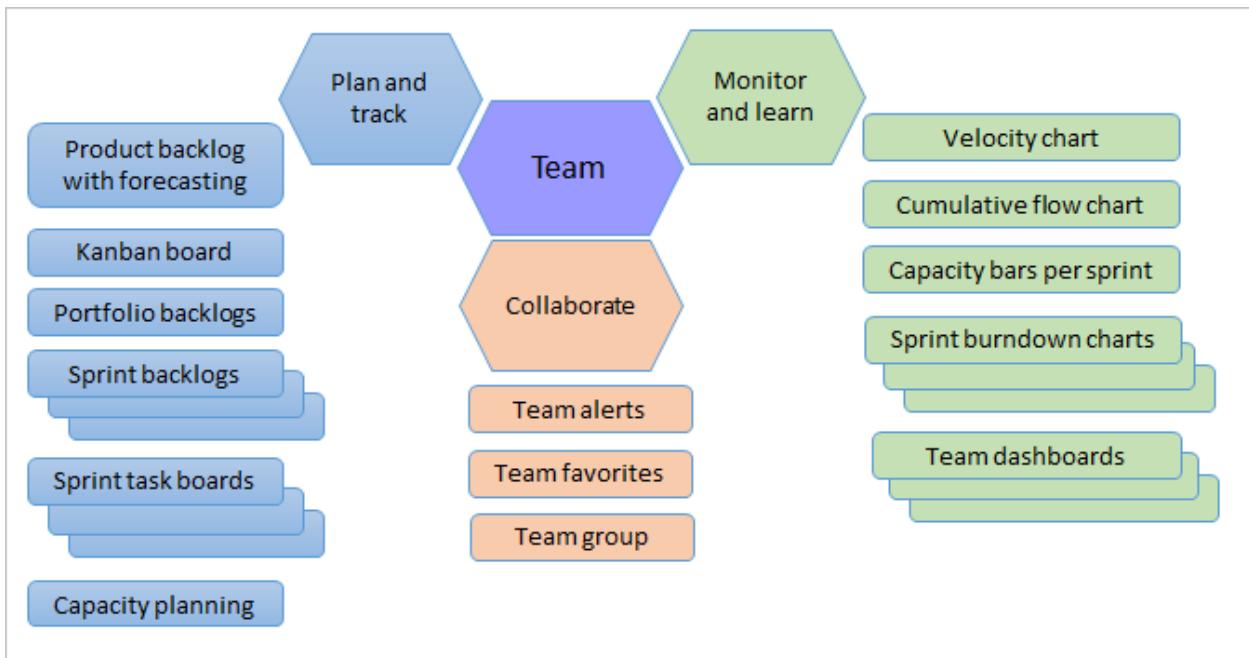
Integrate with other services

Azure DevOps supports integration with Azure, GitHub, and many other services. As a member of the **Project Administrators** group, you can configure integration with many of these services. For more information, see the following articles.

- Azure DevOps and GitHub integration overview

- Azure Boards and GitHub integration

- Microsoft Teams integration:
    - Azure Boards with Microsoft Teams
    - Azure Repos with Microsoft Teams
    - Azure Pipelines with Microsoft Teams

- Slack integration:
    - Azure Boards with Slack
    - Azure Repos with Slack
    - Azure Pipelines with Slack

- Integrate with service hooks

Add teams to scale your project

As your organization grows, we recommend that you add teams to scale your project. Each team gets access to their own set of customizable Agile tools.

To learn more, see the following articles:

- [About projects and scaling your organization](#)
- [Add a team, move from one default team to several teams](#)
- [Add a team administrator](#)

Next steps

Share your project vision

Related articles

- [Project and team quick reference](#)
- [Get started managing your organization or project collection](#)
- [About user, team, project, and organization-level settings](#)

Manage your organization or collection

Article • 10/19/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

After you create an organization or project collection, you'll want to add contributors and configure policies, settings, and other options available to you. This article provides an overview of tasks to ensure you set up your organization or collection to get maximal use of your services.

Each organization is associated with one and only one collection. If you need to create another organization, see Plan your organizational structure and Create an organization.

> ⓘ **Note**
>
> This article provides an overview of tasks that require membership in the **Project Collection Administrators** group. For information on tasks performed by members of a **Project Administrators** group, see **Manage your project**.

Add users to your organization

For large enterprises, connect Azure DevOps to Microsoft Entra ID and use its security groups to control user access. This way, you can sync users and groups between Microsoft Entra ID and Azure DevOps, and reduce the overhead of managing permissions and user access.

You can add users and security groups to your organization through the web portal **Organization settings** > **Users** interface, regardless of the size of your enterprise. You can also assign these users and groups to one or more projects within your organization.

When you add users, you specify their *access level*, which determines the features they can use through the web portal. For more information, review these resources:

- Get started with permissions, access, and security groups
- About access levels
- Add organization users and manage access
- Connect your organization to Microsoft Entra ID

> ⓘ **Note**
>
> If the **Limit user visibility and collaboration to specific projects** preview feature is turned on the organization, users added to the **Project-Scoped Users** group can't access projects that they haven't been added to. For more information including important security-related call-outs, see **Limit user visibility for projects and more**, later in this article.

Set up billing

Azure DevOps charges for the following services as described in Pricing for Azure DevOps ⧉ .

- Individual services:
  - Microsoft-hosted CI/CD parallel jobs
  - Self-hosted CI/CD parallel jobs
  - Storage of Azure Artifacts feeds
- User licenses for **Basic** or **Basic + Test Plans**.

All organizations are granted five free **Basic** licenses and unlimited users with **Stakeholder** access. For information on each access level, see About access levels.

If your organization requires more than five contributors, then you need to set up billing. Users that have a Visual Studio subscription can be added without incurring any further billing charges. Billing is based on the access level, **Basic** or **Basic + Test Plans**, that you assign to the user. For more information, see Set up billing.

Manage security and permissions

Permissions and security groups control access to select tasks.

The following table lists the permissions assigned at the organization or collection-level. All of these permissions, except for the **Make requests on behalf of others** permission, are granted to members of the **Project Collection Administrators** group. For a description of each permission, see Permissions and groups reference, Groups.

General
- Alter trace settings
- Create new projects
- Delete team project

- Edit instance-level information
- View instance-level information

Service Account
- Make requests on behalf of others
- Trigger events
- View system synchronization information

Boards
- Administer process permissions
- Create process
- Delete field from organization or account
- Delete process
- Edit process

Repos (TFVC)
- Administer shelved changes
- Administer workspaces
- Create a workspace

Pipelines
- Administer build resource permissions
- Manage build resources
- Manage pipeline policies
- Use build resources
- View build resources

Test Plans
- Manage test controllers

Auditing
- Delete audit streams
- Manage audit streams
- View audit log

Policies
- Manage enterprise policies

For more information about security and setting permissions at the collection-level, review the following articles:

- Get started with permissions, access, and security groups
- Change permissions at the organization or collection-level.

Add members to the Project Collection Administrators group

When you create an organization, you become a member of the Project Collection Administrators group. This group has the authority to manage the organization's settings, policies, and processes. It can also create and manage all the projects and extensions in the organization.

It's always a good idea to have more than one person who has administrative privileges. To add a user to this group, see Change permissions at the organization level,Add members to the Project Collection Administrators group.

Limit user visibility for projects and more

By default, users added to an organization can view all organization and project information and settings.

> ⓘ **Important**
>
> - The limited visibility features described in this section apply only to interactions through the web portal. With the REST APIs or `azure devops` CLI commands, project members can access the restricted data.
> - Guest users who are members in the limited group with default access in Microsoft Entra ID, can't search for users with the people picker. When the preview feature's turned *off* or when guest users aren't members of the limited group, guest users can search all Microsoft Entra users, as expected.

To restrict select users, such as Stakeholders, Microsoft Entra guest users, or members of a particular security group, you can turn on the **Limit user visibility and collaboration to specific projects** preview feature for the organization. Once it's turned on, any user or group added to the **Project-Scoped Users** group, are restricted in the following ways:

- Restricted users to only access those projects to which they've been explicitly added.
- Restricts views that display list of users, list of projects, billing details, usage data, and more that is accessed through **Organization Settings**.
- Limits the set of people or groups that appear through people-picker search selections and the ability to **@mention** people.

> ⚠ **Warning**

For more information, see Manage preview features.

All security groups are organization-level entities, even those groups that only have permissions to a specific project. From the web portal, visibility of some security groups might be limited based on user permissions. However, you can discover the names of all groups in an organization using the **azure devops** CLI tool or our REST APIs. For more information, see Add and manage security groups.

Limit identity search and selection

With Microsoft Entra ID, you can use people pickers to search for any user or group in your organization, not just the ones in your current project. People pickers support the following Azure DevOps functions:

- Selection of a user identity from a work tracking identity field such as **Assigned To**
- Selection of a user or group using **@mention** in a work item discussion or rich-text field, a pull request discussion, commit comments, or changeset or shelveset comments
- Selection of a user or group using **@mention** from a wiki page

As shown in the following image, you simply start typing into a people picker box until you find a match to a user name or security group.

Users and groups who are added to the **Project-Scoped Users** group can only see and select users and groups in the project they're connected to from a people picker. To scope people pickers for all project members, see Limit user visibility for projects and more earlier in this article.

To limit the identity selection to only users and groups added to a project, perform the following procedure for your organization and projects.

1. Turn on the **Limit user visibility and collaboration to specific projects** preview feature for the organization. For more information, see Manage preview features.
2. Add the users to your project(s) as described in Add users to a project or team. Users added to a team are automatically added to the project and team group.
3. Open **Organizations Settings>Security>Permissions** and choose **Project-Scoped Users**. Choose the **Members** tab. Add all users and groups that you want to scope to the project(s) you've added them to. For more information, see Set permissions at the project- or collection-level. The **Project-Scoped Users** group only appears under the **Permissions>Groups** once **Limit user visibility and collaboration to specific projects** preview feature is turned on.

Set security policies

Configure the security policies for your organization through the **Organization settings>Policies** page. These policies let you grant or restrict the following features:

- Third-party application access via OAuth
- SSH authentication
- Creation of public projects
- Invitation of GitHub user accounts

For more information, see Change application connection & security policies for your organization.

Manage extensions

An extension is an installable unit that adds new capabilities to your projects. Azure DevOps extensions support the following functions:

- Planning and tracking of work items, sprints, scrums, and so on
- Build and release flows
- Code testing and tracking
- Collaboration among team members

For example, to support code search, install the Code Search extension⧉ .

You want to tell your users about extensions and that they can request an extension. To install and manage extensions, you must be an organization Owner, a member of the **Project Collection Administrators** group. Or, you can get added to the Manager role for extensions.

Install Code Search

Code Search is a free Marketplace extension that lets you search across all your source repositories. For more information, see Install and configure Search.

Adjust time zone and other organization settings

When you create an organization, you specify the name of your organization and select the region where your organization is hosted. The default **Time zone** is set to *UTC*. You can update the **Time zone** and specify a Privacy URL from the **Organization settings>Overview** page. For more information about these settings, see the following articles:

- Time zone settings and usage
- Add a privacy policy URL for your organization

Configure DevOps settings

Use the following settings, which get defined at the organization-level, to support your work.

- Add agent pools
- Define pipeline retention settings
- Define repository settings:
  - Default branch name for new repositories
  - Gravatar images.

Customize work-tracking processes

All work-tracking tools are available immediately after you create a project. Often, one or more users might want to customize the experience to meet one or more business

needs. Processes are easily customized through the user interface. However, you might want to establish a methodology for who manages the updates and evaluates requests.

For more information, see the following articles:

- About process customization and inherited processes
- Customize a project
- Add and manage processes

Alert users with information banners

Communicate with your Azure DevOps users quickly through information banners. Use banners to alert your Azure DevOps users to upcoming changes or events without sending mass emails. For more information, see Add and manage information banners.

Review and update notifications

Many notifications are predefined at the organization or collection level. You can manage subscriptions or add new subscriptions.

Scale your organization or collection

To learn about scaling your organization, see the following articles.

- About projects and scaling your organization
- Plan your organizational structure

Related articles

- Project and team quick reference
- FAQs about signing up and getting started
- Organization management
- About user, team, project, and organization-level settings

Add users or groups to a team or project

Article • 04/12/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

You add users to a team or project so they can contribute to the team and project. For enterprise organizations with large user bases, we recommend you use Azure Active Directory to add and manage new users through security groups. However, to enable flexibility for all size organizations, the following operations are supported:

- Team and project administrators can add new users to their team or project, unless the policy Allow team and project administrators to invite new users is disabled. New users are ones that haven't been added to the organization.
- When you add new users through the team and project user interfaces, the system automatically assigns an access level to the user.
- Adding users to a team or project automatically adds them to the Contributors group for the project. Members of the Contributors group have permissions to most features needed to contribute.
- By adding users to a team, you make team-specific tools aware of them, such as the team security group, Team Members widget, and sprint capacity planning tools.
- Once users have been added to a project or organization, you can browse for their display name or user name (email alias) from any people-picker tool.

You add projects to an organization or project collection and you add teams to projects. For more information, see:

- Create a project
- Add team, go from one default team to others

> ⓘ **Important**

To view the content available for your platform, make sure that you select the correct version of this article from the version selector which is located above the table of contents. Feature support differs depending on whether you are working from Azure DevOps Services or an on-premises version of Azure DevOps Server. To learn which on-premises version you are using, see **Look up your Azure DevOps platform and version**

Supported options for adding users

Depending on the interface you use, you can exercise different options for adding new or existing users to teams or projects.

Administrator level

Interface

Supported tasks

Team administrators

Team Members dashboard widget

Add new or existing users to a team. Send new users an invite.

Team administrators

Project Settings > Teams > Team > Members

Add existing users or groups to a team, or remove a member.

Project Administrators

[Project **Summary** page, **Invite**](#)

Add new or existing users. Send new users an invite. Optionally add users to one or more teams.

Project Administrators

[**Project Settings** > **Permissions** > **Groups** > **Group** > **Members**](#)

Add existing users or groups to a security group. By adding to a team group, you effectively add them to the team. Optionally remove a user from a group.

Project Collection Administrators

[**Organization Settings** > **Users**](#)

Add new users to an organization and send an invite. Must specify the access level. Optionally add them to select projects. Can use [Group rules](#) to further manage groups being added.

Project Collection Administrators

[az devops user CLI](#)

Add new users to an organization and send an invite. Must specify the access level.

Azure Active Directory Administrators

Azure Active Directory

Users you add to Azure Active Directory connected to Azure DevOps Services are added to the Project Collection Valid Users group. For more information, see [Connect your organization to Azure Active Directory](#).

Prerequisites

- You must have an organization and project. If you don't have a project yet, [create one](#).

- To add users to or remove users from a team, you must be added as a team administrator, or be a member of one of the administrative groups.
- To add users to or remove users from a project, you must be a member of the **Project Administrators** group.
- When the organization is connected to Azure Active Directory, the Allow team and project administrators to invite new users policy must be enabled for team administrators or members of the Project Administrators group to add new users.
- To add users or manage users for an organization, you must be a member of the **Project Collection Administrators** group. Organization owners are automatically members of this group.
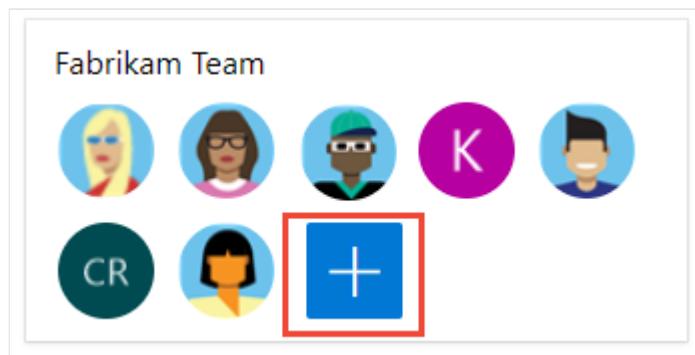
If you're new to Azure DevOps, you may want to familiarize yourself with the information provided in these articles:

- Get started with permissions, access levels, and security groups
- About projects and scaling your organization
- Default permissions and access quick reference
- About teams and Azure Boards tools

Add a user from the Team Members widget

As a team administrator, you can add new or existing members from the **Team Members** dashboard widget. To add this widget to a dashboard, see Add widgets to a dashboard.

1. To invite someone to your team, choose the plus button on the Team Members widget.

2. For new users, enter their email address. For existing users, type their name until it resolves as a known name to the system. You can add several email addresses or account names by separating them with a semicolon (;).

   Choose the entry listed under **Add users** to complete the entry.

Choose the name that appears to complete the entry.

Invite members to Fabrikam Team ✕

Search and add users to your Fabrikam Team

Users

👤 fabrikamfiber11@hotmail.com

Add users

fabrikamfiber11@hotmail.com

Cancel   Add

Invite members to Fabrikam Team ✕

Search and add users to your Fabrikam Team

Users

👤 fabrikamfiber9

JT   Jia-hao Tseng
     fabrikamfiber9@hotmail.com   A=

Cancel   Add

3. Complete the addition.

When the user is unknown, you'll get a notification that an access level must be

assigned. To complete the invitation, choose **Add**.

Choose **Add** to complete adding the user. Known users don't receive an invitation.

When you add a new user, the system assigns Stakeholder as the access level when all free five Basic access levels have been assigned. Active contributors to a project need to have Basic access as a minimum. A Project Collection Administrator can change the access level and resend invitations from the **Organization Settings>Users** page.

4. (Optional) A message will briefly display on the screen to indicate success or
   failure. Choose **Details** to open the notification and review details.

Success: Added 1 user to 2 teams
and assigned an access level to 1
user.   Details

Failed: Added 1 user to 1 team and
assigned an access level to 1 user.   Details

A success message indicates the status of adding the user to the system.

A failure message indicates why the addition of the user failed.

Notifications

System issues are problems in the system that require admin attention. Session
notifications are triggered by user activities in this session.

Session notifications 3   Dismiss all

✔ Added 1 user to 1 team and assigned an access level to 1 user.   Just now ✕

∧ Less details

User   Message

∨ ✔ Success 1

fabrikamfiber11@hotmail.com

"...
...

5. New users receive an email inviting them to sign in to the project. Existing users don't receive any formal notification.

Add users or groups to a team

Add existing users or security groups to a team from the **Project settings> Teams** page. From this interface you can view, add, or remove users and security groups to/from a team. To add a custom security group, see Add or remove users or groups, manage security groups.

> ⓘ **Note**
>
> To enable the new user interface for managing teams, enable the **New Teams Page** from the **Preview features** tool. To learn how, see **Manage or enable features**.
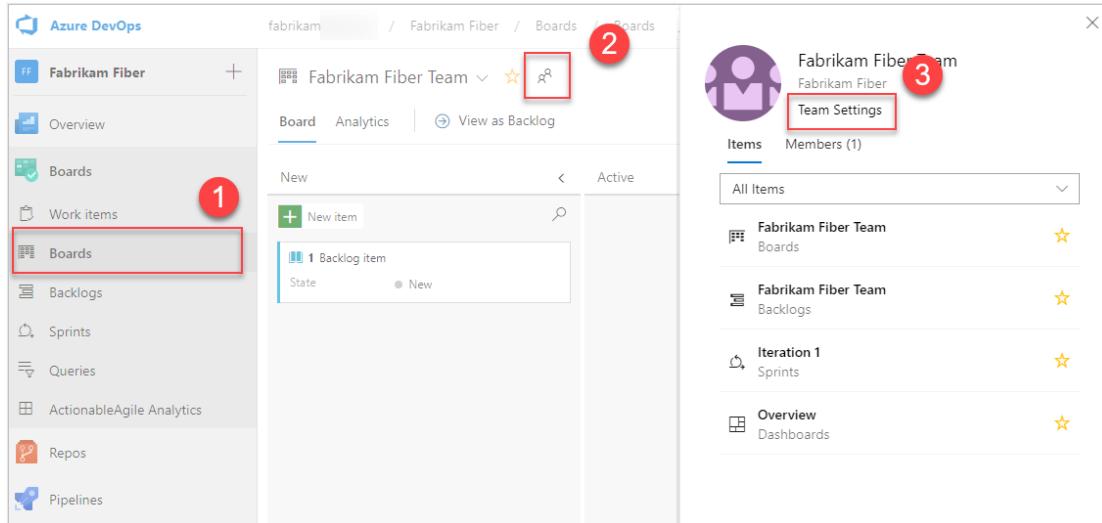
Preview UI

You can toggle between direct or expanded membership views. The **Direct Members** view displays users and groups that have been added to the team. The
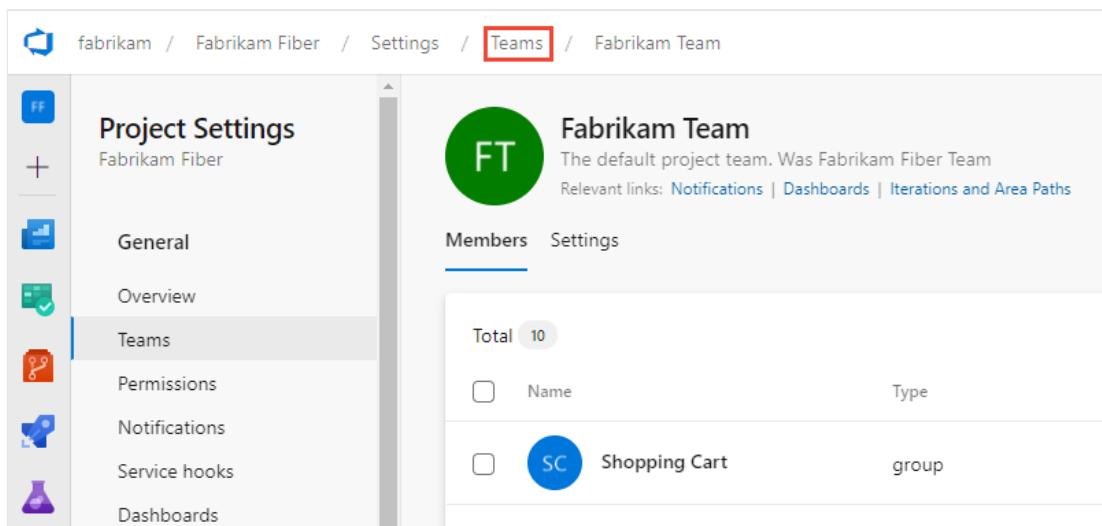
Expanded Members view replaces any Azure DevOps groups with the members that belong to those groups. Azure Active Directory or Active Directory groups aren't expanded.

1. Open a backlog or board for a team and choose the ⚇ team profile icon. Then choose **Team Settings**.

   Here we open the Board for the Web team and from there the team profile.

2. If you need to switch the team context, use the team selector within the breadcrumbs.

3. Choose **Add**.

4. Enter the sign-in addresses or display name for each account you want to add. You can also add a project security group—such as another team group, custom group, or Azure Active Directory group when used by the organization. Add them one at a time or all at the same time. You can enter several identities into the text box, separated by commas.

> 💡 **Tip**
>
> You must enter user and group names one at a time. However, after entering a name, the account is added to the list, and you can enter another name in the Identities text box before choosing to save your changes.

You may need to choose the ↻ refresh icon to see your updates.

5. To add an account as a team administrator, choose the Settings page and then choose **Add** under the Administrators section. For details, see [Add a team administrator](#)

Members   Settings

Total 10

Remove users or groups from a team

From the team's **Members** page, you can remove members.

Preview UI

1. To remove members, open the team's **Members** page, choose **Direct Members**, check the checkbox of the user you want to remove, choose **More options**, and then choose **Remove**.

Members   Settings   🔍 Search users and groups   ✕

Total 11			Direct Members ⌄	Remove
	Name	Type	Username or scope	
☐	Christie Church fabrikamfiber1@hotmail.com	user	fabrikamfiber1@hotmail.com	
☐	Chuck Reinhart fabrikamfiber3@hotmail.com	user	fabrikamfiber3@hotmail.com	
☑	fabrikamfiber11@hotmail.com fabrikamfiber11@hotmail.com	user	fabrikamfiber11@hotmail.com	⋮
☐	fabrikamfiber12@hotmail.com fabrikamfiber12@hotmail.com	user	fabrikamfiber12@hotmail.com	🗑 Remove
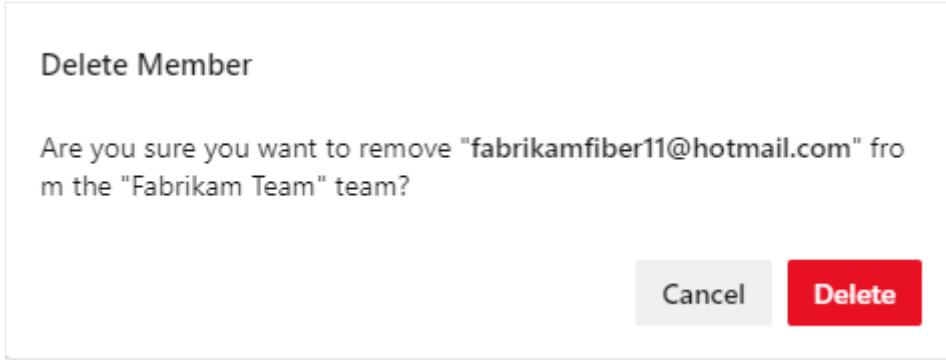
> To remove a team administrator as a team member, you must first remove them as an administrator.

2. Confirm the removal by choosing **Delete** in the confirmation message.

Invite users from the Summary page

As a member of the Project Administrators group, you can add members to a project from the **Summary** page and optionally add them to one or more teams. For more information about the **Summary** page, see Share your project vision, view project activity.
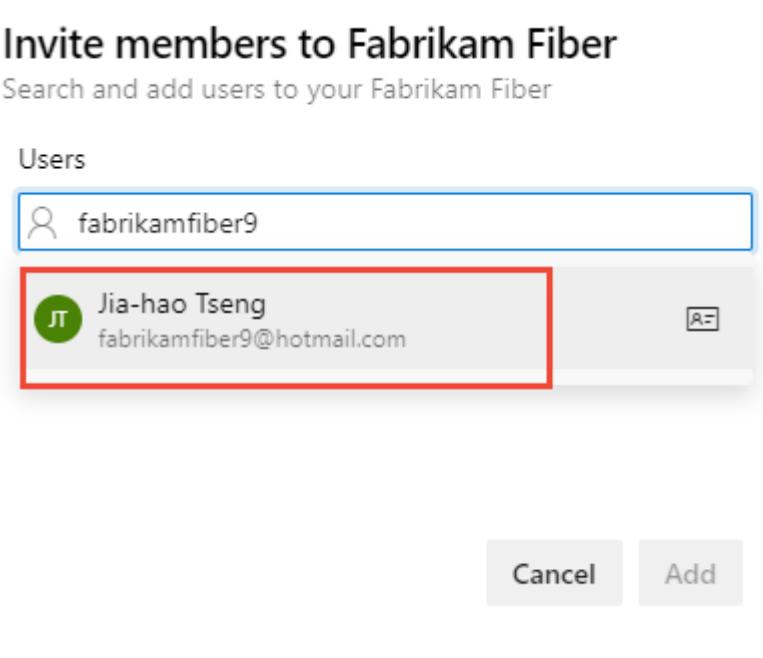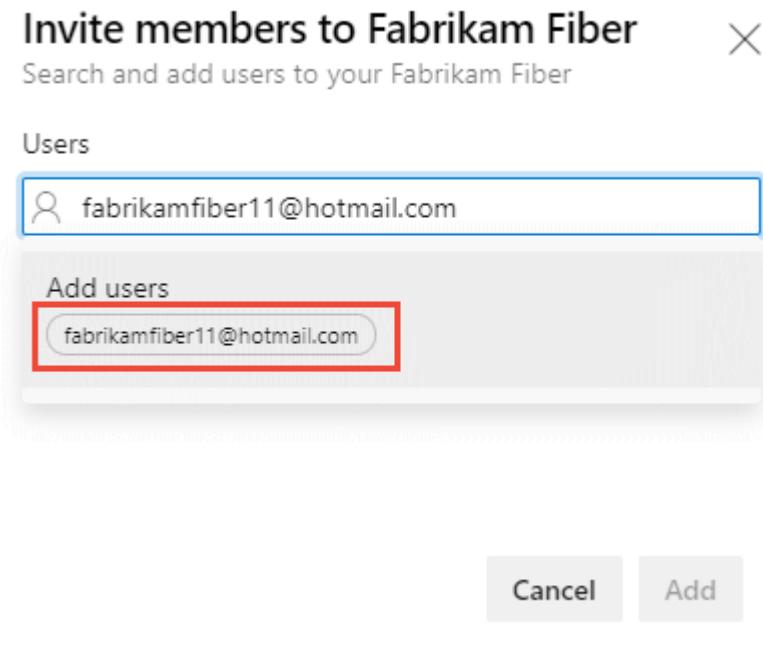
1. Open the **Project>Summary** page, and choose **Invite**.

1. For new users, enter their email address. For existing users, type their name until it resolves as a known name to the system. You can add several email addresses or account names by separating them with a semicolon (;).

   Choose the entry listed under **Add users** to complete the entry.

   If you're adding a user known by the organization or collection, type the name or email address and then choose the name that appears to complete the entry.

> ⓘ **Note**
>
> Any valid email address is acceptable. When the user accepts the invitation and signs into Azure DevOps, they register their email address as a Microsoft account and choose a password.

2. Optionally, select the teams you want to add the user to and then choose **Add** to complete the invitation.

   When the user is unknown, you'll get a notification that an access level must be assigned. To complete the invitation, choose **Add**.

Choose **Add** to complete the invitation.

When you add a new user, the system assigns Stakeholder as the access level when all free five Basic access levels have been assigned. Active contributors to a project need to have Basic access as a minimum. A Project Collection Administrator can change the access level from the **Organization Settings>Users** page.

> ⓘ **Note**
>
> Users that have limited access, such as Stakeholders, won't be able to access select features even if granted permissions to those features. For more information, see **Permissions and access**.

3. (Optional) A message will briefly display on the screen to indicate success or failure. Choose **Details** to open the notification and review details.

Success: Added 1 user to 2 teams and assigned an access level to 1 user. | Details

Failed: Added 1 user to 1 team and assigned an access level to 1 user. | Details

A success message indicates the status of adding the user to the system.

A failure message indicates why the addition of the user failed.

Notifications

System issues are problems in the system that require admin attention. Session notifications are triggered by user activities in this session.

Session notifications 3   Dismiss all

✓ Added 1 user to 1 team and assigned an access level to 1 user.   Just now ✕

∧ Less details

User	Message

∨ ✓ Success 1

fabrikamfiber11@hotmail.com

"...
...

Notifications

System issues are problems in the system that require admin attention. Session notifications are triggered by user activities in this session.

Session notifications 1   Dismiss all

❌ Add 1 user   Just now ✕

⌃ Less details

User	Message
⌄ ❌ Failed 1	
fabrikamfiber11@hotmail.com	You are trying to invite a user from outside your directory, but the security setting of this organization doesn't allow it. Learn more

4. New users receive an email inviting them to sign in to the project. Existing users don't receive any formal notification.

Add users or groups to a project

As a member of the **Project Administrators** group, you can add users or groups to a project from the **Project settings> Permissions** page by adding them to a security group. To add a custom security group, see Add or remove users or groups, manage security groups.

> ⓘ **Note**
>
> To enable the **Project Permissions Settings Page** preview page, see **Enable preview features**.

Preview UI

1. Open the web portal and choose the project where you want to add users or groups. To choose another project, see Switch project, repository, team.

2. Choose **Project settings**, and then **Permissions**.

3. Under **Groups**, choose one of the following options:

- **Readers**: To add users who require read-only access to the project, choose.
- **Contributors**: To add users who contribute fully to this project or who have been granted Stakeholder access.
- **Project Administrators**: To add users who need to administrate the project. For more information, see Change project-level permissions.

Or, you can choose any team group to add users to a specific team.

Here we choose the **Contributors** group.
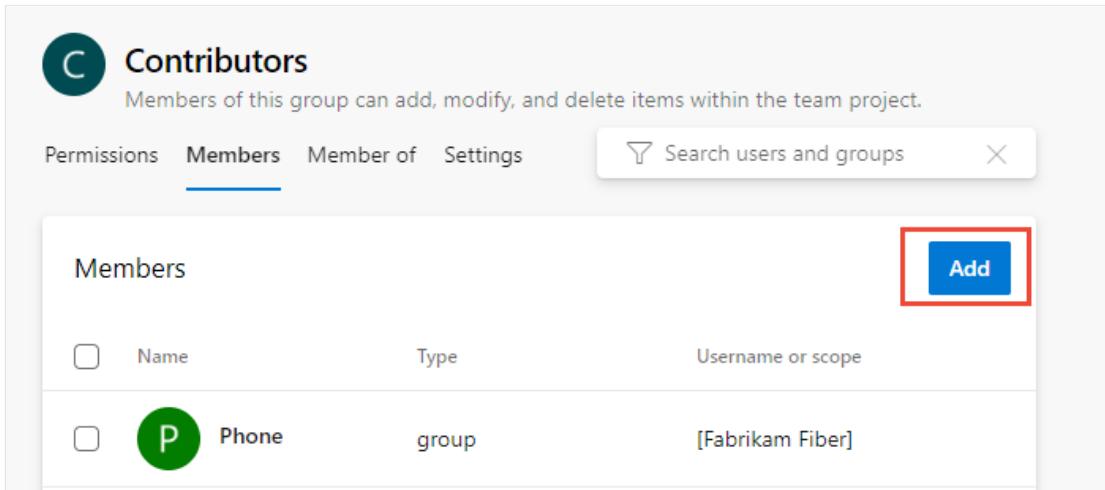
4. Next, choose the **Members** tab.

The default team group, and any other teams you add to the project, get included as members of the **Contributors** group. Add a new user as a member of a team instead, and the user automatically inherits Contributor permissions.

> 💡 **Tip**
>
> Managing users is much easier **using groups**, not individual users.

5. Choose **Add** to add a user or a user group.

6. Enter the name of the user account into the text box. You can enter several identities into the text box, separated by commas. The system automatically searches for matches. Choose the match(es) that meets your requirements.

> **① Note**
>
> The first time you add a user or group to Azure DevOps, you can't browse to it or check the friendly name. After the identity has been added, you can just enter the friendly name.

Choose **Save** when done.

7. You may customize user permissions for other functionality in the project. For example, in areas and iterations or shared queries.

Manage users or resend invitations

Project Collection Administrators can update user assignments and resend invitations. The various options they have are:

- Change the access level
- Manage user - add them to select projects
- Resend invite
- Remove direct assignments

- Remove from organization

For more information, see Add account users for Azure DevOps.

List team members or team details

From the Azure DevOps CLI command, you can see details about a team or list the individual members of that team. To first see a list of all teams in your organization, use the az devops team list command.

List team members | Show team details

> ⓘ **Note**
>
> You can use the **az devops user** command to add users to an organization. There is no comparable command for adding users to a team or project.

List team members

You can list the individual members of a team in your organization with the az devops team list-member command. To get started, see Get started with Azure DevOps CLI.

```
Azure CLI

az devops team list-member --team
                           [--org]
```

```
                                  [--project]
                                  [--skip]
                                  [--top]
```

Parameters

- **team**: Required. Name or ID of the team to show.
- **org**: Azure DevOps organization URL. You can configure the default organization using `az devops configure -d organization=ORG_URL`. Required if not configured as default or picked up using `git config`. Example: `--org https://dev.azure.com/MyOrganizationName/`.
- **project**: Name or ID of the project. You can configure the default project using `az devops configure -d project=NAME_OR_ID`. Required if not configured as default or picked up using `git config`.
- **skip**: Optional. Number of members to skip.
- **top**: Optional. Maximum number of members to return.

Example

The following command lists the first five members of the team named **Fabrikam Team** and returns the details in table format.

Azure CLI

```
az devops team list-member --team "Fabrikam Team" --top 5 --output table

ID                                    Name              Email
-----------------------------------   ----------------  -------------------
-------
3b5f0c34-4aec-4bf4-8708-1d36f0dbc468  Christie Church
fabrikamfiber1@hotmail.com
19d9411e-9a34-45bb-b985-d24d9d87c0c9  Johnnie McLeod
fabrikamfiber2@hotmail.com
8c8c7d32-6b1b-47f4-b2e9-30b477b5ab3d  Chuck Reinhart
fabrikamfiber3@hotmail.com
d291b0c4-a05c-4ea6-8df1-4b41d5f39eff  Jamal Hartnett
fabrikamfiber4@hotmail.com
bd30c189-db0f-4dd6-9418-5d8b41dc1754  Raisa Pokrovskaya
fabrikamfiber5@hotmail.com
```

Show team details

You can view details about a team in your organization with the az devops team show command. To get started, see Get started with Azure DevOps CLI.

```
Azure CLI

az devops team show --team
                    [--org]
                    [--project]
```

Parameters

- **team**: Required. Name or ID of the team to show.
- **org**: Azure DevOps organization URL. You can configure the default organization using `az devops configure -d organization=ORG_URL`. Required if not configured as default or picked up using `git config`. Example: `--org https://dev.azure.com/MyOrganizationName/`.
- **project**: Name or ID of the project. You can configure the default project using `az devops configure -d project=NAME_OR_ID`. Required if not configured as default or picked up using `git config`.

Example

The following command shows information about the team in your organization named **Fabrikam Team** and returns the details in table format.

```
Azure CLI

az devops team show --team "Fabrikam Team" --output table

ID                                    Name          Description
------------------------------------  ------------  ------------------------
                                                    ------------------------
a48cb46f-7366-4f4b-baf5-b3632398ed1e  Fabrikam Team  The default project
team. Was Fabrikam Fiber Team
```

Next steps

Manage your project

Related articles

- Add users and manage access
- Resources granted to project members
- Manage your organization, Limit identity search and selection

- Manage your organization, Limit user visibility for projects and more
- Manage permissions with command line tool
- Grant or restrict access using permissions.

Manage and configure team tools

Article • 01/05/2024

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019

If you're a team administrator, you have the flexibility to tailor your backlogs and boards to align with your team's workflow. In case you require a new team, you can ask a Project Administrator group member to create one for you, which takes only a minute. Team administrators have the ability to set up and oversee all team tools.

Team administrators perform the following tasks for team tools:

- Add team members
- Add another team administrator
- Configure areas and iteration paths
- Configure backlogs, boards, and general settings
- Configure and manage team dashboards
- Configure team notifications

Prerequisites

- To perform any team configuration task, you must be a team administrator for the team to be modified, or be a member of the **Project Administrators** group. For more information, see Change project-level permissions.
- To add a team, you must be a member of the **Project Administrators** group. For more information, see Add teams.

> ⓘ **Note**
>
> For more information, see the following articles:
>
> - **About backlogs**
> - **Configure and customize Azure Boards**
> - **Create a project using the process of your choice**
> - **Customize your work tracking experience (process models)**
> - **Manage inherited processes**

Open your team profile

Open your team profile to quickly access items defined for your team.

1. Sign in to your organization (`https://dev.azure.com/{yourorganization}`), and then open your project.

2. Select **Project settings** > **Teams** > your team name.

Add users to a team

Tools like capacity planning, team alerts, and dashboard widgets operate within the scope of a team. They automatically access the user information of team members to facilitate planning tasks or issue alerts.

To add users to a team, see Add users to a project or specific team.

All members of a team can favorite team artifacts and define work item templates. For more information, see:

- [Set personal or team favorites](#)
- [Use templates to add and update work items](#).

If team members don't have access to all the features they want, make sure they have [the permissions needed for those features](#).

Add an administrator

When you add a team to a project, a Project Administrator should [add one or more team administrators](#).

Configure team areas and iterations

Many Agile tools rely on the team's configured area and iteration paths. For more information, see [About teams and Agile tools](#).

After project administrators add the project's area and iteration paths using [Set area paths](#) and [Set iteration paths](#), team administrators can choose the relevant area and iteration paths for their team. These settings influence a wide range of Agile tools that the team can access.

Settings include making the following associations for each team:

- **Select team area paths**
  Can select the default area path(s) associated with the team. These settings affect many Agile tools available to the team.
- **Select team iteration paths or sprints** Can select the default area path(s) associated with the team. These settings affect many Agile tools available to the team.

For more information, see Define area paths and assign to a team and Define iteration paths and configure team iterations.

Configure team backlogs, boards, and general settings

As a team administrator, you have the flexibility to customize your team's workflow to suit your needs. One way to do so is by choosing which backlog levels are active for your team. For instance, a feature team might only want to display the product backlog, while a management team might prefer to show the feature and epic backlogs only.

Also, you can choose how to treat bugs within your workflow, either as user stories and requirements or as tasks.
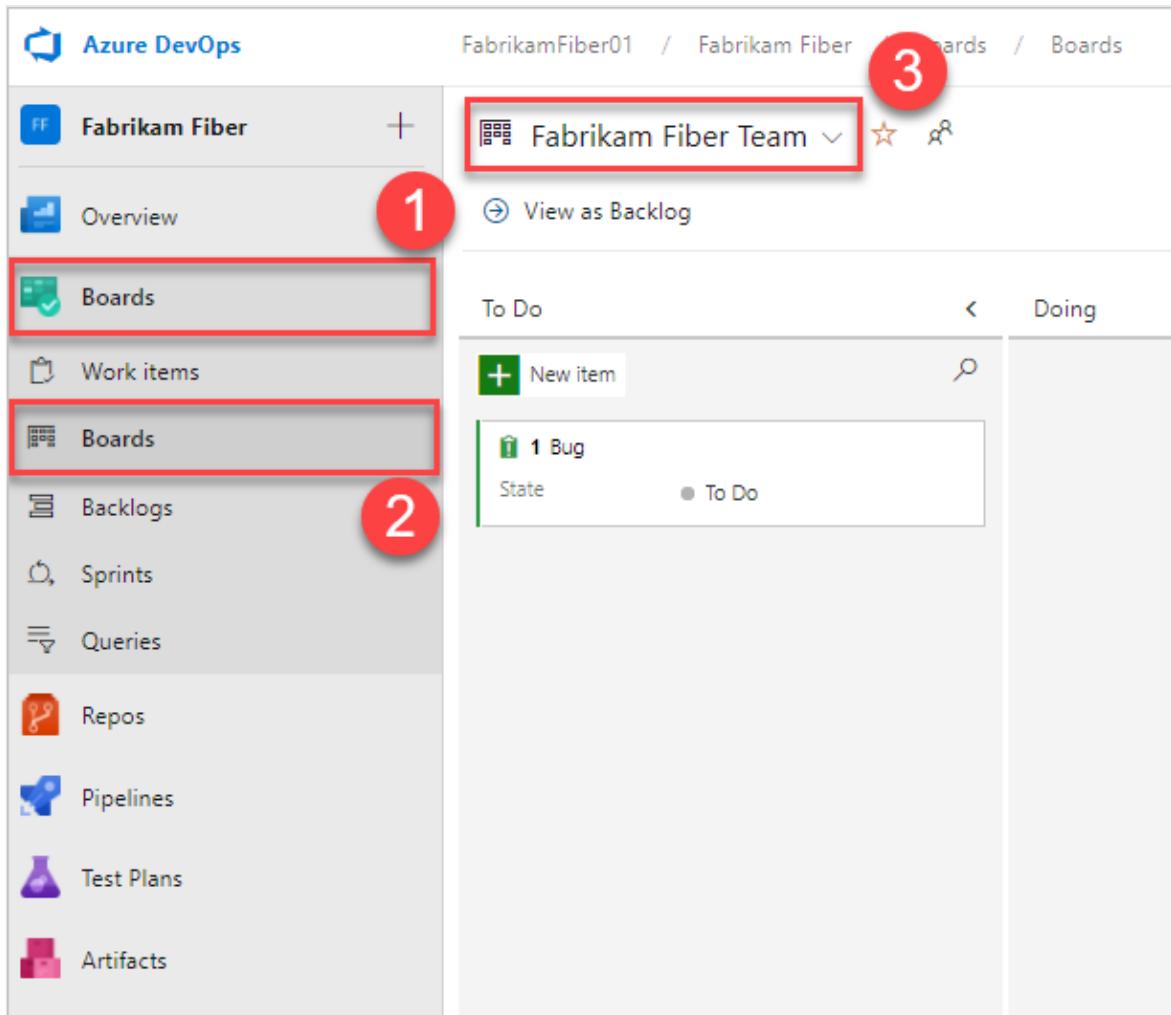
Another way to customize your team's workflow is by selecting non-working days for the team. By doing so, sprint planning and tracking tools can automatically take these days off into account when calculating capacity and sprint burndown.

Most of these team settings can be easily configured from the common configuration dialog, providing a convenient way to manage your team's workflow in one central location. You can also set team automation rules to update work items when child item states change.
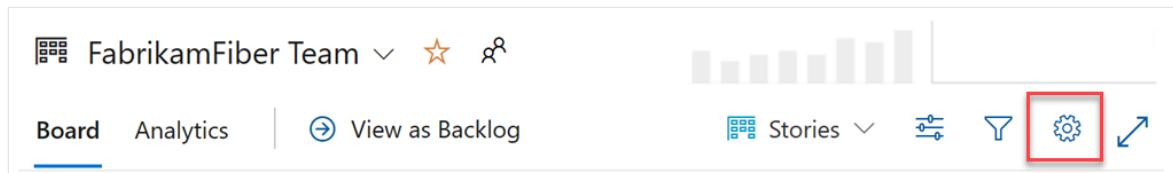
> ⓘ **Note**
>
> For more information, see **Backlogs, boards, and plans**. In case you're not seeing the desired work items on your backlog or board, see **Set up your backlogs and boards** to configure them according to your preferences.
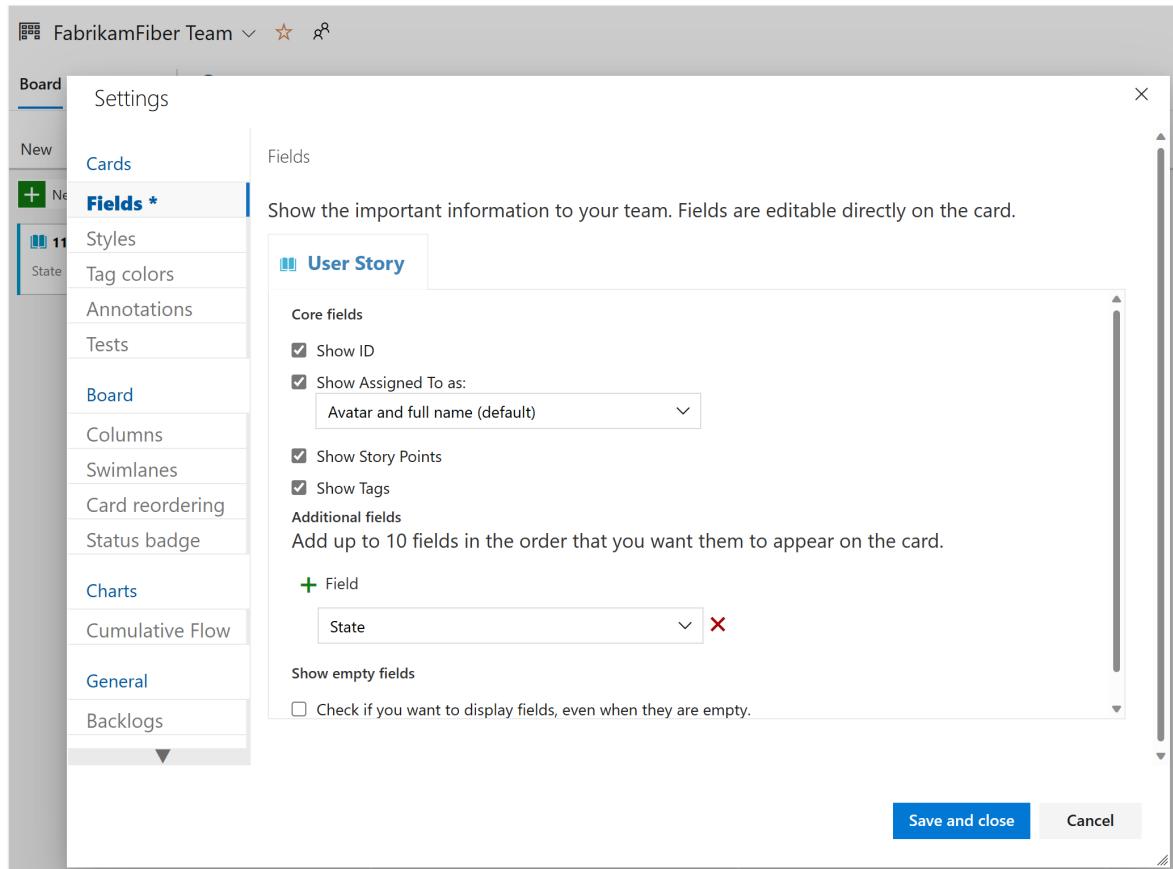
1. Check that you selected the correct project, and then choose **Boards** > **Boards**, and select the correct team from the team selector dropdown menu. For more information, see Use breadcrumbs and selectors to navigate and open artifacts.

2. Choose **Team settings** ⚙ to configure the board and set general team settings.

3. Choose a tab under any of the sections—**Cards, Board**, **Charts**, and **General**—to configure the cards or boards, the cumulative flow chart, or other team settings. When you're done configuring the settings, select **Save and close**.

Team administrators have complete control over customizing their team's Kanban boards for both the product and portfolio backlogs. To set up a Kanban board, you can define the columns and work-in-progress (WIP) limits through the common configuration dialog. For more information, see Kanban overview and Kanban quickstart.

For detailed information on each configuration option, you can explore the following articles:

General
- Backlogs
- Working with bugs

Cards
- Add fields

- [Define styles](#)
- [Add tag colors](#)
- [Enable annotations](#)
- [Configure inline tests](#)
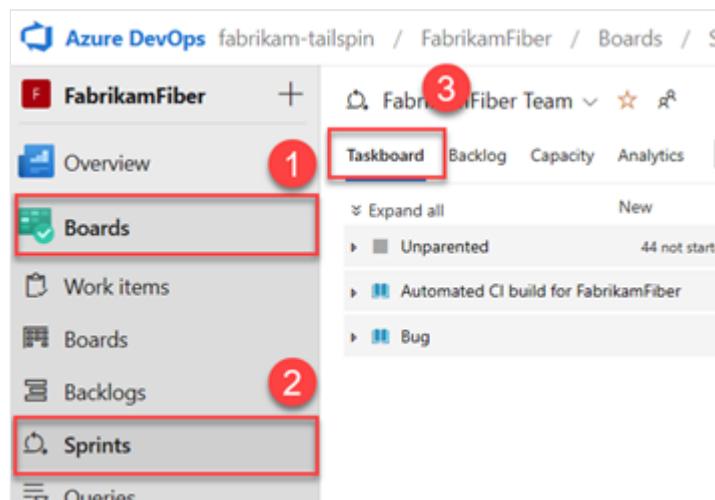
Boards
- [Add columns](#)
- [Split columns](#)
- [WIP limits](#)
- [Definition of Done](#)
- [Add swimlanes](#)
- [Card reordering](#)
- [Configure status badges](#)

Chart
- [Configure cumulative flow chart](#)
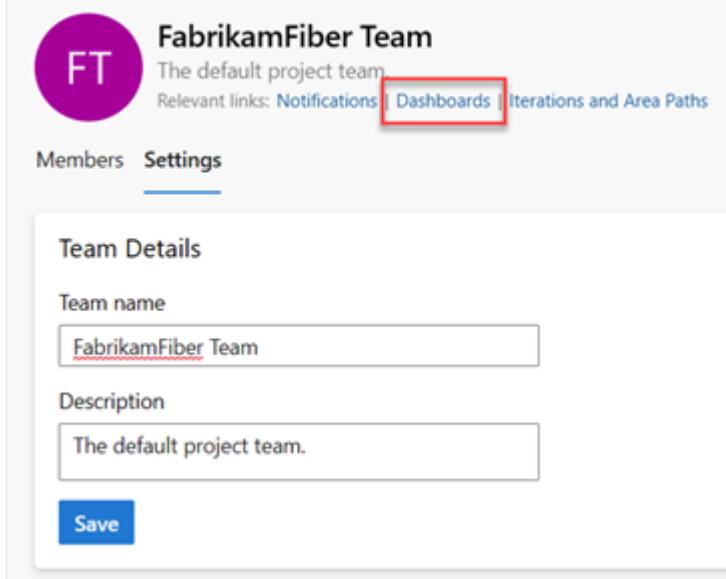
Configure sprint taskboards

Similar to Kanban boards, you can customize each sprint taskboard to support information-rich color-coded cards and columns. For more information, see [Customize sprint taskboards](#).

Add and manage team dashboards

By default, all team members can add and edit team dashboards. In addition, team administrators can manage permissions for team dashboards. For more information, see [Add and manage dashboards](#).
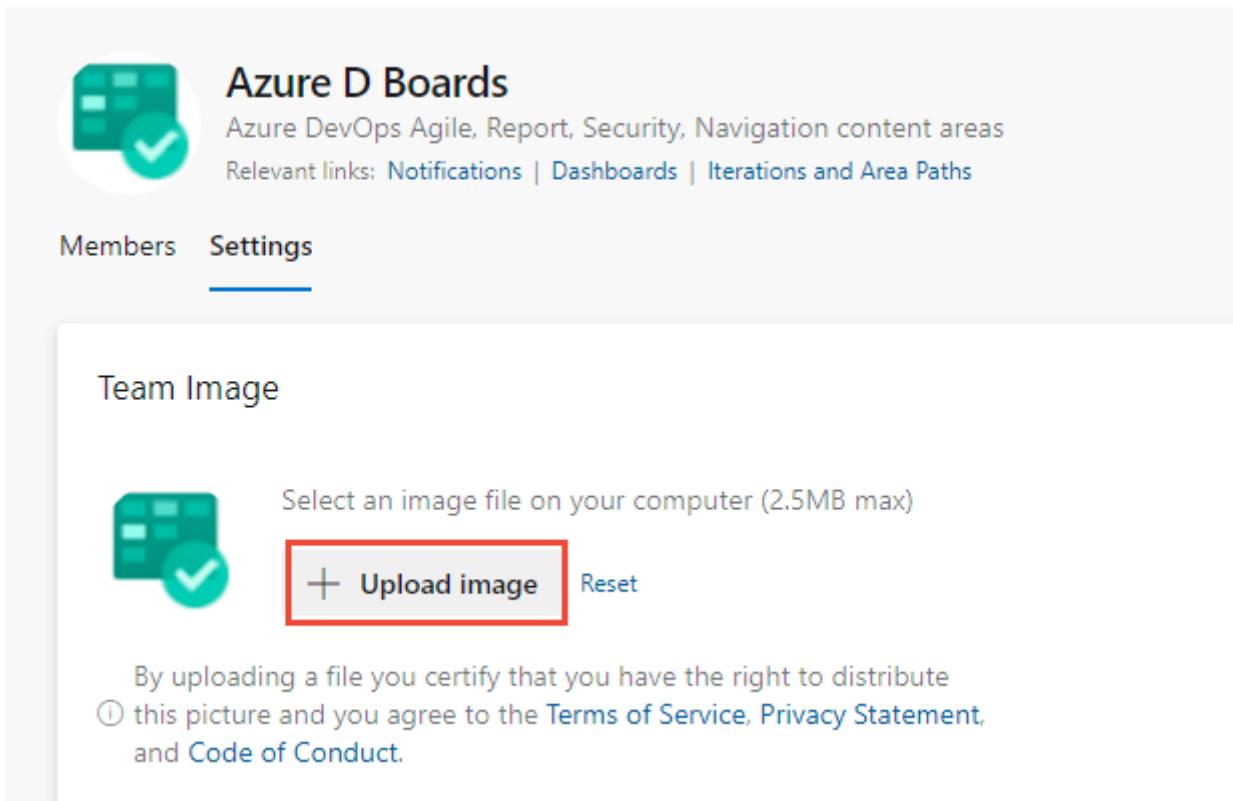
Update team name, description, and image

Team settings also include the team name, description, and team profile image. To add a team picture, select the image icon. The maximum file size is 2.5 MB and 2560 x 1024 px, and then we resize to 200 x 200.

Manage notifications

Team administrators have the ability to add and edit alerts, allowing the team to receive email notifications as changes occur to work items, code reviews, source control files, and builds. Various alerts are pre-defined for each team. For more information, see [Manage team alerts](#).

Related articles

- [About projects and scaling your organization](#)
- [About teams and Agile tools](#)
- [Add teams](#)
- [Add a team administrator](#)
- [Automate work item state transitions](#)

Request an increase in permission levels

Article • 03/23/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

To get access to certain tasks, you might need to request an increase to your permissions or be added to a security role. Typically, you'll do this upon receiving an information or error message indicating you have insufficient permissions. Such a message will indicate the permission levels you need.

Prior to requesting a change in permission levels, make sure you understand the basics by reviewing Get started with permissions, access, and security groups.

Common permissions to request

Most users added to the **Contributors** group are granted the permissions they need to perform most tasks. However, the following tasks require membership in the **Project Administrators** group or a change in permissions.

- **Work tracking**
  - Add or change **Area Paths** or **Iteration Paths**: Requires elevated permissions to an Area Path or Iteration Path node. To learn more, see Set work tracking permissions, Create child nodes.
  - Create shared queries or query folders: Requires elevated permissions set for a shared query folder. To learn more, see Set work tracking permissions, Set permissions on queries or query folders.
  - Change team settings—such as Kanban board settings: Requires addition as a team administrator. To learn more, see Add or remove a team administrator

- **Source code, Git repositories**, the following tasks require elevated permissions for Git repositories or a specific repository. To learn more, see Set Git repository permissions.
  - Create, delete, or rename a Git repository
  - Manage repository permissions
  - Bypass policies

The following tasks require membership in the **Project Collection Administrators** group or a change in permissions at the collection-level or addition to a specific role.

- **Collection-level configurations**

- Create projects: Requires elevated permissions at the [collection level](#).
- Add, edit, or manage a process: Requires elevated permissions at the collection level or [process-level permissions](#).
- Install, uninstall, or disable extensions: Requires addition to the [Manager role](#) for extensions.

For an overview of built-in security groups and default permission assignments, see [Default permissions and access](#).

Prerequisites

- To view permissions, you must be a member of the **Project Valid Users** group. Users added to a project are automatically added to this security group. To learn more, see [View permissions for yourself or others](#).
- To look up an administrator for your project or project collection, you must be a member of the **Project Valid Users** group.

> ⓘ **Note**
>
> Users added to the **Project-Scoped Users** group won't be able to access **Organization Settings** other than the **Overview** section if the **Limit user visibility and collaboration to specific projects** preview feature is enabled for the organization. For more information including important security-related call-outs, see **Manage your organization, Limit user visibility for projects and more**.

Review your permission assignments

Before you request a change to permission levels, review your permission assignments as described in [View permissions for yourself or others](#).

Verify that your permission assignments are preventing you from accomplishing a task you need to perform.

Request a change to a permission level or role change

To request a change or increase in your permission levels, take the following actions:

1. Identify the permissions you need and at what level. Permissions are set at the object, project, and project-collection level. Also, permissions are granted through

various roles. To identify the level and permission you need, review the Permissions lookup guide.

2. Identify a person in your organization who can grant you the permissions you need. For example:

- To get permissions to manage team settings, identify the team administrator for your team or a member of the Project Administrators group.
- To change an object-level permission, identify the owner of the object or a member of the **Project Administrators** group. To learn how, see Set object-level permissions.
- To change a project-level permission, identify a member of the **Project Administrators** group. See Look up a project administrator.
- To change a project collection-level permission, identify a member of the **Project Collection Administrators** group. See Look up a project collection administrator.

3. Contact the person you identified in step 2 and make your request. Make sure you specify the permission you want changed.

Refresh or re-evaluate your permissions

After your permission levels are changed, you may need to refresh your permissions for Azure DevOps to recognize the changes. This step is recommended when a change is made to your permission level, role level, or if you are added to a new or different Azure DevOps, Azure Active Directory, or Active Directory security group. When you are added to a new or different security group, your inherited permissions may change.

By refreshing your permissions, you cause Azure DevOps to re-evaluate your permission assignments. Otherwise, your permission assignments won't be refreshed until you sign-off, close your browser, and sign-in again.

To refresh your permissions, choose **User settings**, on the **Permissions** page, you can select **Re-evaluate permissions**. This function reevaluates your group memberships and permissions, and then any recent changes take effect immediately.

Related articles

- [Permissions lookup guide](#)
- [Default permissions and access](#)
- [Troubleshoot permissions](#)
- [Look up a project administrator](#)
- [Look up a project collection administrator](#)

Grant or restrict access using permissions

Article • 10/16/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

You can grant or restrict access to resources that you manage in Azure DevOps. You may want to open up or close down access to a select set of features and for a select set of users. While the built-in security groups provide a standard set of permission assignments, you may need more security requirements not met by these assignments.

If you're new to administrating permissions and groups, review Get started with permissions, access, and security groups to learn about permission states and inheritance.

In this article you learn how to do the following tasks:

- ✔ Recommended method for granting and restricting permissions
- ✔ Delegate tasks by assigning select permissions to specific roles
- ✔ Limit user visibility to organization information
- ✔ Limit people picker to project users and groups
- ✔ Restrict access to view or modify objects
- ✔ Restrict modification of work items based on a user or group

> 💡 **Tip**
>
> Because you set many permissions at an object-level, such as repositories and area paths, how you structure your project determines the areas you can open up or close down.

Recommended method for granting and restricting permissions

For maintenance purposes, we recommend you use either the built-in security groups or custom security groups to manage permissions.

You can't change the permission settings for the Project Administrators group or the Project Collection Administrators group, which is by design. However, for all other

groups, you can change the permissions.

If you manage a few users, then you may find changing individual permissions a valid option. However, custom security groups allow you to better track roles and permissions assigned to those roles.

Delegate tasks to specific roles

As an administrator or account owner, it's a good idea to delegate administrative tasks to those team members who lead or manage an area. Several of the main built-in roles that come with default permissions and role assignments are:

- Readers
- Contributors
- Team Administrator (role)
- Project Administrators
- Project Collection Administrators

For a summary of permissions for the above roles, see Default permissions and access, or for the Project Collection Administrators, see Change project collection-level permissions.

To delegate tasks to other members within your organization, consider creating a custom security group and then granting permissions as indicated in the following table.

Role

Tasks to perform

Permissions to set to Allow

Development lead (Git)

Manage branch policies

Edit policies, Force push, and Manage permissions
See Set branch permissions.

Development lead (TFVC)

Manage repository and branches

Administer labels, Manage branch, and Manage permissions
See Set TFVC repository permissions.

Software architect (Git)

Manage repositories

Create repositories, Force push, and Manage permissions
See Set Git repository permissions

Team administrators

Add area paths for their team
Add shared queries for their team

Create child nodes, Delete this node, Edit this node See Create child nodes, modify work items under an area path
Contribute, Delete, Manage permissions (for a query folder), See Set query permissions.

Contributors

Add shared queries under a query folder, Contribute to dashboards

Contribute, Delete (for a query folder), See Set query permissions
View, Edit, and Manage dashboards, See Set dashboard permissions.

Project or product manager

Add area paths, iteration paths, and shared queries
Delete and restore work items, Move work items out of this project, Permanently delete work items

Edit project-level information, See Change project-level permissions.

Process template manager (Inheritance process model)

Work tracking customization

Administer process permissions, Create new projects, Create process, Delete field from account, Delete process, Delete project, Edit process
See Change project collection-level permissions.

Process template manager (Hosted XML process model)

Work tracking customization

Edit collection-level information, See Change project collection-level permissions.

Project management ([On-premises XML process model](#))

Work tracking customization

Edit project-level information, See [Change project-level permissions](#).

Permissions manager

Manage permissions for a project, account, or collection

For a project, Edit project-level information
For an account or collection, Edit instance-level (or collection-level) information
To understand the scope of these permissions, see [Permission lookup guide](#). To request a change in permissions, See [Request an increase in permission levels](#).

You can also grant permissions to manage permissions for the following objects:
- [Set Git repository permissions](#)
- [Manage Git branch permissions](#)
- [Set TFVC repository permissions](#)
- [Administer build and release permissions](#)
- [Manage Wiki permissions](#).

Limit user visibility to organization and project information

> ⓘ **Important**
>
> - The limited visibility features described in this section apply only to interactions through the web portal. With the REST APIs or `azure devops` CLI commands, project members can access the restricted data.
> - Guest users who are members in the limited group with default access in Microsoft Entra ID, can't search for users with the people picker. When the preview feature's turned *off* or when guest users aren't members of the limited group, guest users can search all Microsoft Entra users, as expected.

By default, users added to an organization can view all organization and project information and settings. To restrict access to only those projects that you add users to, you can enable the **Limit user visibility and collaboration to specific projects** preview feature for the organization. To enable this feature, see [Manage or enable features](#).

With this feature enabled, users added to the **Project-Scoped Users** group can't view most **Organization settings** and can only connect to those projects to which they've been added.
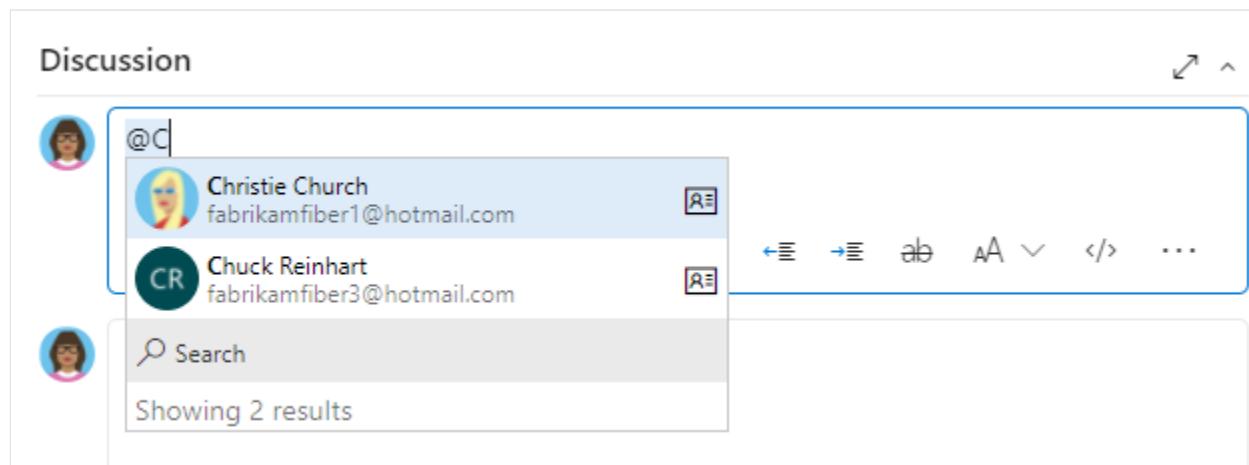
> ⚠ **Warning**
>
> When the **Limit user visibility and collaboration to specific projects** preview feature is enabled for the organization, project-scoped users are unable to search for users who were added to the organization through Microsoft Entra group membership, rather than through an explicit user invitation. This is an unexpected behavior and a resolution is being worked on. To self-resolve this issue, disable the **Limit user visibility and collaboration to specific projects** preview feature for the organization.

Limit people picker to project users and groups

For organizations that manage their users and groups using Microsoft Entra ID, people pickers support searching all users and groups added to Microsoft Entra ID, not just those users or groups added to a project. People pickers support the following Azure DevOps functions:

- Selection of a user identity from a work tracking identity field such as **Assigned To**
- Selection of a user or group using **@mention** in a work item discussion or rich-text field, a pull request discussion, commit comments, or changeset or shelveset comments
- Selection of a user or group using **@mention** from a wiki page

As shown in the following image, you simply start typing into a people picker box until you find a match to a user name or security group.

Users and groups who are added to the **Project-Scoped Users** group can only see and select users and groups in the project they're connected to from a people picker. To scope people pickers for all project members, see Manage your organization, Limit identity search and selection.

Restrict access to view or modify objects

Azure DevOps is designed to enable all valid users to view all objects defined in the system. You can restrict access to resources by setting the permission state to **Deny**. You can set permissions for members that belong to a custom security group or for an individual user. To learn more about how to set these types of permissions, see Request an increase in permission levels.

Area to restrict

Permissions to set to Deny

View or contribute to a repository

View, Contribute
See Set Git repository permissions or Set TFVC repository permissions.

View, create, or modify work items within an area path

Edit work items in this node, View work items in this node
See Set permissions and access for work tracking, Modify work items under an area path.

View or update select build and release pipelines

Edit build pipeline, View build pipeline
Edit release pipeline, View release pipeline
You set these permissions at the object level. See Set build and release permissions.

Edit a dashboard

View dashboards
See Set dashboard permissions.

Restrict modification of work items or select fields

For examples that illustrate how to restrict modification of work items or select fields, see Sample rule scenarios.

Next steps

Remove user accounts

Related articles

- Troubleshoot permissions
- Rules and rule evaluation
- Default permissions and access
- Permission lookup guide
- Get started with permissions, access, and security groups
- Permissions and groups reference
- Change project-level permissions
- Change project collection-level permissions

Security best practices

Article • 11/09/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

When you're working with information and data, particularly in a cloud-based solution like Azure DevOps Services, prioritizing security should always be your primary concern. While Microsoft maintains the security of the underlying cloud infrastructure, it's your responsibility to configure security in Azure DevOps.

Although it's not mandatory, incorporating best practices while using Azure DevOps can enhance your experience and make it more secure. We've compiled the following best practices that aim to keep your Azure DevOps environment secure:

- Securing your Azure DevOps environment
- Restrict access through scoped permissions at the organization/collection, project, or object level
- Maintain tight control of administrators and security groups
- Scope service accounts and service connections
- Learn best practices for authenticating when integrating with Azure DevOps
- Secure specific product areas and associated services, like Azure Artifacts, Azure Boards, Azure Pipelines, Azure Repos, Azure Test Plans, and GitHub integrations.

Secure Azure DevOps environment

Removing users

- If your organization uses MSA accounts, then remove inactive users directly from the organization, as you have no other way to prevent access. When you do so, you can't create a query for work items assigned to the removed user account. For more information, see Delete users from Azure DevOps.
- If your organization is connected to Microsoft Entra ID, then you can disable or delete the Microsoft Entra user account and leave your Azure DevOps user account active. In this way, you can continue to query work item history using your Azure DevOps user ID.
- Revoke user PATs.
- Revoke any special permissions that may have been granted to individual user accounts.
- Reassign work from users you're removing to current team members.

Use Microsoft Entra ID

Integrate Azure DevOps with Microsoft Entra ID to have a single plane for identity. Consistency and a single authoritative source increases clarity and reduces security risks from human errors and configuration complexity. The key to end governance is to have multiple role assignments (with different role definitions and different resource scopes to the same Microsoft Entra groups). Without Microsoft Entra ID, you're solely responsible for controlling organization access.

Using Microsoft Entra ID also allows you to access other security features, like multi-factor authentication or other conditional access policies.

For more information, see the following articles:

- About accessing your organization with Microsoft Entra ID
- Add Active Directory / Microsoft Entra users or groups to a built-in security groups
- Limit access by location or IP addresses
- Manage conditional access
- Require all users to use multi-factor authentication (MFA)

Review auditing events

Once you have a Microsoft Entra backed organization, you can turn on Auditing in your Security policies. Periodically review audit events to monitor and react to unexpected usage patterns by administrators and other users.

Secure your network

A few ways to do so might include:

- Set up an allowlist to restrict specific IPs.
- Always use encryption.
- Validate certificates.
- Implement Web application firewalls (WAFs), so they can filter, monitor, and block any malicious web-based traffic to and from Azure DevOps.
- For more information, see this guidance on application management best practices

Scoped permissions

The system manages permissions at different levels - individual, collection, project, and object - and assigns them to one or more built-in groups by default.

- Only give users and services the minimum amount of access needed to perform their business functions.
- Disable inheritance where possible. Due to the allow-by-default nature of inheritance, unexpected users can get access or permissions. For more information, read about inheritance.
- Learn more about permissions here:
  - Permissions and role lookup guide
  - Permissions, security groups, and service accounts reference
  - Set individual permissions.

Project-level permissions

- Limit access to projects and repos to reduce the risk of leaking sensitive information and deploying insecure code to production.
- Use either the built-in security groups or custom security groups to manage permissions. For more information, see Grant or restrict permissions to select tasks.
- Disable "*Allow public projects*" in your organization's policy settings to prevent every organization user from creating a public project. Azure DevOps Services allows you to change the visibility of your projects from public to private, and vice-versa. If users haven't signed into your organization, they have read-only access to your public projects. If users have signed in, they can be granted access to private projects and make any permitted changes to them.
- Don't allow users to create new projects.

External guest access

- Block external guest access entirely by disabling the "Allow invitations to be sent to any domain" policy. It's a good idea to do so if there's no business need for it.
- Use a different email or user principal name (UPN) for your personal and business accounts, even though it's allowed. This action eliminates the challenge of disambiguating between your business and personal accounts when the email/UPN is the same.
- Put all the external guest users in a single Microsoft Entra group and manage the permissions of that group appropriately. You can easily manage and audit this way.
  - Remove direct assignments so the group rules apply to those users. For more information, see Add a group rule to assign access levels.
  - Reevaluate rules regularly on the Group rules tab of the Users page. Clarify whether any group membership changes in Microsoft Entra ID might affect your organization. Microsoft Entra ID can take up to 24 hours to update dynamic

group membership. Every 24 hours and anytime a group rule changes, rules get automatically reevaluated in the system.

- For more information, see B2B guests in the Microsoft Entra ID.

Manage security groups

Security and user groups

See the following recommendations for assigning permissions to security groups and users groups.

Do ✔️	Don't ❌
Use Microsoft Entra ID, Active Directory, or Windows security groups when you're managing lots of users.	Don't change the default permissions for the *Project Valid Users* group. This group can access and view project information.
When you're adding teams, consider what permissions you want to assign to team members who need to create and modify area paths, iteration paths, and queries.	Don't add users to multiple security groups that contain different permission levels. In certain cases, a *Deny* permission level may override an *Allow* permission level.
When you're adding many teams, consider creating a *Team Administrators* custom group where you allocate a subset of the permissions available to *Project Administrators*.	Don't change the default assignments made to the *Project Valid Users* groups. If you remove or set *View instance-level information* to *Deny* for one of the *Project Valid Users* groups, no users in the group can access whatever project, collection, or deployment you set the permission on.
Consider granting the work item query folders *Contribute* permission to users or groups who require the ability to create and share work item queries for the project.	Don't assign permissions that are noted as *Assign only to service accounts* to user accounts.
Keep groups as small as possible. Access should be restricted, and the groups should be frequently audited.	
Take advantage of built-in roles and default to Contributor for developers. Admins get assigned to the Project Administrator security group for elevated permissions, allowing them to configure security permissions.	

For more information, see Valid user groups.

Just-in-time access for admin groups

You can change the configuration of your organization or project if you have Project Collection Administrator and Project Administrator access. To protect access to these built-in administrator groups, require just-in-time access using a Microsoft Entra Privileged Identity Management (PIM) group.

Configure access

1. Create a role-assignable group in Microsoft Entra ID.
2. Add your Microsoft Entra group to the Azure DevOps group.

> ⓘ **Note**
>
> Make sure any user with elevated access using a PIM group also has standard access to the organization, so they can view the page to refresh their permissions.

Use access

1. Activate your access.
2. Refresh your permissions in Azure DevOps.
3. Take the action requiring administrator access.

> ⓘ **Note**
>
> Users have elevated access in Azure DevOps for up to 1 hour after their PIM group access gets deactivated.

Scope service accounts

- Ensure service accounts have zero interactive sign-in rights.
- Restrict service account privileges to the bare minimum necessary.
- Use a different identity for the report reader account, if you use domain accounts for your service accounts. For more information, see Service accounts and dependencies.
- Use local accounts for user accounts, if you're installing a component in a workgroup. For more information, see Service account requirements.
- Use service connections when possible. Service connections provide a secure mechanism to connect to assorted services without the need to pass in secret

variables to the build directly. - Restrict these connections to the specific place they should be used and nothing more.

- Monitor service account activity and create audit streaming. Auditing allows you to detect and react to suspicious sign-ins and activity.
- For more information, see Common service connection types.

Scope service connections

- Scope Azure Resource Manager, and other service connections, only to the resources and groups to which they need access. Service connections shouldn't have broad contributor rights on the entire Azure subscription.
- Don't give users generic or broad contributor rights on the Azure subscription.
- Don't use Azure Classic service connections, as there's no way to scope the permissions.
- Make sure the resource group only contains Virtual Machines (VMs) or resources that the build needs access to.
- Use a purpose-specific team service account to authenticate a service connection.
- For more information, see Common service connection types.

Choose the right authentication method

Select your authentication methods from the following sources:

- Consider service principals and managed identities
- Use personal access tokens (PATs) sparingly

Consider service principals

Explore alternatives like service principals and managed identities that enable you to use Microsoft Entra tokens to access Azure DevOps resources. Such tokens carry less risk when leaked compared to PATs and contain benefits like easy credential management.

Use PATs sparingly

If possible, we recommended to always use identity services for authentication instead of cryptographic keys since managing keys securely with application code is challenging and can lead to mistakes like accidentally publishing sensitive access keys to public code repositories like GitHub. However, if you must use personal access tokens (PATs), consider the following guidelines:

- PATs should always be scoped to specific roles.

- PATs shouldn't provide global access to multiple organizations.

- PATs shouldn't grant write or manage permissions on builds or releases.

- PATs should have an expiration date and be kept secret since they're as critical as passwords.

- PATs should never be hardcoded in the application code, even if it's tempting to do so to simplify the code.

- Administrators should regularly audit all PATs using the REST APIs and revoke any that don't meet the above criteria.

- Keep your PATs a secret. Your tokens are as critical as passwords.

- Store your tokens in a safe place.

- Don't hard code tokens in applications. It can be tempting to simplify code to obtain a token for a long period of time and store it in your application, but don't do that.

- Give tokens an expiration date.

- For more information, check out the following articles:
  - Manage PATs with policies - for administrators
  - Use PATs

Secure Azure Artifacts

- Make sure you understand the difference between feeds, project, and project collection administrators. For more information, see Configure Azure Artifacts settings.
- For more information, see Set feed permissions.

Secure Azure Boards

- Review Configure and customize Azure Boards before you customize a process.
- See the following articles:
  - Set work tracking and plan permissions
  - Default permissions and access to Azure Boards
  - Set query permissions

Secure Azure Pipelines

- Use extends templates.
- For more information about how to set permission levels for pipelines, see Set pipeline permissions.
- For more information about Azure Pipelines security best practices, see Securing Azure Pipelines.

Policies

- Require at least one reviewer outside of the original requester. The approver shares coownership of the changes and should be held equally accountable for any potential impact.
- Require CI build to pass. This requirement is useful for establishing baseline code quality, through code linting, unit tests, and security checks, like virus and credential scans.
- Ensure that the original pull requester can't approve the change.
- Disallow completion of a PR (Pull Request), even if some reviewers vote to wait or reject.
- Reset code reviewer votes when recent changes get pushed.
- Lock down release pipelines by running them only on specific production branches.
- Enable "Enforce settable at queue time for variables" in your organization's pipeline settings.
- Don't allow "Let users override this value when running this pipeline," for variables set in the editor.

Agents

- Grant permissions to the smallest possible number of accounts.
- Have the most restrictive firewall that leaves your agents usable.
- Update pools regularly to ensure the build fleet isn't running vulnerable code that a malicious actor can exploit.
- Use a separate agent pool for build artifacts that get shipped or deployed to production.
- Segment "sensitive" pool from nonsensitive pools, and only allow the use of credentials in build definitions that are locked to that pool.

Definitions

- Manage pipeline definitions with YAML (Yet Another Markup Language). YAML is the preferred method for managing pipeline definitions, as it provides traceability for changes and can follow approval guidelines.
- Secure the pipeline definition *Edit* access to the minimum number of accounts.

Input

- Include sanity checks for variables in build scripts. A sanity check can mitigate a command injection attack through the settable variables.
- Set as few build variables as possible to "Settable at release time."

Tasks

- Avoid remotely fetched resources, but, if necessary, use versioning and hash checking.
- Don't log secrets.
- Don't store secrets in pipeline variables, use Azure Key Vault. Regularly scan your build pipelines to ensure secrets aren't being stored in build pipeline variables.
- Don't let users run builds against arbitrary branches or tags on security-critical pipelines.
- Disable inheritance on the pipeline, as inherited permissions are broad and don't accurately reflect your needs for permissions.
- Limit job authorization scopes in all cases.

Repositories and branches

- Set the "Require a minimum number of reviewers," policy to *on*, so that every pull request gets reviewed by at least two approvers.
- Configure security policies specific to each repository or branch, instead of project wide. Security policies reduce risk, enforce change management standards, and improve your team's quality of code.
- Store production secrets in a separate Key Vault and ensure that access is only granted on a need-to-know basis to keep nonproduction builds separate.
- Don't mix test environments with production, including use of credentials.
- Disable forking. The more forks there are, the harder it's to keep track of each fork's security. Also, a user can easily fork a copy of a repository to their own private account.
- [Don't provide secrets to fork builds](#).
- [Consider manually triggering fork builds](#).
- [Use Microsoft-hosted agents for fork builds](#).

- For Git, check your production build definitions in the project's git repository, so they can be scanned for credentials.
- Configure a branch control check so that only pipelines running in the context of the `production` branch may use the `prod-connection`.
- For more information, see Other security considerations.

Secure Azure Repos

- Improve code quality with branch policies. For more information about branch permissions and policies, see Set branch permissions.

Secure Azure Test Plans

- Set permissions and access for testing
- Supported scenarios and access requirements

Secure GitHub Integrations

- Disable Personal Access Token (PAT)-based authentication, so the OAuth flow gets used with the GitHub service connection.
- Never authenticate GitHub service connections as an identity that's an administrator or owner of any repositories.
- Never use a full-scope GitHub PAT (Personal Access Token) to authenticate GitHub service connections.
- Don't use a personal GitHub account as a service connection with Azure DevOps.

Plan your organizational structure

Article • 10/16/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Use your business structure as a guide for the number of organizations, projects, and teams that you create in Azure DevOps. This article helps you plan for different structures and scenarios for Azure DevOps.

Consider the following structures for your business and collaborative work in Azure DevOps:

- Number of organizations
- Number of projects under an organization

You also may want to plan for the following scenarios:

- Map your organizations and projects in Azure DevOps to your enterprise, business unit, and team structure
- Structure your repositories (repos)
- Structure your teams- it can either help or hinder teams to be Agile and autonomous
- Manage access to data - who needs to have access and who doesn't?
- Reporting needs
- Promote common practices - use foundational elements to create an agile mindset and culture

Have at least one organization, which may represent your company, your larger collection of code projects, or even multiple related business units.

What's an organization?

An organization in Azure DevOps is a mechanism for organizing and connecting groups of related projects. Examples include business divisions, regional divisions, or other enterprise structures. You can choose one organization for your entire company, one organization for yourself, or separate organizations for specific business units.

Each organization gets its own *free tier* of services (up to five users for each service type) as follows. You can use all the services, or choose only what you need to complement your existing workflows.

- [Azure Pipelines]🗗: One hosted job with 1,800 minutes per month for CI/CD and one self-hosted job
- [Azure Boards]🗗: Work item tracking and Kanban boards
- [Azure Repos]🗗: Unlimited private Git repos
- [Azure Artifacts]🗗: Package management
- Unlimited Stakeholders
  - First five users free (Basic license)
  - **Azure Pipelines:**
    - One [Microsoft-hosted CI/CD] (one concurrent job, up to 30 hours per month)
    - One self-hosted CI/CD concurrent job
  - **Azure Boards:** Work item tracking and Kanban boards
  - **Azure Repos:** Unlimited private Git repos
  - **Azure Artifacts:** Two GiB free per organization

> ⓘ **Note**
>
> While Azure DevOps cloud-based load testing service is deprecated, **Azure Load Testing** is available. Azure Load Testing is a fully managed load testing service that enables you to use existing Apache JMeter scripts to generate high-scale load. To learn more, see **What is Azure Load Testing?**. To learn more about the deprecation of Azure DevOps load testing and other, alternative services see **Changes to load test functionality in Visual Studio and cloud load testing in Azure DevOps**.

How many organizations do you need?

Start with one organization in Azure DevOps. Then, you can add more organizations—which may require different security models—later. A single code repo or project only needs one organization. If you have separate teams that need to work on code or other projects in isolation, consider creating separate organizations for those teams. They'll have different URLs. Add projects, teams, and repos, as necessary, before you add another organization.

Take some time to review your work structure and the different business groups and participants to be managed. For more information, see [Map your projects to business units] and [Structure considerations].

> 💡 **Tip**
>
> For company-owned Microsoft Entra organizations, consider restricting users from creating new organizations as a way to protect your IP. For more information, see

What's a team?

A team is a unit that supports many team-configurable tools. These tools help you plan and manage work, and make collaboration easier.

Create a team for each distinct product or feature team
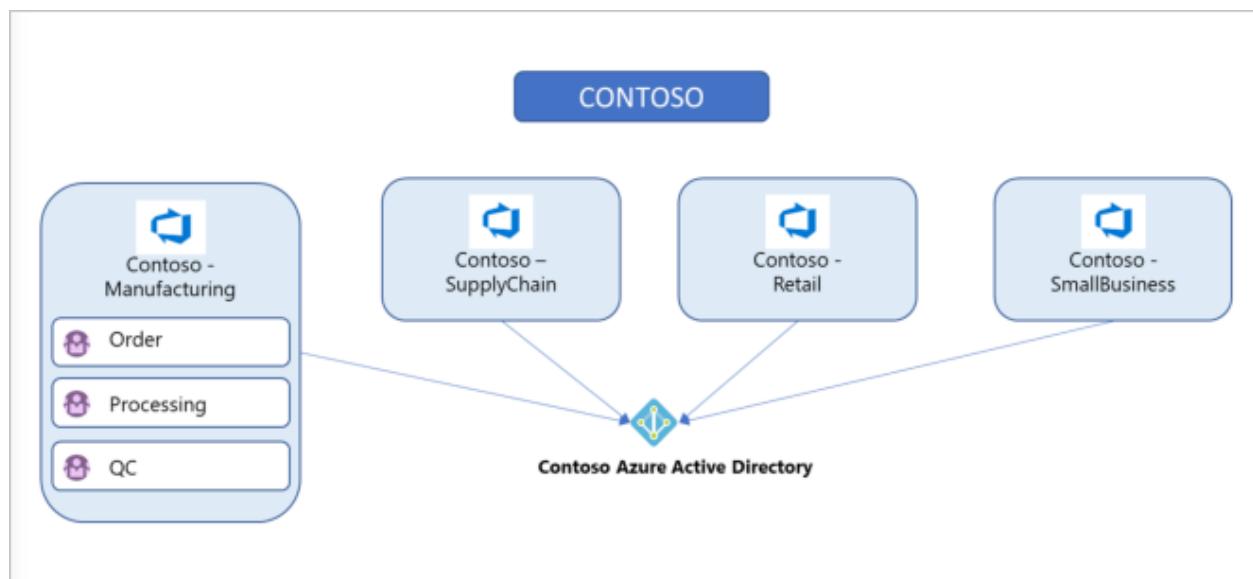
Each team owns their own backlog. To create a new backlog, you create a new team. Configure teams and backlogs into a hierarchical structure, so program owners can more easily track progress across teams, manage portfolios, and generate rollup data. A team group gets created when you create a team. You can use this group in queries or to set permissions for your team.

What's a project?

A project in Azure DevOps contains the following set of features:

- Boards and backlogs for agile planning
- Pipelines for continuous integration and deployment
- Repos for version control and management of source code and artifacts
- Continuous test integration throughout the project life cycle Each organization contains one or more projects

In the following image, the fictitious Contoso company has four projects within their Contoso-Manufacturing organization.

How many projects do you need?

Have at least one project to start using an Azure DevOps service, such as Azure Boards, Azure Repos, or Azure Pipelines. When you create your organization, a default project gets created for you. In your default project, there's a code repo to start working in, backlog to track work, and at least one pipeline to begin automating build and release.

Within an organization, you can do either of the following approaches:

- Create a single project that contains many repos and teams
- Create many projects, each with its own set of teams, repos, builds, work items, and other elements

Even if you have many teams working on hundreds of different applications and software projects, you can manage them within a single project in Azure DevOps. However, if you want to manage more granular security between your software projects and their teams, consider using many projects. At the highest level of isolation is an organization, where each organization is connected to a single Microsoft Entra tenant. A single Microsoft Entra tenant, however, can be connected to many Azure DevOps organizations.

> ⓘ **Note**
>
> If the **Limit user visibility and collaboration to specific projects** preview feature is enabled for the organization, users added to the **Project-Scoped Users** group won't be able to access projects that they haven't been added to. For more information and important security-related call-outs, see **Manage your organization, Limit user visibility for projects and more**.

Single project

A single project puts all of the work at the same "portfolio" level for the entire organization. Your work has the same set of repos and iteration paths. With a single project, teams share source repos, build definitions, release definitions, reports, and package feeds. You might have a large product or service that's managed by many teams. Those teams have tight inter-dependencies across the product life cycle. You create a project and divide the work using teams and area paths. This setup gives your teams visibility into each other's work, so the organization stays aligned. Your teams use the same taxonomy for work item tracking, making it easier to communicate and stay consistent.

A high volume of queries and boards can make it hard to find what you're looking for. Depending on the architecture of your product, this difficulty can bleed into other areas such as builds, releases, and repos. Make sure to use good naming conventions and a simple folder structure. When you add a repo to your project, consider your strategy and determine whether that repo could be placed into its own project.

Many projects

You can best determine project structure by how you ship the product. Having several projects shifts the administration burden and gives your teams more autonomy to manage the project as the team decides. It also provides greater control of security and access to assets across the different projects. Having team independence with many projects creates some alignment challenges, however. If each project is using a different process or iteration schedule, it can make communication and collaboration difficult if the taxonomies aren't the same.

Azure DevOps provides cross-project experiences for managing work.

You may want to add another project due to the following scenarios:

- To prohibit or manage access to the information within a project
- To support custom work tracking processes for specific business units within your organization
- To support entirely separate business units that have their own administrative policies and administrators
- To support testing customization activities or adding extensions before rolling out changes to the working project

When you're considering many projects, keep in mind that Git repo portability makes it easy to migrate repos (including full history) between projects. Other history can't be migrated between projects. Examples are push and pull request history.

When you map projects to business units, your company gets a single organization and sets up many projects with one or more projects representing a business unit. All Azure DevOps assets of the company are contained within this organization and located within a given region (for example, Western Europe). Consider the following guidance for mapping your projects to business units:

⛶ **Expand table**

	One project, many teams	One organization, many projects, and teams	Many organizations
General guidance	Best for smaller organizations or larger organizations with highly aligned teams.	Good when different efforts require different processes.	Useful as part of TFS legacy migrations and for hard security boundaries between organizations. Used with multiple projects and teams within each organization.
Scale	Supports tens of thousands of users and hundreds of teams, but best at this scale if all teams are working on related efforts.	Same as with one project, but many projects may be easier.	
Process	Aligned processes across teams; team flexibility to customize boards, dashboards, and so on.	Independent processes for each project. For example, different work item types, custom fields, and so on.	Same as many projects.
Collaboration	Highest default visibility and reuse between work and assets of different teams.	Good visibility and reuse are possible, but it's easier to hide assets between projects whether intentional.	Poor visibility, collaboration, and reuse between organizations.
Roll-up reporting and portfolio	Best ability to roll up across teams and	Good reporting possible across	No roll-up or coordination between

	One project, many teams	One organization, many projects, and teams	Many organizations
management	coordinate between teams.	projects. More difficult for cross-project roll-up and team coordination.	organizations.
Security/isolation	Can lock down assets at a team level, but default is open visibility and collaboration.	Better ability to lock down between projects. By default, provides good visibility within projects and good isolation across projects.	Hard boundaries across organizations; excellent isolation and minimal ability to share across organizations.
Context switching	Easiest for teams to work together and for users to switch between efforts.	Relatively easy for users to work together and switch contexts between efforts.	More difficult for users having to work across different organizations.
Information overload	By default, all assets are visible to users who make use of "favorites" and similar mechanisms to avoid "information overload."	Reduced risk of information overload; most project assets hidden across project boundaries.	Assets across organizations are isolated, reducing risk of information overload.
Administrative overhead	Much administration is delegated down to individual teams. Easiest for user licensing and org-level administration. More work may be needed if alignment is required between efforts.	More administration at the project level. More overhead, but can be useful when projects have different administrative needs.	As with more projects, there's more administrative overhead, which enables more flexibility between orgs.

Structure repos and version control within a project

Consider the specific strategic work scoped to one of the organizations you created previously and who needs access. Use this information to name and create a project.

This project has a URL defined under the organization you created it in and can be accessed at https://dev.azure.com/{organization-name}/{project-name}.

Configure your project in **Project settings**.

For more information about managing projects, see Manage projects in Azure DevOps. You can move a project to a different organization by migrating the data. For more information about migrating your project, see Migration options.

Manage version control

In projects where the Azure Repos service is enabled, version control repos can store and revise code. Consider the following options when you're configuring repos.

Git vs. Team Foundation Version Control (TFVC)

Azure Repos offers the following version control systems for teams to choose from:

- Git and TFVC. Projects can have repos of each type. By default, new projects have an empty Git repo. Git enables a great amount of flexibility in developer workflows and integrates with nearly every relevant tool in the developer ecosystem. Any project can use Git repos. There's no limit on the amount of Git repos that can be added to a project.

TFVC is a centralized version control system that is also available. Unlike Git, only one TFVC repository is allowed for a project. But, within that repo, folders, and branches are used to organize code for multiple products and services, if wanted. Projects can use both TFVC and Git, if appropriate.

One vs. many repos

Do you need to set up multiple repos within a single project or have a repo set up per project? The following guidance relates to the planning and administration functions across those repos.

One project containing multiple repos works well if the products/services are working on a coordinated release schedule. If developers are frequently working with multiple repos, keep them in a single project to ensure the processes remain shared and consistent. It's easier to manage repo access within a single project, as access controls and options like case enforcement and max file size get set at the project level. You can manage the access controls and settings individually, even if your repos are in a single project.

If the products stored in multiple repos work on independent schedules or processes, you can split them into multiple projects. Git repo portability makes it easy to move a repo between projects and still keep full-fidelity commit history. Other history, such as pull requests or build history, aren't easily migrated.

Base your decision for one vs. many repos on the following factors and tips:

- code dependencies and architecture
- put each independently deploy-able product or service in its own repo
- don't separate a codebase into many repos if you expect to make coordinated code changes across those repos, as no tools can help coordinate those changes
- if your codebase is already a monolith, keep it in one repo. For more information about monolithic repos, see How Microsoft develops modern software with DevOps articles
- if you have many disconnected services, one repo per service is a good strategy
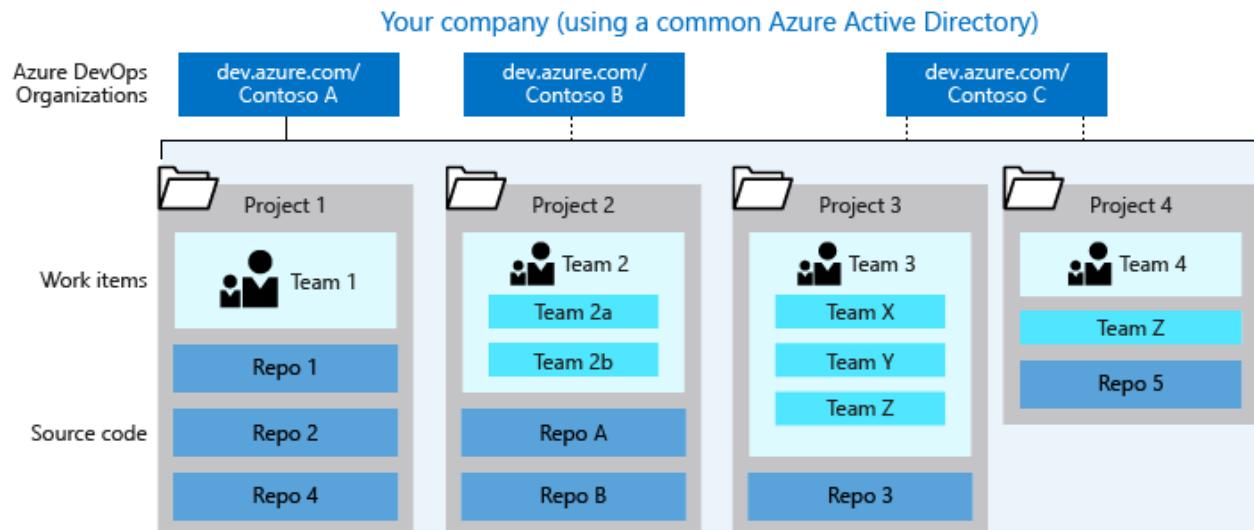
Shared repo vs. forked repos

We recommend using a shared repo within a trusted organization. Developers use branches to maintain isolation of their changes from one another. With a good branching and release strategy, a single repo can scale to support concurrent development for more than a thousand developers. For more information about branching and release strategy, see Adopt a Git branching strategy and Release Flow: Our Branching Strategy.

Forks can be useful when you're working with vendor teams that shouldn't have direct access to update the main repository. Forks can also be useful in scenarios where many developers contribute infrequently, such as in an open-source project. When you're working with forks, you may want to maintain a separate project to isolate the forked repos from the main repo. There may be added administrative overhead, but it keeps the main project cleaner. For more information, see the Forks article.

The following image displays a sample of how "your company" could structure its organizations, projects, work items, teams, and repos.
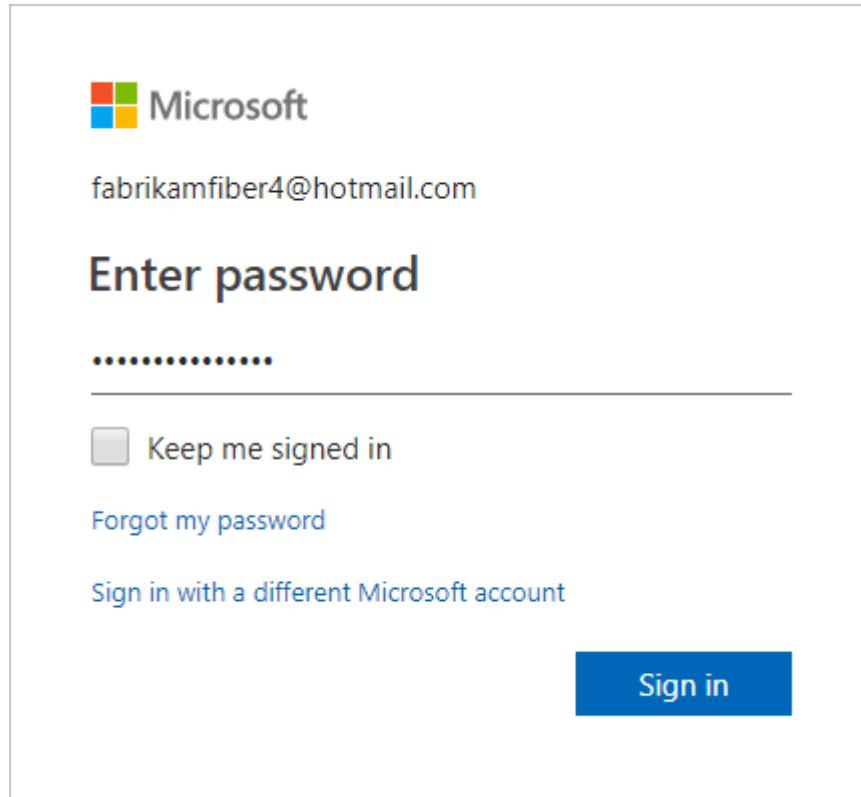
More about organizational structure

Choose your organization administrator account type

When you create an organization, the credentials that you sign in with define which identity provider your organization uses. Create your organization with a Microsoft account or Microsoft Entra instance. Use those credentials to sign in as an administrator to your new organization at `https://dev.azure.com/{YourOrganization}`.

Use your Microsoft account

Use your Microsoft account if you don't need to authenticate users for an organization with Microsoft Entra ID. All users must sign in to your organization with a Microsoft account. If you don't have one, create a Microsoft account ⧉.

If you don't have a Microsoft Entra instance, create one for free from the Azure portal ⧉ or use your Microsoft account to create an organization. Then, you can connect your organization to Microsoft Entra ID.

Use your Microsoft Entra account

You might have a Microsoft Entra account already if you use Azure or Microsoft 365. If you work for a company that uses Microsoft Entra ID to manage user permissions, you probably have a Microsoft Entra account.

If you don't have a Microsoft Entra account, sign up for Microsoft Entra ID ⧉ to automatically connect your organization to your Microsoft Entra ID. All users must be members in that directory to access your organization. To add users from other organizations, use Microsoft Entra B2B collaboration.

Azure DevOps authenticates users through your Microsoft Entra ID, so that only users who are members in that directory have access to your organization. When you remove users from that directory, they can no longer access your organization. Only specific Microsoft Entra administrators manage users in your directory, so administrators control who accesses your organization.

For more information on managing users, see Manage users.

Map organizations to business units

Each business unit within your company gets its own organization in Azure DevOps, along with its own Microsoft Entra tenant. You can set up projects within those individual organizations, as required, based on teams or ongoing work.

For a larger company, you can create multiple organizations using different user accounts (most likely Microsoft Entra accounts). Consider what groups and users share strategies and work, and group them into specific organizations.

For example, the fictional Fabrikam company created the following three organizations:

- Fabrikam-Marketing
- Fabrikam-Engineering
- Fabrikam-Sales

Each organization has a separate URL, such as:

- https://dev.azure.com/Fabrikam-Marketing
- https://dev.azure.com/Fabrikam-Engineering
- https://dev.azure.com/Fabrikam-Sales

The organizations are for the same company, but are mostly isolated from each other. You don't need to separate anything this way. Only create boundaries when it makes sense to your business.

> 💡 **Tip**
>
> You can more easily partition an existing organization with projects, than combine different organizations.

Related articles

- Create an organization

- Create a project
- Connect your organization to Microsoft Entra ID
- Set up billing
- Set user preferences

- Create a project
- Connect your organization to Microsoft Entra ID
- Set up billing
- Set user preferences

What is source control?

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

A source control system, also called a *version control* system, allows developers to collaborate on code and track changes. Source control is an essential tool for multi-developer projects.

Our systems support two types of source control: Git (distributed) and Team Foundation Version Control (TFVC). TFVC is a centralized, client-server system. In both Git and TFVC, you can check in files and organize files in folders, branches, and repositories.

Manage your repos, branches, and other code development operations from **Azure Repos**.

With Git, each developer has a copy of the source repository on their dev machine. The source repo includes all branch and history information. Each developer works directly with their local repository. Changes are shared between repositories as a separate step.

Developers can commit each set of changes and perform version control operations, such as history and compare without a network connection. Branches are lightweight. When developers need to switch contexts, they create a private local branch. Developers can quickly switch from one branch to another to pivot among different variations of the code base. Later, developers can merge, publish, or dispose of the branch.

> ⓘ **Note**
>
> Git in Visual Studio and Azure DevOps is standard Git. You can use Visual Studio with third-party Git services. You can also use third-party Git clients with Azure DevOps Server.

With TFVC, developers have only one version of each file on their dev machines. Historical data is maintained only on the server. Branches are path-based and are created on the server.

Next steps

Start sharing your code or get your code by using source control.

Code with Git

Related articles

- Azure Repos documentation
- Git repositories documentation

Tools and clients that connect to Azure DevOps

Article • 01/18/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Our platform of software development tools began more than 20 years ago. We released Visual Basic and Visual Studio as an integrated development environment (IDE). Visual Studio supports many plug-ins that extend its functionality. In particular, the Team Explorer plug-in allows the Visual Studio client to connect to Azure DevOps to support source control, work tracking, build, and test operations.

Desktop client developer tools

Developers have access to many tools through these versions of Visual Studio and plug-ins. To download any version of Visual Studio, go to the Visual Studio Downloads page ⧉ . To understand what features you get with the Visual Studio versions, see Compare Visual Studio offerings ⧉ .

- **Visual Studio Community**: A fully featured and extensible IDE for creating modern applications for Android, iOS, and Windows, including web applications and cloud services. (Replaces Visual Studio Express.)
- **Visual Studio Professional**: Development tools and services to support individual developers or small teams.
- **Visual Studio Enterprise**: Integrated, end-to-end development tools and solutions for teams of any size, and with a need to scale. It supports designing, building, and managing complex enterprise applications.
- **Visual Studio Test Professional**: Provides access to Microsoft Test and development tools to support quality and collaboration throughout the development process.
- Visual Studio Code ⧉ : Free, open-source code editor with a free extension to support connecting to Git repositories on Azure DevOps.
- Android Studio with the Azure DevOps Services Plug-in for Android Studio: Free plug in to support Android developers and connect to Git repositories on Azure DevOps.
- IntelliJ with the Azure DevOps Services Plugin for IntelliJ: Free plug in to support developers who use IntelliJ IDEA or Android Studio to connect to Git repositories on Azure DevOps.

To get started with client libraries, see Client library samples.

Team Explorer plug-in

Team Explorer, a plug-in to all Visual Studio versions, connects Visual Studio to projects defined in Azure DevOps. You can manage source code, work items, and builds. To learn more, see Work in Team Explorer.

Home page with Git	Home page with TFVC

Visual Studio Git experience

Visual Studio 2019 and later versions provide a new Git experience through the **Git** menu as shown below. To learn more, see Git experience in Visual Studio and Side-by-side comparison of Git and Team Explorer.

Office integration tools

You can integrate the following Microsoft Office tools with Azure DevOps.

- Excel: Use Excel to add and bulk modify work items.

> ⓘ **Important**
>
> Starting with Visual Studio 2019, the Team Foundation plug-in for Office is deprecating support for Microsoft Project. Project integration and the TFSFieldMapping command is not supported for Azure DevOps Server 2019 nor for Azure DevOps Services. However, you can continue to use Microsoft Excel.

Task-specific clients

The following clients support specific tasks, such as managing testing efforts, providing feedback, or modifying work items:

- Azure Test Plans: Manage your test efforts, create and run manual tests, and create and track bugs that are found during test efforts.
- Test & Feedback extension (previously called the Exploratory Testing extension): This extension provides a lightweight plug-in to a web browser. Stakeholders can

respond to feedback requests for user stories and features created in Azure DevOps. This extension is free to Stakeholders.

- **Microsoft Feedback Client**: Your Stakeholders can use this client to record feedback for your application as video, audio, or type-written comments. This client is installed with all versions of Visual Studio, or it can be installed from the free download ⬀ . All feedback is stored in the work item data store and requires Stakeholders to have permissions.

Browser-based web tools

Web portal

The collaboration tools supported through the web portal are summarized under Essential services. New features are deployed every three weeks for Azure DevOps Services, and quarterly for Azure DevOps Server. For release notes, see Azure DevOps Services Features Timeline.

You can use the following browsers to access the web portal:

Version	Edge	Internet Explorer	Safari (Mac)	Firefox	Chrome
Azure DevOps Services Azure DevOps Server 2020.1	Most recent	Not supported	14.1 and later	Most recent	Most recent
Azure DevOps Server 2020 Azure DevOps Server 2019 TFS 2018 TFS 2017	Most recent	11 and later	14.1 and later	Most recent	Most recent
TFS 2015	Most recent	9 and later	5 and later	Most recent	Most recent
TFS 2013		9 and later	5 and later	Most recent	Most recent

Microsoft Edge, Firefox, and Chrome automatically update themselves, so Azure DevOps supports the most recent version.

For more information, see Web portal navigation.

Browser-based extensions

Several extensions are built and maintained by the Azure DevOps Services product team:

- Code search ↗: Increase cross-team collaboration and code sharing. Enables developers to quickly locate relevant information within the code base of all projects that are hosted within an organization or collection. You can discover implementation examples, browsing definitions, and error text.
- Work item search ↗: To quickly find relevant work items, search across all work item fields over all projects in an organization. Do full-text searches across all fields to efficiently locate relevant work items. Use inline search filters, on any work item field, to quickly narrow down a list of work items.

Find more extensions in Azure DevOps **Organization settings** > **Extensions** > **Browse marketplace**. See also, Overview of extensions for Azure Boards.

Command-line tools

You can do many code development and administrative tasks by using the following command-line tools:

- az devops commands
- Git commands
- TFVC commands
- TCM commands
- Manage permissions with command line tool (az devops security)
- witadmin (work item tracking)

Integrated tool support for third-party applications

The following tools provide support for monitoring and interacting with Azure DevOps from a third-party application.

- **Azure Boards**:
  - Use the Azure Boards app with Slack to manage work items
  - Use the Azure Boards app in Microsoft Teams

- **Azure Repos**:
  - Azure Repos with Slack

- Azure Repos with Microsoft Teams

- **Azure Pipelines**:
  - Use Azure Pipelines with Microsoft Teams
  - Azure Pipelines with Slack
  - Integrate with ServiceNow change management
  - Continuously deploy from a Jenkins build

Marketplace extensions

Visual Studio and Azure DevOps provide a wealth of features and functionality. They also provide a means to extend and share that functionality.

Extensions are simple add-ons that you can use to customize and extend your DevOps and work tracking experiences. They're written with standard technologies—HTML, JavaScript, and CSS. You can develop your own extensions by using your preferred dev tools.

You build extensions by using our RESTful API library. Publish your extensions to the Azure DevOps Marketplace. You can privately maintain or share them with millions of developers who use Visual Studio and Azure DevOps.

To learn more, visit the Azure DevOps Marketplace ⧉ and see Overview of extensions.

REST APIs

The Azure DevOps APIs are based on REST, OAuth, JSON, and service hooks—all standard web technologies broadly supported in the industry.

REST APIs are provided to support building extensions to Azure DevOps. To learn more, see REST API overview.

Related articles

- A tour of services
- Software development roles
- Pricing ⧉
- Azure DevOps data protection overview

Software development roles supported by Azure DevOps

Article • 10/04/2022

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

If you're a sole developer or work in a small setting, you track issues, plan features, code, test, build, and deploy.

If you work in a large setting, you might be more focused on a specific set of tasks that aligns with specific roles. These specific roles could be software development, product and scrum management, or DevOps.

The following article describes the features and tasks available to you, based on your role.

Contributor roles

Team members are contributors who have access to the following areas and more:

- code base
- work item tracking
- Agile tools
- build pipelines
- test tools

If you need to lock down specific areas to a select set of contributors, see permission management.

Software developers

Developers use Visual Studio or other tools to develop their applications. They then check in their changes to a Git or Team Foundation Version Control (TFVC) repository hosted in Azure DevOps. From the web portal or a supported IDE, they can view repositories, check history, and more.

To get started with using Git, see one of the following resources:

- Share your code with Git and Visual Studio
- Share your code in Git by using Eclipse

- [Share your code in Git by using Xcode](#)
- [Share your code in Git by using IntelliJ](#)
- [Get started with using Git and Azure DevOps Services](#)

To get started with using TFVC, see one of the following resources:

- [Develop and share your code in TFVC by using Visual Studio](#)
- [Share your code in TFVC by using Eclipse](#)
- [Share your code in TFVC by using Xcode](#)

Product owners

Product owners typically plan the feature set to deliver, set priorities, and track the status of work, code defects, and customer issues. The suite of web-based Agile tools in Azure DevOps provides product owners with the views and features that they need to do these tasks. All work gets captured within a work item. Each work item represents a specific type such as a user story, task, or bug.

- Use the product backlog to quickly define and prioritize user stories, features, and other work items
- Use the sprint backlog and task board to implement Scrum practices
- Use the Kanban board to work with Kanban methods
- Use queries to list and update work items, create status and trend charts, and post charts to dashboards
- Use dashboards to share information, status, and trends with your team or organization

For more information about getting started, see [About Azure Boards and Agile tools](#).

You can integrate Microsoft Excel with Azure DevOps to plan and track your work. For more information, see [Bulk modify by using Excel](#).

Scrum masters

Scrum masters help to facilitate scrum to the larger team by ensuring the scrum framework gets followed. They're committed to the practices, but stay flexible and open to opportunities for the team to improve their workflow. Scrum masters utilize the same features as [product owners](#).

DevOps: builders, testers, and release managers

An advantage of working with Azure DevOps is the suite of tools and integrated functionality that support build, testing, and deploying software applications. See the following general DevOps-associated tasks that Azure DevOps supports.

- Define builds
- Unit test your code
- Run tests with your builds
- Perform exploratory tests
- Define, manage, track, and approve releases
- Deploy applications to Azure, a virtual machine, Docker containers, and more

To get started, see the overviews in Azure Pipelines and Azure Test Plans.

Stakeholders

With Stakeholder access, anyone in your organization can check project status and provide feedback. Stakeholders can track project priorities and provide direction, feature ideas, and business alignment to a team. Stakeholders also contribute to plans by adding and modifying work items. They can't, however, contribute to the code base or exercise test tools.

Stakeholder access essentially provides free access to a limited set of feature to project sponsors and supporters. To learn more, see Work as a Stakeholder.

Administrator roles

A distinct advantage to working in Azure DevOps Services is the reduced overhead of server maintenance. But there are several administrative tasks required to support a collaborative, integrated software development environment.

The main tasks are grouped as follows by membership in a security group or role.

Team administrators

Responsible for configuring team settings, which include:

- Backlog and board settings
- Team areas and iterations (sprints)
- Team members
- Team dashboards
- Team work item templates
- Team alerts

To get started, see Manage teams and configure team tools.

Project administrators

Responsible for configuring project-level resources, including:

- Area paths and iteration paths
- Project permissions and repository security
- Build agents, pools, and service connections
- Test and release retention policies

Organization owners and Project Collection Administrators

Organization owners are automatically members of the Project Collection Administrators group. Responsible for configuring organization-level resources, including the following tasks:

- Manage billing
- Add and manage projects
- Manage collection-level permissions
- Customize work tracking processes
- Install and manage extensions

To get started, see Manage organizations and Settings.

Related articles

- A tour of services
- Plan your organizational structure in Azure DevOps

Troubleshoot connecting to a project

Article • 11/22/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Troubleshoot connectivity
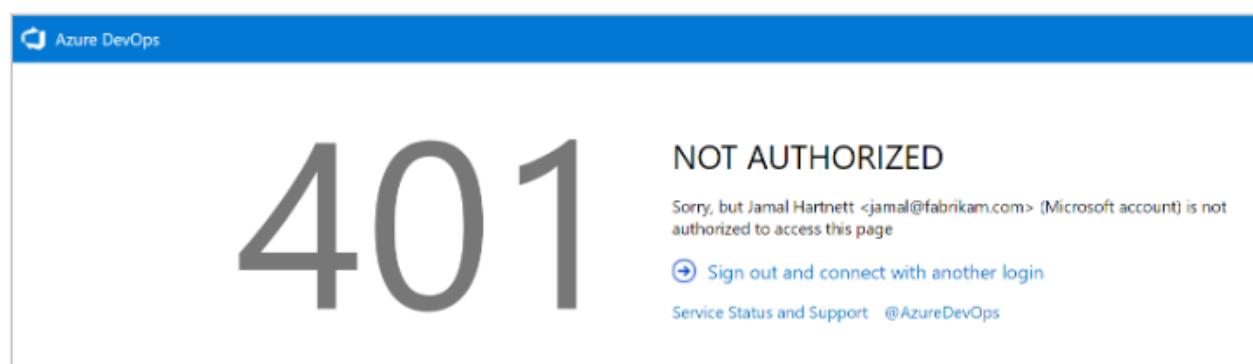
As a first step in resolving connectivity issues with Azure DevOps, complete the following steps:

1. Sign out of your browser. To do so, select the [Visual Studio sign out ↗](#) link.

2. Delete the cookies in your browser. To delete cookies in most browsers, select Ctrl+Shift+Del.

3. Open Internet Explorer and delete the browser cookies. The Visual Studio IDE uses Internet Explorer cookies.

4. Close all browsers and close the Visual Studio IDE.

5. Use a private browser session to retry the connection. If the issue is with the Visual Studio IDE, remove the connection, and then readd it in Team Explorer.

Troubleshoot signing in

Two types of identities can sign in: Microsoft accounts and Microsoft Entra accounts. Depending on your account, you might experience one of the following errors.

> 401 - Not Authorized

The most common error page is the *401 Not Authorized* error, which occurs when your identity doesn't have permissions to enter an organization. See the following common

reasons for the error:
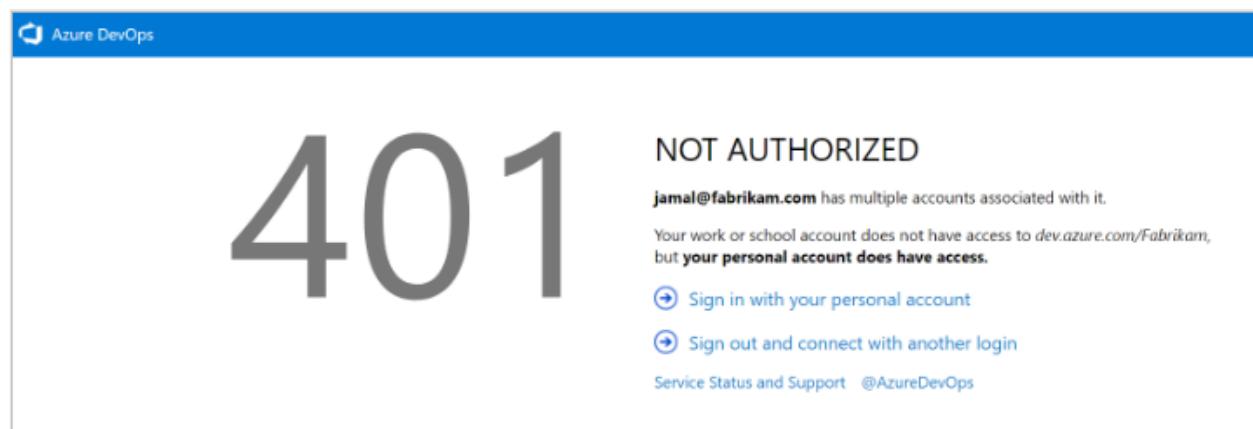
- Your identity isn't a member of the organization.
- Your identity has an invalid or missing license assignment.
- Your identity doesn't have enough memberships to access the resource. For example, membership to the Reader/Contributors group.
- Your identity is a B2B guest in the tenant, and the invitation isn't accepted.

If you think you're a member of the organization, but get this error page, contact Support ☐ .

Scenario 1

Your work or school Microsoft Entra account doesn't have access, but your personal Microsoft account does.

> 401 - Work or school, or Personal account

A highly specific 401 error case. In this case, both a personal Microsoft account and a work or school account (Microsoft Entra ID) that have the same sign-in address exist. You signed in with your work or school account, but your personal account is the identity with access to the organization.

Mitigation

In some cases, you might not know you have two identities with the same sign-in address. It's possible that an administrator created the work or school Microsoft Entra account when you were added to Office 365 or Microsoft Entra ID.

To sign out of your current work or school Microsoft Entra account, select **Sign in with your personal MSA account**, and then sign in by using your personal Microsoft account. After authentication, you should have access to the organization.

- If you can't access to the organization, make sure that your Microsoft Entra ID still exists and that your work or school account is in the Microsoft Entra tenant.
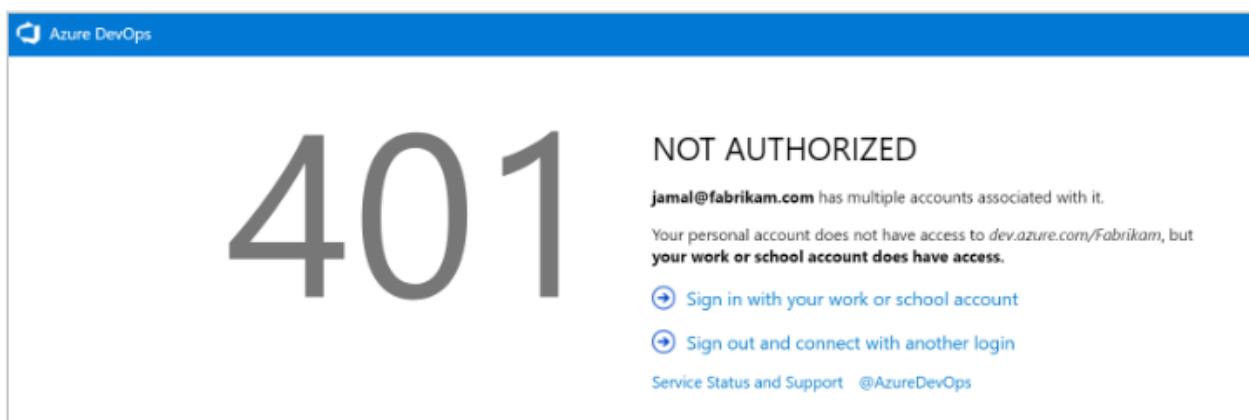
> 💡 **Tip**
>
> To avoid seeing this prompt, you can rename your Microsoft account. Then, only one identity, your work or school account, or Microsoft Entra account, uses your sign-in address.

Scenario 2

Your personal Microsoft account doesn't have access, but your Microsoft Entra account does. This scenario is an opposite version of the 401 error page. In this case, the personal account (Microsoft account identity) doesn't have access to the organization and the work or school account (Microsoft Entra identity) does. The same guidance from Scenario 1 applies, but in reverse.

> 401 - Work or school, or Personal account

Mitigation

When you get redirected back to the original sign-in page, we recommend that you clear all cookies, and then reattempt to sign in. If that doesn't fix the issue, contact Support🔗.

Switch organizations

When you use two or more organizations that are linked to Microsoft Entra ID, the sign-out function might not work as expected. For example, you can't switch between

different organizations to connect to multiple organizations that are linked to directory tenants.

When this problem occurs, a blank screen flashes several times. Then, one of the following error messages appears after you connect to or add a new connection in the **Connect to Azure DevOps Server** dialog box:

> TF31003: Either you have not entered the necessary credentials, or your user account does not have permission to connect to the Azure DevOps Server
>
> TF31002: Unable to connect to this Azure DevOps Server

To resolve this issue, apply Visual Studio 2013.2 or install a later version from the Visual Studio download website ⧉ .

Another solution is to delete your browser cookies. For more information, see the support article You can't switch between different organizations in Visual Studio Codespaces ⧉ .

TF31002: Unable to connect

Article • 02/24/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

You might receive this error when you try to connect to Azure DevOps Services or an on-premises Azure DevOps Server from Visual Studio.

You receive this error when you try to connect to Azure DevOps Services

Problem	Resolution
You don't have an active account or license.	Check with your administrator that you're a member of the account and have an active, valid license. See Assign licenses to users for details.
Your Azure DevOps Services organization is connected to the Azure Active Directory.	When your Azure DevOps Services organization is connected to a directory that is associated with a Microsoft 365 or Microsoft Azure subscription, only members in the directory can access the account. Check with your directory administrator to have them create an organizational account for you or add your account to the directory as external member.
You can't switch between different organizational accounts.	If you work with several organizations that connect to different directories, such as accounts created from the Microsoft Azure Portal, the sign-out function might not work as expected. For example, you can't switch between different organizational accounts to connect to multiple accounts that are linked to directory tenants. When this problem occurs, you see a flashing blank sign in dialog box several times. Then, you receive either TF31002 or TF31003 error after you connect to or add a new connection in "Connect to Team Foundation Server" dialog box. To resolve this problem, apply the most recent Visual Studio update ⧉ . To learn more, see You can't switch between different organizational accounts in Visual Studio Online.

Problem	Resolution
You want to sign in to Azure DevOps Services from Visual Studio using different credentials.	See Connect to projects, Sign in with different credentials.

When you try to connect to an on-premises Azure DevOps Server from your client computer

If you determine that you're receiving this error from one computer but not others, or others aren't receiving this error, then check the problem resolutions that are outlined below.

Problem	Resolution
Your password has expired.	Verify that you entered your user ID and password correctly, and that your password hasn't expired.
You've entered an incorrect server URL.	Verify that you've entered the server URL correctly including the server name, port number, and protocol (http/https). See Connect to projects to learn more.
The TFS configuration has changed.	If the configuration for the on-premises Azure DevOps Server has changed, you must create a new connection. You might also need to clear the client cache.
You work remotely and need to connect to a TFS Proxy server to check in files to Team Foundation version control.	Configure Visual Studio to connect to TFS Proxy.
You're connecting to a later version of TFS than your Visual Studio client version.	Your version of Visual Studio or Team Explorer might be incompatible with Team Foundation Server. You might need to install one or more GDR packs. See Requirements and compatibility for details.
Your firewall is blocking TFS services.	See Allow a program to communicate through Windows Firewall.

Problem	Resolution
Visual Studio stops responding when you run a query in Visual Studio.	Your computer might be configured to bypass the proxy server. Verify the configuration of the BypassProxyOnLocal setting on your computer. For more information, see BypassProxyOnLocal Configuration.

Several users can't connect to an on-premises Azure DevOps Server

If the problem occurs on more than one computer, contact your administrator to confirm whether the server is operational and available on the network.

As an administrator, check the event logs for the application-tier server to try to pinpoint the problem. Also, you can use the following table to determine whether the server is misconfigured. In the table, problems that are more likely to occur appear first. Try the resolutions in the order in which they appear, which increases the chance that you can solve the problem quickly.

Problem	Resolution
The *TFSService* account password has expired or is incorrect.	Many services for Team Foundation Server will stop running when the service account for Team Foundation has expired. For more information, see Change the service account or password for Team Foundation Server.
The application-tier server for Team Foundation is unavailable.	Verify whether each required service is running. If a required service isn't running, you must restart it. If necessary, set it to start automatically. For more information, see Stop and start services, application pools, and websites.
The network is unavailable.	Verify whether your network is operational.
A website identity for Team Foundation is configured incorrectly.	Verify or correct the server binding assignments that are made to websites for Team Foundation.
Access to a website for Team Foundation has been restricted.	Verify or correct restrictions that are made to those websites that are based on IP addresses and domain names.

Problem	Resolution
The firewall or ports are configured incorrectly.	Verify or correct port binding assignments for websites and port assignments for the firewall. First, you should open the administration console for Team Foundation, display the Application Tier page, and review the URL assignments. If necessary, you can click **Change URL** to modify the URL of a website. Next, you should verify the port assignments for Internet Information Services (IIS) and the ports that are allowed through the firewall. For more information, see Review Server Status and Settings and Verify or Correct Port Assignments.
Trust relationships between domains aren't configured correctly.	If a group of users can't access Team Foundation Server, you might have trust issues between domains.
When users connect to different versions of TFS from Visual Studio, for example, they connect to TFS 2012 and then TFS 2008, they can get the TF31002 error.	This error can occur because the GUIDs for the TFS 2012 collection are the same as TFS 2008. The local client cache gets confused because it tries to maintain the same GUID-based local cache for both the 2008 server and the new Project Collection in 2012. To fix, run the **TFSConfig ChangeServerID** command. See TFSConfig ChangeServerID command.

Troubleshoot access and permission issues

Article • 03/23/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Due to the extensive security and permission structure of Azure DevOps, you might investigate why a user doesn't have access to a project, service, or feature that they expect. Find step-by-step guidance to understand and address problems a project member may be having in connecting to a project or accessing an Azure DevOps service or feature.

Before using this guide, we recommend that you're familiar with the following content:

- Get started with permissions, access, and security groups
- Default permissions and access quick reference.
- Quick reference index to Azure DevOps security

> 💡 **Tip**
>
> When you're creating an Azure DevOps security group, label it in a way that is easy to discern if it's created to limit access.

Permissions get set at one of the following levels:

- object level
- project level
- organization or project collection level
- security role
- team administrator role

Common access and permission issues

See the following most common reasons a project member can't access a project, service, or feature:

Issue	Troubleshooting action

Issue	Troubleshooting action
Their access level doesn't support access to the service or feature.	To determine whether it's the cause, determine the user's access level and subscription status.
Their membership within a security group doesn't support access to a feature or they have been explicitly denied permission to a feature.	To determine whether it's the cause, trace a permission.
The user has been recently granted permission, however a refresh is required for their client to recognize the changes.	Have the user refresh or reevaluate their permissions.
The user's trying to exercise a feature granted only to a team administrator for a specific team, however they haven't been granted that role.	To add them to the role, see Add, remove team administrator.
The user hasn't enabled a preview feature.	Have the user open the Preview features and determine the on/off status for the specific feature. For more information, see Manage preview features.
Project member has been added to a limited scope security group, such as the Project-Scoped Users group.	To determine whether it's the cause, look up the user's security group memberships.

Less common access and permission issues

Less common reasons for limited access are when one of the following events has occurred:

Issue	Troubleshooting action
A project administrator disabled a service. In this case, no one has access to the disabled service.	To determine whether a service is disabled, see Turn an Azure DevOps service on or off.
A Project Collection Administrator disabled a preview feature, which disables it for all project members in the organization.	See Manage preview features.
Group rules governing the user's access level or project membership are restricting access.	See Determine a user's access level and subscription status.
Custom rules have been defined to a work item type's workflow.	see Rules applied to a work item type that restrict select operation.

Determine a user's access level and subscription status

You can assign users or groups of users to one of the following access levels:

- Stakeholder
- Basic
- Basic + Test Plans
- Visual Studio subscription

For more information about access level restriction in Azure DevOps, see Supported access levels.

To use Azure DevOps features, users must be added to a security group with the appropriate permissions. Users also need access to the web portal. Limitations to select features get based on the access level and security group to which a user is assigned.

Users can lose access for the following reasons:

Reason for loss of access	Troubleshooting action
The user's Visual Studio subscription has expired.	Meanwhile, this user can work as a Stakeholder, or you can give the user Basic access until the user renews their subscription. After the user signs in, Azure DevOps restores access automatically.
The Azure subscription used for billing is no longer active.	All purchases made with this subscription are affected, including Visual Studio subscriptions. To fix this issue, visit the Azure account portal ⬀.
The Azure subscription used for billing was removed from your organization.	Learn more about linking your organization

Otherwise, on the first day of the calendar month, users who haven't signed in to your organization for the longest time lose access first. If your organization has users who don't need access anymore, remove them from your organization.

For more information about permissions, see Permissions and groups and the Permissions lookup guide.

Trace a permission

Use permission tracing to determine why a user's permissions aren't allowing them access to a specific feature or function. Learn how a user or an administrator can investigate the inheritance of permissions. To trace a permission from the web portal, open the permission or security page for the corresponding level. For more information, see Request an increase in permission levels.

If a user's having permissions issues and you use default security groups or custom groups for permissions, you can investigate where those permissions are coming from by using our permissions tracing. Permissions issues could be because of delayed changes. It can take up to 1 hour for Azure AD group memberships or permissions changes to propagate throughout Azure DevOps. If a user's having issues that don't resolve immediately, wait a day to see if they resolve. For more information about user and access management, see Manage users and access in Azure DevOps.

Users can receive their effective permissions either directly or via groups.

Complete the following steps so administrators can understand where exactly those permissions are coming from and adjust them, as needed.

1. Select **Project settings** > **Permissions** > **Users**, and then select the user.

You should now have a user-specific view that shows what permissions they have.

2. To trace why a user does or doesn't have any of the listed permissions, select the information icon next to the permission in question.

The resulting trace lets you know how they're inheriting the listed permission. You can then adjust the user's permissions by adjusting the permissions that are provided to the groups that they're in.

For more information, see Grant or restrict access to select features and functions or Request an increase in permission levels.

Refresh or reevaluate permissions
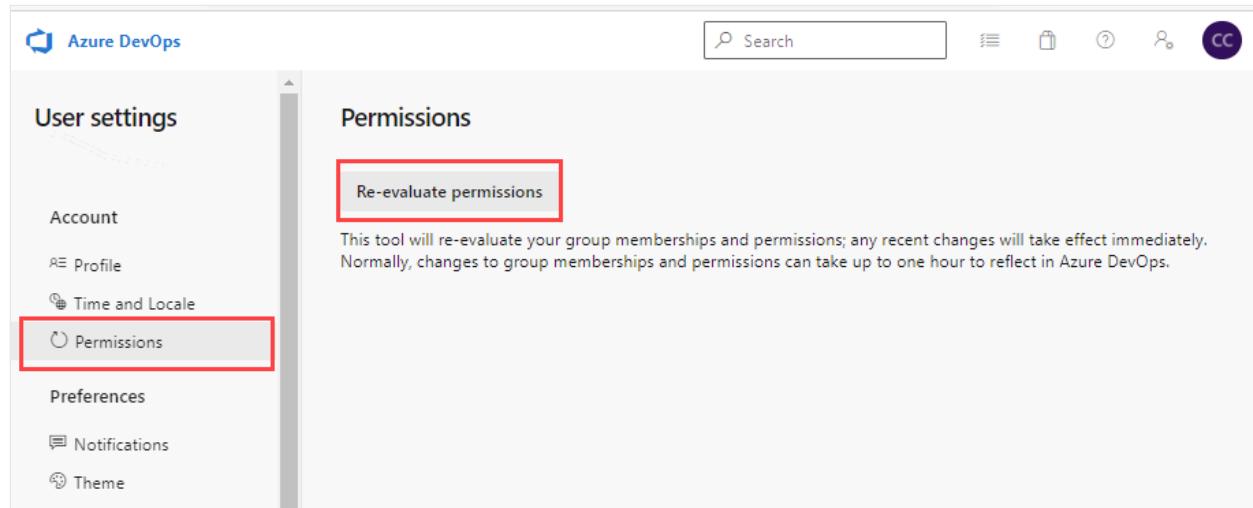
See the following scenario where refreshing or reevaluating permissions may be necessary.

Problem

Users get added to an Azure DevOps or Azure AD group. This action grants inherited access to an organization or project. But, they don't get access immediately. Users must either wait or sign out, close their browser, and then sign back in to get their permissions refreshed.

Solution

Within **User settings**, on the **Permissions** page, you can select **Re-evaluate permissions**. This function reevaluates your group memberships and permissions, and then any recent changes take effect immediately.

Rules applied to a work item type that restrict select operations

Before you customize a process, we recommend that you review Configure and customize Azure Boards, which provides guidance on how to customize Azure Boards to meet your business needs.

For more information about work item type rules that apply toward restricting operations, see:

- Apply rules to workflow states (Inheritance process)
- Sample rule scenarios
- Define area paths and assign to a team

Hide organization settings from users

If a user's limited to seeing only their projects, or from seeing the organization settings, the following information may explain why. To restrict users from accessing organization settings, you can enable the **Limit user visibility and collaboration to specific projects** preview feature. For more information including important security-related call-outs, see Manage your organization, Limit user visibility for projects and more.

Examples of restricted users include Stakeholders, Azure Active Directory (Azure AD) guest users, or members of a security group. Once enabled, any user or group added to

the Project-Scoped Users group gets restricted from accessing the Organization Settings pages, except for Overview and Projects. They're restricted to accessing only those projects to which they've been added.

Examples of restricted users include Stakeholders, or members of a security group. Once enabled, any user or group added to the Project-Scoped Users group gets restricted from accessing the Organization Settings pages, except for Overview and Projects. They're restricted to accessing only those projects to which they've been added.

For more information about hiding organization settings from users, see Manage your organization, Limit user visibility for projects and more.

View, add, and manage permissions with CLI

You can view, add, and manage permissions at a more granular level with the `az devops security permission` commands. For more information, see Manage permissions with command line tool.

Group rules with lesser permissions

Group rule types get ranked in the following order: Subscriber > Basic + Test Plans > Basic > Stakeholder. Users always get the best access level between all the group rules, including Visual Studio (VS) subscription.

> ⓘ **Note**
>
> We recommend that you regularly review the rules listed on the "Group rules" tab of the "Users" page. If there are any changes made to the Active Directory (AD) group membership, these changes will be reflected in the next re-evaluation of the group rules, which can be done on demand, when a group rule is modified, or automatically every 24 hours. Azure DevOps updates Azure AD group membership every hour, but it may take up to 24 hours for Azure AD to update **dynamic group membership**.

See the following examples, showing how subscriber detection factors into group rules.

Example 1: Group rule gives me more access

If I have a VS Pro subscription and I'm in a group rule that gives me Basic + Test Plans – what happens?

Expected: I get Basic + Test Plans because what the group rule gives me is greater than my subscription. Group rule assignment always provides the greater access, rather than limiting access.

Example 2: Group rule gives me the same access

I have a Visual Studio Test Pro subscription and I'm in a group rule that gives me Basic + Test Plans – what happens?

Expected: I get detected as a Visual Studio Test Pro subscriber, because the access is the same as the group rule. I'm already paying for the Visual Studio Test Pro, so I don't want to pay again.

Work with GitHub

See the following troubleshooting information for when you're trying to deploy code in Azure DevOps with GitHub.

Problem

You can't bring the rest of your team into the organization and project, despite adding them as organization and project members. They receive emails but when signing in they receive an error 401.

Solution

You're likely signed into Azure DevOps with an incorrect identity. Complete the following steps.

1. Close all browsers, including browsers that aren't running Azure DevOps.

2. Open a private or incognito browsing session.

3. Go to the following URL: https://aka.ms/vssignout ⬈ .

   A message displays that says, "Sign out in progress." After you sign out, you're redirected to dev.azure.microsoft.com.

4. Sign in to Azure DevOps ⬈ again. Select your other identity.

Other areas where permissions might be applied

- Area path permissions
- Work item tags
- Moved work items out of a project
- Deleted work items
- Quick guide to default permissions and access for Azure Boards
- Custom rules
- Sample custom rule scenarios
- Custom backlogs and boards
- Custom controls

Related articles

- Manage permissions with the command line tool
- Change individual or group permissions
- Security best practices
- Security and permission management tools
- Add users to an administrator role

Allowed IP addresses and domain URLs

Article • 01/02/2024

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

If your organization is secured with a firewall or proxy server, you must add certain internet protocol (IP) addresses and domain uniform resource locators (URLs) to the **allowlist**. Adding these IPs and URLs to the allowlist helps to ensure that you have the best experience with Azure DevOps. You know that you need to update your allowlist if you can't access Azure DevOps on your network. See the following sections in this article:

- Allowed domain URLs
- IP addresses and range restrictions

> 💡 **Tip**
>
> So that Visual Studio and Azure Services work well with no network issues, you should open select ports and protocols. For more information, see **Install and use Visual Studio behind a firewall or proxy server, Use Visual Studio and Azure Services**.

IP addresses and range restrictions

Outbound connections

Outbound connections target other dependent sites. Examples of such connections include:

- Browsers connecting to Azure DevOps website as users go to and use features of Azure DevOps
- Azure Pipelines agents installed on your organization's network connecting to Azure DevOps to poll for pending jobs
- CI events sent from a source code repository hosted within your organization's network to Azure DevOps

Ensure the following IP addresses are allowed for outbound connections, so your organization works with any existing firewall or IP restrictions. The endpoint data in the

following chart lists requirements for connectivity from a machine in your organization to Azure DevOps Services.

IP V4 ranges

IP
13.107.6.0/24 13.107.9.0/24 13.107.42.0/24 13.107.43.0/24

If you're currently allowing the `13.107.6.183` and `13.107.9.183` IP addresses, leave them in place, as you don't need to remove them.

> ⓘ **Note**
>
> **Azure Service Tags** aren't supported for *outbound* connections.

Inbound connections

Inbound connections originate from Azure DevOps and target resources within your organization's network. Examples of such connections include:

- Azure DevOps Services connecting to endpoints for Service Hooks
- Azure DevOps Services connecting to customer-controlled SQL Azure VMs for Data Import
- Azure Pipelines connecting to on-premises source code repositories such as GitHub Enterprise or Bitbucket Server
- Azure DevOps Services Audit Streaming connecting to on-premises or cloud-based Splunk

Ensure the following IP addresses are allowed for inbound connections, so your organization works with any existing firewall or IP restrictions. The endpoint data in the following chart lists requirements for connectivity from Azure DevOps Services to your on-premises or other cloud services.

⌞⌝ Expand table

Geography	Region	IP V4 ranges
Australia	Australia East	20.37.194.0/24
	Australia South East	20.42.226.0/24
Brazil	Brazil South	191.235.226.0/24
Canada	Central Canada	52.228.82.0/24
Asia Pacific	Southeast Asia (Singapore)	20.195.68.0/24
India	South India	20.41.194.0/24
	Central India	20.204.197.192/26
United States	Central United States	20.37.158.0/23
	West Central United States	52.150.138.0/24
	East United States	20.42.5.0/24
	East 2 United States	20.41.6.0/23
	North United States	40.80.187.0/24
	South United States	40.119.10.0/24
	West United States	40.82.252.0/24
	West 2 United States	20.42.134.0/23
	West 3 United States	20.125.155.0/24
Europe	Western Europe	40.74.28.0/23
	North Europe	20.166.41.0/24
United Kingdom	United Kingdom South	51.104.26.0/24

Azure Service Tags are supported only for *inbound* connections. Instead of allowing the previously listed IP ranges, you may use the **AzureDevOps** service tag for Azure Firewall and Network Security Group (NSG) or on-premises firewall via a JSON file download.

> ⊙ **Note**
>
> The Service Tag or previously mentioned inbound IP addresses don't apply to Microsoft Hosted agents. Customers are still required to allow the **entire geography for the Microsoft Hosted agents**. If allowing the entire geography is a concern, we recommend using the **Azure Virtual Machine Scale Set agents**. The

> Scale Set agents are a form of self-hosted agents that can be auto-scaled to meet your demands.
> Hosted macOS agents are hosted in GitHub's macOS cloud. IP ranges can be retrieved using the **GitHub metadata API** ☐ using the instructions provided **here** ☐.

Other IP addresses

Most of the following IP addresses pertain to Microsoft 365 Common and Office Online.

```
Microsoft365Common&OfficeIPs


40.82.190.38
52.108.0.0/14
52.237.19.6
52.238.106.116/32
52.244.37.168/32
52.244.203.72/32
52.244.207.172/32
52.244.223.198/32
52.247.150.191/32
```

For more information, see Worldwide endpoints and Adding IP address rules.

Azure DevOps ExpressRoute connections

If your organization uses ExpressRoute, ensure the following IP addresses are allowed for both outbound and inbound connections.

IP V4 ranges

```
IP

13.107.6.175/32
13.107.6.176/32
13.107.6.183/32
13.107.9.175/32
13.107.9.176/32
13.107.9.183/32
13.107.42.18/32
13.107.42.19/32
13.107.42.20/32
13.107.43.18/32
```

```
13.107.43.19/32
13.107.43.20/32
```

For more information about Azure DevOps and ExpressRoute, see ExpressRoute for Azure DevOps☒ .

Allowed Domain URLs

Network connection issues could occur because of your security appliances, which may be blocking connections - Visual Studio uses TLS 1.2 and above. When you're using NuGet or connecting from Visual Studio 2015 and later, update the security appliances to support TLS 1.2 and above for the following connections.

To ensure your organization works with any existing firewall or IP restrictions, ensure that `dev.azure.com` and `*.dev.azure.com` are open.

The following section includes the most common domain URLs to support sign in and licensing connections.

CommonDomainURLs

```
https://dev.azure.com
https://*.dev.azure.com
https://aex.dev.azure.com
https://aexprodea1.vsaex.visualstudio.com
https://*vstmrblob.vsassets.io
https://amp.azure.net
https://app.vssps.dev.azure.com
https://app.vssps.visualstudio.com
https://*.vsblob.visualstudio.com
https://*.vssps.visualstudio.com
https://*.vstmr.visualstudio.com
https://azure.microsoft.com
https://go.microsoft.com
https://graph.microsoft.com
https://login.microsoftonline.com
https://management.azure.com
https://management.core.windows.net
https://microsoft.com
https://microsoftonline.com
https://static2.sharepointonline.com
https://visualstudio.com
https://vsrm.dev.azure.com
https://vstsagentpackage.azureedge.net
https://*.windows.net
https://{organization_name}.visualstudio.com
https://{organization_name}.vsrm.visualstudio.com
```

```
https://{organization_name}.vstmr.visualstudio.com
https://{organization_name}.pkgs.visualstudio.com
https://{organization_name}.vssps.visualstudio.com

Azure DevOps uses content delivery network (CDN) to serve static content.
The following URLs are part of that.
https://cdn.vsassets.io
https://*.vsassets.io
https://*gallerycdn.vsassets.io
https://aadcdn.msauth.net
https://aadcdn.msftauth.net
https://amcdn.msftauth.net
https://azurecomcdn.azureedge.net
```

The following endpoints are used to authenticate Azure DevOps organizations using a Microsoft Account (MSA). These endpoints are only needed for Azure DevOps organizations backed by Microsoft Accounts (MSA). Azure DevOps organizations backed a Microsoft Entra tenant doesn't need the following URLs.

```
https://live.com
https://login.live.com
```

The following URL is required if you're migrating from Azure DevOps server to the cloud service using our data migration tool.

```
https://dataimport.dev.azure.com
```

> ⓘ **Note**
>
> Azure DevOps uses Content Delivery Networks (CDNs) to serve static content. Users in **China** should also add the following domain URLs to an allowlist:
>
> NuGetDomainURLs
>
> ```
> https://*.vsassetscdn.azure.cn
> https://*.gallerycdn.azure.cn
> ```

We recommend you open port `443` to all traffic on the following IP addresses and domains. We also recommend you open port `22` to a smaller subset of targeted IP addresses.

More domain URLs	Descriptions
https://login.microsoftonline.com	Authentication and sign-in related
https://app.vssps.visualstudio.com	Authentication and sign-in related
https://*.vssps.visualstudio.com	Authentication and sign-in related
https://*gallerycdn.vsassets.io	Hosts Azure DevOps extensions
https://*vstmrblob.vsassets.io	Hosts Azure DevOps TCM log data
https://cdn.vsassets.io	Hosts Azure DevOps Content Delivery Networks (CDNs) content
https://static2.sharepointonline.com	Hosts some resources that Azure DevOps uses in "office fabric" UI kit for fonts, and so on
https://vsrm.dev.azure.com	Hosts releases
https://vstsagentpackage.azureedge.net	Required to set up self-hosted agent in machines within your network
https://amp.azure.net	Needed for deploying to Azure app service
https://go.microsoft.com	Accesses go links

Azure Artifacts

Ensure the following domain URLs are allowed for Azure Artifacts:

```
AzureArtifactsDomainURLs


https://*.blob.core.windows.net
https://*.visualstudio.com
```

Also allow all IP addresses in the "name": "Storage.{region}" section of the following file (updated weekly): Azure IP ranges and Service Tags - Public Cloud ⧉ . {region} is the same Azure Geography as your organization.

NuGet connections

Ensure the following domain URLs are allowed for NuGet connections:

```
NuGetDomainURLs
```

```
https://azurewebsites.net
https://nuget.org
```

> ⓘ **Note**
>
> Privately owned NuGet server URLs might not be included in the previous list. You
> can check the NuGet servers you're using by opening
> `%APPData%\Nuget\NuGet.Config`.

SSH connections

If you need to connect to Git repositories on Azure DevOps with SSH, allow requests to
port 22 for the following hosts:

```
SSHDomainHosts


ssh.dev.azure.com
vs-ssh.visualstudio.com
```

Also allow IP addresses in the "name": "AzureDevOps" section of this downloadable
file ⧉ (updated weekly) named: **Azure IP ranges and Service Tags - Public Cloud**

Azure Pipelines Microsoft-hosted agents

If you use Microsoft-hosted agent to run your jobs and you need the information about
what IP addresses are used, see Microsoft-hosted agents IP ranges. See all Azure Virtual
Machine Scale Set agents.

For more information about hosted Windows, Linux and macOS agents, see Microsoft-
hosted agent IP ranges.

Azure Pipelines Self-hosted agents

If you're running a firewall and your code is in Azure Repos, see Self-hosted Linux agents
FAQs, Self-hosted macOS agents FAQs or Self-hosted Windows agents FAQs. This article
has information about which domain URLs and IP addresses your private agent needs to
communicate with.

Azure DevOps import service

During the import process, we highly recommend that you restrict access to your virtual machine (VM) to only IP addresses from Azure DevOps. To restrict access, allow only connections from the set of Azure DevOps IP addresses, which were involved in the collection database import process. For information about identifying the correct IP addresses, see (Optional) Restrict access to Azure DevOps Services IPs only.

Related articles

- Available service tags
- Microsoft-hosted agents IP address ranges
- Self-hosted Windows agents FAQs
- Configure Azure Storage firewalls and virtual networks
- Install and use Visual Studio behind a firewall or proxy server

Get support and provide feedback

Article • 11/10/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Share your feedback and ideas with us, or join our communities. We're always working to improve Azure DevOps, and we want you to be part of the process!

Action	More info
Get advice	Visit StackOverflow for Azure DevOps Services ⧉ or Azure DevOps Server ⧉.
Report a bug	Submit it through our Developer Community for Azure DevOps Services ⧉ or Azure DevOps Server ⧉.
Suggest a feature or fix	Submit your idea or issue through our Developer Community for Azure DevOps Services ⧉ or Azure DevOps Server ⧉.
Contact Support	You can contact Support by creating a support ticket. To create a support ticket, visit the Microsoft Azure Support Ticket page ⧉ and follow the instructions to create and manage support requests for Azure DevOps Services.
Find out what's new in Azure DevOps	Check out the current Azure DevOps Release Notes. These notes are updated every three weeks.
Chat with our virtual support agent	Get help with common issues, troubleshooting steps, or create a request to change the region your Azure DevOps instance is hosted in using our virtual support agent ⧉.

Documentation feedback

All articles on Microsoft Learn have a ratings tool. Select **Feedback** beneath the title of the article, and in the resulting "Was this page helpful?" pane, select **Yes** or **No** depending on your experience.

Enter more detailed feedback in the **Tell us more** section. Make sure you don't include sensitive or personal information. Although we can't reply back, we collect and review this feedback regularly, and use your sentiments in our content planning.

Tips for effective feedback

If you just want to vent about the product or the documentation, that's okay. It helps us a lot to know when you're happy or unhappy with an experience. Provide details so we can better understand what we're doing right or wrong.

- Provide a little context. What problem were you trying to solve? At what point did it go wrong?
- What's your role? We don't need personal or professional details. Are you a dev? A manager? A business owner? When we understand our audience, we can come up with better solutions for you and other customers doing similar work.
- What version of the product were you using? What other products were you using with it?

The best feedback we get is clear and precise. For example:

- Product feedback: "I'm a project manager for a small start-up. I'm using Azure DevOps. I'm trying to create work item templates through the UI, but my changes don't seem to persist. It's not clear what I'm doing wrong."
- Doc feedback: "I'm a dev in a large organization that works on Java apps. I tried to use Maven with your build system in Azure DevOps Server 2017 Update 1 (15.112.26307.0), but I couldn't get the configuration shown in the documentation to work."

The more details, the better!

Related articles

- [Azure DevOps features timeline](#)
- [Report a problem with Visual Studio](#)

Look up your Azure DevOps platform and version

Article • 02/21/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

You can tell what platform you use by opening the **About** page for Azure DevOps Services or Azure DevOps Server.

Azure DevOps Services

Enter the following URL for your organization, specifying the organization name.

`https://dev.azure.com/YourOrganizationName/_home/About`.

A page similar to the following example opens showing the version number.

Azure DevOps Server

Open the **About** page from the profile menu as shown in the following image.

The corresponding browser URL is:

```
https://ServerName/CollectionName/_home/About
```

A page similar to the following image opens showing the version number.

Version numbers

On-premises release	Version	About help page version	Build number
Azure DevOps Server 2022	RC1	(AzureDevOpsServer_20220720.1)	19.205.32728.1
Azure DevOps Server 2020	Update 1.1	Version Azure DevOps Server 2020 Update 1.2	18.181.31230.2
Azure DevOps Server 2020	RTW	Version Dev 18.M170.9	18.170.30830.2

On-premises release	Version	About help page version	Build number
Azure DevOps Server 2019	Update 1	Version Dev17.M153.6	17.153.29522.3

Related articles

- Azure DevOps features timeline
- Report a problem with Visual Studio

Navigate in Visual Studio Team Explorer

Article • 02/21/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Visual Studio 2019 | Visual Studio 2022

You use Team Explorer to coordinate your code efforts with other team members to develop a software project. In addition, you can manage work and that is assigned to you, your team, or your projects. Team Explorer is a plug-in that installs with Visual Studio. Developers can effectively collaborate using Team Explorer connected to projects hosted on Azure DevOps Services or an on-premises Azure DevOps Server.

> 💡 **Tip**
>
> You can install the latest version of Visual Studio clients from the **Visual Studio downloads page**.
>
> Additional options for connecting to Azure DevOps Services or TFS include:
>
> - **Azure DevOps Plugin for Android Studio**
> - **Azure DevOps Plugin for IntelliJ**
> - **Visual Studio Code** ⧉
>
> For information about compatibility among client and server versions, see **Requirements and compatibility**.

If you don't need Visual Studio, but want to connect to a project in Azure DevOps, you can install the free Visual Studio Community ⧉ .

Prerequisites

- You must have a project in Azure DevOps. If you need to add a project, see Create a project.
- You must be a member of the project you connect to. To get added, see Add users to a project or team.
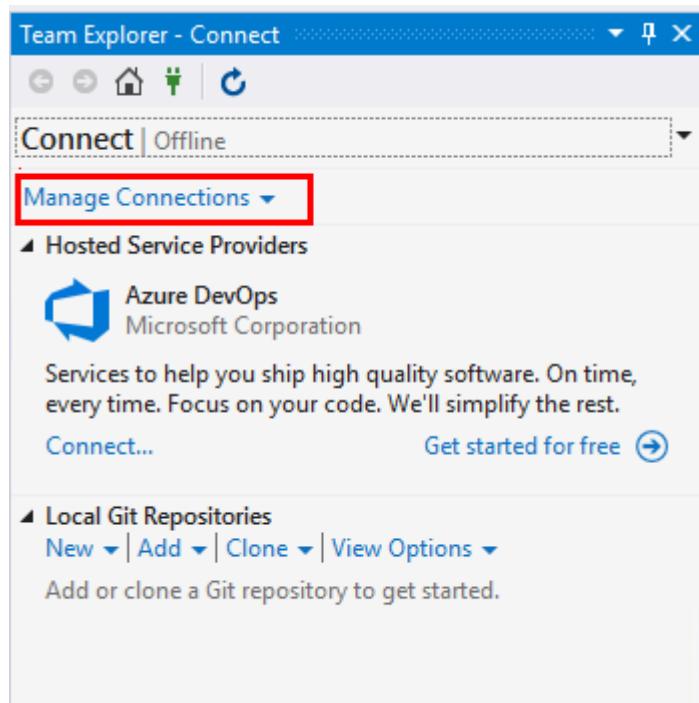
Connect to a project or repository

Team Explorer connects Visual Studio to projects in Azure DevOps. You can manage source code, work items, and builds. The operations available to you depend on which source control option—Git or Team Foundation version control (TFVC) —was selected to manage source code when the project was created.

> 💡 **Tip**
>
> If you open Visual Studio and the Team Explorer pane doesn't appear, choose the **View**>**Team Explorer** menu option from the tool bar.

From the **Connect** page, you can select the projects you want to connect to and quickly switch connection to a different project and or repository. For details, see Connect to a project.

The Git and TFVC repos support different pages and functions. For a comparison of the two version control systems, see Choosing the right version control for your project.

Git version control and repository

The following image shows the pages available when you connect to a Git repository from Visual Studio Team Explorer.

> ⓘ **Note**
>
> Visual Studio 2019 version 16.8 and later versions provide a new Git menu for managing the Git workflow with less context switching than Team Explorer.

To learn more about each page, see the following articles.

Home and Builds

Git version control

Work items

Home
- Web portal
- Task Board

Builds
- Create build pipelines
- View and manage builds
- Manage the build queue
- Create a new repo

- Clone an existing repo
- **Changes**: Save work with commits
- **Branches**: Create work in branches
- **Pull Requests**: Review code with pull requests"
- **Sync**: Update code with fetch and pull
- **Tags**: Work with Git tags
- Git preferences
- Git command reference

Default experience (Visual Studio 2019 and later versions)
- View and add work items
- Set the Work Items experience in Visual Studio
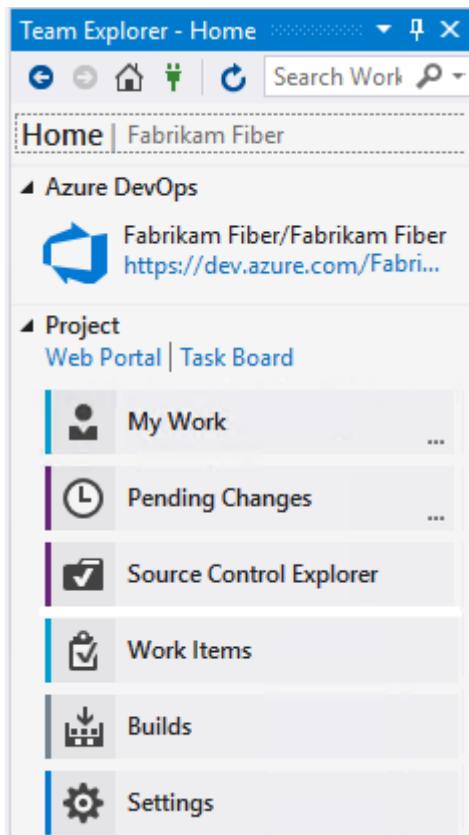
Legacy experience (All Visual Studio versions)
- Add work items
- Query editor
- Query folders
- Query permissions
- Open query in Excel
- Email query results using Outlook
- Create reports from query in Excel

Team Foundation version control

The following image shows the pages available when you connect to a TFVC repository from Visual Studio Team Explorer.

To learn more about each page, see the following articles.

Home and Builds

TFVC

Work items

Home
- Web portal
- Task Board

Builds
- Create build pipelines
- View and manage builds
- Manage the build queue
- Configure workspace
- Suspend/resume work, Code review
- **Pending Changes**: Manage pending changes, Find shelvesets, Resolve conflicts
- **Source Control Explorer**: Add/view files and folders
- Add Check-In Policies
- Version control commands

Default experience (Visual Studio 2019 and later versions)
- View and add work items

- Set the Work Items experience in Visual Studio

Legacy experience (All Visual Studio versions)

- Add work items
- Query editor
- Query folders
- Query permissions
- Open query in Excel
- Email query results using Outlook
- Create reports from query in Excel

Settings

From the **Settings** page, you can configure administrative features for either a project or project collection. To learn more about each page, see the following articles. Most of the links open to a web portal administration page. Not all settings are available from the Team Explorer plug-in for Eclipse.

Project

- Security, Group Membership
- Security, Source Control (TFVC)
- Work Item Areas
- Work Item Iterations
- Portal Settings
- Project Alerts

Project Collection

- Security, Group Membership
- Source Control (TFVC)
- Process Template Manager

Other

- Git Global Settings
- Git Repository Settings

To learn more about settings, see About team, project, and organizational-level settings.

Refresh Team Explorer

If data doesn't appear as expected, the first thing to try is to refresh your client. Refreshing your client updates the local cache with changes that were made in another client or in Azure DevOps. To refresh Team Explorer, do one of the following actions:

- To refresh a page that you are currently viewing, choose 🔃 **Refresh** in the menu bar (or choose **F5**).
- To refresh the project you currently have selected, choose 🏠 **Home**, and then choose 🔃 **Refresh** (or choose **F5**).
- To refresh the set of teams defined for the project that you currently have selected, choose **Connect**, and then choose 🔃 **Refresh** (or enter **F5**).

To avoid potential errors, you should refresh your client application under the following circumstances:

- Process changes are made.
- Work item type definitions are added, removed, renamed, or updated.
- Area or iteration paths are added, removed, renamed, or updated.
- Users are added to or removed from security groups, or permissions are updated.
- A team member adds a new shared query or changes the name of a shared query.
- A build pipeline is added or deleted.
- A team or project is added or deleted.

Resolve images that don't display in Team Explorer

If an inline image isn't displayed in a work item form that you view in Visual Studio Team Explorer, but the image is displayed in the web portal, your credentials might have expired. You can resolve this by completing the following steps:

1. In Visual Studio, select **View** > **Other Windows** > **Web Browser** (or use the shortcut **Ctrl+Alt+R**).
2. In the web browser, locate your organization.
3. Sign in with your credentials.
4. Refresh your work item in Team Explorer.

Related articles

- [Troubleshoot connection](#)
- [Create a project](#)

Rate and usage limits

Article • 12/04/2023

Azure DevOps Services

Azure DevOps Services uses multi-tenancy to reduce costs and improve performance. This design leaves users vulnerable to performance issues and even outages when other users of their shared resources have spikes in their consumption. So, Azure DevOps limits the resources individuals can consume, and the amount of requests they can make to certain commands. When these limits are exceeded, future requests might be either delayed or blocked.

For more information, see Git limits and Best practices to avoid hitting rate limits.

Global consumption limit

Azure DevOps currently has a global consumption limit, which delays requests from individual users beyond a threshold when shared resources are in danger of being overwhelmed. This limit is focused exclusively on avoiding outages when shared resources are close to being overwhelmed. Individual users typically only get delayed requests when one of the following incidents occurs:

- One of their shared resources is at risk of being overwhelmed
- Their personal usage exceeds 200 times the consumption of a typical user within a (sliding) five-minute window

The amount of the delay depends on the user's sustained level of consumption. Delays range from a few milliseconds per request up to 30 seconds. Once consumption goes to zero or the resource is no longer overwhelmed, the delays stop within five minutes. If consumption remains high, delays might continue indefinitely to protect the resource.

When a user request gets delayed by a significant amount, that user receives an email and a warning banner in the web. For the build service account and others without an email address, members of the Project Collection Administrators group get the email. For more information, see Usage monitoring.

When an individual user's requests get blocked, responses with HTTP code 429 (too many requests) are received, with a message similar to the following message:

```
TF400733: The request has been canceled: Request was blocked due to exceeding usage
of resource <resource name> in namespace <namespace ID>.
```

Azure DevOps throughput units (TSTUs)

Azure DevOps users consume many shared resources, and consumption depends on the following factors:

- Uploading a large number of files to version control creates a large amount of load on databases and storage accounts
- Complex work item tracking queries create database load based on the number of work items they search through
- Builds drive load by downloading files from version control, producing log output
- All operations consume CPU and memory on various parts of the service

To accommodate, Azure DevOps resource consumption is expressed in abstract units called Azure DevOps throughput units, or TSTUs. TSTUs eventually incorporate a blend of the following items:

- Azure SQL Database DTUs as a measure of database consumption
- Application tier and job agent CPU, memory, and I/O as a measure of compute consumption
- Azure Storage bandwidth as a measure of storage consumption

For now, TSTUs are primarily focused on Azure SQL Database DTUs, since Azure SQL Databases are the shared resources most commonly overwhelmed by excessive consumption. A single TSTU is the average load we expect a single normal user of Azure DevOps to generate per five minutes. Normal users also generate spikes in load. These spikes are typically 10 or fewer TSTUs per five minutes. Less frequently, spikes go as high as 100 TSTUs.

The global consumption limit is 200 TSTUs within a sliding five-minute window.

We recommend that you at least respond to the `Retry-After` header. If you detect a `Retry-After` header in any response, wait until some time passes before you send another request. Doing so helps your client application experience fewer enforced delays. Keep in mind that the response is 200, so you don't need to apply retry logic to the request.

If possible, we further recommend that you monitor `X-RateLimit-Remaining` and `X-RateLimit-Limit` headers. Doing so allows you to approximate how quickly you're approaching the delay threshold. Your client can intelligently react and spread out its requests over time.

> ⊙ **Note**

Identities that are used by tools and applications to integrate with Azure DevOps might need higher rate and usage limits beyond the allowed consumption limit from time to time. You can get additional rate and usage limits by assigning the **Basic + Test Plans** access level to the desired identities used by your application. Once the need for higher rate limits are fulfilled, you can go back to the access level that the identity used to have. You're charged for the cost of **Basic + Test Plans** access level only for the time it's assigned to the identity.

Identities that are already assigned a Visual Studio Enterprise subscription cannot be assigned **Basic + Test Plans** access level till they are removed.

Pipelines

Rate limiting is similar for Azure Pipelines. Each pipeline gets treated as an individual entity with its own resource consumption tracked. Even if build agents are self-hosted, they generate load in the form of cloning and sending logs.

We apply a 200 TSTU limit for an individual pipeline in a sliding 5-minute window. This limit is the same as the global consumption limit for users. If a pipeline gets delayed or blocked by rate limiting, a message appears in the attached logs.

API client experience

When requests get delayed or blocked, Azure DevOps returns response headers to help API clients react. While not fully standardized, these headers are broadly in line with other popular services ⧉ .

The following table lists the headers available and what they mean. Except for `X-RateLimit-Delay`, all of these headers get sent before requests start getting delayed. This design gives clients the opportunity to proactively slow down their rate of requests.

Header name

Description

`Retry-After`

The RFC 6585-specified header sent to tell you how long to wait before you send your next request to fall under the detection threshold. Units: seconds.

`X-RateLimit-Resource`

A custom header indicating the service and type of threshold that was reached. Threshold types and service names might vary over time and without warning. We recommend displaying this string to a human, but not relying on it for computation.

`X-RateLimit-Delay`

How long the request was delayed. Units: seconds with up to three decimal places (milliseconds).

`X-RateLimit-Limit`

Total number of TSTUs allowed before delays are imposed.

`X-RateLimit-Remaining`

Number of TSTUs remaining before being delayed. If requests are already being delayed or blocked, it's 0.

`X-RateLimit-Reset`

Time at which, if all resource consumption stopped immediately, tracked usage would return to 0 TSTUs. Expressed in Unix epoch time.

Work tracking, process, & project limits

Azure DevOps imposes limits for the number of projects you can have in an organization and the number of teams you can have within each project. Also be aware of limits for work items, queries, backlogs, boards, dashboards, and more. For more information, see Work tracking, process, and project limits.

Wiki

In addition to the usual repository limits, wikis defined for a project are limited to 25 MB per single file.

Service connections

There are no per-project limits placed on creating service connections. However, there might be limits, which are imposed through Microsoft Entra ID. For additional information, review the following articles:

- Microsoft Entra service limits and restrictions
- Azure subscription and service limits, quotas, and constraints

Related articles

- Work tracking, process, and project limits
- Configure and customize Azure Boards
- Usage monitoring
- Git limits
- Best practices to avoid hitting rate limits

Get started with Azure DevOps CLI

Article • 03/27/2023

Azure DevOps Services

With the Azure DevOps extension for Azure Command Line Interface (CLI), you can manage many Azure DevOps Services from the command line. CLI commands enable you to streamline your tasks with faster and flexible interactive canvas, bypassing user interface workflows.

> ⓘ **Note**
>
> The Azure DevOps Command Line Interface (CLI) is only available for use with Azure DevOps Services. The Azure DevOps extension for the Azure CLI does not support any version of Azure DevOps Server.

To start using the Azure DevOps extension for Azure CLI, perform the following steps:

1. Install Azure CLI: Follow the instructions provided in Install the Azure CLI to set up your Azure CLI environment. At a minimum, your Azure CLI version must be 2.10.1. You can use `az --version` to validate.

2. Add the Azure DevOps extension:

   ```
   az extension add --name azure-devops
   ```

   You can use `az extension list` or `az extension show --name azure-devops` to confirm the installation.

3. Sign in: Run `az login` to sign in. Note that we support only interactive or log in using user name and password with `az login`. To sign in using a Personal Access Token (PAT), see Sign in via Azure DevOps Personal Access Token (PAT).

4. Configure defaults: We recommend you set the default configuration for your organization and project. Otherwise, you can set these within the individual commands themselves.

```
  az devops configure --defaults
organization=https://dev.azure.com/contoso project=ContosoWebApp
```

Command usage

Adding the Azure DevOps Extension adds `devops`, `pipelines`, `artifacts`, `boards`, and `repos` groups. For usage and help content for any command, enter the **-h** parameter, for example:

Azure CLI

```
az devops -h
```

Output

```
Group
    az devops : Manage Azure DevOps organization level operations.
        Related Groups
        az pipelines: Manage Azure Pipelines
        az boards: Manage Azure Boards
        az repos: Manage Azure Repos
        az artifacts: Manage Azure Artifacts.

Subgroups:
    admin            : Manage administration operations.
    extension        : Manage extensions.
    project          : Manage team projects.
    security         : Manage security related operations.
    service-endpoint : Manage service endpoints/service connections.
    team             : Manage teams.
    user             : Manage users.
    wiki             : Manage wikis.

Commands:
    configure        : Configure the Azure DevOps CLI or view your
configuration.
    feedback         : Displays information on how to provide feedback to
the Azure DevOps CLI team.
    invoke           : This command will invoke request for any DevOps area
and resource. Please use
                       only json output as the response of this command is
not fixed. Helpful docs -
                       https://learn.microsoft.com/rest/api/azure/devops/.
    login            : Set the credential (PAT) to use for a particular
organization.
    logout           : Clear the credential for all or a particular
organization.
```

Open items in browser

You can use `--open` switch to open any artifact in Azure DevOps portal in your default browser.

For example :

```
Azure CLI

az pipelines build show --id 1 --open
```

This command shows the details of build with `id 1` on the command-line and also opens it in the default browser.

Related articles

- Sign in via Azure DevOps Personal Access Token (PAT)
- Output formats
- Index to az devops examples
- Azure DevOps CLI Extension GitHub Repo

Cross-service overview

Article • 10/11/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Azure DevOps lets you connect to and collaborate across its core services. You can use various features to link and track your devops tasks across Azure Boards, Azure Repos, Azure Pipelines, and Azure Test Plans. This article shows you options for how to use the cross-service integration of Azure DevOps to improve your workflow and productivity.

Links to more information:

- Power Automate, Azure DevOps ⬀
- Power Automate templates for Azure DevOps ⬀
- Microsoft Power Automate documentation

Collaboration across Azure DevOps

The following table summarizes some of the features that help you work with your team and other teams.

Feature

Description

`@mentions` (add to discussions and comments)

You can @mention a team member or an entire team within a work item form discussion or the comment section of a commit, pull request, or changeset.

`#ID` (link to a work item)

To support end-to-end traceability, you can link to work items from commits, pull requests, and changesets.

Teams

Each team gets access to a suite of Agile tools and team assets. These tools let teams work autonomously and collaborate with other teams across the enterprise. Each team

can configure and customize each tool to support how they work. For quick navigation, they can favorite repositories, pipelines, and test plans.

Set up alerts

Configure or opt out of personal, team, project, or organization-level alerts. Subscribe to email alerts when changes occur to work items, code reviews, pull requests, source control files, builds and more.

Share summaries by email
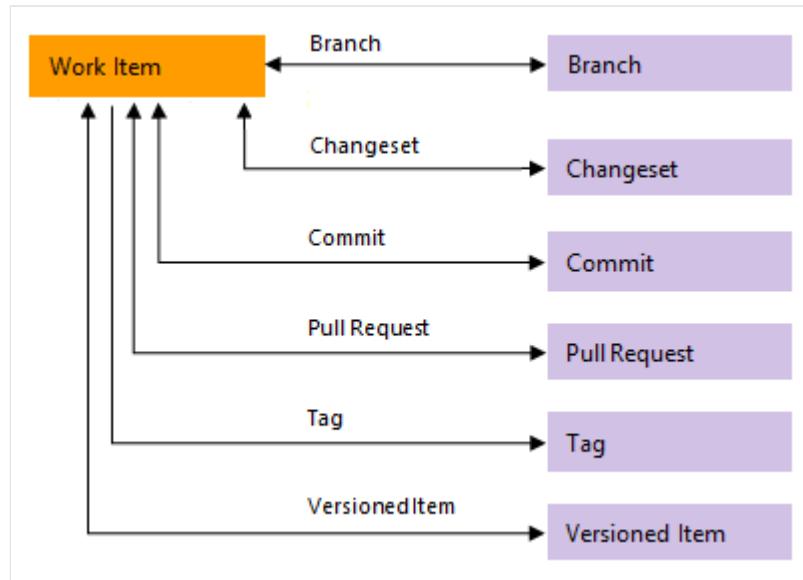
- Email a list of work items
- Email query items
- Send release summaries by email

Wiki

Embed Azure Boards query results in Wiki.

Azure Boards - Azure Repos

You can link code changes to user stories and features with different link types. For Git, use *Branch*, *Commit*, *Pull Request*, or *Tag*. For TFVC, use *Changeset* or *Versioned Item*.

The following table summarizes the integration points between Azure Boards and Azure Repos.

Feature

Description

Drive Git development from work item(s)

You can initiate a Git branch or link to Git commits or pull requests and drive your Git development cycle for a work item from within the work item form.

Automatically link and transition work items with Git commits

For a Git repository, you can turn on or off the following options:
- Close work items with mentions in commit comments. - Remember user choices for completing work items with pull requests.
- Link work items from commit comments. You can also automate linking from commits or pull requests in repo settings.
- Commit mention linking: Turn on to link commits to work items with #*WorkItemID* in commit messages. Turn off when you push a repo from a different account or service. Azure DevOps automatically turns off this feature when you import a repo.
- Commit mention work item resolution: Turn on to close work items with Fixes #*WorkItemID* in commits.
- Work item transition preferences: On by default, it remembers each user's option to complete linked work items with pull requests. You can turn this feature off to discourage users from completing work items with pull requests. When it's off, users have to choose to complete work items for each pull request.

Check for linked work items in a Git branch

Encourage traceability by checking for linked work items on pull requests.

Auto complete work items with pull requests

When you link a work item to a pull request (PR), you can automatically complete those work items when you successfully complete the PR. The system defaults to your selection for future PRs.

View list of code objects a single work item is linked to

You can link work items to code changes, builds, and releases—providing an audit trail of how a feature has been developed

Query for external links

You can query for work items that contain links to branches, commits, pull requests or tags.

Configure branch policies to support work tracking

To ensure that changes to a branch have links to work items, you configure the branch policy for a Git repository in repo settings. Turn on the **Check for linked work items** option. Choose **Required** to mandate all pull requests have at least one linked work item in order to be completed. Choose **Optional** to allow pull requests without linked work items, but warn about it.

Azure Boards - Azure Pipelines

The following table summarizes the integration points between Azure Boards and Azure Pipelines. Several features provide support for end-to-end traceability as user stories and features move through the development cycle. As with Azure Repos, you can link work items to pipeline objects with the following link types: *Build, Integrated in build*, and *Integrated in release*.

Feature

Description

Manually link work items to builds.

Link work items to builds in the same or other project within the organization or collection.

Set integration option to automatically create *Integrated in build* links to work items linked to a branch, commit, or pull request associated with a pipeline.

Required to populate the **Development** control with *Integrated in build* links. The work items or commits that are part of a release are computed from the versions of artifacts. For example, each build in Azure Pipelines is associated with a set of work items and commits. For more information, see Configure pipelines to support integration.

To link work items to builds and releases, choose an option and a branch for a Classic or YAML pipeline, which creates *Integrated in build* and *Integrated in release stage* links for work items that are linked to a branch, commit, or pull request.

Required to populate the work item form **Development** control with *Integrated in build* links and the **Deployment** control with *Integrated in release stage* links when running a Classic or YAML pipeline. For more information, see Configure pipelines to support integration.

Set integration option to automatically create *Integrated in release stage* links to work items linked to a branch, commit, or pull request associated with a release.

Required to populate **Deployment** control in work item form with **Integrated in release stage** links. For more information, see Release pipelines, How do I integrate and report release status?.

View and open list of work items linked to a Classic or YAML pipeline.

Lists all work items linked to a release since the previous selected release. Can sort the list by each column.

View list of build or release objects a single work item is linked to

You can link work items to builds and releases—providing an audit trail of how a feature has been built and deployed.

Query for external links.

You can query for work items that contain external links.

View and quickly navigate to release stages a work item is linked to.

The **Deployment** control on the work item form shows the stages that the work item is linked to. You can see the status of some runs and open each stage or run by expanding a stage. For more information, see Link and view work items to builds and deployments.

Create a work item on failure (Classic or YAML), optionally set values for a work item field (Classic)

Automatically create a work item and set fields when a build fails. For more information, see Build options for Classic pipelines, and Customize pipelines, Create work item on failure.

Query Work Items task. Ensure the number of matching work items returned from a query is within a threshold.

Use this task to ensure the number of matching items returned by a work item query is within the configured thresholds. For more information, see Query Work Items task, Control deployments with gates and approvals.

Azure Repos - Azure Pipelines

Azure Pipelines provides support for building code stored in Azure Repos, either a Git or Team Foundation Version Control (TFVC) repository. Other repositories that Azure Pipelines supports are listed in Supported source repositories.

The following table summarizes the integration features between Azure Repos and Azure Pipelines.

Feature

Description

Report deployment status

Indicates the status of a deployment on the **Files**, **Commits**, and **Branches** pages for Git repositories. This feature improves the traceability from code commit to deployment. You can configure the release environments to report deployment status.

Release status badge

Post the status of your most recent pipeline build in your repository.

Code coverage

Publish and review code coverage results that indicate the proportion of your project's code that is actually being tested.

Azure Boards - Azure Repos - Azure Test Plans

Several collaboration scenarios are supported through Azure Boards work item types. As with other work item types, you can use managed queries and the Azure DevOps search function to find and list work items.

> ⓘ **Note**

> Several of these work item types—such as Feedback Request, Code Review Request, Shared Steps, and Shared Parameters—are designed to be created through a specific tool or form. They aren't meant to be created manually. Therefore, they are added to the Hidden Types category. Work item types that are added to the Hidden Types category don't appear in the menus used to add work items.
>
> Also, for the Inherited process model, you can only customize the following work item types: Test Plan, Test Suite, Test Case.

Scenario

Work item type

Description

Request code review

Code Review Request

Tracks information entered into the TFVC New Code Review form. For more information, see Get your code reviewed with Visual Studio.

Provide code review

Code Review Response

Tracks review comments provided by code reviewers in response to a code review request.

Request feedback

Feedback Request

Tracks information entered into a request feedback form. Use the following forms to initiate a feedback request.
- Request stakeholder feedback
- Get feedback.

Provide feedback

Feedback Review

Lets stakeholders provide feedback based on requests for feedback or by volunteering feedback using the Microsoft Test & Feedback Marketplace extension.

Manual testing

Test Plan

Groups one or more test suites and individual test cases together. Test plans include static test suites, requirement-based suites, and query-based suites. To get started, see Create test plans and test suites.

Manual testing

Test Suite

Groups one or more test cases into separate testing scenarios within a single test plan. Grouping test cases makes it easier to see which scenarios are complete.

Manual testing

Test Case

Defines steps used to validate individual parts of your code to ensure your code works correctly, has no errors, and meets business and customer requirements. You can add individual test cases to a test plan without creating a test suite. More than one test suite or test plan can refer to a test case. You can effectively reuse test cases without having to copy or clone them for each suite or plan.

Manual testing

Shared Steps

Enables sharing steps across several test cases.
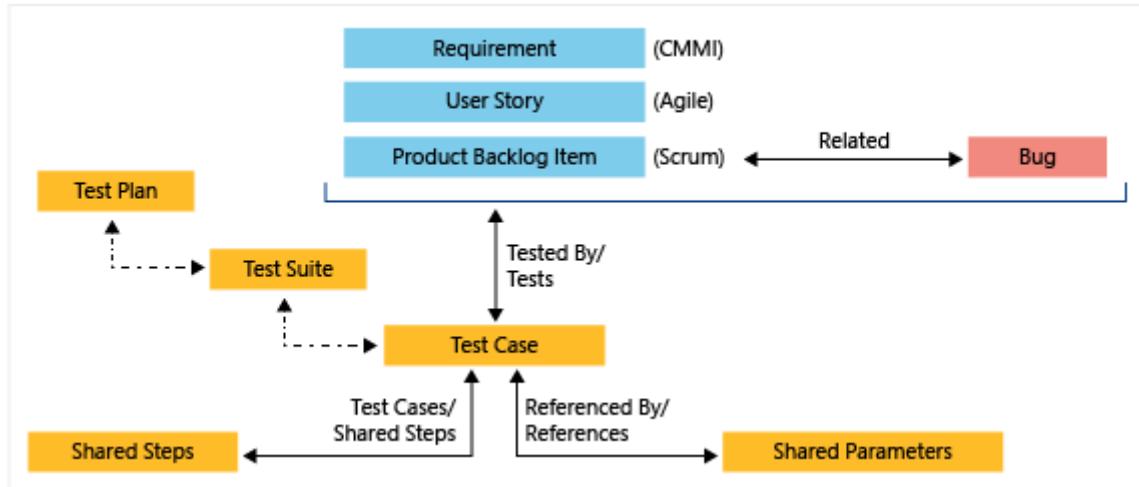
Manual testing

Shared Parameters

Enables repeating the same test cases with different data.

Test work item types

Work item types that support the test experience are linked together using the link types shown in the following image. These include *Tested By/Tests*, *Test Cases/Shared Steps*, and *Reference By/References*.

You can use the web portal to see the test cases that are defined for a test suite, and the test suites that are defined for a test plan. But, there's no specific link type that connects these objects to each other.

Track bugs

The Bug work item type supports the following integrations that you should be aware of when you're tracking bugs.

Scenario	Description
Create a bug from a testing tool	You can add a bug from Test Runner or the Test & Feedback extension. For more information, see Define, capture, triage, and manage bugs.
Create inline tests linked to bugs or user stories	When your team tracks bugs as requirements, you can use the Kanban board to add tests to verify bug fixes or user stories.
Track build information with bugs	The Bug work item form contains System Info, Found in Build, and Integrated in Build, which support tracking code defects found and resolved within pipeline builds. For more

information, see [Query based on build and test integration fields](#).

Azure Pipelines - Azure Test Plans

Azure Test Plans is fully integrated with Azure Pipelines to support testing within continuous integration/continuous deployment (CI/CD). You can associate test plans and test cases with build or release pipelines. Add pipeline tasks to pipeline definitions to capture and publish test results. Review test results via built-in progress reports and pipeline test reports. The following table summarizes the integration points between Azure Pipelines and Azure Test Plans.

Feature

Description

Test plans setting

With test plan settings, you can [configure the Test Run settings](#) to associate build or release pipelines and Test Outcome settings.

Pipeline test-enable tasks

Specify test-enable tasks within a pipeline definition. Azure Pipelines provides several tasks, including the following tasks, that support a comprehensive test reporting and analytics experience.

- [Publish Test Results task](#): Use to publish test results to Azure Pipelines.
- [Visual Studio Test task](#): Use to run unit and functional tests (Selenium, Appium, Coded UI test, and more) using the Visual Studio Test Runner.
- [.NET Core CLI task](#): Use to build, test, package, or publish a dotnet application. For other tasks, see [Publish Test Results task](#)

Run automated tests in build pipelines

[Associate test plans with a build pipeline](#). so that they run with each build.

Associate automated tests with test cases

[Associate automated tests with test cases](#).

Set retention policy for automated test results associated with builds

You can set the test retention policy for automated builds from the **Pipelines** > **Retention** page.

Requirements traceability

The Requirements quality widget supports tracking quality continuously from a build or release pipeline. The widget shows the mapping between a requirement and latest test results executed against that requirement. It provides insights into requirements traceability.

Test results trend

The Test results trend configurable widget displays the trend of test results for the selected build or release pipeline. The widget helps you visualize the test trends over a period of time, thereby surfacing patterns about test failures, test duration etc. For more information, see Configure the Test Results Trend (Advanced) widget

Deployment status

The Deployment status configurable widget shows a combined view of the deployment status and test pass rate across multiple environments for a recent set of builds. You configure the widget by specifying a build pipeline, branch, and linked release pipelines. To view the test summary across multiple environments in a release, the widget provides a matrix view of each environment and corresponding test pass rate. See Associate automated tests with test cases

View test results in builds and releases

Both build and release summaries provide details of test execution. Review these summaries to assess pipeline quality, review traceability, and troubleshoot failures. Choose **Test summary** to view the details in the **Tests** tab.

Test analytics for builds

Each build summary includes an **Analytics** tab that hosts the Test analytics report.

Dashboards, reporting, and Analytics

Dashboards provide an easy way to monitor progress and status. Teams can add configurable widgets to support their goals. The Analytics service is the reporting platform for Azure DevOps, and replaces the previous platform based on SQL Server

Reporting Services. Analytics is optimized for fast read-access and server-based aggregations and provides the following benefits:

- Analytics widgets that you can add to your dashboards
- In-context Analytics reports available from select Azure DevOps pages
- Rollup bars and counts for Azure Boards backlogs
- Custom reports you can create using Power BI
- Custom reports you can create using OData queries
- Support to develop and add your custom Analytics widgets you can add to dashboards

You can add the following built-in widgets to your dashboard. They're organized under the service they support. You might find more widgets from the Azure DevOps Marketplace ☒.

Widgets are annotated as follows:

- **Analytics**: Widget derives data from Analytics data
- **Build**: Widget derives data for a selected build pipeline
- **Project**: indicates you can select the project and team when configuring the widget
- **Release**: Widget derives data for a selected release pipeline
- **Team**: Widget is scoped to a single team
- **Teams**: Widget is scoped to one or more teams
- **User**: Widget is scoped to the signed in user account

Boards
- Assigned to me (User)
- Burndown chart (Analytics, Project, Teams)
- Burnup chart (Analytics, Project, Teams)
- Chart for work items
- Cumulative flow diagram (Team)
- Cycle time (Analytics) (Analytics, Team)
- Lead time (Analytics) (Analytics, Team)
- New Work item
- Query results
- Query tile
- Sprint burndown (Analytics, Team)
- Sprint burndown - Legacy (Team)
- Sprint capacity (Team)
- Sprint overview (Team)
- Velocity (Analytics, Team)

- [Work links](#)

Code
- [Code tile](#) (Repository, Branch, Folder)
- [Pull request](#) (Team)

Pipelines
- [Build history](#) (Build pipeline)
- [Deployment status](#) (Build pipeline)
- [Release pipeline overview](#) (Release pipeline)
- [Requirements quality](#) (Query, Build or Release pipeline)

Test Plans
- [Chart for test plans](#)
- [Test results trend (Advanced)](#) (Analytics, Build or Release pipeline)
- [Test results trend](#) (Build or Release pipeline)

Information and links
- [Embedded web page](#)
- [Markdown](#)
- [Other links](#)
- [Team members](#) (Team)
- [Visual Studio Shortcuts](#)
- [Welcome](#)

Data available from Analytics

Analytics provides the reporting platform for Azure DevOps. Analytics is generally available for Azure DevOps Services and Azure DevOps Server 2020 and is in preview for Azure DevOps Server 2019.

You can access the following data from Analytics.

Service

Data availability

Azure DevOps Services
Azure DevOps Server 2020

Azure DevOps Server 2019

Boards

Widgets
In-context reports
OData Power BI

✔
✔
✔
✔

✔

✔
✔

Repos

None

Pipelines

Test analytics
Pipeline analytics
OData preview

✔
✔
✔

✔

Test Plans

Progress report

✔

Artifacts

None

Automation and connectors

Microsoft products support automation or integration with several other applications and services. For more information, see the following articles.

- Power Automate, Azure DevOps ↗
- Power Automate templates for Azure DevOps ↗
- Microsoft Power Automate documentation

Related articles

- End-to-end traceability
- Data model for Analytics
- GitHub integration

GitHub integration overview

Article • 10/11/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019

Azure Boards and Azure Pipelines provide several integration points with GitHub and GitHub Enterprise.

Sign-in with GitHub credentials

Azure DevOps simplifies deployment from your repository with seamless access to the Azure portal and Azure DevOps using your GitHub account credentials.

Feature

Description

Invite GitHub collaborators into Azure DevOps

Provides support for inviting GitHub account users to collaborate within an Azure DevOps project. For more information, see Invite GitHub collaborators into Azure DevOps (Release Notes).

Sign into Azure DevOps using your GitHub credentials

Allows users to sign in using their GitHub credentials and link their GitHub account to a Microsoft account. For more information, see Signing into Azure DevOps using your GitHub credentials (Release Notes).

Connect to a GitHub repository from Visual Studio

Provides a user interface to support cloning GitHub repositories, pushing and pulling commits, and more. For more information, see Side-by-side comparison of Git and Team Explorer.

Azure Boards and GitHub integration

By connecting Azure Boards with GitHub repositories, you enable linking between GitHub commits, pull requests, and issues to work items. You can use GitHub for

software development while using Azure Boards to plan and track your work. To get started, see Azure Boards-GitHub integration.

Feature

Description

Connect Azure Boards project to GitHub repos

Supports establishing connection of one or more GitHub repositories to an Azure Boards project.

Connect Azure Boards project to repositories hosted in a GitHub Enterprise Server instance

Supports establishing connection of one or more GitHub repositories hosted in a GitHub Enterprise Server.

Link work items to GitHub commits, pull requests, and issues. Quickly view and open linked objects from the Kanban board.

Supports linking GitHub commits, pull requests, and issues to Azure Boards work items. Mentioned work items in GitHub comments are configured as hyperlinks to support quick navigation to Azure Boards work items.

Add status badges of Azure Boards to a GitHub repository README file.

Supports adding Markdown syntax to a GitHub repo README.md file to display the status of a Kanban board. For more information, see Configure status badges to add to GitHub README files.

Work items linked to GitHub commit in Release Summary

Review list of all work items linked to GitHub commits in the Release summary page, which helps teams track and retrieve more information about the commits that have

been deployed to an environment.
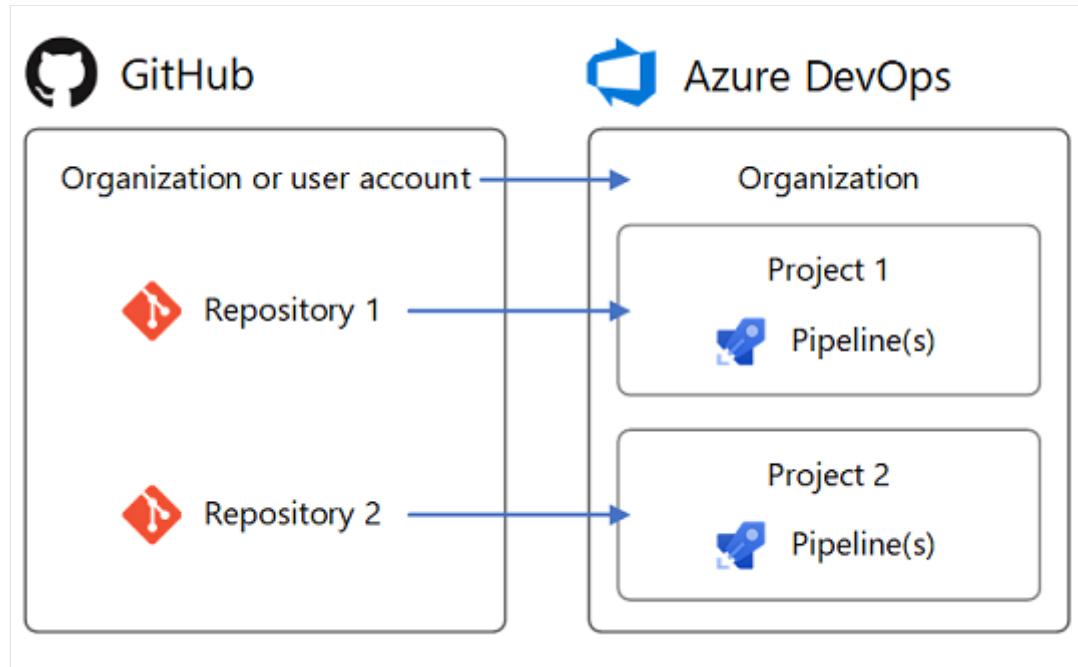
Sync GitHub Issues to Azure Boards Work Items

Using the GitHub Action, GitHub Issues to Azure DevOps you can sync your GitHub Issues to your Azure Boards. For more information, see Sync GitHub Issues to Azure DevOps Work Items (Release Notes).

Azure Pipelines and GitHub integration

You can use Azure Pipelines to automatically build, test, package, release, and deploy your GitHub repository code. To get started, see Build GitHub repositories.

You can map your GitHub repositories to one or more projects in Azure DevOps.

Feature

Description

GitHub repository and pull request builds
- Automatically build your GitHub pull requests. After the build is done, status is reported back with a comment in your GitHub pull request.
- Manually run a pipeline or test suite triggered by a GitHub pull request comment.
- Configure draft PR validation for GitHub repository. Supports adding `drafts` to the `pr` trigger YAML syntax for GitHub draft pull requests. You can choose if you want your draft PRs to queue a build. The default option is true (a build is queued) like it currently is for GitHub PRs.

- Rebuild GitHub pull request builds upon failure. Provides support for queueing a failed build.
- Configure draft PR validation for GitHub repositories
- Automatically build pull requests from repository forks to ensure changes successfully build and tests pass before they get merged. For more information, see Build GitHub repositories.

GitHub Enterprise builds

- Supports continuous integration (CI) builds for GitHub Enterprise repositories.
- Create a pipeline to build code contained within a GitHub Enterprise repository using the build pipeline wizard. For more information, see Build GitHub repositories, CI triggers.

GitHub service connections

The pipeline wizard automatically creates and reuses a service connection for the repository you choose. If you wish to manually choose a connection other than the one that is automatically selected, follow the **Choose connection** hyperlink. For more information, see Build GitHub repositories.

GitHub-specific tasks and utilities

Supported:

- Download GitHub Release task
- GitHub Release task
- Open source Azure Pipelines tasks

Manage GitHub releases

- Inline GitHub connection as a release artifact source.
- Automate GitHub releases using the **GitHub Release** task.
- Link your GitHub releases as an artifact source in release pipelines. This function lets you consume the GitHub release as part of your deployments.
  For more information, see:
- CI triggers
- Download GitHub Release task
- GitHub Release task

Filter GitHub branches for GitHub, GitHub Enterprise, or external Git artifacts

When you release from GitHub, GitHub Enterprise, or external Git repositories, you can configure the specific branches to release. For example, you might want to deploy only

builds coming from a specific branch to production. For more information, see Release triggers, Continuous deployment triggers.

Use build tags to trace GitHub sources or trigger GitHub releases

Use build tags to trace GitHub sources to builds. While choosing a GitHub repository in a build definition, you can select the types of builds you want to tag, along with the tag format.

- Use build tags to trace GitHub sources to builds. While choosing a GitHub repository in a build definition, you can select the types of builds you want to tag, along with the tag format.
- Specify a tag pattern to determine when to trigger a GitHub release. By specifying a tag regular expression, you can control when a GitHub release is created based on the triggering commit.
  For more information, see Build GitHub repositories, Label sources.

GitHub packages support in YAML pipelines

In your YAML pipeline, specify a package type (NuGet or npm) that you want to consume from GitHub. For more information, see Resources: packages.

Status checks, tracking, and traceability
- **GitHub Checks**: Display status for each pipeline job: Run a pipeline or test suite to validate a GitHub pull request from the comments section of the GitHub pull request.
- **GitHub Checks**: Send detailed information about the pipeline status, test, code coverage, and errors. Status is posted to GitHub Checks for each job in the pipeline.
- **Status badges**: Add Markdown syntax to a GitHub repo README.md file to display the pipeline status.
- GitHub artifacts: Show associated commits deployed in a release. To enhance traceability, you can see all the commits that were deployed to an environment for GitHub repositories, as a part of a specific release.
- Track GitHub commits and associated issues in releases. List commits made in GitHub repos and the associated GitHub issues that are being deployed with a release. For more information, see Track GitHub commits and associated issues in releases (Release Notes).

For more information, see:
- Create your first pipeline, Add a status badge to your repository
- GitHub Checks API

- Display status for each pipeline job in GitHub Checks (Release Notes)

Related articles

- Azure Boards-GitHub integration
- Build GitHub repositories
- Git experience in Visual Studio

Deploy to Azure

Article • 12/15/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Azure Pipelines combines continuous integration (CI) and continuous delivery (CD) to test and build your code and ship it to any target. While you don't have to use Azure services with Pipelines, Pipelines can help you take advantage of Azure. You can use Pipelines to integrate your CI/CD process with most Azure services.

Learn more about the Azure services that support continuous integration (CI) and continuous delivery (CD) with Azure Pipelines.

Azure App Configuration

- Push settings to App Configuration with Azure Pipelines
- Pull settings to App Configuration with Azure Pipelines

Azure App Service

- Deploy an Azure Web App
- Deploy an Azure Web App Container
- Use CI/CD to deploy a Python web app to Azure App Service on Linux

Azure Container Registry

- Build and push Docker images to Azure Container Registry

Azure Cosmos DB

- Set up a CI/CD pipeline with the Azure Cosmos DB Emulator build task in Azure DevOps

Azure Data Factory

- Build a data pipeline by using Azure Data Factory, DevOps, and machine learning; Configure Azure Databricks and Azure Data Factory

Azure Government

- Deploy an app in Azure Government with Azure Pipelines

Azure IoT Edge

- Continuous integration and continuous deployment to Azure IoT Edge devices

Azure Kubernetes Service

- Build and deploy to Azure Kubernetes Service

Azure Monitor

- Define approvals and checks, Query Azure Monitor Alerts

Azure MySQL Database

- Quickstart: Deploy to Azure MySQL

Azure Service Fabric

- Tutorial: Deploy an application with CI/CD to a Service Fabric cluster

Azure Static Web Apps

- Tutorial: Publish Azure Static Web Apps with Azure DevOps

Azure SQL Database

- Deploy to Azure SQL Database

Azure Virtual Machines

- Build an Azure virtual machine using an Azure RM template
- Deploy to Azure VMs using deployment groups in Azure Pipelines
- Tutorial: Deploy a Java app to a Virtual Machine Scale Set

For a complete list of Azure Pipelines tasks, see Build and release tasks.

Web portal navigation in Azure DevOps

Article • 02/21/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

The web portal for Azure DevOps is organized around a set of services as well as administrative pages and several task specific features such as the search box. The service labels differ depending on whether you work from Azure DevOps Services or Azure DevOps on premises and its version.

> ⓘ **Important**
>
>
>
> To view the content available for your platform, make sure that you select the correct version of this article from the version selector which is located above the table of contents. Feature support differs depending on whether you are working from Azure DevOps Services or an on-premises version of Azure DevOps Server. To learn which on-premises version you are using, see **Look up your Azure DevOps platform and version**

Each service provides you with one or more pages which support a number of features and functional tasks. Within a page, you may then have a choice of options to select a specific artifact or add an artifact.

Here's what you need to know to get up and running using the web portal.

- **Open a service, page, or settings**: use to switch to a different service or functional area
- **Add an artifact or team**: use to quickly add a work item, Git repo, build or release pipelines, or a new team
- **Open another project or repo**: use to switch to a different project or access work items and pull requests defined in different projects, or items you've favorited
- **Open team artifacts, use breadcrumbs, selectors and directories**: use to navigate within a service, to open other artifacts, or return to a root function
- **Work with favorites**: favorite artifacts to support quick navigation
- **Search box**: use to find code, work items, or wiki content
- **Your profile menu**: use to set personal preferences, notifications, and enable preview features
- **Settings**: use to add teams, manage security, and configure other project and organization level resources.

> ⓘ **Note**
>
> Only those services that are enabled will appear in the user interface. For example, if **Boards** is disabled, then **Boards** or **Work** and all pages associated with that service won't appear. To enable or disable a service, see **Turn an Azure DevOps service on or off**.

You select services —such as **Boards**, **Repos**, and **Pipelines**— from the sidebar and pages within those services.

Now that you have an understanding of how the user interface is structured, it's time to get started using it. As you can see, there are a lot of features and functionality.

If all you need is a code repository and bug tracking solution, then start with Get started with Git and Manage bugs.

To start planning and tracking work, see About Agile tools.

Connect to the web portal, user accounts, and licensing

You connect to the web portal through a supported web browser —such as the latest versions of Edge, Chrome, Safari, or Firefox. Only users who have been added to a project can connect. This is typically done by the organization owner.

Five account users are free as are Visual Studio subscribers and stakeholders. After that, you need to pay for more users. Find out more about licensing from Azure DevOps pricing ⤴ .

Limited access is available to an unlimited number of stakeholders for free. For details, see Work as a Stakeholder.

Refresh the web portal

If data doesn't appear as expected, the first thing to try is to refresh your web browser. Refreshing your client updates the local cache with changes that were made in another client or the server. To refresh the page or object you're currently viewing, refresh the page or choose the ↻ **Refresh** icon if available.

To avoid potential errors, you should refresh your client application under the following circumstances:

- Process changes are made
- Work item type definitions are added, removed, renamed, or updated
- Area or iteration paths are added, removed, renamed, or updated
- Users are added to or removed from security groups or permissions are updated
- A team member adds a new shared query or changes the name of a shared query
- A build definition is added or deleted
- A team or project is added or deleted

Differences between the web portal and Visual Studio

Although you can access source code, work items, and builds from both clients, some task specific tools are only supported in the web browser or an IDE but not in both. Supported tasks differ depending on whether you connect to a Git or TFVC repository from Team Explorer.

Web portal

Visual Studio

- Product backlog, Portfolio backlogs, Sprint backlogs, Taskboards, Capacity planning
- Kanban boards
- Dashboards, Widgets, Charts
- Request feedback
- Web-based Test Management
- Administration pages to administer accounts, team projects, and teams
- Git: Changes, Branches, Pull Requests, Sync, Work Items, Builds
- TFVC: My Work, Pending Changes | Source Control Explorer, Work Items | Builds
- Greater integration with work items and Office integration clients. You can open a work item or query result in an office supported client.

> **ⓘ Note**
>
> Visual Studio 2019 version 16.8 and later versions provide a new Git menu for managing the Git workflow with less context switching than Team Explorer. Procedures provided in this article under the Visual Studio tab provide information for using the Git experience as well as Team Explorer. To learn more, see **Side-by-side comparison of Git and Team Explorer**.

Resources

- Manage projects
- Project & Organizational Settings

Open a service, page, or settings

Article • 02/21/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

The web portal for Azure DevOps provides support for software development teams to collaborate through the planning, development, and release cycles. You can manage source code, plan and track work, define builds, run tests, and manage releases.

This article shows you how to navigate to functional and administrative tasks available from the web portal. There are three levels of administrative tasks: team, project, and organization.

If you don't have a project yet, create one. If you don't have access to the project, get invited to the team.

Open a service or functional task page

Services support getting work done—managing code, planning and tracking work, defining and managing pipelines, creating and running tests, and so on.

> ⓘ **Note**
>
> Only those services that are enabled will appear in the user interface. For example, if **Boards** is disabled, then **Boards** or **Work** and all pages associated with that service won't appear. To enable or disable a service, see **Turn an Azure DevOps service on or off**.

You open a service by choosing the service from the sidebar and then selecting from the available pages.

For example, here we select **Boards>Backlogs**.

Within the page you may select a specific view or artifact, such as a team backlog or choose another page.

Open team settings

Select configurations are made to teams through the team settings pages. For an overview of all team settings, see About user, team, project, and organization-level settings.

1. Choose **Project Settings**.

2. Expand **Boards** and choose **Team configuration**.

3. Choose one of the pages **General**, **Iterations**, **Areas**, or **Templates** to configure settings for the team. To learn more, see Manage teams.

4. If you need to switch to a different team, use the team selector within the breadcrumbs.
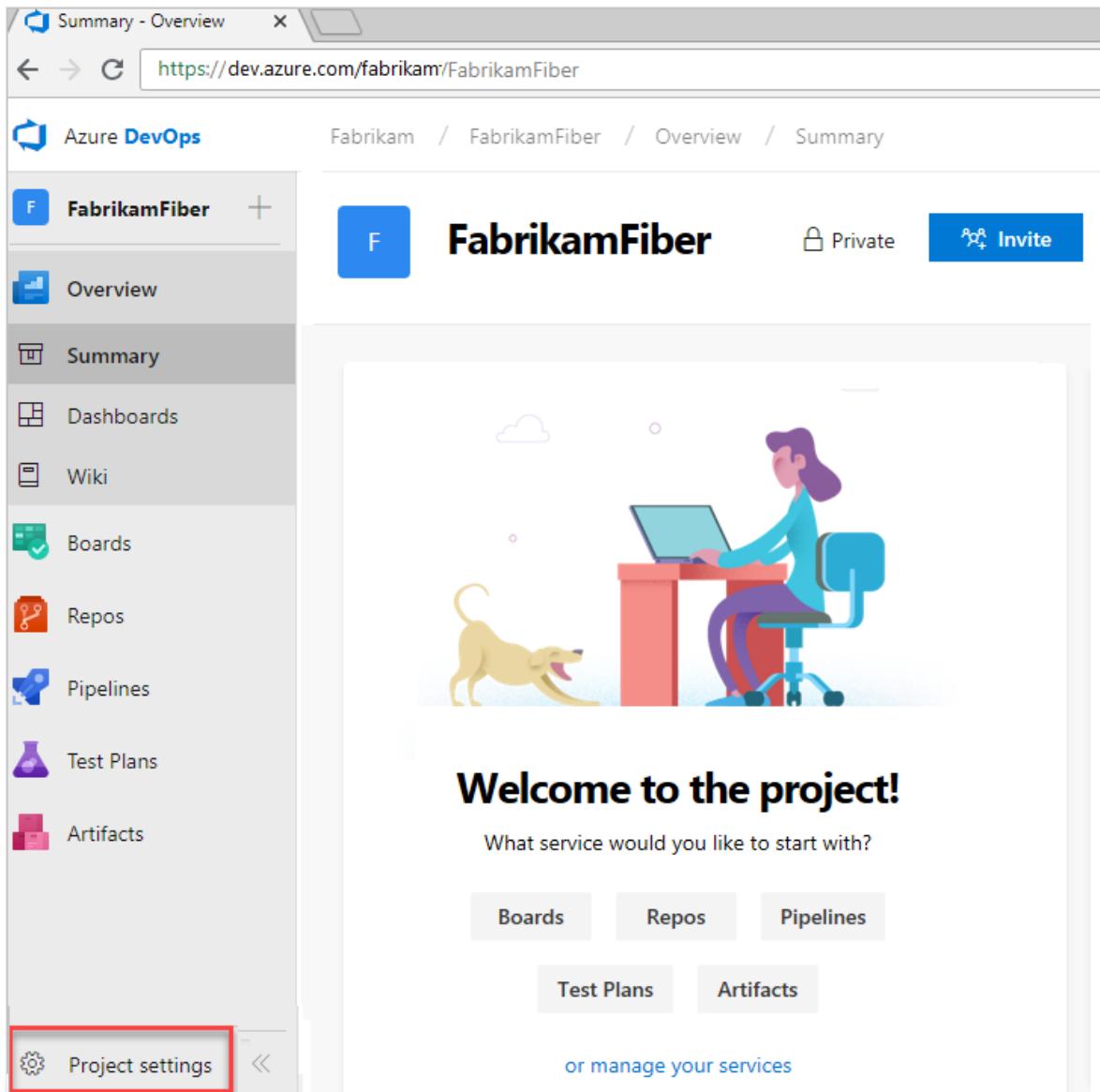
5. To add a team administrator, add team members, or change the team profile, choose **Teams** from the vertical sidebar, and then choose the name of the team you want to configure.

Open project settings

Administrators configure resources for a project and manage project-level permissions from the **Project settings** pages. Tasks performed in this context can impact the project and team functions. For an overview of all project settings, see Project administrator role and managing projects.

1. Choose **Project Settings**.

2. From there, you can choose a page from the list. Settings are organized based on the service they support. Expand or collapse the major sections such as **Boards**, **Build and release**, **Code**, **Test**, and **Extensions** to select from the list.

Open Organization settings

Organization owners and members of the Project Collection Administrators group configure resources for all projects or the entire organization, including adding users, from the Organization settings pages. This includes managing permissions at the organization-level. For an overview of all organization settings, see Project collection administrator role and managing collections of projects.

Related articles

- Manage projects
- About team, project, and admin settings

Add an artifact or team artifacts

Article • 02/21/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Select the service of interest to get started adding new artifacts or objects. For example, to add work items, choose **Boards** or **Work**. Some artifacts—such as a product backlog, Kanban board, portfolio backlogs—are added when you add a team.

Prior to adding an artifact, make sure that you've selected the project and repository that you want to work in.

Add work items, queries, or other work tracking artifacts

You can quickly add a query or work item when working from a **Boards** or **Work** page.

Choose a **Boards** page—such as **Work Items**, **Boards**, or **Backlogs**. Then choose the +
plus icon and select from the menu of options.

To add other work tracking artifacts, see one of the following articles:
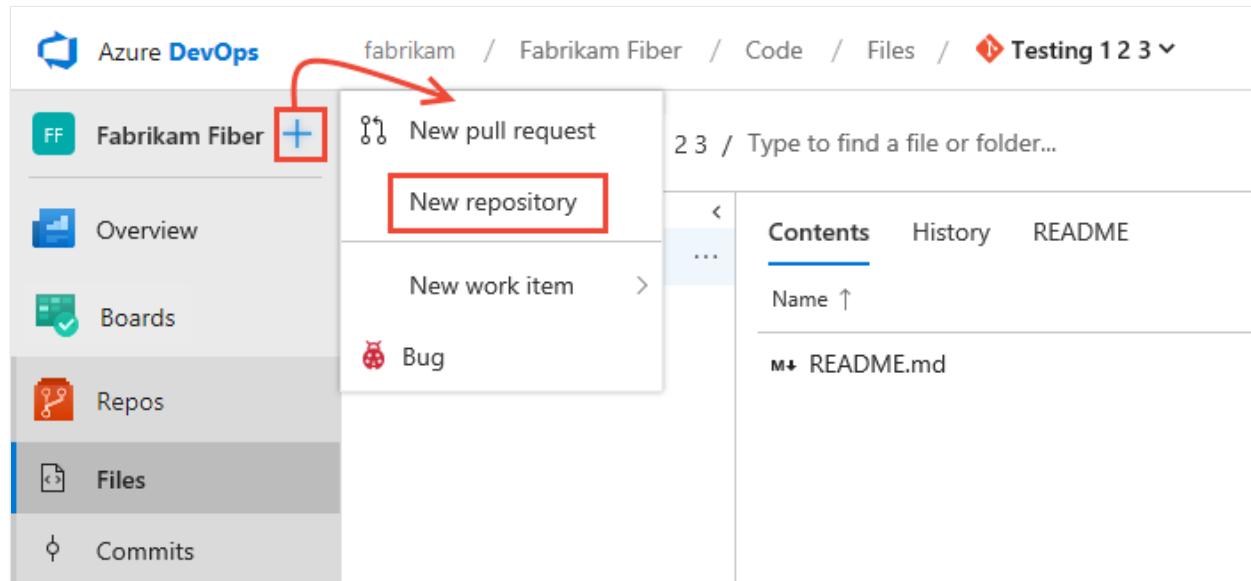
- To add a board, backlog, or sprint backlog, first add a team which will be associated with those artifacts
- Add a delivery plan
- Add a managed work item query

- Add work items.

Add a pull request or Git repository

You can quickly add a pull request, Git repository, or work item using the **Add** menu when working from **Code**.

Expand the **Repos** service and choose **Files**, **Commits**, or **Pull Requests** (Git repos) or **Files**, **Changesets**, or **Shelvesets** (TFVC). Then, choose the ✚ plus icon and select from the menu of options.
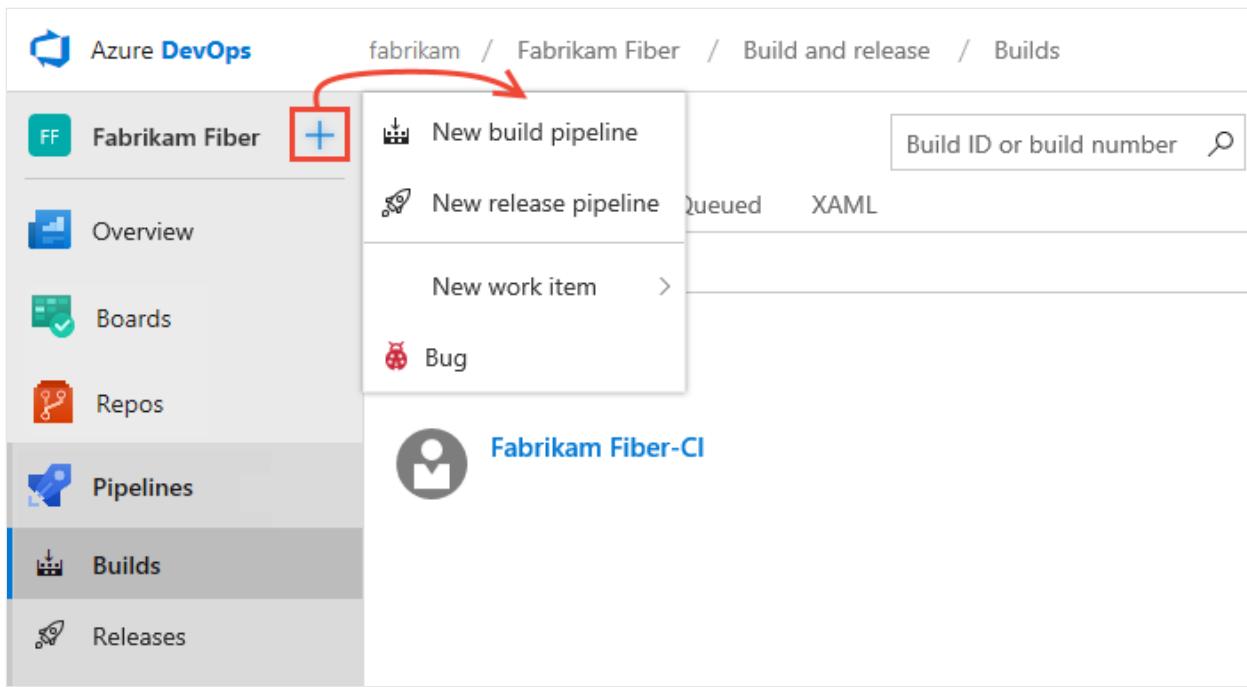
For details on adding a Git repository, see Git repository.

Note that you can only add one TFVC repository per project, but an unlimited number of Git repositories. To learn more about Git artifacts, see one of the following articles:

- Git repository
- Git branch
- Git pull request
- Add work items

Add build and release pipelines

Expand **Pipelines** and choose **Builds** or **Releases**. Then choose the ✚ plus icon and select from the menu of options.

To learn more about adding other pipeline related artifacts, see the following articles:

- Deployment groups
- Task groups
- Variable groups
- Secure files

Add a team

Agile tools and dashboards are typically associated with teams. You add teams to a project. To learn more about teams, see About teams and Agile tools. To add a team, see Add a team and team members.

View teams already defined

To view the set of defined teams, open **Project settings**, and choose **Overview**.

Add a dashboard

Dashboards are associated with a team or a project. Each team can create and configure a number of dashboards. And, any team member can create one or more project dashboards. To learn how, see Add a dashboard.

Add a wiki

If you don't have a wiki yet, you can add one. Once added, you can add and update pages to that wiki.

- Create a wiki
- Add and edit wiki pages
- Publish a Git repository to a wiki

Related articles

- Azure Artifacts
- Exploratory & Manual Testing

Use breadcrumbs, selectors, and directories to navigate and open artifacts
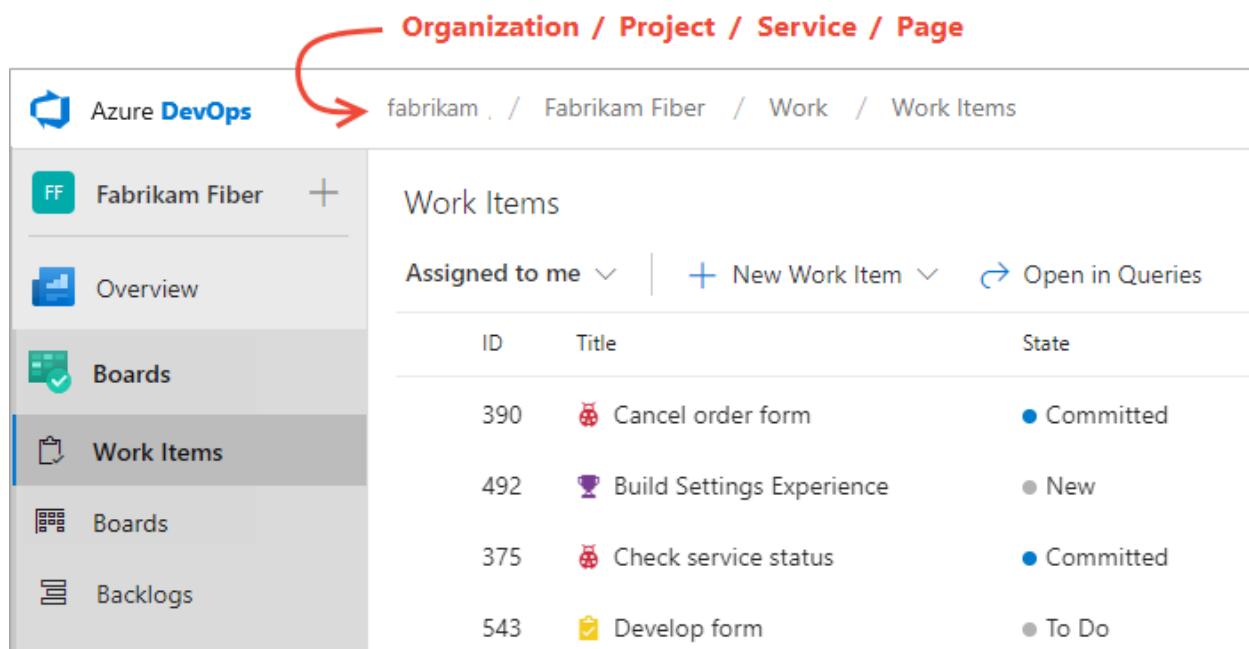
Article • 02/21/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

To quickly navigate to a feature or artifact—such as a dashboard, repository, product backlog, Kanban board, build pipeline—you can use breadcrumbs, selectors, and directories.

Organization and project breadcrumbs

To navigate to the project summary page, choose the project link within the breadcrumbs. To navigate to the organization page with all projects defined for the organization, choose the organization link.
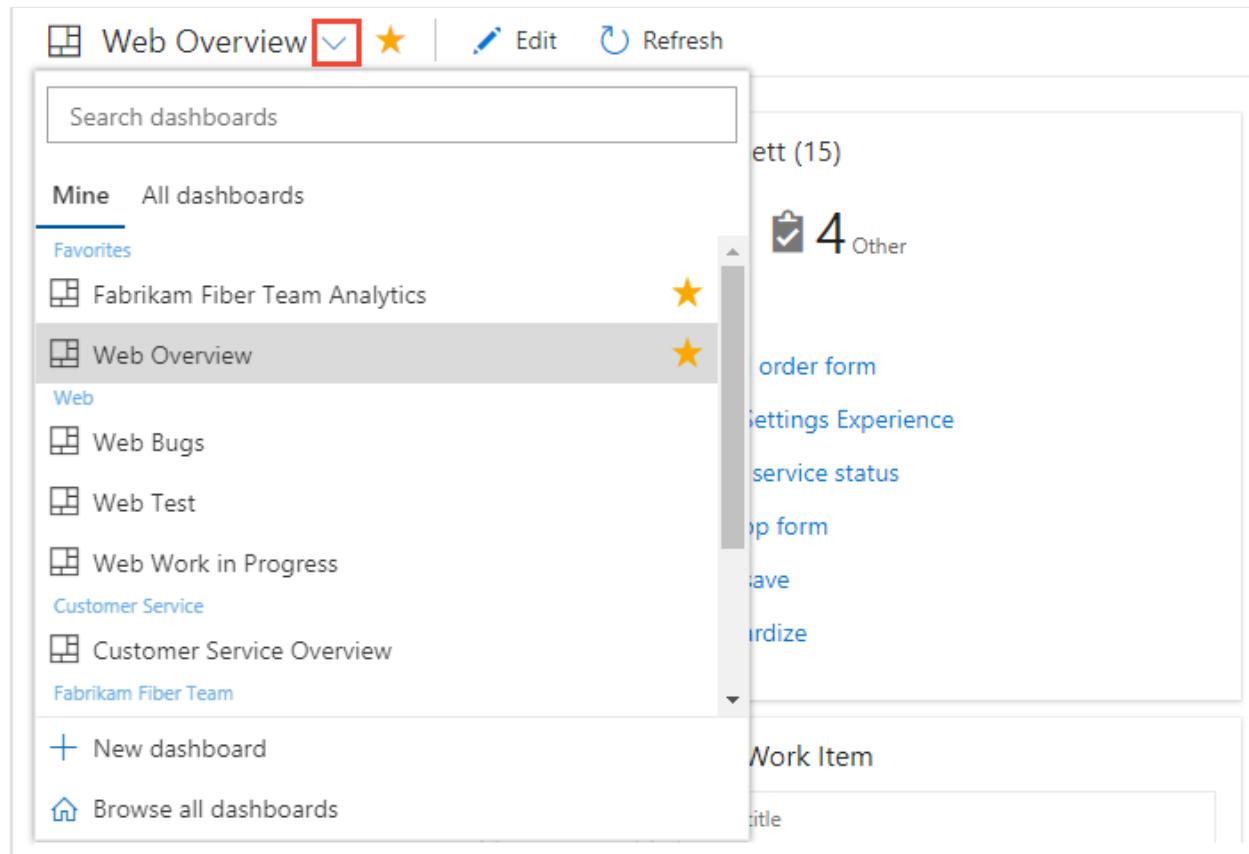
Selectors

Selectors are used to select an artifact within the current page. Most Agile tools are defined for a team and therefore require selection of the team artifact or tool.

> ⓘ **Note**

When you navigate to a specific page or artifact, the system remembers your selection. You use selectors to choose a different artifact within the current page.

Example: Dashboard selector

Within **Dashboards**, you open a specific dashboard from the selector.

This particular selector features these navigational elements:

- Search box for filtering dashboards based on a team name or keyword
- Two pages you can choose from:
  - **Mine** (dashboards you created) which are organized by team
  - **All** (dashboards created by everyone) which are listed alphabetically
- Dashboards you've favorited will appear at the top of the selector
- **Add new dashboard** feature
- **Browse all dashboards** - opens **Dashboards>All**

Example: Backlogs

From the **Boards>Backlogs** page, you use the selector to switch to another team's backlog. Again, favorited backlogs appear towards the top of the menu. You can also filter the list based on a team name or keyword.

Or, choose **Browse all team backlogs** to open the **Backlogs>All** page.

Artifact breadcrumbs and selectors

Within select pages, breadcrumbs are provided to support navigating within the page or opening an artifact.

Example: Queries folders and breadcrumbs

For example, when working in the **Queries** pages, you can navigate to a subfolder, folder, or page.

Also, you can choose a query that you've favorited from the selector menu, Or, you can choose to browse all queries which returns you to the **All Queries** page.

Example: Pipeline folders and breadcrumbs

Breadcrumb-and-selector navigation elements are used within most services that support defining and organizing artifacts within folders. This includes **Pipelines** or **Build and Release** applications pages.

Directories

Directories provide a filterable list of all artifacts defined for a service area. Often when you navigate to an application, it will open the application's directory.

For example, here is the **Boards>Boards** directory.

It lists boards in the following order:

- Your last visited board
- Your favorited boards
- All boards of teams that you belong to
- All boards defined for the project in alphabetical order.

Choose the ▽ filter icon to filter the list as described in Filter basics.

From a specific page, you can open the directory from the breadcrumbs or a selector. For example, choose **Browse all boards** from the Boards selector.

Open from breadcrumb

Open from selector

Team profiles

- Open a team profile to quickly access items defined for a team. The team profile is available from the **Overview>Dashboards**, **Boards>Boards**, **Boards>Backlogs**, and **Boards>Sprints** pages.
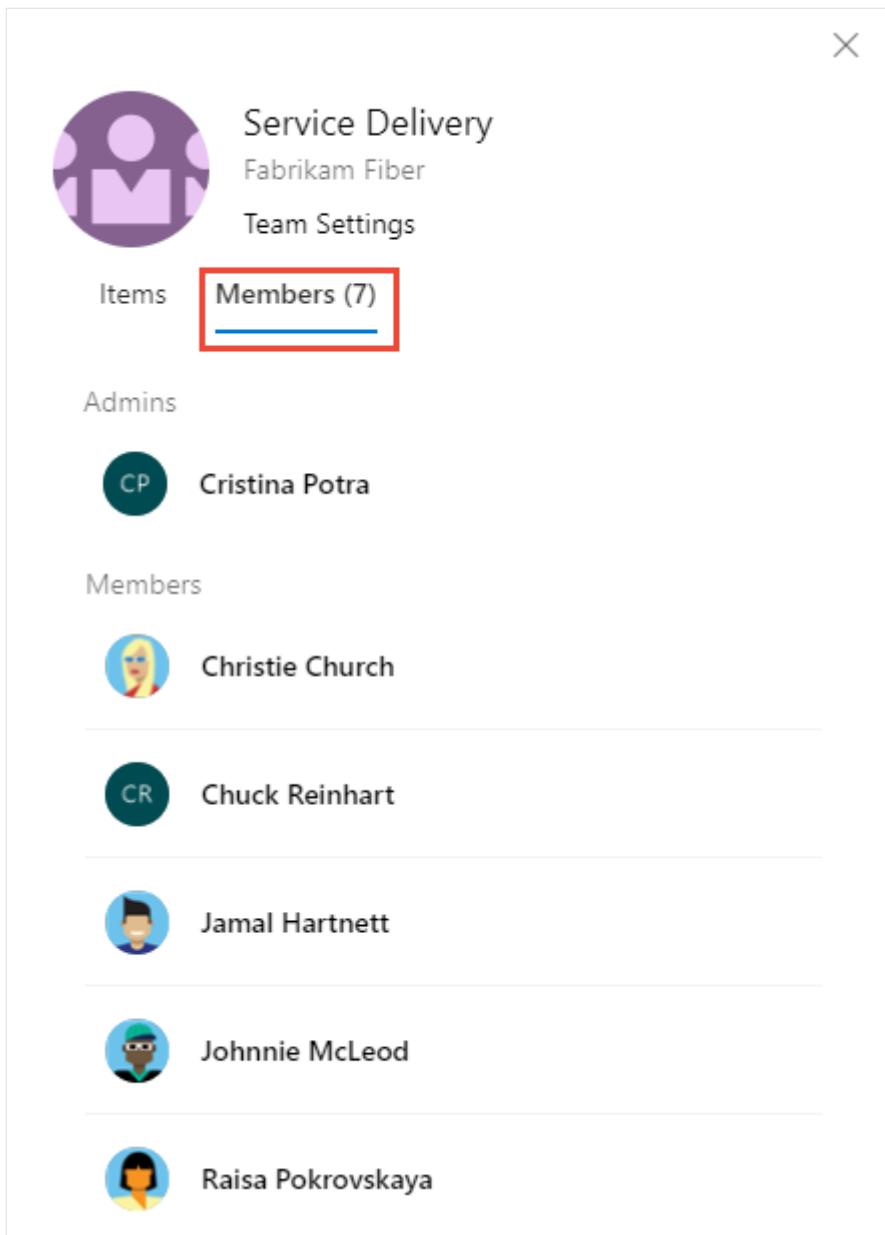
A panel opens that shows all items defined for the team.

- You can filter the list to show only **Dashboards**, **Boards**, **Backlogs**, or **Sprints** by choosing from the menu.

- To view the team admins and members of the team, choose **Members**.

- To view or change the team configuration, choose **Team Settings**.
  You can then add team members, team admins, or navigate to team notifications, or team iterations and area paths.

See also Manage and configure team tools.

Related articles

- About teams and Agile tools
- Add an artifact or team
- Set favorites
- Open a service or page
- Filter basics

Switch project, repository, team

Article • 03/23/2023

**Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS
2018**

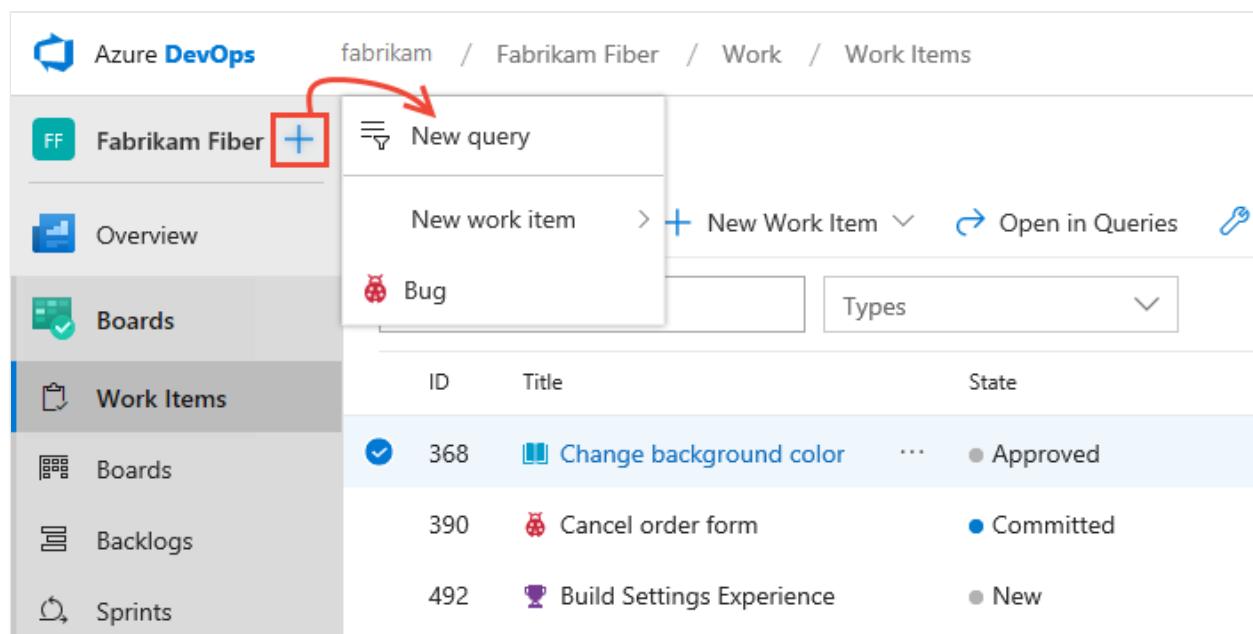Several features depend on the project, repository, or team that you have selected. For
example, dashboards, backlogs, and board views will change depending on the project
and team you select.

Also, when you add a work item, the system references the default area and iteration
paths defined for the team context. Work items you add from the team dashboard (new
work item widget) and queries page are assigned the team default iteration. Work items
you add from a team backlog or board, are assigned the team default backlog iteration.
To learn more, see About teams and Agile tools.

Prerequisites

- You must be added to a project as a member of the **Contributors** or administrator
  security group. To get added, Add users to a project or team.

> ⓘ **Note**
>
> If the **Limit user visibility and collaboration to specific projects** preview feature is
> enabled for the organization, users added to the **Project-Scoped Users** group won't
> be able to access projects that they haven't been added to. For more information
> including important security-related call-outs, see **Manage your organization,
> Limit user visibility for projects and more**.

View and open a project

From the **Projects** page you can quickly navigate to a project that you have permissions
to view.

1. Choose the ☁ Azure DevOps logo to open **Projects**.

The projects you most recently viewed are displayed, followed by a list of all projects in alphabetic order.

2. Hover over the dots and you can open the service of interest for that project.
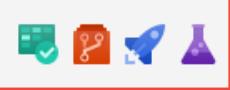
3. You can filter the project and team list using the *Filter projects* search box. Simply type a keyword contained within the name of a project or team. Here we type **Fabrikam** to find all projects or teams with *Fabrikam* in their name.

4. Choose **Create Project** to add a project. You must be an account administrator or a member of the Project Collection Administrators group to add a project.

View and open a repository

1. Choose **Repos>Files**.

2. Select the repository of interest from the repository selector.

Switch to a different team

From a user page, one under—**Boards**, **Repos**, **Pipelines**, or **Test Plans**—you can't switch to a different team, you can only select team artifacts.

From a **Project Settings**>**Work**>**Team configuration** page, you select a team from the team selector breadcrumb.

Related articles

- Work across projects
- Add teams

Tutorial: Set personal or team favorites

Article • 02/21/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Favorite ⭐ those views that you frequently access. You can favorite all sorts of Azure DevOps features and tools—such as a project, repository, build pipeline, dashboard, backlog, board, or query. You can set favorites for yourself or your team.

As your code base, work tracking efforts, developer operations, and organization grows, you'll want to be able to quickly navigate to those view of interest to you and your team. Setting favorites allows you to do just that.

Team favorites are a quick way for members of your team to quickly access shared resources of interest. You favorite an item for yourself by choosing the ⭐ star icon. The favorited item will then show up easily from one or more directory lists. You set favorites for a team through the context menu for the definition, view, or artifact.

In this tutorial you'll learn how to view your personal favorites and to favorite or unfavorite the following views:

- ✔ Project or team
- ✔ Dashboard
- ✔ Team backlog, board, shared query, or other Azure Boards view
- ✔ Repository
- ✔ Build and release definition
- ✔ Test plans
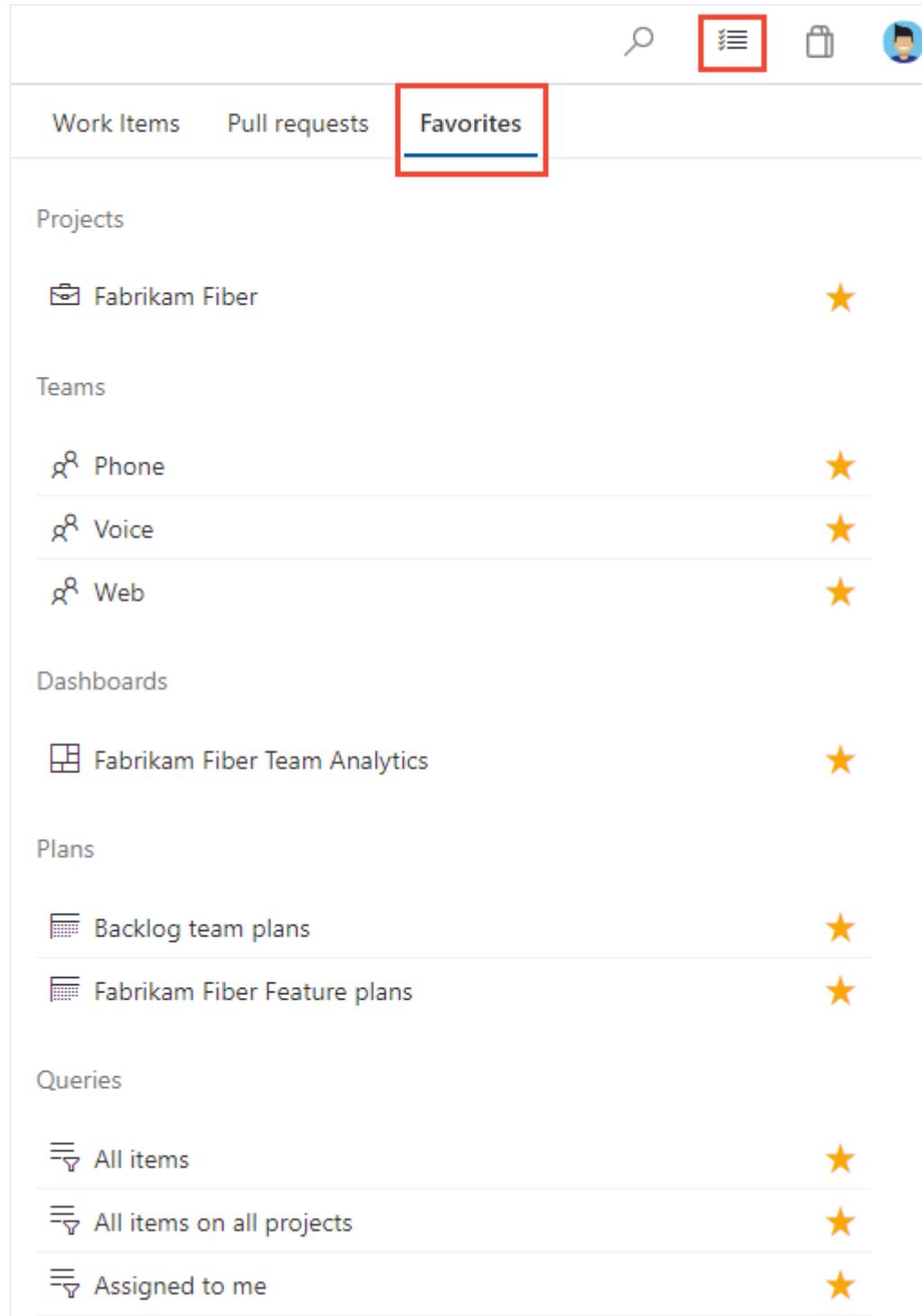
Prerequisites

- You must connect to a project through the web portal. If you don't have a project yet, create one. To connect to the web portal, see Connect to a project.
- You must be a member of the **Contributors** or an administrators security group of the project. To get added, Add users to a project or team.
- To favorite projects, backlogs, boards, queries, dashboards, or pipeline views, you must have **Stakeholder** access or higher.
- To favorite repositories, or delivery plans, you must have **Basic** access or higher.
- To favorite test plans, you must have **Basic + Test Plans** access level or equivalent.

For details about the different access levels, see About access levels.

View personal favorites

Access views that you have favorited by choosing the ≣ inbox icon, and then choosing **Favorites**.

> ⓘ **Note**
>
> If a service is disabled, then you can't favorite an artifact or view of that service. For example, if **Boards** is disabled, then the favorite groups—Plans, Boards, Backlogs, Analytics views, Sprints, and Queries and all Analytics widgets—are disabled. To re-enable a service, see **Turn an Azure DevOps service on or off**.

Favorite a project or team

1. To favorite a project, open the project **Summary** page and choose the ⭐ star icon.
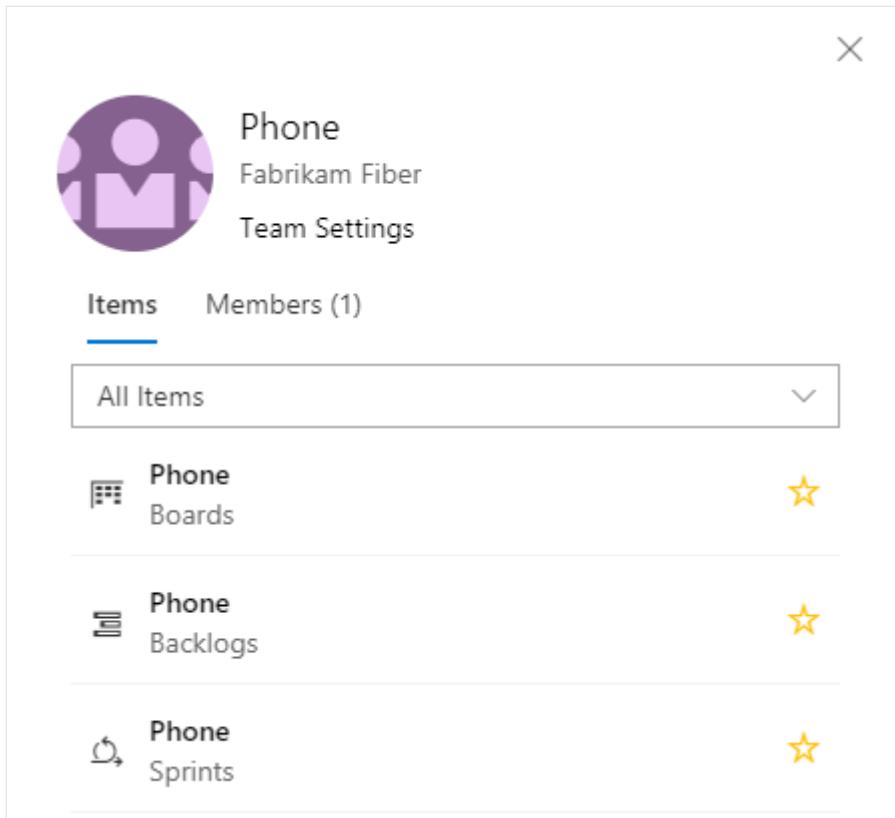
2. To favorite a team artifact, open **Boards>Boards** or **Boards>Backlogs**. Select the team you want to favorite from the team selector and choose the ⭐ star icon.

3. To favorite other team artifacts, choose the 👥 team icon, and then choose the ⭐ star icon next to one of the listed artifacts.

Favorite a dashboard

1. From **Overview**>**Dashboards**, open the selector and choose the **Browse all dashboards** option.

2. The **Mine** page shows your favorited dashboards, and all dashboards of teams that you belong to. The **All** page (shown below) lists all dashboards defined for the project in alphabetical order. You can filter the list by team or by keyword.

> 💡 **Tip**
>
> You can change the sort order of the list by choosing the column label.

3. To favorite a dashboard, hover over the dashboard and choose the ⭐ star icon.

Favoriting a dashboard will cause it to appear on your **Favorites** page and towards the top in the **Dashboards** selection menu.

Favorite a team's backlog, Kanban board, or other view

You can favorite several Agile tools for a team from a **Boards** page.

1. Choose **Boards**, and then choose the page of interest, such as **Boards**, **Backlogs**, or **Sprints**.

   For example, here we choose (1) **Work** and then (2) **Backlogs**.

To choose a specific team backlog, open the selector and select a different team or choose the 🏠 **Browse all team backlogs** option. Or, you can enter a keyword in the search box to filter the list of team backlogs for the project.

2. Choose the ☆ star icon to favorite a team backlog. Favorited artifacts (★ favorited icon) appear on your **Favorites** page and towards the top of the team backlog selector menu.

Favorite a shared query

Open **Boards>Queries** and choose the **All** page. Expand a folder as needed. Choose the ⭐ star icon next to the query you want to favorite.

Or, open the context menu of the query, and then select **Add to Team Favorites**, and then select from the list of teams.

> ⓘ **Note**
>
> You must be a member of at least one team for the **Add to Team Favorites** option to be visible. If not visible, ask your project administrator or team administrator to add you to a team.

You can also set a query as a personal favorite by opening the query and choosing the ⭐ star icon.
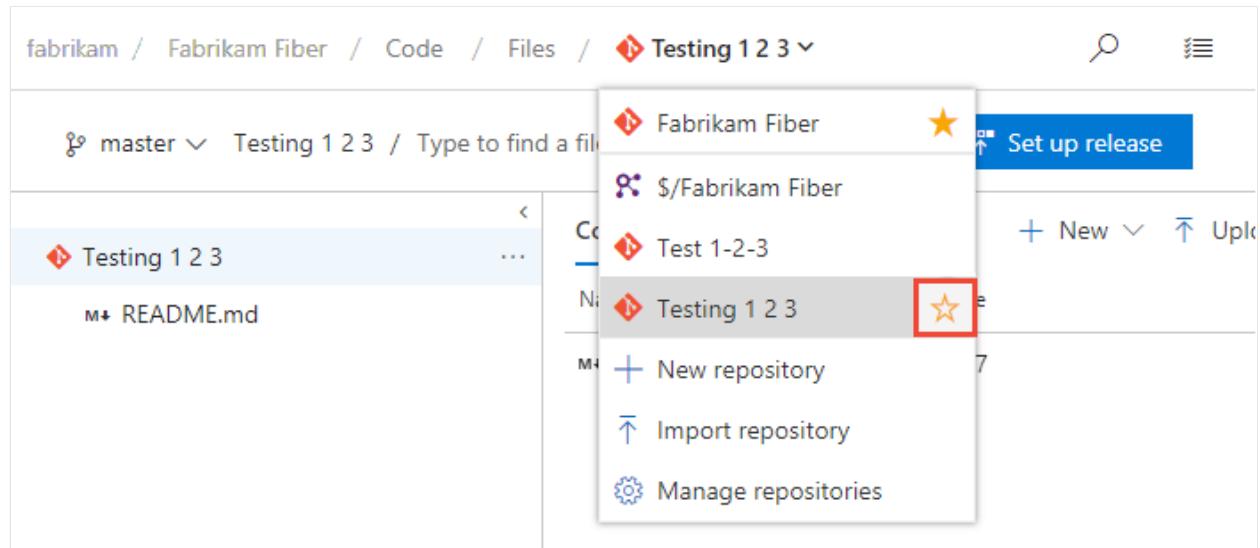
Favorite a delivery plan

To learn more about delivery plans, see [Review team Delivery Plans](#).

To mark a delivery plan as a favorite, open the **Boards>Plans** page and choose the ☆ star icon next to the Delivery Plan.

Favorite a repository

From any **Repos** page, open the repository selector and choose the ☆ star icon for the repository you want to favorite.

Favorite a build pipeline

Open **Pipelines>Builds** and choose either **Mine** or **Definitions** page. Choose the ☆ star icon next to the build definition you want to favorite. Or, open the context menu of the build definition, and then select **Add to my favorites** or **Add to team favorites**.

Favorite a test plan

To learn more about test plans, see [Create a test plan and test suite](#).

To mark a test plan as a favorite, open **Test Plans>Test Plans** and choose the ☆ star icon next to a test plan from the menu that shows All test plans.

Unfavorite a view you've favorited

You can unfavorite an artifact from your **Favorites** page. Choose the inbox icon, and then choose **Favorites**. Choose the ⭐ favorited icon of a currently favorited artifact.

Similarly, you can unfavorite an artifact from the same page where you favorited it.

Try this next

Follow a user story, bug, issue, or other work item or pull request

Related articles

- Manage personal notifications
- Set your preferences

Filter lists, boards, and directories

Article • 02/21/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Several applications and pages support filtering, which is very useful when a large number of artifacts or items have been defined. Most directory views provide one or more filter functions.

You can filter most items using keywords or a user name for an author of an item or where work is assigned to them. You can filter lists and boards in the following areas:

- Git repositories: Branches, Commits, Commit history, Pull Requests, Pushes, and Repositories
- Work tracking: Work Items, Kanban boards, Backlogs, Sprint Backlogs, and Taskboards
- Directories: Dashboards, Boards, Backlogs, Sprints, Queries, Builds, Releases

> ⓘ **Note**
>
> You may have fewer or additional filter options based on the **features you've enabled** or the platform and version that you are working from.

Filter based on keywords, tags, or fields

To turn filtering on, choose the ▽ filter icon.

You can filter work items by typing a keyword or using one or more of the fields provided, such as work item type, assigned to, state, and tags. Based on the keyword that you enter, the filter function will list work items based on any visible/displayed column or field, including tags. Also, you can enter a value for an ID, whether or not the ID field is visible.

The filtered set is always a flat list, even if you've selected to show parents.

Characters ignored by keyword filter criteria

The filter criteria ignores the following characters: `,` (comma), `.` (period), `/` (forward slash), and `\` (back slash).

Filter directories

Choose the ⧩ filter icon to filter a directory list by keyword, team, or other supported field. Keywords apply to titles, descriptions, and team names.

For example, here we turn filtering on for the dashboard directory.

Dashboards

Name	Team
Continue where you left off	
⊞ Analytics	☌ Fabrikam Team
⌄ **My favorite dashboards (2)**	
⊞ Overview ★	☌ Account Management
⊞ Team Guidance ★	☌ Fabrikam Team
⌄ **Account Management (1)**	
⊞ Overview ★	☌ Account Management
⌄ **Customer Profile (1)**	
⊞ Overview	☌ Customer Profile
⌄ **Fabrikam Team (5)**	
⊞ Analytics	☌ Fabrikam Team

Filter by text — Team ⌄ ✕

🔍 |

☐ Account Management
☐ Customer Profile
☐ Fabrikam Team
☐ Phone
☐ Service Delivery
☐ Service Status
☐ Shopping Cart
☐ TV

✕ Clear

Related articles

- [Commit history](#)
- [Working with Git tags](#)
- [Filter backlogs and queries](#)
- [Filter your Kanban board](#)
- [Add tags to work items](#)

Get started with search

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

You can quickly find work items, code files, wiki pages, or packages based on a keyword, wildcards, and other supported search filters with the search function.

See the following quick links to more information:

- Code Search Marketplace extension ↗
- Search features, further in this article
- Functional code search

Prerequisites

- Every project member can use the search functions, including project members granted Stakeholder, Basic, and higher levels of access.
- When you search across the organization or collection, only results for which a project member has access are listed.
- Stakeholder wiki search results are limited to provisioned wikis. Because published wikis require access to regular repositories, which Stakeholders don't have access to, results for published wikis don't appear in the search results. Similarly, Code search results don't appear for Stakeholders.

Start your search with a keyword

Start your search using a keyword. You can then apply other options, as needed, to broaden or narrow your search results.

- To get results that match the input, you may need to remove filters and search again. After you see the search results, you can narrow them down by applying appropriate filters and searching again.
- Make sure your search terms are spelled correctly. Work item search doesn't ignore spelling errors.
- You might get a message that no matching files are found if you use a wildcard search with too many hits, such as a simple wildcard search string. In this situation, make your search more specific to decrease the number of matches. Add more characters of the word or words you want to find, or use a condition or filter to restrict the number of possible matches.
- Searches aren't case sensitive.

Search features, usage, and examples

The following features apply to all searches, including work items, code, wikis, and packages.

Search feature

Usage

Example

Keyword

Search based on one or more keywords.

`validate` finds instances that contain the word *validate*.

Exact match

Search based on an exact match, enclosed in double-quotes.

`"Client not found"` finds instances that contain the exact phrase match *Client not found*.

Wildcard
- Add wildcard characters, `*` and `?`, to keywords to extend the search criteria.
- Add `*` at the end of a keyword to find items that start with the keyword.
- Add `?` in the middle to represent any alphanumeric character.
- Use wildcard characters anywhere in your search string except as a prefix. You can use prefix wildcards with the other search filter functions.
- You can use more than one wildcard to match more than one character.
- `alpha?version` finds instances of alpha1version and alphaXversion.
- `Browser*` finds instances of BrowserEdge, BrowserIE, and BrowserFirefox.
- `CodeSenseHttp*` finds files containing words that start with *CodeSenseHttp*, such as CodeSenseHttpClient and CodeSenseHttpClientTest.

Boolean operators
- Find two or more keywords using Boolean operators: `AND`, `OR`, and `NOT` (must be uppercase).
- Add parenthesis to clauses to support logical groupings.
- Because `AND` is the default operator, an entry of two keywords with no operator is the same as an `AND` search.
- `Validate AND revisit` finds files that contain both the words *validate* and *revisit*.
- `Validate OR revisit` finds files that contain either of the words *validate* or *revisit*.

- `Validate NOT revisit` finds files that contain the word *validate* but not the word *revisit*.
- `(Validate NOT revisit) OR "release delayed"` finds files that contain the word *validate* but not the word *revisit* or files that contain the phrase *release delayed*.

Proximity

- Search for files based on vicinity using proximity operators: NEAR, BEFORE, and AFTER (must be uppercase).
- By default, proximity search looks for terms within five tokens distance.
- `term1 BEFORE term2` returns all files where term1 occurs BEFORE term2 within a distance of five tokens between them.
- `term1 AFTER term2` returns the same results as term2 BEFORE term1.
- `term1 NEAR term2` returns all files where term1 is within five token distance from term2 in any direction. `term1 NEAR term2` returns the same results as `term1 BEFORE term2` OR `term2 BEFORE term1`.

Special characters

- Escape the special characters `(`, `)`, `[`, `]`, `:`, `*`, and `?` by enclosing them in a phrase delimited with double-quotes.
- Include special characters in a search string, or search specifically for special characters, according to the following rules:
- CodeA23?R finds files containing words that start with CodeA23
- Have any alphanumeric character next, and end with R. For example, CodeA234R and CodeA23QR.
- Search for any special character that isn't a part of the query language.
- `"flatten()"` finds the literal string *flatten()*. Search for a literal occurrence of the double-quote character " by preceding it with the escape character `\` and enclosing the search string in double-quotes.
- `"\"react-redux\""` finds the literal string "react-redux."

Search from a different page

You can search from any of the following pages:

- Organization project page: Starts a search across all projects.
- Project overview page: Automatically applies a filter to search within the selected project.

- Boards page for a project: Automatically displays recent work items and backlogs accessed by the user.
- Azure Repos, Pipelines, Test Plans, or an Artifacts page for a project: Automatically displays functional filters for code searches.
- Wiki page: Automatically go to a wiki page you recently opened.

For more information, see the following articles:

- Filter backlogs, boards, and plans.
- Provisioned vs. published wiki.

> 💡 **Tip**
>
> **No results found for ...**
> Too many hits from a simple wildcard search can result in no matching files. You can narrow your search by adding more characters or using a condition or filter.

More search functions

See the following table for more search tasks and actions.

Search task

Action

Find an organization setting

Go to your organization and select **Organization settings**.

Find a project setting

Go to your project and select **Project settings**.

Find a user setting

Go to your **User settings page**.

Find a user

Go to your organization and select **Organization settings** > **Users**, and then enter the name in the filter box.

Find an organization

Scroll through the left side of your screen, which lists all organizations.

Find a project

Go to your organization, and then enter the project name in the Filter projects box.

View file history and compare versions

Go to **Repos** > **Files**, highlight your file, and then select **History**.

> ⓘ **Note**
>
> When you search from the **Organization settings** page, your search results include both organization-level and project-level settings.

Marketplace extensions

- Code search ↗ - Extends search with fast, flexible, and precise search results across all your code. Required for searching repositories.
- Azure Paths Search ↗ - Adds a special search hub to Boards for searching within iterations and area paths without having to create and maintain custom queries.

> ⓘ **Note**
>
> Azure DevOps doesn't support some extensions. For more information or assistance, go to the **Visual Studio Marketplace** ↗.

Next steps

Functional code search

Related articles

- Functional work item search
- Functional artifact or package search
- Code search blog posts ↗

- Work item search blog posts ↗

Manage preview features

Article • 10/19/2023

Azure DevOps Services | Azure DevOps Server 2022 | Azure DevOps Server 2020
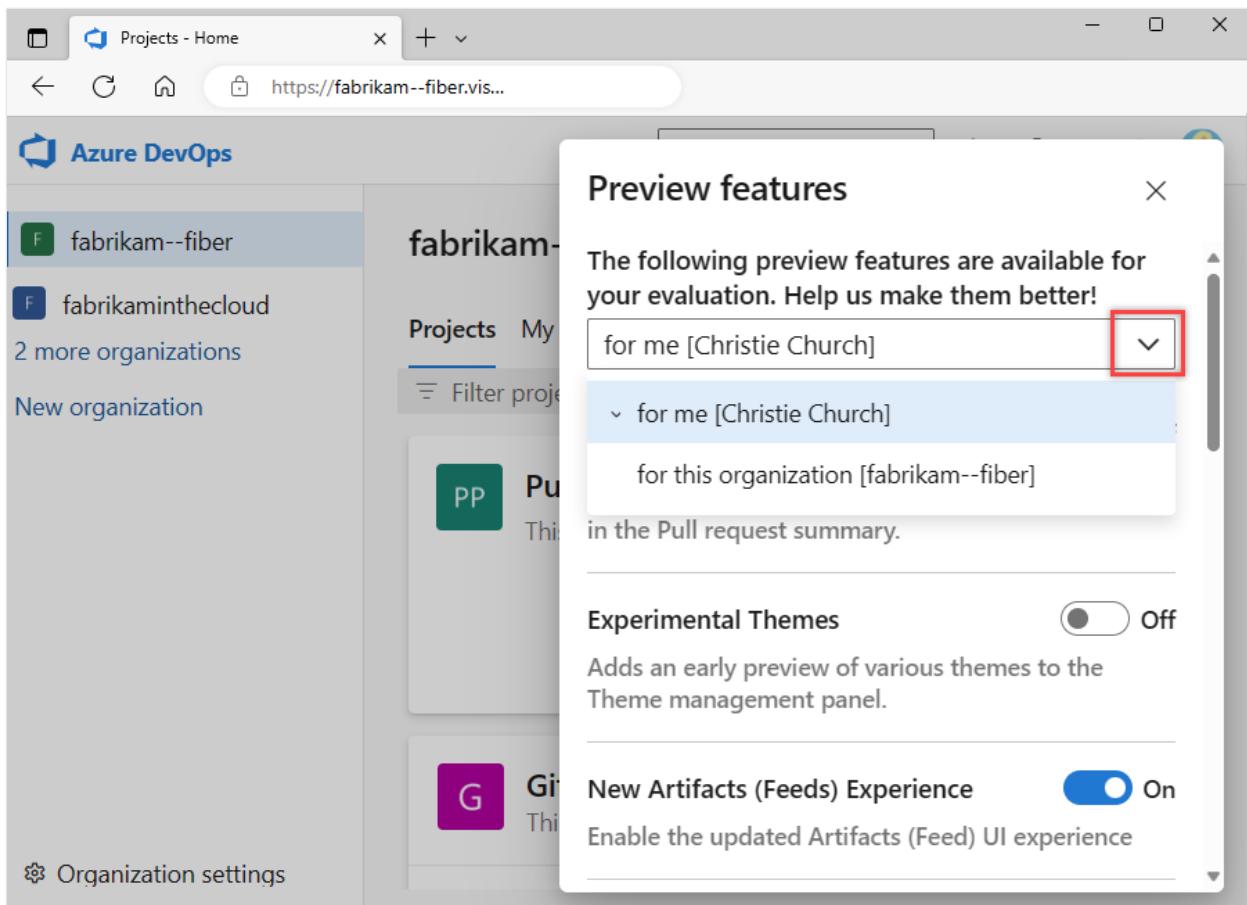
As some new features get introduced, you can turn them on or off. You can try them out, provide feedback, and work with the ones that meet your requirements. Some preview features provide access to new functionality, while others reflect a change to the user interface, but little or no change in functionality.

> ⓘ **Note**
>
> It might take up to three weeks after a release to Azure DevOps for the preview feature to appear in your organization. The **latest release notes** usually provide information on new preview features. You can turn on or off select features for Azure DevOps. Preview features become available first on Azure DevOps Services and then become standard features with an update to Azure DevOps Server. At some point, the preview feature moves out of preview status and becomes a regular feature of the web portal.

Turn on or off preview features

Select either your organization or personal settings from the dropdown menu and slide the toggle to **on** or **off** to change the feature status.

Enabling a feature at the organization level activates it for all users. They can still turn it off individually. Disabling a feature at the organization level doesn't affect user settings. Users can change feature status on their own. For more information, see Set user preferences.

> 💡 **Tip**
>
> If you don't see the **for this account** menu option, then you aren't a member of the Project Collection Administrators group. To get added as one, see **Change project collection-level permissions**.

Preview features

The following table shows the preview features you can turn on or off based on your role: user or team member, or for the organization. Only Project Collection Administrators can manage organization-level features.

Preview features	Per user	Per organization
Pull Request Summary - Load of large files	✔	
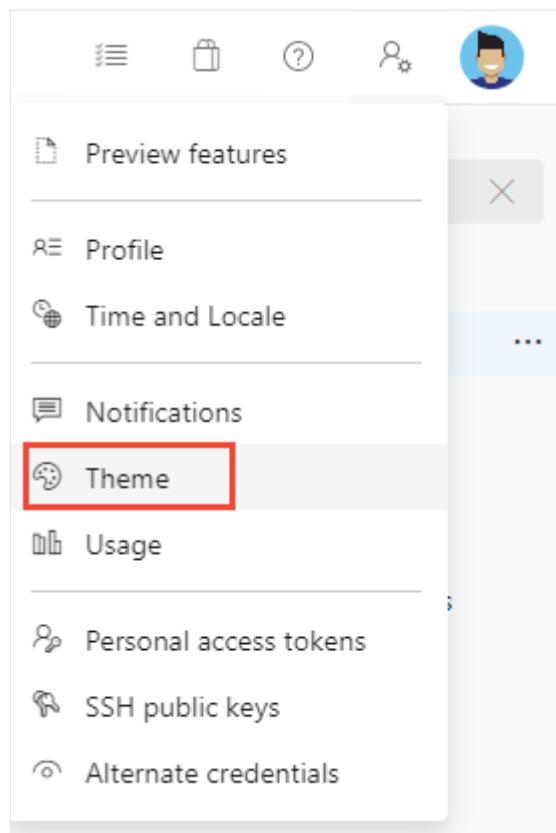
Preview features	Per user	Per organization
Experimental themes	✔	✔
Limit user visibility and collaboration to specific projects		✔
New Artifacts (Feeds) Experience (accessibility updates)	✔	✔
New release progress views	✔	✔
New service connections experience	✔	✔
New Settings Search in the organization settings panel	✔	✔
New workflow identity authentication option	✔	✔
New Teams page	✔	✔
New Wiki experience	✔	✔
Organization Permissions Settings Page v2	✔	✔
Project Permissions Settings page	✔	✔
Task Insights for Failed Pipeline Runs	✔	✔
Workload Identity federation for Azure Resource Manager service connections		✔
YAML templates editor	✔	✔

Pull request summary - Load of large files

Turns on a file content load experience for large files in the Pull request summary.

Experimental themes

When you select **Theme** from the Profile menu, you can select between **Dark** and **Light** themes for the display of Azure DevOps web portal.

With **Experimental themes** on, you can select from many other themes.
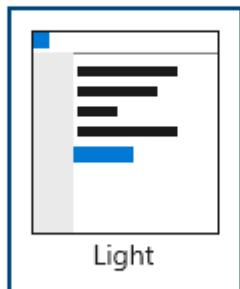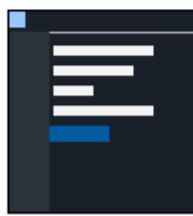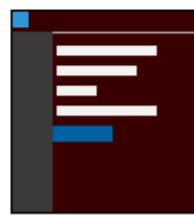
GA features for Azure DevOps

The following features are generally available for Azure DevOps.

General

- New user hub
- New PAT experience
- New Navigation 🗗
- Wiki
- Combine email recipients
- New experience in Code, Work Item, & Wiki search
- Out of the box notifications
- Team expansion for notifications
- Streamlined User Management

Azure Artifacts

- NuGet.org upstream sources
- Updated package experience

Azure Boards, Dashboards, and Analytics

- Copy Dashboard Experience
- New Delivery Plans Experience
- Enable group by tags for work item chart widget on dashboard
- New Queries Experience
- New Work Items
- New Dashboards Experience
- New boards reports
- New Boards Hub on by default
- Analytics views

Azure Repos

- New TFVC pages
- Git Forks
- New Repos pull request experience ⧉
- New Repos settings experience
- New Repos landing pages
- Pull Request Status Policy

Azure Pipelines

- Historical graph for agent pools
- Pipeline decorators
- Multi-stage pipelines
- Test tab in new web platform
- Test analytics in new web platform
- New builds hub
- Build with multiple queues ⧉
- New Releases Hub
- Approval gates in releases - New Release Definition Editor
- Symbol server
- Task tool installers

Azure Test Plans

- New Test Plans Page
- New Test Plan Experience

Related articles

- Set user preferences
- Azure DevOps Feature Timeline

Get started with search

Article • 09/28/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

You can quickly find work items, code files, wiki pages, or packages based on a keyword, wildcards, and other supported search filters with the search function.

See the following quick links to more information:

- Code Search Marketplace extension ⧉
- Search features, further in this article
- Functional code search

Prerequisites

- Every project member can use the search functions, including project members granted Stakeholder, Basic, and higher levels of access.
- When you search across the organization or collection, only results for which a project member has access are listed.
- Stakeholder wiki search results are limited to provisioned wikis. Because published wikis require access to regular repositories, which Stakeholders don't have access to, results for published wikis don't appear in the search results. Similarly, Code search results don't appear for Stakeholders.

Start your search with a keyword

Start your search using a keyword. You can then apply other options, as needed, to broaden or narrow your search results.

- To get results that match the input, you may need to remove filters and search again. After you see the search results, you can narrow them down by applying appropriate filters and searching again.
- Make sure your search terms are spelled correctly. Work item search doesn't ignore spelling errors.
- You might get a message that no matching files are found if you use a wildcard search with too many hits, such as a simple wildcard search string. In this situation, make your search more specific to decrease the number of matches. Add more characters of the word or words you want to find, or use a condition or filter to restrict the number of possible matches.
- Searches aren't case sensitive.

Search features, usage, and examples

The following features apply to all searches, including work items, code, wikis, and packages.

Search feature

Usage

Example

Keyword

Search based on one or more keywords.

`validate` finds instances that contain the word *validate*.

Exact match

Search based on an exact match, enclosed in double-quotes.

`"Client not found"` finds instances that contain the exact phrase match *Client not found*.

Wildcard
- Add wildcard characters, `*` and `?`, to keywords to extend the search criteria.
- Add `*` at the end of a keyword to find items that start with the keyword.
- Add `?` in the middle to represent any alphanumeric character.
- Use wildcard characters anywhere in your search string except as a prefix. You can use prefix wildcards with the other search filter functions.
- You can use more than one wildcard to match more than one character.
- `alpha?version` finds instances of alpha1version and alphaXversion.
- `Browser*` finds instances of BrowserEdge, BrowserIE, and BrowserFirefox.
- `CodeSenseHttp*` finds files containing words that start with *CodeSenseHttp*, such as CodeSenseHttpClient and CodeSenseHttpClientTest.

Boolean operators
- Find two or more keywords using Boolean operators: `AND`, `OR`, and `NOT` (must be uppercase).
- Add parenthesis to clauses to support logical groupings.
- Because `AND` is the default operator, an entry of two keywords with no operator is the same as an `AND` search.
- `Validate AND revisit` finds files that contain both the words *validate* and *revisit*.
- `Validate OR revisit` finds files that contain either of the words *validate* or *revisit*.

- `Validate NOT revisit` finds files that contain the word *validate* but not the word *revisit*.
- `(Validate NOT revisit) OR "release delayed"` finds files that contain the word *validate* but not the word *revisit* or files that contain the phrase *release delayed*.

Proximity

- Search for files based on vicinity using proximity operators: NEAR, BEFORE, and AFTER (must be uppercase).
- By default, proximity search looks for terms within five tokens distance.
- `term1 BEFORE term2` returns all files where term1 occurs BEFORE term2 within a distance of five tokens between them.
- `term1 AFTER term2` returns the same results as term2 BEFORE term1.
- `term1 NEAR term2` returns all files where term1 is within five token distance from term2 in any direction. `term1 NEAR term2` returns the same results as `term1 BEFORE term2` OR `term2 BEFORE term1`.

Special characters

- Escape the special characters `(`, `)`, `[`, `]`, `:`, `*`, and `?` by enclosing them in a phrase delimited with double-quotes.
- Include special characters in a search string, or search specifically for special characters, according to the following rules:
- CodeA23?R finds files containing words that start with CodeA23
- Have any alphanumeric character next, and end with R. For example, CodeA234R and CodeA23QR.
- Search for any special character that isn't a part of the query language.
- `"flatten()"` finds the literal string *flatten()*. Search for a literal occurrence of the double-quote character " by preceding it with the escape character `\` and enclosing the search string in double-quotes.
- `"\"react-redux\""` finds the literal string "react-redux."

Search from a different page

You can search from any of the following pages:

- Organization project page: Starts a search across all projects.
- Project overview page: Automatically applies a filter to search within the selected project.

- Boards page for a project: Automatically displays recent work items and backlogs accessed by the user.
- Azure Repos, Pipelines, Test Plans, or an Artifacts page for a project: Automatically displays functional filters for code searches.
- Wiki page: Automatically go to a wiki page you recently opened.

For more information, see the following articles:

- [Filter backlogs, boards, and plans](#).
- [Provisioned vs. published wiki](#).

> 💡 **Tip**
>
> **No results found for ...**
> Too many hits from a simple wildcard search can result in no matching files. You can narrow your search by adding more characters or using a condition or filter.

More search functions

See the following table for more search tasks and actions.

Search task

Action

Find an organization setting

Go to your organization and select **Organization settings**.

Find a project setting

Go to your project and select **Project settings**.

Find a user setting

Go to your **User settings page**.

Find a user

Go to your organization and select **Organization settings** > **Users**, and then enter the name in the filter box.

Find an organization

Scroll through the left side of your screen, which lists all organizations.

Find a project

Go to your organization, and then enter the project name in the Filter projects box.

View file history and compare versions

Go to **Repos** > **Files**, highlight your file, and then select **History**.

> ⓘ **Note**
>
> When you search from the **Organization settings** page, your search results include both organization-level and project-level settings.

Marketplace extensions

- Code search ↗ - Extends search with fast, flexible, and precise search results across all your code. Required for searching repositories.
- Azure Paths Search ↗ - Adds a special search hub to Boards for searching within iterations and area paths without having to create and maintain custom queries.

> ⓘ **Note**
>
> Azure DevOps doesn't support some extensions. For more information or assistance, go to the **Visual Studio Marketplace** ↗ .

Next steps

Functional code search

Related articles

- Functional work item search
- Functional artifact or package search
- Code search blog posts ↗

- Work item search blog posts ⬈

Functional code search

Article • 12/15/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Find the code you need faster with functional code search. This article explains how to refine your search across repositories using code types and other functions with the Code Search ⧉ Marketplace extension for Azure DevOps.

Prerequisites

- Install Code Search ⧉

- To use Code Search, you must have at least Basic access.
- Users with Stakeholder access don't have access to code, so they can't search for code.

- Users with Stakeholder access for a public project have full access to code, so they can search for code. To access code in a private project, you must have at least Basic access.

- When you're searching across the organization or collection, only results for which a project member has access are listed.

Code search best practices

- Start with a broad search and then use filter operators to narrow it down by project, repository, path, file name, and more.
- If you don't know the exact term, use wildcards to expand your search and Boolean operators to refine it.
- To get more information about a code item, hover over it and use the shortcut menu to search for that text in all your projects and files.
- To trace how your code works, use the shortcut menu to search for related items like definitions and references in a file or in the search results.
- To find the implementation of an API or other code element, use code type filters to search for specific kinds of code such as:
  - definitions
  - references
  - functions

- comments
  - strings
  - namespaces, and more.

> ⊘ **Note**
>
> Code search does not work for forked repositories.

Functions to find specific types of code

To create your query faster, choose functions and keywords from the drop-down list as you type. Select **Show more** to see all the options. You can combine different functions as needed.

You can also use filters from the left column to narrow your search. **Show more** shows you all the functions and keywords.

Or, you can type the functions and parameters in the search box. The table below lists the functions for finding specific types or members in your C#, C, C++, Java, and Visual Basic.NET code.

⟦ ⟧ **Expand table**

To find code where *findThis* appears as a ...	... search for argument arg:*findThis*
Argument	**arg:***findThis* `Deprecated in July 2019`
Base type	**basetype:***findThis*
Calling function	**caller:***findThis* `Deprecated in July 2019`
Class definition or declaration	**class:***findThis*
Class declaration	**classdecl:***findThis* `Merged with class:`
Class definition	**classdef:***findThis* `Merged with class:`
Comment	**comment:***findThis*
Constructor	**ctor:***findThis* `Merged with method:`
Declaration	**decl:***findThis*
Definition	**def:***findThis*
Destructor	**dtor:***findThis* `Merged with method:`

To find code where *findThis* appears as a ...	... search for argument arg:*findThis*
Enumerator	**enum:**findThis
Extern	**extern:**findThis `Deprecated in July 2019`
Field	**field:**findThis
Friend function	**friend:**findThis `Deprecated in July 2019`
Function	**func:**findThis `Merged with method:`
Function declaration	**funcdecl:**findThis `Merged with method:`
Function definition	**funcdef:**findThis `Merged with method:`
Global	**global:**findThis `Deprecated in July 2019`
Header	**header:**findThis `Deprecated in July 2019`
Interface	**interface:**findThis
Macro	**macro:**findThis
Macro definition	**macrodef:**findThis `Merged with macro:`
Macro reference	**macroref:**findThis `Merged with macro:`
Method	**method:**findThis
Method declaration	**methoddecl:**findThis `Merged with method:`
Method definition	**methoddef:**findThis `Merged with method:`
Namespace	**namespace:**findThis
Property	**prop:**findThis
Reference	**ref:**findThis
String literal	**strlit:**findThis
Struct	**struct:**findThis `Merged with type:`
Struct declaration	**structdecl:**findThis `Merged with type:`
Struct definition	**structdef:**findThis `Merged with type:`
Template argument	**tmplarg:**findThis `Deprecated in July 2019`
Template specification	**tmplspec:**findThis `Deprecated in July 2019`
Type	**type:**findThis

To find code where *findThis* appears as a ...	... search for argument arg:*findThis*
Typedef	**typedef:***findThis* `Merged with type:`
Union	**union:***findThis* `Deprecated in July 2019`

Functions to select projects, repositories, paths, and files

Functions make it easy to narrow the search to specified locations, specific types of files within these locations, or specified filenames. Narrow the search to a specific location using the `proj`, `repo`, or `path` filters. Mix and match the following functions as required.

⌞⌝ **Expand table**

Usage	Example
Find all occurrences of the word *QueueJobsNow* in the Fabrikam project.	`QueueJobsNow proj:Fabrikam`
Find all occurrences of the word *QueueJobsNow* in the Contoso repository.	`QueueJobsNow repo:Contoso`
Find all occurrences of the word *QueueJobsNow* in the path *VisualStudio/Services/Framework* and its subpaths.	`QueueJobsNow` `path:VisualStudio/Services/Framework`
Find all occurrences of the word *QueueJobsNow* in the path */Doc*/Framework/** and */Doc*/*/*/Framework/** and its subpaths. Globbing Pattern (**) matches zero or more characters across multiple segments. For example, path:**/Doc**/Framework will also match abc/*Doc*Test/gh/ijk/mnop/*Framework*/	`QueueJobsNow path:**/Doc**/Framework`
Find all occurrences of the word *QueueJobsNow* in the path */Doc*/Framework/** and its subpaths and file name Test*.txt (Use Globbing Pattern **). For example, path:**/Doc**/Framework/**/Test*.txt also matches abc/def/*Doc*A/gh/*Framework*/*Test*Misc.*txt*	`QueueJobsNow` `path:**/Doc**/Framework/**/Test*.txt`

Enclose the argument to the filter in double-quotes if it contains a space.	`QueueJobsNow`
	`path:"VisualStudio/Windows Phones and`
	`Devices/Services"`

Enclose the argument to the filter in double-quotes if it contains a space.	`QueueJobsNow`
	`path:"VisualStudio/Windows Phones and`
	`Devices/Services"`

Usage	Example
Find all occurrences of the word *QueueJobsNow* in all files where the filename starts with *queueRegister*.	`QueueJobsNow file:queueRegister*`
Find all files with the name *QueueRegister* without an extension. Use quotes to find files without extensions.	`file:"queueRegister"`
Find all occurrences of the word *QueueJobsNow* in only C# source files. A plain text search string that doesn't include file type functions also finds files where the string matches part of the filename.	`QueueJobsNow ext:cs`

Find related items or other terms

Code Search lets you interactively expand your search based on previous results. For example, you can widen your search to related files when you are tracing or debugging code.

Right-click on a term in the file and start a new search for other files with the same term. You can search for it as text, or as a definition or reference if it is an object name.

For more information about the following search functions, see Get started with search

- Keyword
- Exact match
- Wildcard
- Boolean operators
- Proximity

More code search operations

Here are some more code search functions. You can search for code types in C#, C, C++, Java, and Visual Basic.NET files. To open the search results in a new tab, select **Ctrl** + **Enter** from the main search box. To switch to the new tab in Google Chrome, select **Ctrl** + **Shift** + **Enter**.

⌗ Expand table
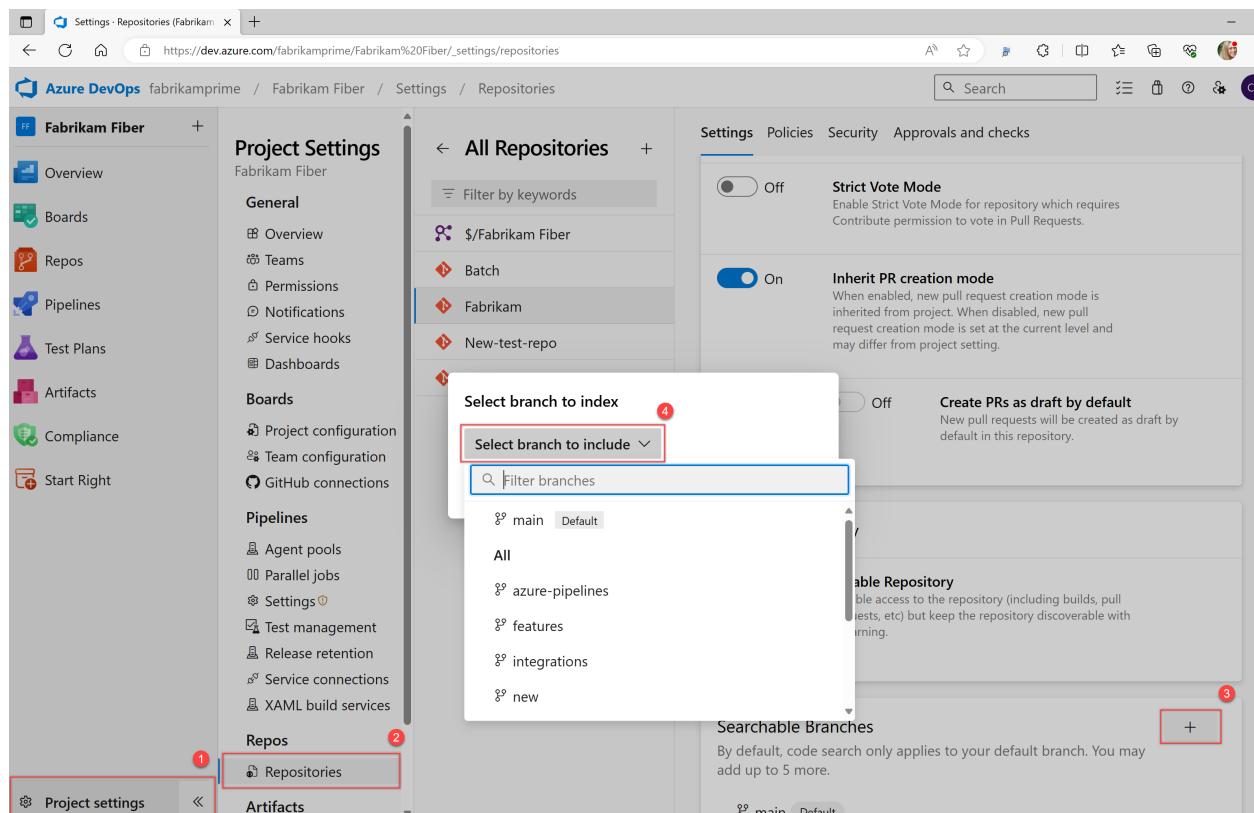
Usage	Example
Find all comments	History:Keyword

Usage	Example
Find all instances of "ToDo" comments in your code	Select `comment:` and enter `todo`
Search in specific locations, such as within a particular path	Use a search string such as `Driver path:MyShuttle/Server`
Search for files by name or just by file extension	`Driver file:GreenCabs.cs`. The search string `error ext:resx` could be useful if you want to review all error strings in your code. Even if your plain text search string matches part of a filename, the file appears in the list of found files. This search works without matching specific file type functions.

Search Git projects and repositories

A Git project has a list of repositories. To expand your search, check the project and repository boxes. You can search all or more projects, or fewer projects and repositories. If there are many projects or repositories, select **Show more** to see them all.

Code Search can index different branches in a Git repository. It only indexes files in the default branch of your Git repositories by default. The default branch is usually main. To index other branches, go to the **Options** tab in the **Repositories** section of the project settings page.

Search TFVC projects

TFVC projects display only the folders that you can read. You can't see any other projects or folders. To filter your search, choose folders from the tree.

> 💡 **Tip**
>
> Code Search saves your last settings, such as the project and repository or path that you searched in. When you want to search in a different scope, select **Clear all links** to clear the checkboxes and search across all projects. The first 100 hits or matches in the target files are highlighted by Code Search in the results pane.

Search code with REST API

You can use APIs to extend or supplement the capabilities listed in this article. For information about Code Search with REST API, see Fetch Code Search Results.

Next steps

Search work items

Related articles

- Get started with Search
- Search artifacts and packages
- Search work items

Functional work item search

Article • 09/28/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Functional work item search command filters let you search for work items more precisely by assignment, work item type, specific fields, and more. For more filter functions, see Get started with search.

Work Item Search lets you do the following tasks and more.

Search task	Description
Search over all your projects	Search in your own and your partner teams' backlog. Use cross-project searches over all the work items to search across your enterprise's entire work items. Narrow your search by using project and area path filters.
Search across all work item fields	Quickly and easily find relevant work items by searching across all work item fields, including custom fields. Use a full text search across all fields to efficiently locate relevant work items. The snippet view indicates where matches were found.
Search in specific fields	Use the quick in-line search filters to narrow down to a list of work items in seconds. Use the filters on any work item field. The list of suggestions helps complete your search faster. For example, a search such as **AssignedTo:Chris WorkItemType:Bug State:Active** finds all active bugs assigned to a user named Chris.
Search across test	Search across Test Plans, Test Suites, and other test work item types.
Take advantage of integration with work item tracking	The Work Item Search interface integrates with familiar controls for managing your work items; letting you view, edit, comment, share, and more.

Prerequisites

All users can use work item search.

You can use Work Item Search by default without any installation when the Boards service is installed and enabled in Azure DevOps Services.

Search by work item ID

Enter the work item ID in the Azure DevOps title bar. You can read and edit the work item in a modal dialog.

Search across all fields

Search all work item fields, including custom ones, for natural searches. The snippet view shows matches.

- Use simple search strings for words or phrases. Work item search matches derived forms of your search terms; for example, a search for "updating" also finds instances of the word "updated" and "update." Searches aren't case-sensitive.

- Search from within a project to, by default, search only within that project.
- Search from inside a team to, by default, search only within the default area path of that team.
  - Select a project to view a list of area paths in that project for which you have read access.
  - Select area paths in the tree to narrow your search if necessary.
- View hit counts for all projects, even one that you don't select.
- Open the search results in a new browser tab from either the main search function or by selecting **Ctrl** + **Shift** + **Enter**.

Apply best practices

- Use a text search across all fields to efficiently locate relevant work items. Text search is useful when you're trying to, for example, search for all work items that had similar exception trace.
- Use the quick in-line search filters on any work item field to narrow down to a list of work items in seconds. The list of suggestions helps complete your search faster.

Compare search vs. managed work item queries

The main search function and managed queries are two ways to find and list work items. For a single work item, use the main search. For a list of work items that you want to triage, update, chart, or share, use a managed query.

You can search more fields with the main search function than with managed queries.

Use a managed query

Search

- List items to perform bulk updates to fields.
- Review work that's in progress or recently closed.
- Triage work: set priority, review, update.
- Create a chart and add it to a dashboard.
- Create a chart to get a count of items or sum a field.
- Create a chart that shows a burndown or burnup over time.
- View a tree of parent-child related work items.
- List work items with link relationships.
- List work items for a single project, multiple projects, or across all projects.
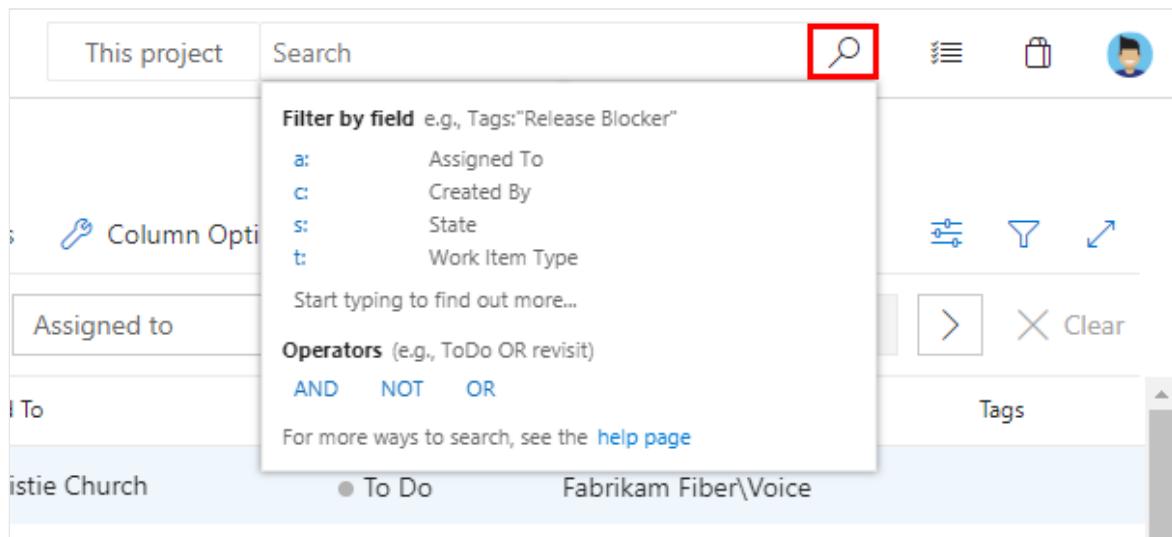
- Find a specific work item using its ID or a keyword.
- Find one or more work items across all projects in a fast, flexible manner.
- Perform full text search across all work item fields.
- Review work items assigned to a specific team member.
- Search against specific work item fields to quickly narrow down a list of work items.
- Determine what key words support a managed search.
- List work items for a single project, multiple projects, or across all projects.

For more information, see the following articles:

- View and run a query
- Use search
- Define a query
- Query quick reference, Example queries

Fine-tune your search

1. Specify fields to fine-tune your search. Search all assigned items by entering `a:` and a user name.

   Quick filters:

   - `a:` for **Assigned to:**
   - `c:` for **Created by:**
   - `s:` for **State**
   - `t:` for **Work item type**

2. Start entering the name of a field in your work items; for example, enter `ta`.
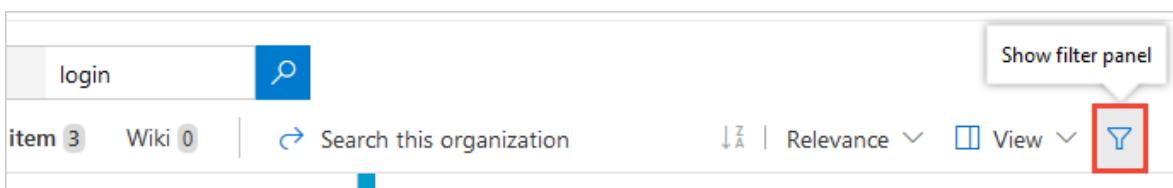
The dropdown list shows work item field name suggestions that match user input. These suggestions help you complete the search faster. For example, a search such as `tags:Critical` finds all work items tagged 'Critical.'

3. Add more filters to further narrow your search, and use Boolean operators to combine terms if necessary. For example, **a: Chris t: Bug s: Active** finds all active bugs assigned to a user named Chris.

4. Narrow your search to specific types and states, by using the selector lists at the top of the results page.

5. Widen your search across all projects, or narrow it to specific types and states. Use the filter to show the selector lists.

6. Select the criteria you want in the drop-down selector lists, or search across the entire organization.

7. Sort the results with the drop-down list of field names, work item types, or by relevance.

Quick filters for matching in specific fields

Quick inline search filters let you refine work items in seconds. The dropdown list of suggestions helps complete your search faster. Mix and match the functions to create quick powerful searches.

Usage	Example
Scope your search terms to match in any work item field including custom fields. Enter the field name followed by the search terms.	`tags:Critical` finds work items having a field 'tags' containing the term 'Critical.'

Usage	Example
Use multiple inline search filters to scope your search by any work item field, including custom fields.	`t: Bug path:"project\search"` finds all bugs in the area path "project\search."
Use the operators `>`, `>=`, `<`, `<=`, `=`, and `!=` for date, integer, and float fields.	`t: Bug CreatedDate> @Today-7` finds all bugs created in the last week.
For the search query that contains multiple terms and users looking for exact match, embed the search term inside `" "`	`BuildPath: "tools.demoproject.com"` finds all work items that necessarily contain the path "tools.demoproject.com."

Quick Filters

Quick in-line search filters lets you refine work items by specific criteria on any work item field, in seconds!

Search Work Items

Scope projects and area and iteration paths using filters

Filters make it easy to narrow the search to specified projects and area paths.

Narrow the search to a specific location using the `proj`, `area`, `iteration`, `path`, and `comment` filters:

Usage	Example
Finds all occurrences of the word **Wiki** in the **Fabrikam** project.	`Wiki proj:Fabrikam`
Finds all occurrences of the word **Wiki** in the area path **Contoso/Mobile** and its subpaths.	`Wiki area:Contoso/Mobile`
Finds all occurrences of the word **Wiki** in the iteration path **Contoso/Sprint101** and its subpaths.	`Wiki iteration:Contoso/Sprint101`

Usage	Example
Enclose the argument to the filter in double-quotes if it contains a space.	`Wiki path:"Contoso/Windows Phones and Devices/Services"`
Finds backlog comments	`comment:todo`

See more of the work item

You can quickly get a full screen view of the selected work item using ⊡ **expand** and ⛶ **shrink** in the toolbar. However, another way to see more of the work item, while you can still select work items from the list of matching results, is to hide the left column filter pane by choosing < at the top left of the column. Use > to restore the filter pane.

If you're using a portrait orientation screen, use the **Preview pane: Right** link at the top right of the window to display the code below the search results list.

> 💡 **Tip**
>
> Search remembers the state of the filter pane, configuration of the work item view pane, and its position between sessions as part of your user preferences.

Search Work Items with REST API

You can use APIs to extend or supplement the capabilities listed in this article. For information about Work Item Search with REST API, see Fetch Work Item Search Results.

Next steps

Supported filter functions and more for work items

Related articles

- Get started with Search
- Search code
- Search artifacts and packages

Migrate data from Azure DevOps Server to Azure DevOps Services

Article • 05/10/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

The data migration tool for Azure DevOps provides a high fidelity way to migrate collection databases from Azure DevOps Server to Azure DevOps Services. It's recommended that you download the [migration guide and tool](#) ⧉ if you're looking to use this service to import your collection(s). The guide serves as a walk through of the different steps involved in an import. Providing best practices, checklists, and helpful tips to make your import as easy as possible. The guide should be used with the more technical documentation referenced below to successfully import to Azure DevOps Services.

Supported Azure DevOps Server versions for import

> ⓘ **Important**
>
> It can take up to 2-3 weeks after a new RTW version of Azure DevOps Server is released for import support to come online for that version. It's important to take this into consideration when choosing to upgrade shortly after a new RTW Azure DevOps Server release.

The data migration tool for Azure DevOps supports the two latest releases of Azure DevOps Server at a given time. Releases include updates and major releases. Currently the following versions of Azure DevOps Server are supported for import:

- Azure DevOps Server 2022.0.1
- Azure DevOps Server 2022.1

> ⓘ **Note**
>
> The data migration tool doesn't support imports from Azure DevOps Server release candidates (RC). If you're planning on importing your collection database to Azure DevOps Services using this service, it's important that you don't upgrade your

production database to an RC release. If you do upgrade, then you'll need to wait and upgrade to the release to web (RTW) version when it's available or restore a backup copy of your database from a previous Azure DevOps Server version to import.

Normal release cadence for new Azure DevOps Server versions is once every three-to-four months. Meaning that support for a given version of Azure DevOps Server for migration to Azure DevOps Services should last for anywhere between six-to-eight months. It's important to ensure that your planning accounts for this support window to avoid having to suddenly upgrade to migrate.

Preview features

> ⓘ **Note**
>
> If you're not including preview features when running the migration tool, then you'll need to re-run the migration tool prepare to generate a new import.json to queue an import. You DO NOT need to include preview features when you re-generate your import.json.
>
> If you had previously been including preview features then you DO NOT need to take any additional actions after Monday, April 23, 2020.

The following features can be included with your import, but are currently in a preview state.

- Analytics - Note this is only supported for Azure DevOps Server 2019 and later.

When queueing an import, you can elect to include preview features with your import. If you do, data related to these features will be copied into your new organization along with all your other data. If you choose not to include these features then their data won't be copied.

For a list of items not included with an import, see the migration guide and tool ⧉ .

Data migration tool for Azure DevOps resources

In general, you should use the Migration guide and tool ⧉ when going through an import. When it's required, the guide links back to the following articles. These articles

offer deeper technical knowledge on various import topics.

Import process

- Validate a collection for import
- Prepare a collection for import
- Prepare for import
  - Prepare large collections for import
- Run an import
- Post import steps

Troubleshooting

- Troubleshooting validation errors
- Troubleshooting process errors
- Troubleshooting import errors

Q & A

Q: Will my Personal Access Tokens also migrate when I migrate from on-premises to Azure DevOps Services?

A: **No**. Your tokens won't migrate and you'll need to regenerate your Personal Access Tokens on Azure DevOps Services.

Q: If I have feedback or other questions is there somewhere I can reach out?

A: You can search the developer community portal ⧉ to see if your question is asked and answered and if not, open up a new issue. If you need assistance with a failed import, contact Azure DevOps customer support ⧉ .

Related articles

- Migration and process model FAQs

Migration options

Article • 02/21/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

When you decide to make the move from Azure DevOps Server to Azure DevOps Services, you might start fresh with an empty organization. Often, however, you will have existing code, work items, and other assets that you want to move. There are many approaches to doing this which vary in both the fidelity of the data transfer and the complexity of the process.

Prior to migrating data, review the differences that exist between Azure DevOps Server and Azure DevOps Services.

Option 1: Copy the most important assets manually

By far the easiest option for moving data into Azure DevOps Services is to manually copy your most important assets and start relatively fresh. This can be difficult when you are in the middle of a large project, but you can make it easier if you do some advance planning and schedule your move when it makes sense for your team.

For example, when the Azure DevOps team chose to move from Azure DevOps Server to Azure DevOps Services, we also decided to move from Team Foundation Version Control (TFVC) to Git. This required a fair bit of planning, but when we actually performed our migration, we created a new Git repo using the "tip" version of our TF VC sources, and left our history behind in Azure DevOps Server. We also moved our active work items, and left behind all our old bugs, completed user stories and tasks, and so on.

Here's the general process:

1. Identify the most important assets that you need to migrate - typically source code, work items, or both. Other assets in Azure DevOps Server - build pipelines, test plans, and so forth - are harder to manually migrate.
2. Identify a good time to make the transition.
3. Prepare your target organizations. Create the organizations and team projects that you need, provision users, and so on.
4. Migrate your data.
5. Consider making the source Azure DevOps Server deployments read-only.

Option 2: High fidelity database migration.

The Azure DevOps Server & Azure DevOps Services product team provides a high fidelity data migration tool. A downloadable Migration Guide is available at https://aka.ms/AzureDevOpsImport ⧉ .

Because the data migration tool operates at a database level, it can provide a very high fidelity migration. If you want to move your existing Azure DevOps Server data into Azure DevOps Services, we strongly recommend using this option.

Option 3: Using public API-based tools for higher fidelity migration

If for some reason you cannot use the data migration tool but still want a higher fidelity migration than Option 1, you can choose from a variety of tools that use public APIs to move data. Generally these tools can provide a higher fidelity migration than a manual copy of "tip" data, but they are still relatively low fidelity. For example:

- None of them will preserve the dates of TF VC changesets.
- Many of them will not preserve the changed dates of work item revisions.
- None of them will migrate all Azure DevOps Server artifacts.

In general, we only recommend this approach if the extra fidelity beyond a manual copy is critical. If you decide to take this approach, you might consider hiring a consultant who has experience with one or more of the tools. You should definitely consider doing a test migration before doing your final migration.

Many organizations need a very high fidelity migration for only a subset of their work. New work could potentially start directly in Azure DevOps Services. Other work, with less

stringent fidelity requirements, could be migrated using one of the other approaches. You will have to weigh the pros and cons of the various approaches against your motivations for moving into Azure DevOps Services and decide for yourself what is the right strategy.

Related articles

- About Azure DevOps Services and Azure DevOps Server
- Pricing, Azure DevOps Services ⧉
- Pricing, Azure DevOps Server ⧉

Validation and import processes

Article • 01/04/2024

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

This article walks you through the preparation required to get an import to Azure DevOps Services ready to run. If you encounter errors during the process, see Troubleshoot import and migration errors.

Prerequisites

- You must set up a Microsoft Entra tenant as described in Microsoft Entra Connect Sync: Make a change to the default configuration. The data migration tool sets up a link to your Microsoft Entra tenant when your Azure DevOps Services organization is created as part of the beginning of the import process. When you synchronize your on-premises Active Directory with Microsoft Entra ID, your team members can use the same credentials to authenticate and your Azure DevOps Services administrators can use your Active Directory groups for setting permissions within your Azure DevOps Services organization. To set up the synchronization, use the Microsoft Entra Connect technology.

- Before you begin the import tasks, check to ensure that you're running a supported version of Azure DevOps Server.

- We recommend that you use the Step-by-step migration guide ☒ to progress through your import. The guide links to technical documentation, tools, and best practices.

Validate a collection

Validate each collection that you want to migrate to Azure DevOps Services. The validation step examines various aspects of your collection, including, but not limited to, size, collation, identity, and processes.

Run the validation by using the data migration tool.

1. Download the tool ☒

2. Copy the zip file to one of your Azure DevOps Server application tiers

3. Unzip the file. You can also run the tool from a different machine without Azure DevOps Server installed, as long as the machine can connect to the configuration

database of the Azure DevOps Server instance.

4. Open a Command Prompt window on the server, and enter a cd command to change to the directory where the data migration tool is stored. Take a few moments to review the help content for the tool.

a. To view the top-level help and guidance, run the following command:

```cmdline
Migrator /help
```

b. View the help text for the command:

```cmdline
Migrator validate /help
```

5. As your first time validating a collection, let's keep it simple. Your command should have the following structure:

```cmdline
Migrator validate /collection:{collection URL} /tenantDomainName:
{name} /region:{region}
```

Where `{name}` provides the name of your Microsoft Entra tenant, for example, to run against the *DefaultCollection* and the *fabrikam* tenant, the command would like the example:

```cmdline
Migrator validate /collection:http://localhost:8080/DefaultCollection
/tenantDomainName:fabrikam.OnMicrosoft.com /region:{region}
```

6. To run the tool from a machine other than the Azure DevOps Server, you need the **/connectionString** parameter. The connection string parameter points to your Azure DevOps Server configuration database. As an example, if the validated command runs by the Fabrikam corporation, the command would look like:

```cmdline
Migrator validate /collection:http://fabrikam:8080/DefaultCollection
/tenantDomainName:fabrikam.OnMicrosoft.com /region:{region}
```
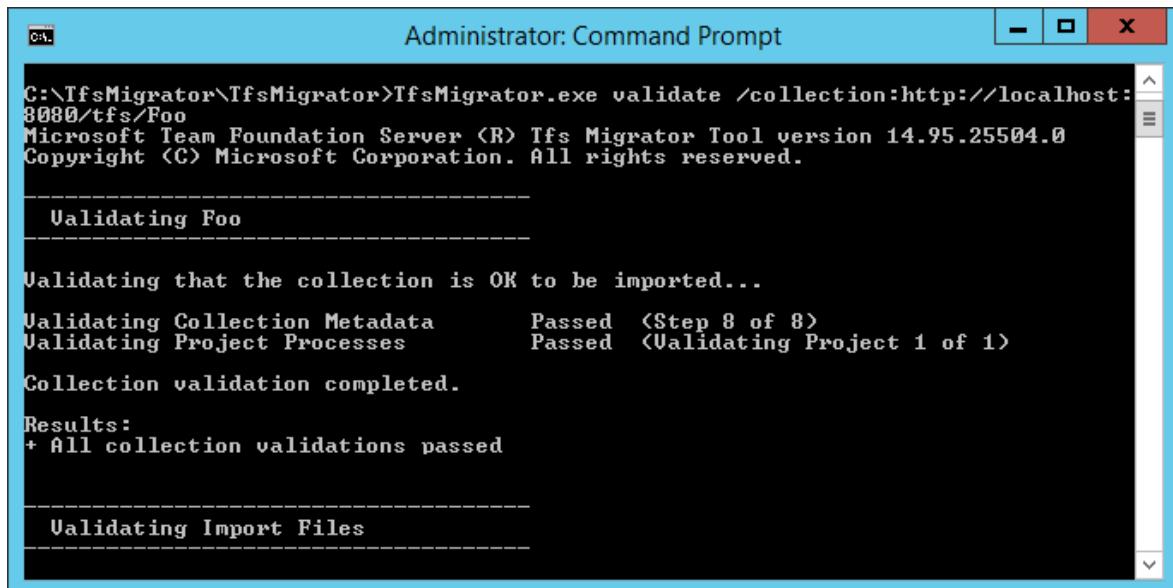
```
/connectionString:"Data Source=fabrikam;Initial
Catalog=Configuration;Integrated Security=True"
```

> ⓘ **Important**
>
> The data migration tool *does not* edit any data or structures in the collection.
> It reads the collection only to identify issues.

7. After the validation is complete, you can view the log files and results.

During validation, you receive a warning if some of your pipelines contain per-pipeline retention rules. Azure DevOps Services uses a project-based retention model and doesn't support per-pipeline retention policies. If you proceed with the migration, the policies aren't carried over to the hosted version. Instead, the default project-level retention policies apply. Retain builds important to you to avoid their loss.

After all the validations pass, you can move to the next step of the import process. If the data migration tool flags any errors, correct them before you proceed. For guidance on correcting validation errors, see Troubleshoot import and migration errors.

Import log files

When you open the log directory, you might notice several logging files.

The main log file is named *DataMigrationTool.log*. It contains details about everything that was run. To make it easier for you to focus on specific areas, a log generates for each major validation operation.

For example, if TfsMigrator reports an error in the "Validating Project Processes" step, you can open the *ProjectProcessMap.log* file to view everything that was run for that step instead of having to scroll through the entire log.

Review the *TryMatchOobProcesses.log* file only if you're trying to import your project processes to use the [inherited model](). If you don't want to use the inherited model, you can ignore these errors, because they don't prevent you from importing to Azure DevOps Services.

Generate import files

The data migration tool validated the collection and it's returning a result of "All collection validations passed." Before you take a collection offline to migrate it, generate the import files. When you run the `prepare` command, you generate two import files:

- *IdentityMapLog.csv*: Outlines your identity map between Active Directory and Microsoft Entra ID.
- *import.json*: Requires you to fill out the import specification you want to use to kick off your migration.

Prepare command

The `prepare` command assists with generating the required import files. Essentially, this command scans the collection to find a list of all users to populate the identity map log, *IdentityMapLog.csv*, and then tries to connect to Microsoft Entra ID to find each identity's match. To do so, your company needs to use the [Microsoft Entra Connect tool]() (formerly known as the Directory Synchronization tool, Directory Sync tool, or DirSync.exe tool).

If directory synchronization is set up, the data migration tool should find the matching identities and mark them as *Active*. If there are no matches, the identity is marked as *Historical* in the identity map log, so you must investigate why the user isn't included in your directory sync. The import specification file, *import.json*, should be populated prior to the import.

Unlike the `validate` command, `prepare` *does* require an internet connection, because it needs to connect to Microsoft Entra ID to populate the identity map log file. If your Azure DevOps Server instance doesn't have internet access, run the tool from a machine that does. As long as you can find a machine with an intranet connection to your Azure DevOps Server instance and an internet connection, you can run this command. For help with the `prepare` command, run the following command:

```cmdline
```

```
Migrator prepare /help
```

Included in the help documentation are instructions and examples for running the `Migrator` command from the Azure DevOps Server instance itself and a remote machine. If you're running the command from one of the Azure DevOps Server instance's application tiers, your command should have the following structure:

```cmdline
Migrator prepare /collection:{collection URL} /tenantDomainName:{name}
/region:{region}
```

```cmdline
Migrator prepare  /collection:{collection URL} /tenantDomainName:{name}
/region:{region} /connectionString:"Data Source={sqlserver};Initial
Catalog=Configuration;Integrated Security=True"
```

The **connectionString** parameter is a pointer to the configuration database of your Azure DevOps Server instance. As an example, if the Fabrikam corporation runs the `prepare` command, the command looks like the following example:

```cmdline
Migrator prepare /collection:http://fabrikam:8080/DefaultCollection
/tenantDomainName:fabrikam.OnMicrosoft.com /region:{region}
/connectionString:"Data Source=fabrikam;Initial
Catalog=Configuration;Integrated Security=True"
```

When the data migration tool runs the `prepare` command, it runs a complete validation to ensure that nothing changed with your collection since the last full validation. If any new issues are detected, no import files are generated.

Shortly after the command starts running, a Microsoft Entra sign-in window displays. Sign in with an identity that belongs to the tenant domain, which is specified in the command. Make sure that the specified Microsoft Entra tenant is the one you want your future organization to be backed with. In our Fabrikam example, a user enters credentials that are similar to the following example screenshot.

ⓘ **Important**

> Don't use a test Microsoft Entra tenant for a test import and your production
> Microsoft Entra tenant for the production run. Using a test Microsoft Entra tenant
> can result in identity import issues when you begin your production run with your
> organization's production Microsoft Entra tenant.

When you run the `prepare` command successfully in the data migration tool, the results window displays a set of logs and two import files. In the log directory, find a logs folder and two files:

- *import.json* is the import specification file. We recommend that you take time to fill it out.
- *IdentityMapLog.csv* contains the generated mapping of Active Directory to Microsoft Entra identities. Review it for completeness before you kick off an import.

The two files are described in greater detail in the next sections.

The import specification file

The import specification, *import.json*, is a JSON file that provides import settings. It includes the desired organization name, storage account information, and other information. Most of the fields are autopopulated, and some fields require your input before you attempt an import.

The *import.json* file's displayed fields and required actions are described in the following table:

⎵⎴ Expand table

Field	Description	Required action
Source	Information about the location and names of the source data files that are used for import.	No action required. Review information for the subfield actions to follow.
Location	The shared access signature key to the Azure storage account that hosts the data-tier application package (DACPAC).	No action required. This field is covered in a later step.

Field	Description	Required action
Files	The names of the files containing import data.	No action required. Review information for the subfield actions to follow.
DACPAC	A DACPAC file that packages the collection database to be used to bring in the data during the import.	No action required. In a later step, you create this file by using your collection and then upload it to an Azure storage account. Update the file based on the name you use when you generate it later in this process.
Target	Properties of the new organization to import into.	No action required. Review information for the subfield actions to follow.
Name	The name of the organization to be created during the import.	Provide a name. The name can be quickly changed later after the import completed. **NOTE**: *Don't* create an organization with this name before you run the import. The organization is created as part of the import process.
ImportType	The type of import that you want to run.	No action required. In a later step, select the type of import to run.
Validation Data	Information needed to help drive your import experience.	The data migration tool generates the "ValidationData" section. It contains information to help drive your import experience. Don't* edit the values in this section, or your import could fail to start.

After you complete the preceding process, you should have a file that looks like the following example.

In the preceding image, the planner of the Fabrikam import added the organization name *fabrikam-import* and selected CUS (Central United States) as the geographical location for import. Other values were left as is to be modified just before the planner took the collection offline for the migration.

> ⓘ **Note**
>
> Dry-run imports have a '-dryrun' automatically appended to the end of the organization name, which you can change after the import.

Supported Azure geographical locations for import

Azure DevOps Services is available in several [Azure geographical locations](link) ↗. But, not all locations where Azure DevOps Services is available are supported for import. The

following table lists the Azure geographical locations that you can select for import. Also included is the value that you need to place in the import specification file to target that geography for import.

Geographical location	Azure geographical location	Import specification value
United States	Central United States	CUS
Europe	Western Europe	WEU
United Kingdom	United Kingdom South	UKS
Australia	Australia East	EAU
South America	Brazil South	SBR
Asia Pacific	South India	MA
Asia Pacific	Southeast Asia (Singapore)	SEA
Canada	Central Canada	CC

The identity map log

The identity map log is of equal importance to the actual data that you migrate to Azure DevOps Services. As you're reviewing the file, it's important to understand how identity import operates and what the potential results could entail. When you import an identity, it can become either *active* or *historical*. Active identities can sign in to Azure DevOps Services, but historical identities can't.

Active identities

Active identities refer to user identities in Azure DevOps Services post-import. In Azure DevOps Services, these identities are licensed and are displayed as users in the organization. The identities are marked as *active* in the **Expected Import Status** column in the identity map log file.

Historical identities

Historical identities are mapped as such in the **Expected Import Status** column in the identity map log file. Identities without a line entry in the file also become historical. An

example of an identity without a line entry might be an employee who no longer works at a company.

Unlike active identities, historical identities:

- *Don't* have access to an organization after migration.
- *Don't* have licenses.
- *Don't* show up as users in the organization. All that persists is the notion of that identity's name in the organization, so that its history can be searched later. We recommend that you use historical identities for users who no longer work at the company or who don't need further access to the organization.

> ⓘ **Note**
>
> After an identity is imported as historical, it *can't* become active.

Understand the identity map log file

The identity map log file is similar to the example shown here:

The columns in the identity map log file are described in the following table:

You and your Microsoft Entra admin must investigate users marked as *No Match Found (Check Microsoft Entra ID Sync)* to understand why they aren't part of your Microsoft Entra Connect Sync.

⌐⌐ **Expand table**

Column	Description
Active Directory: User (Azure DevOps Server)	The friendly display name used by the identity in Azure DevOps Server. This name makes it easier to identify which user the line in the map is referencing.
Active Directory: Security Identifier	The unique identifier for the on-premises Active Directory identity in Azure DevOps Server. This column is used to identify users in the collection.
Microsoft Entra ID: Expected Import User (Azure DevOps Services)	Either the expected sign-in address of the matched soon-to-be-active user or *No Match Found (Check Microsoft Entra ID Sync)*, which

Column	Description
	indicates that the identity isn't found during the Microsoft Entra ID Sync and is imported as historical.
Expected Import Status	The expected user import status: either *Active* if there's a match between your Active Directory and Microsoft Entra ID, or *Historical* if there isn't a match.
Validation Date	The last time the identity map log was validated.

As you read through the file, notice whether the value in the **Expected Import Status** column is *Active* or *Historical*. *Active* indicates that the identity on this row maps correctly on import becomes active. *Historical* means that the identities become historical on import. It's important to review the generated mapping file for completeness and correctness.

> ⓘ **Important**
>
> The import fails if major changes occur to your Microsoft Entra Connect security ID sync between import attempts. You can add new users between dry runs, and you can make corrections to ensure that previously imported historical identities become active. However, changing an existing user that was previously imported as active isn't supported at this time. Doing so causes your import to fail. An example of a change might be completing a dry-run import, deleting an identity from your Microsoft Entra ID that was imported actively, re-creating a new user in Microsoft Entra ID for that same identity, and then attempting another import. In this case, an active identity import attempts between the Active Directory and newly created Microsoft Entra identity, but causes an import failure.

1. Review the correctly matched identities. Are all the expected identities present? Are the users mapped to the correct Microsoft Entra identity?

   If any values must be changed, contact your Microsoft Entra administrator to verify that the on-premises Active Directory identity is part of the sync to Microsoft Entra ID and is set up correctly. For more information, see Integrate your on-premises identities with Microsoft Entra ID.

2. Next, review the identities that are labeled as *historical*. This labeling implies that a matching Microsoft Entra identity couldn't be found, for any of the following reasons:

   - The identity isn't set up for sync between on-premises Active Directory and Microsoft Entra ID.

- The identity isn't populated in your Microsoft Entra ID yet (for example, there's a new employee).
- The identity doesn't exist in your Microsoft Entra instance.
- The user who owns that identity no longer works at the company.

To address the first three reasons, set up the intended on-premises Active Directory identity to sync with Microsoft Entra ID. For more information, see Integrate your on-premises identities with Microsoft Entra ID. You must set up and run Microsoft Entra Connect for identities to be imported as *active* in Azure DevOps Services.

You can ignore the fourth reason, because employees who are no longer at the company should be imported as *historical*.

Historical identities (small teams)

> ⊘ **Note**
>
> The identity import strategy proposed in this section should be considered by small teams only.

If Microsoft Entra Connect isn't configured, all users in the identity map log file are marked as *historical*. Running an import this way results in all users being imported as *historical*. We strongly recommended that you configure Microsoft Entra Connect to ensure that your users are imported as *active*.

Running an import with all historical identities has consequences that need to be considered carefully. Only teams with a few users and for which the cost of setting up Microsoft Entra Connect is deemed too high should consider.

To import all identities as historical, follow the steps outlined in later sections. When you queue an import, the identity used to queue the import is bootstrapped into the organization as the organization owner. All other users are imported as historical. Organization owners can then add the users back in by using their Microsoft Entra identity. The added users are treated as new users. They don't* own any of their history, and there's no way to reparent this history to the Microsoft Entra identity. However, users can still look up their preimport history by searching for their <domain><Active Directory username>.

The data migration tool displays a warning if it detects the complete historical identities scenario. If you decide to go down this migration path, you need to consent in the tool to the limitations.

Visual Studio subscriptions

The data migration tool can't detect Visual Studio subscriptions (formerly known as MSDN benefits) when it generates the identity map log file. Instead, we recommend that you apply the auto license upgrade feature after the import. As long as users' work accounts are linked correctly, Azure DevOps Services automatically applies their Visual Studio subscription benefits at their first sign-in after the import. You're never charged for licenses that are assigned during the import, so you can safely handle subscriptions afterward.

You don't need to repeat a dry-run import if users' Visual Studio subscriptions aren't automatically upgraded in Azure DevOps Services. Visual Studio subscription linking happens outside the scope of an import. As long as their work account is linked correctly before or after the import, users' licenses are automatically upgraded on their next sign-in. After their licenses are upgraded successfully, the next time you run an import, the users are upgraded automatically on their first sign-in to the organization.

Prepare for import

Now you have everything ready to execute on your import. Schedule downtime with your team to take the collection offline for the migration. When you agree upon a time to run the import, upload the required assets you generated and a copy of the database to Azure. Preparing for import consists of the following five steps.

Step 1: Take the collection offline and detach it.

The collection size limit for the DACPAC method is 150 GB. If the data migration tool displays a warning that you can't use the DACPAC method, you have to perform the import by using the SQL Azure virtual machine (VM) method. Skip steps 2 to 5 in that case and follow instructions provided in Import large collections and then continue to section determine the import type.

Step 2: Generate a DACPAC file from the collection you're going to import.
Step 3: Upload the DACPAC file and import files to an Azure storage account.
Step 4: Generate an SAS token to access the storage account.
Step 5: Complete the import specification.

> ⓘ **Note**
>
> Before you perform a production import, we *strongly* recommend that you complete a dry-run import. With a dry run, you can validate that the import process

Step 1: Detach your collection

Detaching the collection is a crucial step in the import process. Identity data for the collection resides in the Azure DevOps Server instance's configuration database while the collection is attached and online. When a collection is detached from the Azure DevOps Server instance, it takes a copy of that identity data and packages it with the collection for transport. Without this data, the identity portion of the import *can't* be executed. We recommend that you keep the collection detached until the import completes, because there isn't a way to import the changes that occurred during the import.

For a dry run (test) import, we recommend that you reattach your collection after you back it up for import, so you aren't concerned about having the latest data for this type of import. To avoid offline time altogether, you can also choose to employ an offline detach for dry runs.

It's important to weigh the cost of choosing to incur zero downtime for a dry run. It requires taking backups of the collection and configuration database, restoring them on a SQL instance, and then creating a detached backup. A cost analysis could prove that taking just a few hours of downtime to directly take the detached backup is better in the end.

Step 2: Generate a DACPAC file

DACPACs offer a fast and relatively easy method for moving collections into Azure DevOps Services. However, after a collection database size exceeds a certain threshold, the benefits of using a DACPAC start to diminish.

> ⓘ **Note**
>
> If the data migration tool displays a warning that you can't use the DACPAC method, you have to perform the import by using the SQL Azure virtual machine (VM) method provided in **Import large collections**.
>
> If the data migration tool doesn't display a warning, use the DACPAC method described in this step.

DACPAC is a feature of SQL Server that allows databases to be packaged into a single file and deployed to other instances of SQL Server. A DACPAC file can also be restored directly to Azure DevOps Services, so you can use it as the packaging method for getting your collection's data in the cloud.

> ⓘ **Important**
>
> - When you use SqlPackage.exe, you must use the .NET Framework version of SqlPackage.exe to prepare the DACPAC. The MSI Installer must be used to install the .NET Framework version of SqlPackage.exe. Do not use the dotnet CLI or .zip (Windows .NET 6) versions of SqlPackage.exe because those versions may generate DACPACs that are incompatible with Azure DevOps Services.
> - Version 161 of SqlPackage encrypts database connections by default and might not connect. If you receive a login process error, add `;Encrypt=False;TrustServerCertificate=True` to the connection string of the SqlPackage statement.

Download and install SqlPackage.exe using the latest MSI Installer from the SqlPackage release notes.

After you use the MSI Installer, SqlPackage.exe installs in a path similar to `%PROGRAMFILES%\Microsoft SQL Server\160\DAC\bin\`.

When you generate a DACPAC, keep two considerations in mind: the disk that the DACPAC is saved on and the disk space on the machine that's generating the DACPAC. You want to ensure that you have enough disk space to complete the operation.

As it creates the package, SqlPackage.exe temporarily stores data from your collection in the temp directory on drive C of the machine you're initiating the packaging request from.

You might find that your drive C is too small to support creating a DACPAC. You can estimate the amount of space you need by looking for the largest table in your collection database. DACPACs are created one table at a time. The maximum space requirement to run the generation is roughly equivalent to the size of the largest table in the collection's database. If you save the generated DACPAC to drive C, consider the size of the collection database as reported in the *DataMigrationTool.log* file from a validation run.

The *DataMigrationTool.log* file provides a list of the largest tables in the collection each time the command is run. For an example of table sizes for a collection, see the following output. Compare the size of the largest table with the free space on the drive that hosts your temporary directory.

> ⓘ **Important**
>
> Before you proceed with generating a DACPAC file, ensure that your collection is **detached**.

cmdline
```
[Info   @08:23:59.539] Table name                      Size in MB
[Info   @08:23:59.539] dbo.tbl_Content                 38984
[Info   @08:23:59.539] dbo.tbl_LocalVersion            1935
[Info   @08:23:59.539] dbo.tbl_Version                 238
[Info   @08:23:59.539] dbo.tbl_FileReference           85
[Info   @08:23:59.539] dbo.Rules                       68
[Info   @08:23:59.539] dbo.tbl_FileMetadata            61
```

Ensure that the drive that hosts your temporary directory has at least as much free space. If it doesn't, you need to redirect the temp directory by setting an environment variable.

cmdline
```
SET TEMP={location on disk}
```

Another consideration is where the DACPAC data is saved. Pointing the save location to a far-off remote drive could result in longer generation times. If a fast drive such as a solid-state drive (SSD) is available locally, we recommend that you target the drive as the DACPAC save location. Otherwise, it's always faster to use a disk that's on the machine where the collection database resides rather than a remote drive.

Now that you identified the target location for the DACPAC and ensured that you have enough space, it's time to generate the DACPAC file.

Open a Command Prompt window and go to the SqlPackage.exe location. To generate the DACPAC, replace the placeholder values with the required values, and then run the following command:

cmdline

```
SqlPackage.exe /sourceconnectionstring:"Data Source={database server
name};Initial Catalog={Database Name};Integrated Security=True" /targetFile:
{Location & File name} /action:extract /p:ExtractAllTableData=true
/p:IgnoreUserLoginMappings=true /p:IgnorePermissions=true /p:Storage=Memory
```

- **Data Source**: The SQL Server instance that hosts your Azure DevOps Server collection database.
- **Initial Catalog**: The name of the collection database.
- **targetFile**: The location on the disk and the DACPAC file name.

A DACPAC generation command that's running on the Azure DevOps Server data tier itself is shown in the following example:

cmdline

```
SqlPackage.exe /sourceconnectionstring:"Data Source=localhost;Initial
Catalog=Foo;Integrated Security=True" /targetFile:C:\DACPAC\Foo.dacpac
/action:extract /p:ExtractAllTableData=true /p:IgnoreUserLoginMappings=true
/p:IgnorePermissions=true /p:Storage=Memory
```

The output of the command is a DACPAC file, generated from the collection database *Foo* called *Foo.dacpac*.

Configure your collection for import

After your collection database restores on your Azure VM, configure a SQL sign-in to allow Azure DevOps Services to connect to the database to import the data. This sign-in allows only *read* access to a single database.

To start, open SQL Server Management Studio on the VM, and then open a new query window against the database to be imported.

Set the database's recovery to simple:

SQL

```
ALTER DATABASE [<Database name>] SET RECOVERY SIMPLE;
```

Create a SQL sign-in for the database, and assign that sign-in the 'TFSEXECROLE':

SQL

```
USE [<database name>]
CREATE LOGIN <pick a username> WITH PASSWORD = '<pick a password>'
```

```
CREATE USER <username> FOR LOGIN <username> WITH DEFAULT_SCHEMA=[dbo]
EXEC sp_addrolemember @rolename='TFSEXECROLE', @membername='<username>'
```

Following our Fabrikam example, the two SQL commands would look like the following example:

SQL

```
ALTER DATABASE [Foo] SET RECOVERY SIMPLE;

USE [Foo]
CREATE LOGIN fabrikam WITH PASSWORD = 'fabrikamimport1!'
CREATE USER fabrikam FOR LOGIN fabrikam WITH DEFAULT_SCHEMA=[dbo]
EXEC sp_addrolemember @rolename='TFSEXECROLE', @membername='fabrikam'
```

ⓘ **Note**

Be sure to enable **SQL Server and Windows authentication mode** in SQL Server Management Studio on the VM. If you don't enable authentication mode, the import fails.

Configure the import specification file to target the VM

Update the import specification file to include information about how to connect to the SQL Server instance. Open your import specification file and make the following updates.

1. Remove the DACPAC parameter from the source files object.

   The import specification before the change is shown in the following code.

   ```
   "Source": {
     "Location": "<Provide the SASKey to the Azure storage container with the collection and
     import files.>",
     "Files": {
       "Dacpac": "Tfs_DefaultCollection.dacpac"
     }
   },
   ```

   The import specification after the change is shown in the following code.

   ```
   "Source": {
     "Properties": {
       "ConnectionString": "Data Source=8.8.8.8;Initial Catalog=Tfs_Foo;Integrated Security=False;
       User ID=fabrikam;Password=fabrikam1!;Encrypt=True;TrustServerCertificate=True"
     }
   },
   ```

2. Fill out the required parameters and add the following properties object within your source object in the specification file.

JSON

```json
"Properties":
{
    "ConnectionString": "Data Source={SQL Azure VM Public IP};Initial
Catalog={Database Name};Integrated Security=False;User ID={SQL Login
Username};Password={SQL Login
Password};Encrypt=True;TrustServerCertificate=True"
}
```

After you apply the changes, the import specification looks like the following example.

Your import specification is now configured to use a SQL Azure VM for import. Proceed with the rest of preparation steps to import to Azure DevOps Services. After the import

finishes, be sure to delete the SQL sign-in or rotate the password. Microsoft doesn't retain the sign-in information after the import finished.

Step 3: Upload the DACPAC file

> ⓘ **Note**
>
> If you're using the SQL Azure VM method, you need to provide only the connection string. You don't have to upload any files, and you can skip this step.

Your DACPAC must be placed in an Azure storage container, which can be an existing container or one created specifically for your migration effort. It's important to ensure that your container is created in the right geographical locations.

Azure DevOps Services is available in multiple geographical locations ⬀. When you're importing to these locations, it's critical to place your data correctly to ensure that the import can start successfully. Your data must be placed in the same geographical location that you're importing to. Placing the data anywhere else results in the import being unable to start. The following table lists the acceptable geographical locations for creating your storage account and uploading your data.

⌗ Expand table

Desired import geographical location	Storage account geographical location
Central United States	Central United States
Western Europe	Western Europe
United Kingdom	United Kingdom South
Australia East	Australia East
Brazil South	Brazil South
India South	India South
Canada Central	Canada Central
Asia Pacific (Singapore)	Asia Pacific (Singapore)

Although Azure DevOps Services is available in multiple geographical locations in the US, only the Central United States location accepts new Azure DevOps Services. You can't import your data into other US Azure locations at this time.

You can create a blob container from the Azure portal. After you create the container, upload the Collection DACPAC file.

After the import finishes, delete the blob container and accompanying storage account with use tools such as AzCopy or any other Azure storage explorer tool, like Azure Storage Explorer ⧉ .

> ⓘ **Note**
>
> If your DACPAC file is larger than 10 GB, we recommend that you use AzCopy. AzCopy has multithreaded upload support for faster uploads.

Step 4: Generate an SAS token

A shared access signature (SAS) token provides delegated access to resources in a storage account. The token allows you to give Microsoft the lowest level of privilege required to access your data for executing the import.

SAS tokens can be generated using the Azure portal. From a security point-of-view, we recommend:

1. Select only **Read** and **List** as permissions for your SAS token. No other permissions are required.
2. Set an expiry time no further than seven days into the future.
3. Restrict access to Azure DevOps Services IPs only.
4. Place the SAS token in a secure location.

Step 5: Complete the import specification

Earlier in the process you partially filled out the import specification file, known as *import.json*. At this point, you have enough information to complete all the remaining fields except for the import type. The import type is covered later, in the import section.

In the *import.json* specification file, under **Source**, complete the following fields.

- **Location**: Paste the SAS key you generated from the script and then copied in the preceding step.
- **Dacpac**: Ensure that the file, including the *.dacpac* file extension, has the same name as the DACPAC file you uploaded to the storage account.

The final import specification file should look like the following example.

Restrict access to Azure DevOps Services IPs only

For more information, see Restrict access to Azure DevOps Services IPs only.

Option 1: Using Service Tags

You can easily allow connections from all Azure DevOps Services geographical locations by adding the `azuredevops` Service Tag to your network security groups or firewalls either through the portal or programmatically.

Option 2: Using IpList

Use the `IpList` command to get the list of IPs that need to be granted access to allow connections from a specific Azure DevOps Services geographical location.

Included in the help documentation are instructions and examples for running Migrator from the Azure DevOps Server instance itself and a remote machine. If you're running the command from one of the Azure DevOps Server instance's application tiers, your command should have the following structure:

```cmdline
Migrator IpList /collection:{CollectionURI} /tenantDomainName:{name}
/region:{region}
```

You can add the list of IPs to your network security groups or firewalls either through the portal or programatically.

Determine the import type

Imports can be queued as either a dry run or a production run. The **ImportType** parameter determines the import type:

- **DryRun**: Use a dry run for test purposes. The system deletes dry runs after 21 days.
- **ProductionRun**: Use a production run when you want to keep the resulting import and use the organization full time in Azure DevOps Services after the import finishes.

> 💡 **Tip**
>
> We always recommend that you complete a dry-run import first.

```json
{
  "Source": {
    "Location": "https://fabrikam.blob.core.windows.net/fabrikam?st=2017-08-28T17%3A15%3A00Z&se=2017-08-29T17%3A15%3A00Z&sp=rl&sv=2015-04-05&sr=c&sig=FSPF9sgiFCSmAWYz0icd0O9cxrmG9VP5pknFN16MgpY%3D",
    "Files": {
      "Dacpac": "Tfs_DefaultCollection.dacpac"
    }
  },
  "Target": {
    "Name": "fabrikam-import"
  },
  "Properties": {
    "ImportType": "DryRun"
  },
  "ValidationData": {
    "TfsMigratorVersion": "16.255.65000.0",
    "SourceCollectionId": "8b245d37-d41d-4188-a6f1-b5bb397860ba",
    "DataImportCollectionId": "ca970402-9b06-4720-9407-ba32684e9499",
    "DatabaseCollation": "SQL_Latin1_General_CP1_CI_AS",
    "CommandExecutionCount": 0,
    "CommandExecutionTime": 0.0,
    "TfsVersion": "Dev15.M117",
    "DatabaseTotalSize": 181,
    "DatabaseBlobSize": 0,
    "DatabaseTableSize": 181,
    "DatabaseLargestTableSize": 8,
    "ActiveUserCount": 8,
    "TenantId": "72f988bf-86f1-41af-91ab-2d7cd011db47",
    "Region": "CUS",
    "ValidationChecksumVersion": 1,
    "ValidationChecksum":
      "66S16G8u850KY6XKJm6MM5Ty3krNjhUFFCh4zyZMXqm7ZDLVpFpiIi0zDnJcoZmjHgDzvoCNS/9PwGm28hBgPg==",
  },
  "Identities": [
    "S-1-5-21-1374400868-3601225936-2087002269-500",
    "S-1-5-21-2127521184-1604012920-1887927527-11008431",
    "S-1-5-21-2127521184-1604012920-1887927527-15795496"
  ]
}
```

Dry-run organizations

Dry-run imports help teams test the migration of their collections. Organizations are expected not to remain around forever but to exist for a short time. In fact, before a production migration can be run, any completed dry-run organizations must be deleted. All dry-run organizations have a *limited existence and are automatically deleted after a set period of time*. Information about when the organization is deleted is included in the success email you should receive after the import finishes. Be sure to take note of this date and plan accordingly.

Most dry-run organizations have 15 days before they're deleted. Dry-run organizations can also have a 21-day expiration if more than 100 users have a basic or greater license at *import time*. After the specified time period, the dry-run organization is deleted. You can repeat dry-run imports as many times as you need before you do a production migration. You need to delete any previous dry runs before you attempt a new one.

When your team is ready to perform a production migration, you need to manually delete the dry-run organization.

For more information about post-import activities, see the post import article.

If you encounter any import problems, see Troubleshoot import and migration errors.

Run an import

Your team is now ready to begin the process of running an import. We recommend that you start with a successful dry-run import before you attempt a production-run import. With dry-run imports, you can see in advance how an import looks, identify potential issues, and gain experience before you head into your production run.

> ⓘ **Note**
>
> - If you need to repeat a completed production-run import for a collection, as in the event of a rollback, contact Azure DevOps Services **Customer Support** ⧉ before you queue up another import.

- Azure administrators can prevent users from creating new Azure DevOps organizations. If the Microsoft Entra tenant policy is turned on, your import fails to finish. Before you begin, verify that the policy isn't set or that there's an exception for the user that is performing the migration. For more information, see Restrict organization creation via Microsoft Entra tenant policy.
- Azure DevOps Services doesn't support per-pipeline retention policies, and they aren't carried over to the hosted version.

Considerations for rollback plans

A common concern for teams doing a final production run is their rollback plan, if anything goes wrong with import. We highly recommend doing a dry run to make sure that you can test the import settings you provide to the data migration tool for Azure DevOps.

Rollback for the final production run is fairly simple. Before you queue the import, you detach the team project collection from Azure DevOps Server, which makes it unavailable to your team members. If for any reason you need to roll back the production run and bring the on-premises server back online for your team members, you can do so. Attach the team project collection on-premises again and inform your

team that they continue to work normally while your team regroups to understand any potential failures.

Queue an import

> ⓘ **Important**
>
> Before you proceed, ensure that your collection was **detached** prior to generating a DACPAC file or uploading the collection database to a SQL Azure VM. If you don't complete this step, the import will fail. In the event that your import fails, see **Troubleshoot import and migration errors**.

Start an import by using the data migration tool's **import** command. The import command takes an import specification file as input. It parses the file to ensure that the provided values are valid and, if successful, it queues an import to Azure DevOps Services. The import command requires an internet connection, but doesn't* require a connection to your Azure DevOps Server instance.

To get started, open a Command Prompt window, and change directories to the path to the data migration tool. We recommended that you review the help text provided with the tool. Run the following command to see the guidance and help for the import command:

```cmdline
Migrator import /help
```

The command to queue an import has the following structure:

```cmdline
Migrator import /importFile:{location of import specification file}
```

The following example shows a completed import command:

```cmdline
Migrator import /importFile:C:\DataMigrationToolFiles\import.json
```

After the validation passes, you're asked to sign in to Microsoft Entra ID. It's important to sign in with an identity that's a member of the same Microsoft Entra tenant as the

identity map log file was built against. The signed in user is the owner of the imported organization.

> **ⓘ Note**
>
> Each Microsoft Entra tenant is limited to five imports per 24-hour period. Only imports that are queued count against this cap.

When your team initiates an import, an email notification is sent to the user that queued the import. About 5 to 10 minutes after it queues the import, your team can go to the organization to check on the status. After the import finishes, your team is directed to sign in, and an email notification is sent to the organization owner.

Related articles

- Migrate options
- Post-import

Import large collections

Article • 09/13/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

For databases that the data migration tool warns are too large, a different data packaging approach is required to migrate to Azure DevOps Services. If you're unsure whether your collection exceeds the size threshold, you should run a data migration tool validation on the collection. The validation lets you know whether you need to use the SQL Azure VM method for import.

Determine if you can reduce the collection size

Before you proceed, we recommend checking to see whether your old data can be cleaned up. Over time, collections can build up large volumes of data. This growth is a natural part of the DevOps process, but you might find that you don't need to retain all of the data. Some common examples of no longer relevant data are older workspaces and build results.

Cleaning older, no-longer-relevant artifacts could remove a lot more space than you might expect, and it could determine whether you use the DACPAC import method or a SQL Azure VM.

> ⓘ **Important**
>
> After you've deleted older data, it *can't* be recovered unless you restore an older backup of the collection.

If you're under the DACPAC threshold, follow the instructions to generate a DACPAC for import. If you still can't get the database under the DACPAC threshold, you need to set up a SQL Azure VM to import to Azure DevOps Services.

Set up a SQL Azure VM to import to Azure DevOps Services

Let's walk through how to accomplish this. At a high level, you'll:

- Set up a SQL Azure VM.

- (Optional) Restrict access to Azure DevOps Services IPs only.
- Configure IP firewall exceptions.
- Restore your database on the VM.
- Configure your collection for import.
- Configure the import specification file to target the VM

Set up a SQL Azure VM

You can set up a SQL Azure VM from the Azure portal with just a few clicks. To learn how, see Use the Azure portal to provision a Windows virtual machine with SQL Server.

> ⊘ **Note**
>
> While setting up your SQL Azure VM, bear in mind that the performance of the VM and attached data disks will have a significant impact on the performance of the import. For this reason, we *highly* recommend:
>
> - Selecting a VM Size at the level of D8s_v5_* or greater
> - Using managed disks
> - Consulting **Virtual machine and disk performance**. Please ensure your infrastructure is configured so that neither the VM IOPS or storage IOPS become a bottleneck on the performance of the import. For example, ensuring the number of data disks attached to your VM is sufficient to support the IOPS from the VM.

Azure DevOps Services is available in several Azure regions across the globe. These can be seen in the table below.

> ⓘ **Important**
>
> To ensure that the import starts successfully, it's critical to place your data in the correct region. If you set up your SQL Azure VM in a location other than the regions listed in the following table, the import will fail to start.

If you're using this import method, determine where to create your SQL Azure VM by referring to the following table. Creating your VM in a region other than those in this list is not supported for running an import.

Desired import region	SQL Azure VM region
Central United States	Central US
Western Europe	West Europe
Australia East	Australia East
Brazil South	Brazil South
South India	South India
Central Canada	Canada Central
Asia Pacific	Southeast Asia (Singapore)
UK South	UK South

Although Azure DevOps Services is available in multiple regions in the US, only the Central United States region accepts new organizations. Companies can't import their data into other US Azure regions at this time.

> ⓘ **Note**
>
> DACPAC customers should consult the region table in the **"Step 3: Upload the DACPAC file" section**. The preceding guidelines are for SQL Azure VMs only.

Here are a few more SQL Azure VM configurations that we recommend:

- Configure the SQL temporary database to use a drive other than drive C. Ideally the drive should have ample free space; at least equivalent to your database's largest table.
- If your source database is still over 1 terabyte (TB) after you've reduced its size, you need to attach additional 1-TB disks and combine them into a single partition to restore your database on the VM.
- If your collection databases are over 1 TB in size, consider using an SSD for both the temporary database and collection database. Also, consider using larger VMs with 16 virtual CPUs (vCPUs) and 128 GB of RAM.
- You need to have a public facing IP address for the service to reach this machine.

Restrict access to Azure DevOps Services IPs only

See the [Restrict access to Azure DevOps Services IPs only](#) page for more details.

Restore your database on the VM

After you set up and configure an Azure VM, you need to take your detached backup from your Azure DevOps Server instance to your Azure VM. Azure has [several documented methods](#) for how to accomplish this task. The collection database needs to be restored on your SQL instance and doesn't require Azure DevOps Server to be installed on the VM.

Configure your collection for import

After your collection database has been restored on your Azure VM, configure a SQL login to allow Azure DevOps Services to connect to the database to import the data. This login allows only *read* access to a single database.

To start, open SQL Server Management Studio on the VM, and then open a new query window against the database to be imported.

Set the database's recovery to simple:

```SQL
ALTER DATABASE [<Database name>] SET RECOVERY SIMPLE;
```

Create a SQL login for the database, and assign that login the 'TFSEXECROLE':

```SQL
USE [<database name>]
CREATE LOGIN <pick a username> WITH PASSWORD = '<pick a password>'
CREATE USER <username> FOR LOGIN <username> WITH DEFAULT_SCHEMA=[dbo]
EXEC sp_addrolemember @rolename='TFSEXECROLE', @membername='<username>'
```

Following our Fabrikam example, the two SQL commands would be:

```SQL
ALTER DATABASE [Foo] SET RECOVERY SIMPLE;

USE [Foo]
CREATE LOGIN fabrikam WITH PASSWORD = 'fabrikamimport1!'
```

```
CREATE USER fabrikam FOR LOGIN fabrikam WITH DEFAULT_SCHEMA=[dbo]
EXEC sp_addrolemember @rolename='TFSEXECROLE', @membername='fabrikam'
```

> ⓘ **Note**
>
> Be sure to enable **SQL Server and Windows authentication mode** in SQL Server
> Management Studio on the VM. If you don't enable authentication mode, the
> import will fail.

Configure the import specification file to target the VM

Update the import specification file to include information about how to connect to the
SQL Server instance. Open your import specification file and make the following
updates:

1. Remove the DACPAC parameter from the source files object.

   The import specification before the change is shown in the following code:

   ```
   "Source": {
     "Location": "<Provide the SASKey to the Azure storage container with the collection and
     import files.>",
     "Files": {
       "Dacpac": "Tfs_DefaultCollection.dacpac"
     }
   },
   ```

   The import specification after the change is shown in the following code:

   ```
   "Source": {
     "Properties": {
       "ConnectionString": "Data Source=8.8.8.8;Initial Catalog=Tfs_Foo;Integrated Security=False;
       User ID=fabrikam;Password=fabrikam1!;Encrypt=True;TrustServerCertificate=True"
     }
   },
   ```

2. Fill out the required parameters and add the following properties object within
   your source object in the specification file.

   JSON

   ```json
   "Properties":
   {
       "ConnectionString": "Data Source={SQL Azure VM Public IP};Initial
   Catalog={Database Name};Integrated Security=False;User ID={SQL Login
   Username};Password={SQL Login
   ```

```
        Password};Encrypt=True;TrustServerCertificate=True"
    }
```

Following the Fabrikam example, after you apply the changes, the import specification would look like the following:

Your import specification is now configured to use a SQL Azure VM for import. Proceed with the rest of preparation steps to import to Azure DevOps Services. After the import has finished, be sure to delete the SQL login or rotate the password. Microsoft does not retain the login information after the import has finished.

Related articles

- Validation and import processes

Restrict access to Azure DevOps Services IPs only

Article • 09/13/2023

We highly recommend that you restrict access to your Azure Storage account to only IPs from Azure DevOps Services. You can restrict access by only allowing connections from Azure DevOps Services IPs that are involved in the collection database import process. The IPs that need to be granted access to your storage account depend on the region you're importing into.

Option 1: Using Service Tags

You can easily allow connections from all Azure DevOps Services regions by adding the `azuredevops` [Service Tag](#) to your network security groups or firewalls either through the portal or programmatically.

Option 2: Using IpList

Use the `IpList` command to get the list of IPs that need to be granted access to allow connections from a specific Azure DevOps Services region.

Included in the help documentation are instructions and examples for running Migrator from the Azure DevOps Server instance itself and a remote machine. If you're running the command from one of the Azure DevOps Server instance's application tiers, your command should have the following structure:

```cmdline
Migrator IpList /collection:{CollectionURI} /tenantDomainName:{name}
/region:{region}
```

You can add the list of IPs to your network security groups or firewalls either through the portal or programatically.

Configure IP firewall exceptions for SQL Azure
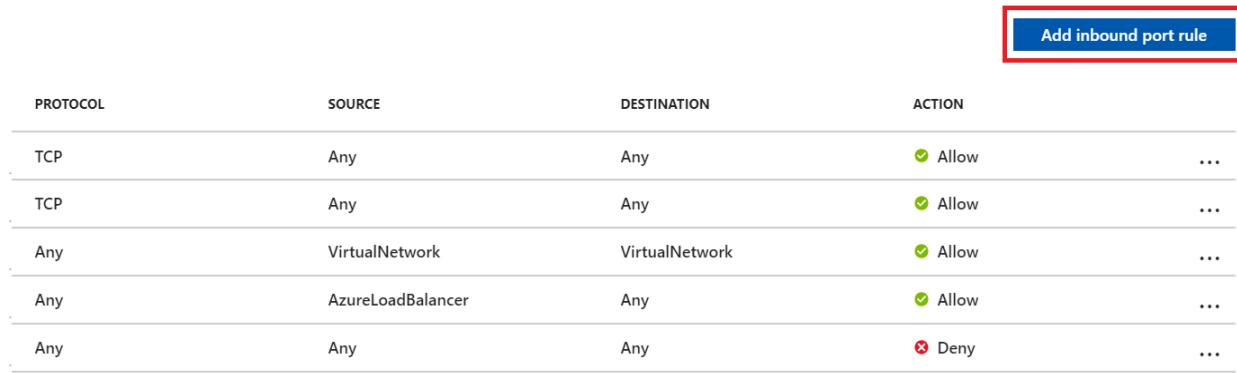
> ⓘ **Note**

This section only applies to configuring firewall exceptions for SQL Azure. For DACPAC imports, see **Configure Azure Storage firewalls and virtual networks**

> ⊙ **Note**
>
> The data migration tool requires the Azure DevOps Services IPs to be configured for inbound connections only on port 1433.

Granting exceptions for the necessary IPs is handled at the Azure networking layer for your SQL Azure VM. To get started, go to your SQL Azure VM in the Azure portal ⧉. In **Settings**, select **Networking**. You can grant exceptions for the IPs by selecting **Add inbound port rule** in the networking settings.

			Add inbound port rule

PROTOCOL	SOURCE	DESTINATION	ACTION	
TCP	Any	Any	✅ Allow	...
TCP	Any	Any	✅ Allow	...
Any	VirtualNetwork	VirtualNetwork	✅ Allow	...
Any	AzureLoadBalancer	Any	✅ Allow	...
Any	Any	Any	❌ Deny	...

On the **Add inbound security rule** pane, select **Advanced** to configure an inbound port rule for a specific IP.

Add inbound security rule   ✕

Service ⓘ

Custom   ⌄

* Port range ⓘ

8080   ✓

* Priority ⓘ

1510

* Name

Port_8080   ✓

Description

In the **Source** drop-down list, select **IP Addresses**, enter an IP address that needs to be granted an exception, set the **Destination port range** to **1433** and, in the **Name** box, enter a name that best describes the exception you're configuring.

Depending on other inbound port rules that have been configured, you might need to change the default priority for the Azure DevOps Services exceptions so they don't get ignored. For example, if you have a "deny on all inbound connections to 1433" rule with a higher priority than your Azure DevOps Services exceptions, the data migration tool might be unable to make a successful connection to your database.

* Source ⓘ

```
IP Addresses                           ⌄
```

* Source IP address range ⓘ

```
168.62.105.45                          ✓
```

* Source port range ⓘ

```
*                                      ✓
```

* Destination ⓘ

```
Any                                    ⌄
```

* Destination port range ⓘ

```
1433                                   ✓
```

* Protocol

| Any | TCP | UDP |

* Action

| Allow | Deny |

* Priority ⓘ

```
1010                                   ✓
```

* Name

```
VSTS_Identity_Service                  ✓
```

Description

```


```

Repeat adding inbound port rules until all necessary Azure DevOps Services IPs have been granted an exception. Missing one IP could result in your import failing to start.

Validate and resolve errors related to process templates

Article • 02/21/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

As part of the migration import process, the data migration tool checks the process used by the projects in the collection. Fix any errors that get flagged.

After resolving the errors, rerun the data migration tool's `validate` command to verify that all errors have been fixed.

> ⓘ **Note**
>
> It's recommended that you use the **Migration Guide** ⬀ to progress through your import. The guide links to the technical documentation as needed.
>
> With the release of Azure DevOps Server 2019 the TFS Database Import Service was rebranded to Migrate to Azure DevOps. This includes TfsMigrator becoming the data migration tool or migrator for short. This service still works exactly the same as the old Import Service. If you're on an older version of on-premises with TFS as the branding you can still use this feature to migrate to Azure DevOps as long as you upgrade to one of the supported versions.

Process validation types

During validation, the data migration tool determines the target process model for each project. It automatically assigns one of the following two process models to each project in the collection:

- **Inherited process model**: If the project was created with the Agile, Scrum, or CMMI process template, and was never customized.
- **Hosted XML process model**: If the project process appears to have been customized. A customized process contains custom fields, work item types, or other types of customizations.

When the Hosted XML process is the targeted process model, the data migration tool validates if the customizations can be migrated. The data migration tool generates two files during the validation:

- **DataMigrationTool.log**: Contains the set of process validation errors found in the collection. Fix all process errors found to proceed with your migration.

- **TryMatchOobProcesses.log**: Lists for each project the target process model - Inheritance or Hosted XML. For projects that are set to target the Hosted XML process model, it explains why they are considered to be customized. You don't have to fix these errors, but they give you guidance what to do in case you want to migrate to the Inheritance process model. Note that once a collection is imported, you can migrate a project to an Inheritance process model.

Most customers have a mix of projects within a collection. Some projects use a default process template and others use custom process templates. The data migration tool checks and validates each project accordingly. It is very possible that you'll have a mix of projects, some mapped to an Inherited process and others to a Hosted XML process.

We recommend that for any project that has not been customized, that you review the **TryMatchOobProcesses.log** to determine if there are any errors. If so, make the adjustments accordingly so that the project can be mapped to an Inherited process upon data import.

Update to a system process

If you started with an older version of Azure DevOps Server, odds are your projects are still using an older process template. If those projects have not been updated using the Configure Features Wizard then the data migration tool will find process errors. In some rare cases, if your process is very old, even the Configure Features Wizard won't be able to resolve the errors.

Here are some examples of error messages you may receive:

```
no-highlight

Invalid process template: WorkItem
Tracking\Process\ProcessConfiguration.xml:: TF402571: Required element
PortfolioBacklog is missing from Process Configuration.
Invalid process template: WorkItem
Tracking\Process\ProcessConfiguration.xml:: TF402571: Required element
BugWorkItems is missing from Process Configuration.
Invalid process template: WorkItem
Tracking\Process\ProcessConfiguration.xml:: TF402571: Required element
FeedbackRequestWorkItems is missing from Process Configuration.
Invalid process template: WorkItem
Tracking\Process\ProcessConfiguration.xml:: TF402571: Required element
FeedbackResponseWorkItems is missing from Process Configuration.
Invalid process template: WorkItem
Tracking\Process\ProcessConfiguration.xml:: TF402574: ProcessConfiguration
```

```
    doesn't specify required TypeField Team.
    Invalid process template: WorkItem
    Tracking\Process\ProcessConfiguration.xml:: TF402574: ProcessConfiguration
    doesn't specify required TypeField RemainingWork.
    Invalid process template: WorkItem
    Tracking\Process\ProcessConfiguration.xml:: TF402574: ProcessConfiguration
    doesn't specify required TypeField Order.
    Invalid process template: WorkItem
    Tracking\Process\ProcessConfiguration.xml:: TF402574: ProcessConfiguration
    doesn't specify required TypeField Effort.
    Invalid process template: WorkItem
    Tracking\Process\ProcessConfiguration.xml:: TF402574: ProcessConfiguration
    doesn't specify required TypeField Activity.
    Invalid process template: WorkItem
    Tracking\Process\ProcessConfiguration.xml:: TF402574: ProcessConfiguration
    doesn't specify required TypeField ApplicationStartInformation.
    Invalid process template: WorkItem
    Tracking\Process\ProcessConfiguration.xml:: TF402574: ProcessConfiguration
    doesn't specify required TypeField ApplicationLaunchInstructions.
    Invalid process template: WorkItem
    Tracking\Process\ProcessConfiguration.xml:: TF402574: ProcessConfiguration
    doesn't specify required TypeField ApplicationType.
    Invalid process template: WorkItem
    Tracking\Process\ProcessConfiguration.xml:: TF400572: The Project Process
    Settings must be configured for this feature to be used.
```

If you have never customized your project (added fields, work item types, etc.), then fixing these errors is actually pretty simple. If you have customized your process, then this approach won't work. You'll need to manually change the process templates so that your customizations don't get overwritten.

First, make sure you know what process your project started as. Is it Scrum, Agile or CMMI? In this example, let us assume Agile. Next, go to the Process Customization Scripts ⬀ provided on GitHub and download the repo. In this instance, we are going to focus on contents in the **Import** folder.

Use the **ConformProject.ps1** script to conform a project of your choosing to the Agile system process. This will update the entire project to be Agile.

```
.\ConformProject.ps1 "<collection url>" "<project name>" "c:\process-
customization-scripts\import\agile"
```

Make sure you do this for each and every project.

Resolve process errors

Are your process templates customized? Are you using an older outdated process template? If so, you'll most likely have process validation errors. The data migration tool does an exhaustive check against your process templates. It checks to make sure that it is valid for Azure DevOps Services. Odds are that you'll need to make some adjustments and apply them to your collection.

> ⓘ **Note**
>
> If you are using an OOB Agile, Scrum, or CMMI process, you probably won't see any errors in the **DataMigrationTool.log**. Instead, check the **TryMatchOobProcesses.log** for errors. If you are error free, then your project will map to an OOB process.

There are several customizations that won't work in Azure DevOps Services. Make sure you review the list of customizations that are supported.

If you have projects that are using an older process template, the data migration tool will find several errors. This is because your process templates hasn't been updated to match the most recent process templates. To start, try running the Configure Features Wizard for each project. This will attempt to update your process templates with the most recent features. Doing so should drastically reduce the error count.

Finally, make sure you have witadmin on the machine that you intend to use to fix the process errors. This can be your local desktop. The `witadmin` command line tool is used in the automated scripts and is required whenever making changes to the process templates.

Step 1 - Review errors

DataMigrationTool.log file will be generated and contains the list of errors that the validation process found. To view the logs, open DataMigrationTool.log file. Search for the string "Validation - Starting validation of project 1". Each project is validated. Scan through all the projects and search for any lines that contain a prefix of **[Error ....**

For a list of validation errors, see Resolve validation errors for process import. For each validation error, we have provided the error number, description, and the method to resolve.

Step 2 - Fix errors

Once you've determined which projects have errors and the error details, fix the errors. Fixing the errors requires that you change the XML syntax and apply the changes back to the project.

> ⓘ **Note**
>
> We recommend you don't use TFS Power Tools to do this work. Instead, we highly recommended that you modify the XML manually.

To get the process template from the project add the **/SaveProcesses** parameter when running the data migration tool command.

cmdline

```
Migrator validate /collection:{collection URL} /SaveProcesses
```

This command will extract the XML from the project and place it into the same folder as the logs. Extract the zip files to your local machine so that you can edit the files.

Now, fix the XML. Use the logs from the `DataMigrationTool.log` file to determine the errors for each project.

Some errors will require you to do use a witadmin changefield command. Changing a field name is the most common example. To save yourself some time, we recommend you run the `witadmin changefield` command and then re-run the data migration tool. Doing this will re-export the XML with the corrected names. Otherwise, you'll need to manually fix the fields in the XML syntax as well.

Once you make a fix, apply the changes back to the Azure DevOps Server. To do this, depending on the changes you made, you'll need to run one or more witadmin commands. To make this easier for you, we created a PowerShell script to automate the process. The script contains all of the `witadmin` commands needed to conform the entire process.

You can get the scripts at Process Customization Scripts ↗. Use the **import/ConformProject.ps1** script.

cmdline

```
.\conformproject.ps1 "<collection url>" "<project name>" "<process template folder>"
```

When the script has completed, re-run the data migration tool to validate the collection. Follow steps 1 through 3 until the data migration tool generates no more validation errors.

> 💡 **Tip**
>
> If you are new to XML and `witadmin`, we suggest you make one fix at a time and then conform. Continue this loop until all errors are resolved.

Common validation errors

VS402841: Field X in work item type Bug has syncnamechanges=false but has rules making it an identity field. Identity fields must have syncnamechanges=true. Please update your process template to include this change.

In Azure DevOps Services we added a rule so that every identity field must have the **syncnamechanges=true** attribute. In Azure DevOps Server that rule does not apply. Therefore, the data migration tool will identify this as an issue. Don't worry, making this change on Azure DevOps Server on-prem will not cause any harm.

Run the `witadmin changefield` command. Syntax for the command looks similar to the following:

```cmdline
witadmin changefield
/collection:http://AdventureWorksServer:8080/tfs/DefaultCollection
/n:fieldname /syncnamechanges:true
```

For more information on the `witadmin changefield` command see [Manage work item fields](#).

TF402556: For field System.IterationId to be well defined, you must name it Iteration ID and set its type to Integer.

This error is typical for old process templates. Try running the [Configure Features Wizard](#) on each project. Alternatively you can run the follow `witadmin` command:

```cmdline
```

```
witadmin changefield
/collection:http://AdventureWorksServer:8080/tfs/DefaultCollection
/n:fieldname /name:newname
```

TF402571: Required element BugWorkItems is missing from Process Configuration.

This error typically occurs when a process hasn't been updated in a while. Try running the configure features wizard on each project to resolve.

TF402564: You've defined XX global lists. Only 64 are allowed.

By default, Azure DevOps Services will support 64 global lists. You'll typically run across this error if you have a large amount of build pipelines. The global list named Builds - **TeamProjectName** gets created for each new build pipeline. You'll need to remove the outdated global lists.

Related articles

- Migration and process model FAQs
- witadmin: Customize and manage objects for tracking work
- Differences between Azure DevOps Services and Azure DevOps Server process template customizations
- Configure features after Azure DevOps Server upgrade
- Resolve validation errors
- Define global lists in Azure DevOps Server
- Process customization PowerShell scripts ⧉

Post import

Article • 02/24/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Once a successful import is completed, an organization is ready to use. However, before you make it available to all users, there are several common tasks you should complete. See the following list of the most typical tasks that should be completed after import, in the recommended order of execution.

> ⓘ **Note**
>
> We recommend that you use the **Migration Guide** ⧉ to progress through your import. The guide links to the technical documentation as needed.
>
> With the release of Azure DevOps Server 2019 the TFS Database Import Service was rebranded to become data migration tool for Azure DevOps. This includes TfsMigrator becoming the data migration tool or migrator for short. This service still works exactly the same as the old Import Service. If you're on an older version of on-premises with TFS as the branding you can still use this feature to migrate to Azure DevOps as long as you upgrade to one of the supported versions.

Immediately after import

Immediately after the organization becomes available, take a small team and perform spot checks on the organization. We recommend that this team consists of the project collection administrators. This shouldn't be an in-depth check, but rather making sure that major pieces from your collection were brought over. Did your source code get imported? Are you seeing your build history? Are all of our area paths still present? It's best to confirm these artifacts are present before opening the organization to the entirety of your user base.

After you check the organization, consider whether you want to rename it. Renaming an organization is a simple operation, but it has large impacts ⧉ on users currently using the organization. Some examples being Team Explore connections breaking or bookmarks no longer working. Getting a rename out of the way while it's just a small group of users using the organization allows the rest of the users to come in and configure their connections once.

Set up billing

To pay for users or services in Azure DevOps, like hosted build and deployment agents, you need to set up billing for your organization. If you import more than one collection, you should ensure all your organizations are set up for billing with the same Azure subscription, and that your subscription is enabled for multi-organization billing. You can then assign as many Basic users as you need free of charge during the calendar month in which you run the import.

Manage users and access

Your organization includes five free users with Basic ⧉ access. Basic includes features like Git and Team Foundation version control, tools for Agile planning and Java teams, and more. Also, you can add Visual Studio subscribers ⧉ for free—they get basic features plus other features—based on their subscription level. Also, you can add Stakeholder for free, which allows them to have partial access to Agile tools, create work items, and view backlogs and boards.

As Visual Studio subscribers sign in to the organization, they're automatically detected. For all other users, you need to assign paid access. Keep in mind, if you automate access using group rules, the rules only apply to existing users if you remove direct assignments, which were applied to users during import.

Behavior change—Starting November 13, 2019, the default access behavior for imports will change. Previously, all imports tried to give users an equivalent access level post import. This means that users that had *Basic* received Basic access, and other users started with *Stakeholder* access. Once this change happens, all users start out with free *Stakeholder* access. **You will continue to be able to assign Basic access to any users who need it at no cost, until the end of the calendar month during which your import is run.** If you have any questions or concerns about this change, feel free to contact us.

Builds

Next, you want to configure your build agents. As part of the migration, all of your build pipelines have been brought over, but agents and pools need to be reconfigured against the new organization. Azure DevOps offers the ability to use a Microsoft-hosted pool of build agents that you can use, or you can connect your self-hosted build agent(s). It's important to note that only one self-hosted build agent is included for free. After that there's a fee ⧉ for having more self-hosted build agents. To pay for Microsoft-

hosted and self-hosted build agents, you need to link a subscription to your organization. See the following resources for details on performing this task:

- Build Agents

If you plan on using your existing on-premises private build agents, there's one more recommended step that needs to be taken after registering them to your new organization. Clearing their cache ensures that you don't encounter any build issues related to older TFVC or Git pointers to your on-premises collection. See refreshing caches on client computers for details on how to accomplish this task.

Release management

If you used Release Management in Azure DevOps Server, then your release pipelines and history data are included with your import. However, like builds, you need to reconfigure your agents and pools against the new organization.

Azure Artifacts

Azure Artifacts is included with Azure DevOps Services for all users granted a **Basic** license. There's no need to install an extension. Your Azure Artifacts data should be available post import.

Azure Boards

If you have an existing GitHub Enterprise Server connection associated with your Azure DevOps Server, it will not work as expected. Work item mentions, within GitHub, may be delayed or never show up in Azure DevOps Services. This problem occurs because the callback URL associated with GitHub is no longer valid.

To resolve the problem, consider the following items:

- **Remove and re-create the connection**: Remove and re-create the connection to the GitHub Enterprise Server repository. Follow the sequence of steps provided in Connect from Azure Boards documentation.

- **Fix the webhook url**: Go to GitHub's repository settings page and edit the webhook url to point out to the migrated Azure DevOps Services organization url: `https://dev.azure.com/{OrganizationName}/_apis/work/events?api-version=5.2-preview`

Notify your teams

After your builds are running and license subscription is configured, we recommend that you open up the organization to all users for validation. Then individual users can ensure that all of the content is in place, they have the right access level, and that they can pull code. Be sure to point users to our documentation on connecting to Azure DevOps Services from all of our supported IDEs and Team Explorer.

Users of TFVC with local workspaces need to remap their workspaces against the new organization and Git users have to reconfigure their remotes to be able to pull code.

If anything is reported as missing from the migrated organization, reach out to AzureDevOpsImport@microsoft.com. For other functional issues, reach out to customer support ☑ .

Troubleshoot import and migration errors

Article • 10/19/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

The data migration tool flags errors that you need to correct prior to performing a migration to Azure DevOps Services. This article describes the most common warnings and errors that you might receive when you're preparing to migrate. After you correct each error, run the **migrator validate** command again to verify resolution.

> ⓘ **Note**
>
> We recommended that you use the **Migration guide** ⧉ to progress through your import. The guide links to the technical documentation as needed.
>
> With the release of Azure DevOps Server 2019, the Team Foundation Server (TFS) Database Import Service was re-branded to become the data migration tool for Azure DevOps. The data migration tool, **TfsMigrator** has been renamed **Migrator** for short. The service still works exactly the same as the previous import service. If you're on an older version of on-premises with TFS as the branding, you can still use **Migrator** to migrate to Azure DevOps as long as you upgrade to one of the supported versions. For details, see **Migrate data from Azure DevOps Server to Azure DevOps Services**.

Resolve size warnings

Extra-large collections might generate one of the following messages after running the data migration tool. If you receive any of these warnings or errors, we recommend that you try to reduce your database's size.

Database size over recommended size

The following warning means you need to use the SQL Azure VM method to complete your import. Once a database reaches a certain size, it becomes faster to set up a SQL Azure VM to complete the import to Azure DevOps Services. To set up the VM and complete your import, follow the instructions linked from the warning message.

```cmdline
The database is currently {Database Size}GBs. This is above the recommended
size of {DACPAC Size Limit}GBs to use the DACPAC import method. Please see
the following page to learn how to import using a SQL Azure VM:
https://aka.ms/AzureDevOpsImportLargeCollection
```

This warning **DOES NOT** mean that your collection is too large for import.

Table size over recommended size

Similar to the previous warning, the following warning means you must use the SQL
Azure Virtual Machine (VM) method to complete the import. Follow the instructions
linked from the warning message to set up the VM and complete your import.

```cmdline
The largest table size is currently {Table size}GBs. This is above the
recommended size of {Size limit}GBs to use the DACPAC import method. Please
see the following page to learn how to import using a SQL Azure VM:
https://aka.ms/AzureDevOpsImportLargeCollection
```

This warning **DOES NOT** mean that your collection is too large for import.

Database metadata size over recommended size

The following warning means that your database is approaching the limit for total
metadata size. Metadata size refers to the size of your database without including files,
code, and other binary data. We recommend that you reduce the size of your database
before import. Reducing the size provides the other benefit of speeding up your import.

```cmdline
The database metadata size is currently {Metadata Size}GBs. This is above
the recommended size of {Warning Size}GBs. It's recommended that you
consider cleaning up older data as described in [Cleaning up old data]
(/azure/devops/server/upgrade/clean-up-data).
```

The warning **DOES NOT** mean that your collection is too large for import, rather its
metadata size is larger than most other databases.

Database metadata size over maximum supported size

Unlike the previous warnings, the following error **WILL** block you from moving forward with your migration.

It indicates that the volume of metadata in your collection is too large. To proceed with the import, you need to reduce the size below the indicated limit.

```cmdline
The database metadata size is currently {Metadata Size}GBs. This is above
the maximum supported size of {Metadata Limit}GBs.
```

Resolve collation warnings

Collation warnings refer to your collection database's collation. Collations control the way string values are sorted and compared. Collections that aren't using either `SQL_Latin1_General_CP1_CI_AS` or `Latin1_General_CI_AS` receive one of the **warning** messages.

No native support

Receiving the following warning means that you need to consider collation implications before performing the import.

```cmdline
The collection database's collation '{collation}' is not natively supported
in Azure DevOps Services. Importing your collection will result in your
collation being converted to one of the supported Azure DevOps Services
collations. See more details at https://aka.ms/AzureDevOpsImportCollations
```

This warning **DOES NOT** mean that you can't import your collection.

This warning requires you to acknowledge acceptance of the warning. Accepting the warning allows the data migration tool to continue import preparations.

When you import a nonsupported collation into Azure DevOps Services, the collation is transformed to a supported collation. While this transform generally works without issue, unexpected results post import or import failures could occur.

For instance, customers might notice different ordering for strings containing non-English characters. Non-English characters like 'é' might become equivalent to the English 'e' after import. It's important that you complete and verify a dry run import when importing a collection with a nonsupported collation.

No native support, no internet connection

If the data migration tool can't connect to the internet, it can't validate conversion of your collation. It's only a warning, so you can continue with your migration process. However, when you run the **prepare** command, an internet connection is required and collation conversion is validated at that time.

```cmdline
The collections database's collation '{collation}' is not natively supported
in Azure DevOps Services. It could not be validated that the collation can
be converted during import to a supported Azure DevOps Services collation,
as there was no internet connection. Please run the command again from a
machine with an internet connection. See more details at
https://aka.ms/AzureDevOpsImportCollations
```

Unsupported database collation

Generally you can convert a nonsupported collation to a supported collation at import time. However, some collations can't be converted. If your collection uses one of these collations, you receive the following **error** message.

```cmdline
The collection database's collation '{collation}' is not supported for
import to Azure DevOps Services. It will need to be changed to a supported
collation before it can be imported. See more details at
https://aka.ms/AzureDevOpsImportCollations
```

In order to continue, you need to change your collection's collation to one of the supported collations on Azure DevOps Services.

Resolve identity errors

Fix identity errors before migration to prevent problems. They're rare and happen when old TFS operations are invalid on new Azure DevOps Server. For instance, some users can't be in valid users group anymore.

The following sections provide guidance for resolving the most common identity errors.

ISVError: 100014

This error indicates that a permission is missing from a system security group. For example, every collection that you create has Project Collection Valid Users and Project Collection Administrators groups. The system creates them by default. These groups don't support editing of their permissions.

This error indicates that one or more groups is missing a permission that it's expected to have. To resolve this error, use the **TFSSecurity.exe** command to apply the expected permissions onto the flagged system groups. Your first step is to identify which TFSSecurity command(s) you need to run.

Project Collection Valid Users error message

Examine the error message(s) the data migration tool highlighted. If the flagged group ends with "**0-0-0-0-3**", such as in the following example, you need to fix a missing permission for the **Project Collection Valid Users** group.

Run the following command, replace the scope with the one from the error message and specify your collection URL.

```cmdline
TFSSecurity.exe /a+ Identity "{scope}\\" Read sid:{Group SID} ALLOW
/collection:{collectionUrl}
```

You determine the scope and group security ID (SID) from the error message.

```cmdline
ISVError:100014 Missing permission for
group:Microsoft.TeamFoundation.Identity;S-1-9-XXXXXXXXXX-XXXXXXXXXX-
XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-3 for scope:397c326b-b97c-4510-8271-
75aac13de7a9. Expected:1 and Actual:0
```

The final command appears similar to the following entry:

```cmdline
TFSSecurity.exe /a+ Identity "397c326b-b97c-4510-8271-75aac13de7a9\\" Read
sid:S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-3 ALLOW
/collection:https://localhost:8080/defaultcollection
```

Project Collection Administrators error message

Carefully examine the error message(s) the data migration tool highlighted. If the flagged group that ends with "**0-0-0-0-1**", such as in the following example, then you need to fix a missing permission for the **Project Collection Administrators** group. Run the following commands against **TFSSecurity.exe**, replace the scope with the one from the error message and specify your collection.

cmdline

```
TFSSecurity.exe /a+ Identity "{scope}\\" Read sid:{Group SID} ALLOW
/collection:{collectionUrl}

TFSSecurity.exe /a+ Identity "{scope}\\" Write sid:{Group SID} ALLOW
/collection:{collectionUrl}

TFSSecurity.exe /a+ Identity "{scope}\\" Delete sid:{Group SID} ALLOW
/collection:{collectionUrl}

TFSSecurity.exe /a+ Identity "{scope}\\" ManageMembership sid:{Group SID}
ALLOW /collection:{collectionUrl}
```

In the following example, take the scope and group `SID` from the error message and add them to the preceding command.

cmdline

```
ISVError:100014 Missing permission for
group:Microsoft.TeamFoundation.Identity;S-1-9-XXXXXXXXXX-XXXXXXXXXX-
XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1 for scope:0c7c2216-fa4b-4107-a203-
82b324a147ef. Expected:15 and Actual:0
```

The final command appears similar to the following entry:

cmdline

```
TFSSecurity.exe /a+ Identity "0c7c2216-fa4b-4107-a203-82b324a147ef\\" Read
sid:S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1 ALLOW
/collection:https://localhost:8080/defaultcollection

TFSSecurity.exe /a+ Identity "0c7c2216-fa4b-4107-a203-82b324a147ef\\" Write
sid:S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1 ALLOW
/collection:https://localhost:8080/defaultcollection

TFSSecurity.exe /a+ Identity "0c7c2216-fa4b-4107-a203-82b324a147ef\\" Delete
sid:S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1 ALLOW
/collection:https://localhost:8080/defaultcollection

TFSSecurity.exe /a+ Identity "0c7c2216-fa4b-4107-a203-82b324a147ef\\"
```

```
ManageMembership sid:S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-
0-0-1 ALLOW /collection:https://localhost:8080/defaultcollection
```

When you need to correct multiple errors, we recommend that you create a batch file to automate execution of the commands. Once you've executed the commands, you need to rerun the data migration **validate** tool to verify resolution. If some errors still persist, contact Azure DevOps Services customer support ⬀ .

ISVError: 300005

ISVError: 300005 indicates that a nongroup identity is a member of an everyone group, more commonly known as the Valid Users groups. Valid Users groups are default groups defined for all projects and collections. These groups aren't editable. They're designed to only contain other Azure DevOps permission or security groups as members. This error indicates that an Active Directory (AD) group or user identity has a direct membership in a Valid Users group.

> ⓘ **Important**
>
> Ensure that you have a backup of your collection and configuration databases before running the following commands to resolve the error.

Since you can't directly edit Valid Users groups, you need to run a SQL statement against the configuration database to remove the offending identity and correct the invalid membership. Carefully examine the error messages highlighted by the data migration tool. Copy the `GroupSid`, `MemberId`, and `ScopeId` as you need to place these values into the following command.

```SQL
DECLARE @p6 dbo.typ_GroupMembershipTable

INSERT into @p6
values('{GroupSid}','Microsoft.TeamFoundation.Identity','{MemberId}',0)

EXEC prc_UpdateGroupMembership
@partitionId=1,@scopeId='{ScopeId}',@idempotent=1,@incremental=1,@insertInac
tiveUpdates=0,@updates=@p6,@eventAuthor='9EE20697-5343-43FC-8FC5-
3D5D455D21C5',@updateGroupAudit=0
```

The following example lists an example of an ISVError: 300005 message from the data migration tool.

```
cmdline
```

```
ISVError:300005 Unexpected non group identity was found to have direct
membership to everyone group. GroupSid:S-1-9-1551374245-3746625149-
2333054533-2458719197-2313548623-0-0-0-0-3, MemberId:76050ddf-4fd8-48c4-
a1ff-859e44364519, ScopeId:7df650df-0f8b-4596-928d-13dd89e5f34f
```

If the error message lists a `MemberSid`, you need to get the `MemberID` from the dbo.tbl_Identity table in the configuration database. With the `MemberID`, you can then look up the GUID for the `MemberSid`.

```
cmdline
```

```
ISVError:300005 Unexpected non group identity was found to have direct
membership to everyone group. GroupSid:S-1-9-1551374245-3746625149-
2333054533-2458719197-2313548623-0-0-0-0-3,
MemberSid:System.Security.Principal.WindowsIdentity;S-1-5-21-124525095-
708259637-1543119021-1737349, ScopeId:7df650df-0f8b-4596-928d-13dd89e5f34f
```

```
SQL
```

```
DECLARE @MemberId uniqueidentifier

SET @MemberId = (Select Id from dbo.tbl_Identity where Sid ='S-1-5-21-
124525095-708259637-1543119021-1737349');

SELECT @MemberId
```

Copy the `GroupSid`, `MemberId`, and `ScopeId` into the SQL command.

```
SQL
```

```
DECLARE @p6 dbo.typ_GroupMembershipTable

INSERT into @p6 values('S-1-9-1551374245-3746625149-2333054533-2458719197-
2313548623-0-0-0-0-3','Microsoft.TeamFoundation.Identity','76050ddf-4fd8-
48c4-a1ff-859e44364519',0)

EXEC prc_UpdateGroupMembership @partitionId=1,@scopeId='7df650df-0f8b-4596-
928d-
13dd89e5f34f',@idempotent=1,@incremental=1,@insertInactiveUpdates=0,@updates
=@p6,@eventAuthor='9EE20697-5343-43FC-8FC5-3D5D455D21C5'
```

Run the completed command against the Azure DevOps Server configuration database. Repeat this command for each ISVError: 300005 instance reported. You can batch errors with the same scope ID into a single command. Once you've executed the commands,

rerun the data migration tool validate again to ensure that the errors have been corrected. If the errors still persist, contact Azure DevOps Services customer support ⬀.

> ⓘ **Important**
>
> To address these errors, the collection must be attached.
>
> If you receive a -1 result when you run the command, ensure that your collection database that produced the error is attached to your Azure DevOps Server instance and that you're running the command on the configuration database.

Microsoft Entra timeout exception

On rare occasions, you might receive a Microsoft Entra timeout error when running the data migration tool prepare command.

```cmdline
Exception Message: Request failed (type AadGraphTimeoutException)
```

This error means that the requests to Microsoft Entra ID to find the matching Microsoft Entra identities for users in your collection timed out. Generally, you can resolve this error by waiting to run the **prepare** command at a less busy time of the day, such as after regular business hours.

To troubleshoot, test Microsoft Entra ID connection from **prepare** machine. Follow these steps to get user info from Microsoft Entra ID.

Open PowerShell in elevated mode and replace 'someone@somecompany.com' in the following command with your Microsoft Entra user identity.

```PowerShell
# Install the Microsoft Graph PowerShell module - ensuring to select Yes to All
Install-Module Microsoft.Graph

# Import Users module
Import-Module Microsoft.Graph.Users

# Connect to Microsoft Entra and use your Microsoft Entra ID credentials
(someone@somecompany.com) to login when the pop-up appears
Connect-MgGraph  -Scopes 'User.Read.All'
```

```
# Try to retrieve information on a user from your Microsoft Entra
Get-MgUser -Filter "UserPrincipalName eq 'someone@somecompany.com'"
```

If the steps fail or you can't find the user, check the connection between the **prepare** machine and Microsoft Entra ID. Run a network trace with **prepare** to see if the network blocks calls. If not, contact Azure support. Check the log file for the user information.

```cmdline
Number of active users is {Number of Users}.
```

If the number of active users is over 50,000, the volume of identities being mapped might require more time than provided by the timeout limit. Inspect your collection for inclusions of large groups such as an 'everyone' group. If possible, remove these groups and try again. If you still can't resolve this error, contact Azure DevOps Services customer support ⧉ .

Resolve process errors

See the separate Process Templates page for details on resolving common process errors.

Resolve field validation errors

VS403310

The following error message can occur when an inconsistency in collection files is detected. Contact customer support if you encounter this error.

```
VS403310: An inconsistency was detected in some of the files in the collection.
```

VS403442

Field name conflicts sometimes occur between your local collection and an Azure DevOps Services system field.

```
In order to migrate successfully, you must rename field *{TFSfieldReferenceName}*.
Given name *{TFSfieldName}* is reserved for field *{VSTSfieldReferenceName}*.
```

To resolve this error, change the name of your collection field. Use the **witadmin changefield** command from witadmin.

```
cmdline

witadmin changefield
/collection:http://AdventureWorksServer:8080/DefaultCollection
/n:TFSfieldReferenceName /name:newFieldName
```

VS403443

The following error indicates a field name conflict exists between your local collection and a specific Azure DevOps Services field.

`In order to migrate successfully, you must rename field *{TFSfieldReferenceName}*`
`to *{VSTSfieldName}*. Given name for *{TFSfieldReferenceName}* is *{TFSfieldName}*`

To resolve this error, use the **witadmin changefield** command. For details, see witadmin.

```
cmdline

witadmin changefield
/collection:http://AdventureWorksServer:8080/DefaultCollection
/n:TFSfieldReferenceName /name:VSTSfieldName
```

VS403444

The following error indicates a field type conflict exists between your local collection and Azure DevOps Services.

Using witadmin, you can change the data type only for HTML or PlainText fields.

`In order to migrate successfully, you must set type of field *`
`{TFSfieldReferenceName}* to *{Type}*. Given type for *{TFSfieldReferenceName}* is *`
`{collectionType}*.`

If your field type is HTML or PlainText, then you can change its type to the required type.

```
cmdline

witadmin changefield
/collection:http://AdventureWorksServer:8080/DefaultCollection
/n:TFSfieldReferenceName  /type:PlainText | HTML
```

> ⓘ **Note**

> If your field type is something different than HTML or PlainText and field data isn't important or the field isn't used in any project, then we recommend you delete the field.

```cmdline
witadmin deletefield
/collection:http://AdventureWorksServer:8080/DefaultCollection
/n:TFSfieldReferenceName
```

> ⓘ **Important**
>
> Deleting a field results in a loss of field data across the collection.

Resolve import errors

Failures that occur during import fall into one of two categories, verification failure and import failure.

Verification failures

Verification failures mean the import didn't start. The data migration tool attempted to queue an import, but got an error instead. Your import request isn't valid. Fix the error messages and then try to import again.

VS403254

The region that you entered for your Azure DevOps Services import isn't supported.

`VS403254: Region {0} might not be used for the Import, it is not a supported region.`

Open your import specification file and update the region that you've provided with the correct short name for the region.

VS403249

The organization name your team has selected is already in use by an existing organization. All Azure DevOps Services imports go into a new organization that is created at import time.

```
VS403249: The organization {0} already exists. Please select a different name and
try the import again.
```

Select a different organization name and update the import specification file before
retrying the import.

VS403250 & VS403286

The DACPAC isn't built off a detached collection.

```
VS403250: The dacpac is not a detached Azure DevOps Server Collection database.
```

```
VS403286: The dacpac is from a Azure DevOps Server Configuration database. You must
use a detached Azure DevOps Server Collection database.
```

Detach your collection database and generate the DACPAC again.

VS403243

Unable to make a connection to the database using the provided SQL Connection
String.

```
VS403243: Unable to connect to the database using the provided SQL Connection
String {0}.
```

Review the parameters that were provided to ensure they're correct and try again.

VS403260 & VS403351

The collection database isn't detached.

```
VS403260: The database is not detached.
```

```
VS403351: The DACPAC or source database is missing an expected table. It's possible
that the database was not correctly detached from Azure DevOps Server.
```

Detach your collection database and retry the import queue.

VS403261

The connection string must be encrypted otherwise the password is sent in the clear.

```
VS403261: The SQL connection string must use encryption.
```

Add **Encrypt=true** to your SQL connection string.

VS403262

The connection string must use SQL Authentication.

```
VS403262: The SQL connection string must use SQL Authentication, Integrated
Authentication is not supported.
```

Add **Integrated Security=False** to your SQL connection string.

VS403263

Your SQL sign in user account doesn't have the required database role.

```
VS403263: The User ID {0} must be member of the database role {1}.
```

Make sure the user account for sign in is assigned the 'TFSEXECROLE' role.

> ⓘ **Note**
>
> There is a known issue with using `sp_addrolemember` to add `TFSEXECROLE` to an
> existing SQL login. The role membership isn't applied until all open connections
> using that identity are closed. If you receive the VS403263 error and have
> confirmed your identity has the role, we recommend that you create a new identity
> for your import. Details on how to create a new SQL login that's ready to be used
> for import can be found at **Import large collections**.

VS403264

The connection string doesn't point to an Azure DevOps Server collection database.

```
VS403264: The database is not a Azure DevOps Server Collection database, it cannot
be used for import.
```

Verify or correct the connection string points to your collection database.

VS40325

The Azure DevOps Server Update has queued the file migration job. Imports can't be
performed until this job has completed. The completion time for this job is dependent
on the size of the collection.

```
VS403255: The collection cannot be imported due to an ongoing post upgrade job.
Please wait and try again later
```

You can track job progress by running the following query on the collection database:

SQL

```
SELECT  COUNT (*) as remaining_files_to_migrate
FROM    tbl_FileReference
WHERE   PartitionId > 0
        AND MigrateFileId IS NOT NULL
```

Once the number of files remaining to migrate is zero, you can run the data migration tool.

VS403282

A new line character exists in the source location value. This character could have remained after copying the SAS key from your windows console.

VS403282: The source location parameter contains a new line character. Please ensure the SAS key is defined on a single line in the import specification file.

Remove the line break and try again.

VS403271

Your import files and DACPAC aren't located in the **required** Azure region to complete the import to your target Azure DevOps Services region.

VS403271: It appears that your DACPAC was uploaded to East US. It's required that customers targeting Central US for import put their DACPACs in Central US. Please move your DACPAC to Central US and requeue the import.

Create a new Microsoft Azure storage account in the required region and copy your files. The following example shows how to copy your data using **AzCopy**.

cmdline

```
AzCopy.exe /Source:https://accountSCUS.blob.core.windows.net/mycontainer
/SourceKey:"primary access key"
/Dest:https://accountCUS.blob.core.windows.net/mycontainer /DestKey:"primary
access key" /S
```

VS403316

Inconsistencies were detected in some Team Foundation version control (TFVC) files within your collection.

VS403316: An inconsistency was detected in some TFVC files for this collection. The inconsistency needs to be corrected prior to running an import to Azure DevOps

Work with Azure DevOps Services [customer support ⧉](#). Open a support ticket and they work with you to resolve the error.

VS403366

The data migration tool was unable to connect to the SQL Azure VM.

```
VS403366: A problem occurred while attempting to connect to your database.
Please verify that your connection string is correct and that all required
IP addresses for Azure DevOps Services have been provided exceptions for
your machines firewall.

List of Azure DevOps Services IPs:
```

Verify that you've entered the information correctly in your connection string and that you can connect to the VM.

The IPs that the error message lists are for Azure DevOps Services. Azure DevOps Services IPs can change temporarily during deployments. Add them to your firewall exceptions and try queuing the import again. For a list of IP addresses, see [Import large collections, Restrict access to Azure DevOps Services IPs only](#)

VS403373

The data migration tool doesn't support importing multiple copies of the **SAME** collection. However, it **DOES** support importing **split** copies of a collection. Change the GUID for the ***DataImportCollectionID***.

From SQL Server Management Studio (SSMS), open the extended properties for the split copies that you haven't imported yet. Add a newly generated GUID to the "TFS_DATAIMPORT_COLLECTIONID" property. Then rerun the **prepare** command and use the new **import.json** file to queue the import.

VS403379

Data import fails as one or more projects found in this collection are in the soft-deleted stage. Restore the soft-deleted project(s) or delete them permanently before running the data import. For details, see [Delete a project](#).

```
them permanently before running the data import.
```

Verify the collection against which you're running the data migration tool has projects in the soft-deleted stage. Once a project is deleted, it remains in a soft-delete state for 28 days during which the deleted project can be restored. You can read about how to restore a deleted project in Restore a project. If you have projects in the soft-deleted stage, remove them completely or restore them back before running data import.

Import failures

Import failures mean that the import queued, but didn't complete. The individual who queued the import receives a failure email notification. Most of the time this email includes a reason for the failure. If it does, use the troubleshooting steps provided in the email and this page to resolve the errors and retry your import.

If the error is more complex, then the email you receive provides instructions on how to file a customer support case ⤢ . After you submit a customer support case, your team must roll back by bringing your Azure DevOps Server instance back online and reattach your collection. Your team members can then continue working. We recommended that you don't try the import again until the failure causing the issue gets resolved.

Related articles

- Validate and import
- Post-import
- Delete a project
- Restore a project

Migration and process model FAQs

FAQ

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Find answers to frequently asked questions when migrating to Azure DevOps Services from on-premises.

Why is my production import taking longer than the corresponding dry-run import of the same organization?

During a production import, additional provisioning steps are undertaken to cement the organization into the Azure DevOps Services hosted environment. In contrast, the dry-run import of the same organization typically completes faster since dry-run organizations are temporary and thus do not require these additional steps.

Why is my dry-run import taking longer than previous dry-run imports of the same organization?

Any dry-run import after the first is expected to take longer given the extra time required to clean up resources from previous dry-run attempts.

How long after deleting or renaming an existing dry-run organization will the name become available so another import can be queued?

It can take up to one hour for an organization name to become available after deleting or renaming.

Does use of the Hosted XML process model pose a future risk in use of Azure DevOps Services?

No. When it comes to service updates, Hosted XML organizations are treated the same as organizations using the Inheritance process model.

Will my organization be stuck in Hosted XML forever?

No. You are using the Hosted XML process because the Inheritance process model doesn't contain all features yet. However, you can now clone a hosted XML process to an Inheritance process.

Will migrating from Hosted XML into Inheritance process model be a manual process?

No. The migration is automated. Follow the steps to clone a hosted XML process to an Inheritance process.

What happens in Hosted XML when Microsoft makes a change to a system process?

This is the same experience with Azure DevOps Server. If we make a change to a system process, it isn't applied to any of your Hosted XML processes. You won't have to update your processes if you don't want to. But if you do, you'll need to make the changes in the XML definition files manually for each process.

Is there a difference between a project that was created manually versus one

that was created from data import?

The features available to each project are the same. The differences occur in how you modify the processes in your organization. When you create an organization, you'll use the Inheritance process model to customize the work tracking experience. Team projects migrated via data import, however, will use the Hosted XML process model to customize the work tracking experience. As described previously, you can clone a Hosted XML process to an Inheritance process model after import.

If my organization is using Hosted XML, can I create new projects to use the Inheritance process model?

Yes. For data import organizations, Azure DevOps Services supports team projects that use Inheritance as well as Hosted XML process models. To learn more about the Inheritance process, see Manage processes.

Where can I find more information on Hosted XML and the Inheritance process model?

- Inheritance Process Model
- Hosted XML

Related articles

- Migration overview
- Process template validation
- Troubleshooting process errors
- Inheritance Process Model
- Hosted XML

Default permissions quick reference

Article • 10/06/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

To use Azure DevOps features, users must be added to a security group with the appropriate permissions and granted access to the web portal. Limitations to select features are based on the *access level* and *security group* to which a user is assigned. The **Basic** access level and higher supports full access to most Azure DevOps services, except for Azure Test Plans. **Stakeholder** access level provides partial support to Azure Boards and Azure Pipelines. To learn more about access levels, see About access levels and Stakeholder access quick reference.

Assign users to a security group

The most common built-in security groups—**Readers**, **Contributors**, and **Project Administrators—and team administrator role grant permissions to specific features.

In general, use the following guidance when assigning users to a security group:

- Add to the **Contributors** security group full-time workers who contribute to the code base or manage projects.
- Add to the **Project Administrators** security group users tasked with managing project resources. I
- Add to the **Project Collection Administrators** security group users tasked with managing organization or collection resources.

To learn more about administrative tasks see About user, team, project, and organization-level settings. For a complete reference of all built-in groups and permissions, see Permissions and groups. For information about access levels, see About access levels.

In the tables provided in this article, a ✅ (checkmark) indicates that the corresponding access level or security group has access to a feature by default.

To assign or change an access level, see Add users and assign licenses. If you need to grant specific users select permissions, you can do so.

Azure Boards

You can plan and track work from the web portal **Boards** hub, and using Visual Studio, Excel, and other clients. For an overview of work tracking features, see About Agile tools. To change permissions, see Set permissions and access for work tracking. In addition to the permissions set at the project level via the built-in groups, you can set permissions for the following objects: area and iteration paths and individual queries and query folders.

> ⓘ **Note**
>
> Team administrators can configure settings for their team's tools. Organization owners and members of the **Project Administrators** group can configure settings for all teams. To be added as an administrator, see **Add team administrators** or **Change project-level permissions**.

Each user's access level or permission assignment controls access to the following tasks . Members of the Readers, Contributors, or Project Administrators group are assumed to have Basic access or greater.

General work item permissions

You can use work items to track anything you need to track. To learn more, see Understand how work items are used to track issues, tasks, and epics.

Task or permission

Readers

Contributors

Project admins

View work items in this node (Area Path permission)

✔

✔

✔

Edit work items in this node (Area Path permission)

✔

✔

Edit work item comments in this node (Area Path permission)

✔

✔

Create tag definition

✔

✔

Change work item type (Project-level permission)

✔

✔

Move work items out of this project (Project-level permission)

✔

✔

Email work items

✔

✔

✔

Apply a work item template

✔

✔

Delete and restore work items (Project-level permission) (able to restore from the Recycle bin)

✔

✅

Permanently delete work items (Project-level permission)

✅

[Provide feedback](#) (through the Microsoft Feedback client)

✅

✅

[Request feedback](#)

✅

✅

> ⓘ **Note**
>
> Work items are subject to rules applied to them. Conditional rules based on user or group membership are cached for your web browser. If you find yourself restricted to update a work item, you may have encountered one of these rules. If you believe you've encountered an issue that doesn't apply to you, see **Work item form IndexDB caching issues**. For more information, see **Rules and rule evaluation**.

Boards

You use [Boards](#) to implement Kanban methods. Boards present work items as cards and support quick status updates through drag-and-drop.

Task

Readers

Contributors

Team admins
Project admins

View boards and open work items

✔

✔

✔

Add work items to a board; update status through drag-and-drop

✔

✔

Reorder work items or reparent child items through drag-and-drop; update a field on a card

✔

✔

Add child items to a checklist

✔

✔

Assign to a sprint (from card field)

✔

✔

Configure board settings

✔

Backlogs features access

[Backlogs](#) display work items as lists. A product backlog represents your project plan and a repository of all the information you need to track and share with your team. Portfolio backlogs allow you to group and organize your backlog into a hierarchy.

Task

Readers

Contributors

Team admins
Project admins

View backlogs and open work items

✔

✔

✔

Add work items to a backlog

✔

✔

Use bulk edit features

✔

✔

Add child items to a backlog item; prioritize or reorder a backlog; parent items using the Mapping pane; Assign items to a sprint using the Planning pane

✔

✔

Configure team settings, backlog levels, show bugs, work days off

✔

Sprints

You use sprint tools to implement Scrum methods. The Sprints set of tools provide filtered views of work items that a team has assigned to specific iteration paths or sprints.

Task

Readers

Contributors

Team admins Project admins

View sprint backlogs, taskboards, and open work items

✔

✔

✔

Add work items to a sprint backlog or taskboard

✔

✔

Prioritize/reorder a sprint backlog or taskboard; add child items to a backlog item; reassign items to a sprint using the Planning pane

✔

✔

View team capacity and work details

✔

✔

Set team capacity

✔

Use bulk edit features

✔

✔

Define team sprints

✔

Queries

Queries are filtered lists of work items based on criteria that you define by using a query editor. Adhoc searches are powered by a semantic search engine.

Task

Readers

Contributors

Project admins

View and run managed queries, view query charts

✔

✔

✔

Create and save managed **My queries**, query charts

✔

✔

Create, delete, and save **Shared queries**, charts, folders

✔

Delivery plans

Delivery plans display work items as cards against a calendar view. This format can be an effective communication tool with managers, partners, and stakeholders for a team.

Task

Readers

Contributors

Team admins
Project admins

View delivery plans

✔

✔

✔

Create, edit, or delete a delivery plan, Contributors can only edit or delete plans that they create

✔

✔

Manage permissions for a delivery plan, Contributors can only manage permissions for plans that they create

✔

✔

Azure Repos

You can manage your source code from the web portal **Repos** hub, or using Xcode, Eclipse, IntelliJ, Android Studio, Visual Studio, or Visual Studio Code.

Stakeholders for private projects have no access to **Repos**. Stakeholders for public projects have the same access to **Repos** as **Contributors**.

Advanced Security

You can use Advanced Security to identify security vulnerabilities in your repository.

Permission

Readers

Contributors

Build Admins

Project Admins

View alerts (ability to view all security alerts under the Advanced Security tab)

✔

✔

✔

Manage and dismiss alerts (ability to dismiss any Advanced Security alert)

✔

Manage settings (toggle on Advanced Security and/or enable push protection for a repository)

Code: Source control

You can connect to your code from the web portal **Code** hub, or using Xcode, Eclipse, IntelliJ, Android Studio, Visual Studio, or Visual Studio Code. Stakeholders for private projects have no access to **Code**.

Git

You can use Git repositories to host and collaborate on your source code. For an overview of code features and functions.

Permission

Readers

Contributors

Build Admins

Project Admins

Read (clone, fetch, and explore the contents of a repository); also, can create, comment on, vote, and **Contribute to pull requests**

✅

✅

✅

✅

Contribute, **Create branches**, **Create tags**, and **Manage notes**

✅

✅

✅

Create repository, **Delete repository**, and **Rename repository**

✅

Edit policies, **Manage permissions**, **Remove others' locks**

✅

Bypass policies when completing pull requests, **Bypass policies when pushing**, **Force push** (rewrite history, delete branches and tags)
(not set for any security group)

TFVC

[Team Foundation Version Control (TFVC)](#) provides a centralized version control system to manage your source control.

> ⓘ **Note**
>
> Tasks such as create, delete, or rename a TFVC repository are not supported. Once a TFVC repository is created you can't delete it. Also, you can only have one TFVC repository per project. This is different from Git repositories which allow for adding, renaming, and deleting multiple repositories.

Permission

Readers

Contributors

Build Admins

Project Admins

Check in, **Label**, **Lock**, **Merge**, **Pend a change in a server workspace**, **Read**

Read only

✔

✔

✔

Administer labels, **Manage branches**, **Manage permissions**, **Revise other users' changes**, **Undo other users' changes**, **Unlock other users' changes**

✔

Azure Pipelines

You can define and manage your builds and releases from the web portal **Pipelines** hub. For an overview of pipelines features and functions, see Continuous integration on any platform.

Task	Readers	Contributors	Build Admins	Project Admins	Release Admins
View release pipelines	✔	✔	✔	✔	✔
Define builds with continuous integration		✔	✔	✔	
Define releases and manage deployments		✔		✔	✔
Approve releases		✔	✔	✔	✔
Azure Artifacts (5 users free)		✔		✔	✔
Queue builds, edit build quality		✔	✔	✔	

Task	Readers	Contributors	Build Admins	Project Admins	Release Admins
Manage build queues and build qualities			✔	✔	
Manage build retention policies, delete and destroy builds		✔	✔	✔	
Administer build permissions			✔	✔	
Manage release permissions				✔	✔
Create and edit task groups		✔	✔	✔	✔
Manage task group permissions			✔	✔	✔
Can view library items such as variable groups	✔	✔	✔	✔	✔
Use and manage library items such as variable groups			✔	✔	✔

Azure Test Plans

Users granted **Basic + Test Plans** or **Visual Studio Enterprise** access level can define and manage manual tests from the web portal. For an overview of manual test features and functions, see Testing overview. You set several test permissions at the project level from **Project Settings>Permissions**.

Permission

Level

Readers

Contributors

Project Admins

View test runs

Project-level

✔

✓

✓

Create test runs
Delete test runs

Project-level

✓

✓

Manage test configurations
Manage test environments

Project-level

✓

✓

Create tag definition
Delete and restore work items

Project-level

✓

✓

Permanently delete work items

Project-level

✓

View work items in this node

Area Path

✓

✓

✔

Edit work items in this node
Manage test plans
Manage test suites

Area Path

✔

✔

> ⓘ **Note**
>
> The **Change work item type** permission doesn't apply to test-specific work items. Even if you choose this feature from the work item form, changing the work item type is disallowed.

Azure Artifacts

You can manage feeds from the web portal, **Artifacts**. Users with Stakeholder or Basic access, or higher can access Azure Artifacts features. To set permissions, see Secure feeds using permissions.

Feeds have four permission roles: Readers, Collaborators, Contributors, and Owners. Owners can add user accounts or security groups to any role.

Permission	Reader	Collaborator	Contributor	Owner
List, install, and restore packages	✔	✔	✔	✔
Push packages			✔	✔
Unlist/deprecate packages			✔	✔
Delete/unpublish package				✔
Promote a package to a view			✔	✔
Add/remove upstream sources				✔
Save packages from upstream sources		✔	✔	✔
Edit feed permissions				✔

By default, the Project Collection Build Service is a Contributor and your project team is a Reader.

Notifications, alerts, and team collaboration tools

To manage notifications, see Manage personal notifications and Manage team notifications.

Task

Readers

Contributors

Team admins

Project admins Project Collection admins

View the project page, navigate using the project page

✔

✔

✔

✔

Edit the project page

✔

Set personal notifications or alerts

✔

✔

✔

Set team notifications or alerts

✔

✔

Set project-level notifications or alerts

✔

View Project READMEs

✔

✔

✔

✔

View Project wikis or code wikis

✔

✔

✔

✔

Provision or create a project wiki

✔

✔

✔

Publish code as a wiki

✔

✔

✔

Request feedback

✔

✔

✔

Provide feedback

✔

✔

✔

✔

Search across projects, organizations, collections

✔

✔

✔

✔

Dashboards, charts, reports, and widgets

You can define and manage team and project dashboards from the web portal, **Dashboards**. For an overview of dashboard and chart features, see Dashboards. You can set individual dashboard permissions to grant or restrict the ability to edit or delete dashboards.

Users granted Stakeholder access to private projects can't view or create query charts. Stakeholder access to public projects can view and create query charts.

Task

Readers

Contributors

Team admins

Project admins

View team and project dashboards

✔

✔

✔

✔

Add and configure project dashboards

✔

✔

Power BI Integration and Analytics views

From the web portal **Analytics views**, you can create and manage Analytics views. An Analytics view provides a simplified way to specify the filter criteria for a Power BI report based on the Analytics Service data store. The Analytics Service is the reporting platform for Azure DevOps. To learn more, see What is the Analytics Service?.

You set permissions for the service at the project level, and for shared Analytics views at the object level. Users with **Stakeholder** access have no access to view or edit Analytics views.

Task

Readers

Contributors

Project admins

View Analytics

✔

✔

✔

View a shared Analytics view

✔

✔

Add a private or shared Analytics view

✔

✔

Edit and delete shared Analytics views

✔

Related articles

- [Add users to a project or team](#)
- [Security and permission management tools](#)
- [Permissions and groups reference](#)
- [About access levels](#)
- [Web portal navigation](#)
- [Troubleshoot permissions](#)
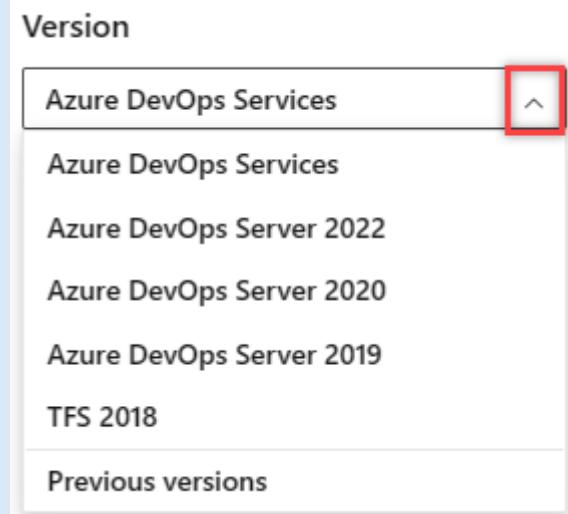
About access levels

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019 | TFS 2018

Access levels control which web portal features are available or not. Access levels supplement security groups, which allow or deny certain tasks. Administrators provide their user-base access to the features they need and only pay for those features. For more information, see Stakeholder access quick reference and Manage users and access.

> ⓘ **Important**
>
> Version
>
> | Azure DevOps Services   ∧ |
> |--|
> | Azure DevOps Services |
> | Azure DevOps Server 2022 |
> | Azure DevOps Server 2020 |
> | Azure DevOps Server 2019 |
> | TFS 2018 |
> | Previous versions |
>
> Select the version of this article that corresponds to your platform and version. The version selector is above the table of contents. **Look up your Azure DevOps platform and version**.

Adding a user or group to a team or project gives them automatic access to the features of the default access level and security group. Most users can access most features by being assigned to the **Basic** access level and **Contributors** security group. For a simplified overview of the permissions assigned to the most common groups **Readers**, **Contributors**, and **Project Administrators**, see Default permissions.

To add user accounts or groups to specific access levels, see Manage users and access. Make sure to set each user's access level based on what you've purchased for that user.

Supported access levels

Assign users or groups of users to one of the following access levels:

- **Basic**: Provides access to most features. Assign to users with a Visual Studio Professional subscription, an Azure DevOps Server CAL, and to users for whom you're paying for Basic access in an organization.
- **Basic + Test Plans**: Provides access to all features included in **Basic** and Azure Test Plans. Assign to users with a Visual Studio Test Professional or MSDN Platforms subscription, and to users for whom you're paying for Basic + Test Plans access in an organization.
- **Stakeholder**: Can be assigned to unlimited users for free. Provides partial access to private projects and mostly full access to public projects. Assign to users with no license or subscriptions who need access to a limited set of features.
- **Visual Studio Subscriber**: Assign to users who already have a Visual Studio subscription. The system automatically recognizes the user's subscription—Visual Studio Enterprise, Visual Studio Professional, Visual Studio Test Professional, or MSDN Platform—and enables any other features that are included in their subscription level. If you assign **Basic** or **Stakeholder**, they also receive their Visual Studio subscription benefits upon sign-in.

> 💡 **Tip**
>
> As a best practice when adding new users, we recommend assigning the **Visual Studio Subscriber** level when appropriate (as opposed to Basic) to prevent being charged the **Basic** rate before the user signs in for the first time.

The following table indicates those features available for each supported access level. Visual Studio Test Professional and MSDN Platform subscriptions grant access to the same features as Visual Studio Enterprise.

Feature

Stakeholder

Basic &
Visual Studio Professional

Basic + Test Plans &
Visual Studio Enterprise

[Administer organization](#)
Can configure resources when also added to a security group or role: team administrator, Project Administrator, or Project Collection Administrator.

✔

✔

✔

Advanced backlog and sprint planning tools

Includes full access to all backlog and sprint planning tools.

✔

✔

Advanced home page

Includes access to projects, work items, and pull requests defined across projects you work in.

✔

✔

Advanced portfolio management

Includes full access to define features and epics from a portfolio backlog or Kanban board.

✔

✔

Agile boards

Stakeholders have limited access to Kanban boards and Taskboards. Stakeholders use drag-and-drop to create and change work items, but only change the State field on cards. They also can't see or adjust capacity.

✔

✔

✔

Agile Portfolio Management

Includes limited access to portfolio backlogs and Kanban boards. Stakeholders can't

change the backlog priority order, can't assign items to an iteration, use the mapping pane, or exercise forecasting.

✔

✔

✔

Artifacts

Includes full access to all Azure Artifacts features, up to 2-GiB free storage.

✔

✔

✔

Author Release Pipelines and Manage Releases

Includes defining release pipelines, multi-stage continuous deployment (CD) pipelines, and using approvals and gates to control deployments; when the **Free access to Pipelines Preview** feature is enabled, Stakeholders gain access to all Azure Pipelines features.

✔

✔

Basic backlog and sprint planning tools

Includes limited access to add and modify items on backlogs and sprint backlogs and Taskboards. Stakeholders can't assign items to an iteration, use the mapping pane, or forecasting.

✔

✔

Build

Includes full access to all features to manage continuous integration and continuous delivery of software.

✔

✔

Chart Authoring

Can create work tracking [query charts](#).

✔

✔

Chart Viewing

Can only view work tracking query charts. Stakeholders can't view query charts from the Queries page, however can view them when added to a dashboard.

✔

✔

Code

Includes full access to all features to manage code using [Git repositories](#) or using [Team Foundation Version Control (TFVC)](#) Team Foundation Version Control (TFVC).

✔

✔

Delivery Plans

Includes full access to add and view Delivery plans.

✔

✔

Request and Manage Feedback Includes full access to request and manage feedback on working software.

✔

✔

Standard Features

Includes [working across projects](#), [View dashboards](#), [View wikis](#), and [Manage personal notifications](#). Stakeholders can't view Markdown README files defined for repositories and can only read wiki pages.

✔

✔

✔

Test services in build and release

Includes [running unit tests with your builds](#), [reviewing](#), and [analyzing](#) test results.

✔

✔

Test Case Management

Includes [adding test plans and test suites](#), [creating manual test cases](#), [deleting test artifacts](#), and [testing different configurations](#).

✔

Test Execution and Test Analysis

Includes running [manual](#), [tracking test status](#), and [automated tests](#).

✔

✔

Test summary access to Stakeholder license

Includes [requesting Stakeholder feedback using the Test & Feedback extension](#).

✔

✔

✔

View My Work Items

Access to [add and modify work items](#), [follow work items](#), [view and create queries](#), and [submit, view, and change feedback responses](#). Stakeholders can only assign existing tags to work items (can't add new tags) and can only save queries under My Queries (can't save under Shared Queries).

✔

✔

✔

View Releases and Manage Approvals

Includes viewing releases and approving releases; when the **Free access to Pipelines Preview** feature is enabled feature is enabled, Stakeholders gain access to all Azure Pipelines features.

✔

✔

✔

Visual Studio subscription access

Visual Studio subscribers are entitled to **Visual Studio subscription** features as a subscriber benefit. When you add those users, be sure to assign them the **Visual Studio subscription** access level.

The system automatically recognizes their subscription and enables any other features that are included, based on their subscription level.

Programmatic mapping of access levels

You can manage access levels programmatically using the az devops user add (Azure DevOps Services only) or the User Entitlement - Add REST API. The following table provides a mapping of the access level selected through the user interface and the `AccountLicenseType`, `licensingSource`, and `msdnLicenseType` parameters.

Access level (user interface) licenseDisplayName	accountLicenseType	licensingSource	msdnLicenseType
Basic	express	account	none
Basic + Test Plans	advanced	account	none
Visual Studio Subscriber	none	msdn	eligible
Stakeholder	stakeholder	account	none
Visual Studio Enterprise subscription	none	msdn	enterprise

> ⓘ **Note**

The `earlyAdopter` accountLicenseType is an internal value used solely by Microsoft.

Related articles

- Stakeholder access quick reference
- Free access to Pipelines Preview
- Get started as a Stakeholder
- Export a list of users and their access levels
- Default permissions and access

Azure DevOps Services status

Article • 12/20/2023

Azure DevOps Services

Our team of engineers around the world work 24/7 to ensure that our customers are always productive and successful with our service. We respond quickly during performance slowdowns and stability issues. Our top priority is to communicate the incident status and our next steps to mitigate the issue. Check the status of our services through the Azure DevOps Services status portal ⧉ .

Our Customer Impact Assessment (CIA) is modeled after our availability model, which measures real customer experiences representing both reliability and performance. Many of the events we post are based on the CIA.

Services health matrix

Azure DevOps ⧉ is a product suite of service offerings. The geography ⧉ indicates where an organization is hosted in the cloud. The data residency, sovereignty, compliance, and resilience requirements are honored within the geographical boundaries. To help clarify which specific aspects of the service are affected, we communicate impact of each of these services by geography in the service matrix.

The status portal ⧉ provides a two-dimensional matrix view of active events mapped to a given service and geography. In addition to the suite of **Azure DevOps Services**, it displays the following items:

- **Core services**: Encompass the set of features that are fundamental to all five services, such as authentication or the web portal
- **Other services**: Correspond to features that complement the suite, such as extensions

Service health indicators

The Azure DevOps Services status portal displays indicators that reflect the severity of a service health event, based on the number of customers affected by the issue. The highest severity events affect a large percentage of our customers and render some parts of the product unusable.

The Azure DevOps Services status portal displays four indicators that reflect the severity of a service health event: Healthy, Degraded, Unhealthy, and Advisory. The highest severity events affect a large percentage of our customers and render some parts of the product unusable.

Service status and event logs

Access detailed information on active and past events from the Status history page ⬈. Each event log contains associated information such as the impacted service, geography, and event duration. You can filter the logs to adjust the scope of your search into past events. Additionally, you can use the REST API to build automated alerting solutions to stay on top of events.

When and how to report availability issues

If you see an issue reported on the Azure DevOps Services health page, we're already working to restore normal operations. If your issue isn't reported, you can ask a question through the Azure DevOps Services virtual support agent ⬈. For issues not related to availability, refer to our Developer Community portal ⬈.

RSS feed

Subscribe to the RSS feed ⬈ to receive updates in your feed reader.

Use REST APIs to build automated solutions

The Azure Resource health REST API can retrieve the current health status of each of the Azure DevOps Services. You can use it to build an automated solution to monitor the infrastructure incidents.

> ⓘ **Note**
>
> Looking for Azure DevOps REST APIs? See the latest **Azure DevOps REST API reference**.
>
> For information about .NET client libraries, see **.NET client libraries for Azure DevOps**.

Related articles

- Azure Service Health overview
- Blog post: How do you measure quality of a service?

Data protection overview

Article • 11/27/2023

Azure DevOps Services

Azure DevOps Services is a cloud-hosted application for your development projects, from planning through deployment. Based on the on-premises capabilities, with additional cloud services, we manage your source code, work items, builds, and tests. Azure DevOps uses platform as a service (PaaS) infrastructure and many Azure services, including Azure SQL, to deliver a reliable, globally available service for your development projects.

This article discusses the steps that Microsoft takes to help keep your projects safe, available, secure, and private. It describes the role that you play in keeping your projects safe and secure.

This article is for organization administrators and IT professionals who manage their project assets daily. It's most useful to individuals who are already familiar with Azure DevOps and want to know more about how Microsoft protects stored assets in Azure DevOps.

Our commitment

Microsoft helps to ensure that your projects remain safe and secure, without exception. When you store your projects in Azure DevOps, they benefit from multiple layers of security and governance technologies, operational practices, and compliance policies. We enforce data privacy and integrity both at rest and in transit.
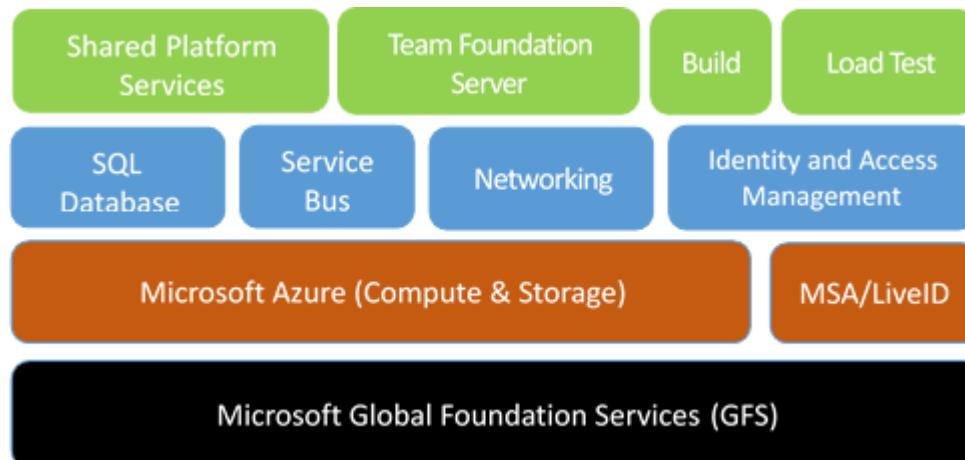
The threats that you face boil down to four basic categories: data availability, service availability, service security, and data privacy. This article explores specific threats within each category and explains what Azure DevOps does to address them. The article begins by describing how data is stored and how Azure DevOps manages access to your data.

Data protection requires the active engagement of administrators and users who know what steps to take to protect your assets from unauthorized disclosure and tampering. Be explicit when you grant permissions to user access points, so only the right people access data within Azure DevOps.

You should consider all data to be potentially at risk, no matter where it is or how it's being used. This statement is true for both data stored in the cloud and data stored in a private datacenter. It's important to classify your data, its sensitivity and risk, and the

damage that it might do if it becomes compromised. Also, categorize your data relative to an overall policy for managing information security.

Built on Azure

We host Azure DevOps entirely in Azure datacenters. Azure DevOps uses many core Azure services, including compute, storage, networking, Azure SQL, identity and access management, and Azure Service Bus.

Azure DevOps uses Azure Storage as the primary repository for service metadata and customer data. Depending on the type of data and the storage and retrieval requirements, Azure DevOps uses Azure Blob Storage and Azure SQL Database storage.

To help you understand the Azure DevOps Services approach to data protection, here's some background on the storage services:

- Azure Blob Storage stores large chunks of unstructured data. All projects use this service. Data includes potentially sensitive or private information, like the contents of source files and attachments for work items. For most projects, the majority of storage in use is this type of unstructured blob storage.

- Azure SQL Database stores the structured and transactional aspects of your organization, including project metadata, the versioned source-control history, and details of work items. Database storage gives you fast access to the important elements of your project. It provides indexes into the blob storage to look up files and attachments.

Administrators can manage access to resources by granting or restricting permissions on user identities or groups. Azure DevOps uses federated authentication of user identities via Microsoft Entra ID and Microsoft accounts.

During authentication, the user is routed to the authentication provider, where they provide their credentials. After the authentication provider verifies the user's credentials, Azure DevOps issues an authentication cookie to the user. This cookie allows the user to remain authenticated against Azure DevOps.

In this way, the user's credential information is never shared directly with Azure DevOps. For each Azure DevOps resource that the user tries to access, validation of permissions is based on the user's explicit permissions and on permissions that the user inherited through group membership.

Administrators can use access controls to help protect access to the organization, project collections, team projects, and team-scoped data and functionality. Administrators can also use access controls for specific assets like folders for version control and area paths for work items.

Data availability

Azure DevOps uses many Azure Storage features to help ensure data availability if there's a hardware failure, service disruption, or regional disaster. Also, the Azure DevOps team follows procedures to help protect data from accidental or malicious deletion.

Data redundancy

To help protect data during hardware or service failures, Azure Storage geo-replicates customer data between two regions in the same geographical location. For example, Azure Storage can geo-replicate data between North and West Europe or between North and South United States.

For Azure Blob Storage, customer data is replicated three times within a single region. Customer data is replicated asynchronously to a second region in the same geographical location. As such, Azure always maintains the equivalent of six copies of your data.

Having multiple copies enables you to fail over to a separate region if there's a major outage or disaster, while also having local redundancy for hardware failures within a region. For Azure SQL Database storage, daily backups are maintained offsite if there's a regional disaster.

Regarding data redundancy and failover:

- There's an inherent delta, measured in minutes, when Microsoft replicates your data between the primary and secondary region.
- Failover to the secondary region is a decision that Microsoft must make centrally, because it affects all customers on a particular scale unit. Except in extreme circumstances, Microsoft opts to avoid failing over so that customer data isn't lost.
- Azure DevOps offers a service-level (SLA) guarantee of 99.9 percent uptime. Azure DevOps refunds a portion of the monthly charges if it misses that SLA in a specific month.
- Because there's only one region in Brazil, customer data in Brazil is replicated to the South Central US region for disaster recovery purposes.

Mistakes happen

To help protect against accidental deletion of data, Microsoft also takes point-in-time backups of both the blobs in Azure Blob Storage and the databases in Azure SQL Database. There's a separate copy of all blobs, and changes are appended to each storage account. Because this data is immutable, you don't need to rewrite any existing storage as part of the backup procedures.

Backups are a standard part of Azure SQL Database, and Azure DevOps makes use of this capability. We maintain your data for 28 days. These backups are also replicated in a paired region, to help with recovery from a regional outage.

Customers can recover their deleted organizations or projects for up to 28 days after deletion. After 28 days, these organizations and projects are permanently deleted and can't be restored.

> ⓘ **Important**
>
> *Accidental deletion* here refers to scenarios that arise as a result of an incident on our services. It doesn't include customers' accidental deletion of assets (for example, repositories, work items, attachments, or artifacts).
>
> We don't support restoring assets that customers accidently delete. These backups are meant only for business continuity and to aid recovery from outage or disaster scenarios.

Practice is critical

Having multiple backups of your data is good, but without practice, restoring can be unpredictable. It's been said that "backups never fail; the restores do." Though the

statement is technically incorrect, the sentiment is right.

Microsoft regularly practices restoring datasets from backup. We regularly test geo-redundant storage from Azure. There are many combinations of disaster and data corruption scenarios. We continue to plan and run new tests for these scenarios regularly.

Service availability

Azure DevOps offers distributed denial-of-service (DDoS) protections and live site response to help ensure that you have access to your organization and associated assets.

DDoS protections

In some cases, a malicious DDoS attack can affect service availability. Azure has a DDoS defense system that helps prevent attacks against our service. It uses standard detection and mitigation techniques such as SYN cookies, rate limiting, and connection limits. The system is designed to withstand attacks not only from the outside but also from within Azure.

For application-specific attacks that can penetrate the Azure defense systems, Azure DevOps establishes application-level and organization-level quotas and throttling. This practice helps prevent any overuse of key service resources during an attack or accidental misuse of resources.

Live site response

In rare circumstances, you might require a live site response to a problem with service availability. We have an operations team that's constantly available to rapidly identify the problem and to engage the necessary development team resources.

The development team resources then address the problem. They also aim to update the service status page within minutes of detecting a problem that affects the service. After development team resources address a problem, they identify the root cause and track the necessary changes to prevent similar problems in the future.

Azure DevOps processes for live site management focus on your experience and the health of the service. These processes minimize the time to detect, respond to, and mitigate problems. All engineering disciplines are involved and responsible, so continual

improvements evolve out of direct experience. Monitoring, diagnostics, resiliency, and quality assurance processes then improve over time.

Live site management in Azure DevOps has three distinct tracks: telemetry, incident management, and live site review. Here's what these tracks entail:

Telemetry	Incident management	Live-site review
• **Alerts** – define health alerts for failure modes • **Diagnostics** – deliver instrumentation data and operational reports • **Troubleshooting guides** – guidance for investigating an alert is defined by the feature, and then refined by the Service Engineer • **Failure mode testing** – the Service Delivery (SD) team performs failure testing to ensure alerts fire as expected • **Onboarding** – the feature team works with their Service Engineer (SE) to onboard new alerts to the 24 x 7 team	• **Detection** – product alerts detect health issues and start the Live Site Incident (LSI) process • **Triage** – The 24 x 7 team receives all critical alerts and confirms impact using Azure DevOps guidance • **Escalation** – both Dev and Ops have individuals in an on-call rotation. SE is the initial escalation path. The SE calls Dev, as needed • **Incident management** – a bridge is managed by the SE who engages Dev. and Partners to troubleshoot • **Resolution** - communication and service restoration are actively driven until customer impact is eliminated	• **Goal** – weekly review of LSI ensure that leadership has visibility into live site health and repeat issues • **Cadence** – Incidents from prior week have root cause documented, then reviewed on weekly basis • **Audience** – VS Leadership. Partner team when they drive impact. Developer attends to provide details on Service incident • **Ownership** – Dev. Owns reviews for App and Deploy issues. SD owns for Platform issues. • **Driving improvements** – Bugs and problem work items are logged for gaps (e.g. – missing alerts) and repeat root cause

The operations team also monitors the availability metrics for individual organizations. These metrics provide insights into specific conditions that might affect only some of our customers. Investigations into this data can often result in targeted improvements to address customer-specific issues. In some cases, Microsoft might even contact you directly to understand your experience and work with you to improve the service.

Microsoft publishes an SLA and provides a financial guarantee to ensure that we meet this agreement each month. For more information, see SLA for Azure DevOps ⟂ .

Sometimes, partner teams or dependencies have incidents that affect Azure DevOps. All partner teams follow similar approaches to identifying, resolving, and learning from these service outages.

Service security

Service security requires constant vigilance, from proper design and coding techniques to operational factors. Microsoft actively invests in the prevention of security holes and in breach detection. If there's a breach, Microsoft uses security response plans to minimize data leakage, loss, or corruption. For more information, see About security, authentication, and authorization.

Security by design

Azure DevOps is designed to be secure. Azure DevOps uses the Microsoft Security Development Lifecycle at the core of its development process. The Microsoft Operational Security Assurance program guides cloud operation procedures in Azure DevOps.

The Azure DevOps team has annual training requirements for all engineers and operations personnel. The team also sponsors informal meetings hosted by Microsoft engineers. After the team solves a problem that surfaces in a meeting, it shares the lessons learned with other teams.

The following methodologies specify the training requirements:

- Threat modeling during service design
- Following best practices for design and code
- Verifying security with standard tooling and testing
- Limiting access to operational and customer data
- Gating rollout of new features through a rigid approval process

A cloud service is only as secure as the host platform. Azure DevOps uses PaaS for much of its infrastructure. PaaS automatically provides regular updates for known security vulnerabilities.

Virtual machines hosted in Azure use infrastructure as a service (IaaS), such as for a hosted build service. Such images receive regular updates to include the latest security patches available from Windows Update. The same update rigor applies for on-premises machines, including those used for deployment, monitoring, and reporting.

The Azure DevOps team conducts regular, security-focused penetration testing of Azure DevOps. Penetration testing tries to exploit the live production services and infrastructure of Azure DevOps by using the same techniques and mechanisms that malicious attackers use. The goal is to identify real-world vulnerabilities, configurations, errors, or other security gaps in a controlled process.

The team reviews the results of these tests to identify other areas of improvement and to increase the quality of the preventative systems and training. You can review the

results of recent Azure DevOps penetration tests on the Microsoft Service Trust Portal ⧉.

Credential security

We use industry best practices to store your credentials in Azure DevOps. Learn more about credential storage.

Reporting security flaws

If you believe that your penetration testing has revealed a potential security flaw related to the Azure DevOps service, report it to Microsoft within 24 hours. For more information, see the Microsoft webpage for reporting a computer security vulnerability ⧉.

> ⓘ **Important**
>
> Although you don't need to notify Microsoft about penetration testing activities, you must comply with the **Microsoft Penetration Testing Rules of Engagement** ⧉.

Bounty program

Azure DevOps participates in the Microsoft Bug Bounty program. This program rewards security researchers who report problems to us, and it encourages more people to help keep Azure DevOps secure. For more information, see Microsoft Azure DevOps Bounty Program ⧉.

Restricting access

Microsoft maintains strict control over who gets access to our production environment and customer data. We grant access at the level of least privilege that's required, and only after verification of a user's justifications. If a team member needs access to resolve an urgent problem or deploy a configuration change, they must apply for just-in-time access to the production service. Access is revoked as soon as the situation is resolved.

We track and monitor access requests and approvals in a separate system. All access to the system correlates against these approvals. If we detect unapproved access, we alert the operations team to investigate.

We use two-factor authentication for all remote system access. If the username and password for one of our developers or operations staff are stolen, the data remains

protected. Additional authentication checks via smart card or a phone call to a preapproved number must occur before we permit any remote access to the service.

To manage and maintain the service, Microsoft uses secrets such as RDP passwords, SSL certificates, and encryption keys. These secrets are all managed, stored, and transmitted securely through the Azure portal. Any access to these secrets requires specific permission, which is logged and recorded securely. All secrets are rotated on a regular cadence, and we can rotate them on demand if there's a security event.

The Azure DevOps operations team uses hardened administrator workstations to manage the service. These machines run a minimal number of applications and operate in a logically segmented environment.

Operations team members must provide specific credentials with two-factor authentication to access the workstations. All access is monitored and securely logged. To isolate the service from outside tampering, we don't permit applications such as Outlook and Office, because they're often targets of spear phishing and other types of attacks.

Intrusion protection and response

We encrypt data via HTTPS and SSL to help ensure that it isn't intercepted or modified while in transit between you and Azure DevOps. Data that we store on your behalf in Azure DevOps is encrypted as follows:

- Data stored in Azure SQL databases is encrypted via transparent data encryption. This feature helps protect against malicious activity by doing real-time encryption of the database, associated backups, and transaction log files at rest.

- Azure Blob Storage connections are encrypted to help protect your data in transit. For data at rest that's stored in Azure Blob Storage, Azure DevOps uses service-side encryption.

> ⓘ **Note**
>
> Azure DevOps is not Federal Information Processing Standards (FIPS) 140-2 or 140-3 compliant.

The Azure DevOps team uses the Azure infrastructure to log and monitor key aspects of the service. Logging and monitoring help ensure that activities within the service are legitimate, and they help detect breaches or attempted breaches.

All deployment and administrator activities are securely logged, as is operator access to production storage. The log information is automatically analyzed, and any potentially malicious or unauthorized behavior raises real-time alerts.

When the Azure DevOps team identifies a possible intrusion or high-priority security vulnerability, it has a clear response plan. This plan outlines responsible parties, required steps for securing customer data, and instructions on how to engage with security experts at Microsoft. The team also notifies any organization owners if data was disclosed or corrupted, so that they can take appropriate steps to remedy the situation.

To help combat emerging threats, Azure DevOps employs an *assume breach* strategy. A highly specialized team of security experts within Microsoft assumes the role of sophisticated adversaries. This team tests breach detection and response, to accurately measure readiness and the impacts of real-world attacks. This strategy strengthens threat detection, response, and defense of the service. It also allows the team to validate and improve the effectiveness of the entire security program.

Ransomware attack protection

Azure DevOps uses Azure controls to help prevent, detect, and respond to a ransomware attack. For more information about how Azure helps protect customers from ransomware attacks, see Ransomware protection in Azure.

Data privacy

You should have confidence that we're handling your data appropriately and for legitimate uses. Part of that assurance involves carefully restricting usage.

General Data Protection Regulation

The General Data Protection Regulation (GDPR) is the biggest change in data protection laws in Europe since the 1995 introduction of the European Union (EU) Data Protection Directive 95/46/EC. For more information about GDPR, see the overview page in the Microsoft Trust Center ⤢ .

Data residency and sovereignty

Azure DevOps is available in the following eight geographical locations across the world: United States, Canada, Europe, United Kingdom, India, Australia, Asia Pacific, and Brazil. By default, your organization is assigned to your closest location. However, you can choose a different location when you create your organization. If you change your mind

later, you can migrate the organization to a different location with the assistance of Microsoft support.

Azure DevOps doesn't move or replicate customer data outside the chosen location. Instead, your data is geo-replicated to a second region within the same location. The only exception is Brazil, which replicates data to the South Central US region for disaster recovery purposes.

> ⓘ **Note**
>
> For builds and releases that run on Microsoft-provided macOS agents, your data is transferred to a GitHub datacenter in the United States.

To learn more, see Data locations for Azure DevOps.

Law enforcement access

In some cases, third parties such as law enforcement entities might approach Microsoft to obtain access to customer data stored in Azure DevOps. We try to redirect the requests to the organization owner for resolution. When a court order compels Microsoft to disclose customer data to a third party, Microsoft makes a reasonable effort to notify the organization owner in advance, unless we're legally prohibited from doing so.

Some customers require their data storage in a particular geographical location to ensure a specific legal jurisdiction for any law enforcement activities. All customer data, such as source code, work items, test results, and geo-redundant mirrors and offsite backups, is maintained within one of the previously mentioned locations.

Microsoft access

From time to time, Microsoft employees need to obtain access to customer data stored in Azure DevOps. As a precaution, all employees who have (or might ever have) access to customer data must pass a background check that includes previous employment and criminal convictions. We permit access to the production systems only when there's a live site incident or other approved maintenance activity, which is logged and monitored.

Because not all data within our system is treated the same way, we classify data into these types:

- **Customer data**: What you upload to Azure DevOps.

- **Organization data**: Information that you submit when you sign up for or administer your organization.
- **Microsoft data**: Information that's required for or collected through the operation of the service.

Based on the classification, we control usage scenarios, geo-location requirements, access restrictions, and retention requirements.

Microsoft promotional use

Microsoft occasionally wants to contact customers to let them know about additional features and services that might be useful. Because not all customers want to be contacted about these offers, you can opt in and opt out of marketing email communications.

Microsoft never uses customer data to target specific offers for specific users or organizations. Instead, we use organization data and aggregate usage statistics at the organization level to determine groups that should receive specific offers.

Building confidence

You can be confident in other efforts that Microsoft makes on behalf of Azure DevOps. These efforts include internal adoption policies at Microsoft, the level of transparency into the state of our service, and progress toward receiving certification of our systems for managing information security.

Internal adoption

Teams across Microsoft are adopting Azure DevOps internally. The Azure DevOps team moved into an organization in 2014 and uses it extensively. We established guidelines to enable the adoption plans for other teams.

Large teams move more gradually than smaller ones, because of their investments in existing DevOps systems. For teams that move quickly, we established a project classification approach. It assesses risk tolerance, based on project characteristics, to determine if the project is appropriate for Azure DevOps. For larger teams, the adoption typically occurs in phases, with more planning.

Additional requirements for internal projects include associating the organization with Microsoft Entra ID to ensure the proper user-identity lifecycle and password complexity. Projects that are more sensitive also require two-factor authentication.

Compliance certifications

You might be interested in understanding third-party evaluation of our procedures for data security. Azure DevOps has achieved the following certifications:

- ISO 27001:2013
- ISO 27018:2019
- ISO 26262:2023
- Health Insurance Portability and Accountability Act (HIPAA)
- Business Associate Agreement (BAA)
- EU Model Clauses
- System and Organization Controls (SOC) 1 Type 2
- SOC 2 Type 2
- Germany C5

The SOC audit for Azure DevOps covers controls for data security, availability, processing integrity, and confidentiality. The SOC reports for Azure DevOps are available through the Microsoft Service Trust Portal ⧉ .

The Consensus Assessment Initiative Questionnaire (CAIQ) helps organizations assess and evaluate the security practices and capabilities of cloud service providers. In alignment with our commitment to security and transparency, we recently completed the CAIQ assessment for Azure DevOps. We invite you to review the full report on the Microsoft Service Trust Portal ⧉ .

Steps you can take

Proper data protection requires active engagement from you, your administrators, and your users. Your project data stored in Azure DevOps is only as secure as the user access points. Match the level of permission strictness and granularity for those organizations with your project's sensitivity level.

Classify your data

The first step is to classify your data. Classify data based on sensitivity and risk horizon, along with the damage that might occur if it's compromised. Many enterprises have existing classification methods that they can reuse when projects move to Azure DevOps. For more information, you can download Data classification for cloud readiness ⧉ from Microsoft Trustworthy Computing.

Adopt Microsoft Entra ID

Use Microsoft Entra ID to manage your organization's access to Azure DevOps. Microsoft Entra ID provides another way to improve the security of your users' credentials.

Microsoft Entra ID allows your IT department to manage its user access policy, password complexity, password refreshes, and expiration when users leave your organization. Through Active Directory federation, you can directly link Microsoft Entra ID to your organization's central directory, so you have only one location to manage these details for your enterprise.

The following table compares Microsoft account and Microsoft Entra characteristics relative to Azure DevOps access:

⌷ Expand table

Property	Microsoft account	Microsoft Entra ID
Identity creator	User	Organization
Single username and password for all work assets	No	Yes
Password lifetime and complexity control	User	Organization
Azure DevOps membership limits	Any Microsoft account	Organization's directory
Traceable identity	No	Yes
Organization and IP ownership	Unclear	Organization
Two-factor authentication enrollment	User	Organization
Device-based conditional access	No	Organization

Learn more about configuring this support for your organization.

Require two-factor authentication

You might want to restrict access to your organization by requiring more than one factor to sign in. You can require multiple factors by using Microsoft Entra ID. For example, you can require phone authentication, in addition to a username and password, for all authentication requests.

Use BitLocker

For sensitive projects, you can use BitLocker on your Windows laptop or desktop computer. BitLocker encrypts the entire drive on which Windows and your data reside. When BitLocker is enabled, it automatically encrypts any file you save on that drive. If your computer falls into the wrong hands, BitLocker prevents unauthorized access of local copies of data from your projects.

Limit use of alternate authentication credentials

The default authentication mechanism for Git-related tooling is alternate authentication (sometimes called *basic authentication*). This mechanism allows a user to set up an alternate username and password for use during Git command-line operations. The user can use this username/password combination to access any other data for which that user has permissions. By its nature, alternate authentication credentials are less secure than the default federated authentication.

You can still make choices for increased security. All communication is sent over HTTPS, and there are password complexity requirements. Your organization should continue to evaluate whether it needs additional policies to meet your projects' security requirements.

You can disable alternate authentication credentials if you decide that it doesn't meet your organization's security requirements. For more information, see Change application connection & security policies for your organization.

Secure access to your organization

Administrators can use Microsoft Entra ID to control access to Azure resources and applications, such as Azure DevOps. With conditional access control in place, Microsoft Entra ID checks for the specific conditions that you set for a user to access an application. After the user meets access requirements, the user is authenticated and can access the application.

Azure DevOps supports enforcing certain types of conditional access policies (for example, IP fencing) for custom Azure DevOps authentication mechanisms. These mechanisms include personal access tokens, alternate authentication, OAuth, and Secure Shell (SSH) keys. If your users access Azure DevOps through a third-party client, only IPv4-based policies are honored.

Additional resources

- Azure DevOps home page ⬈

- Data locations for Azure DevOps
- Microsoft privacy statement ⧉
- Azure DevOps support ⧉
- Features and services included with Azure DevOps
- Azure Trust Center ⧉
- Microsoft Security Development Lifecycle ⧉

Data locations for Azure DevOps

Article • 12/15/2023

Azure DevOps Services

You can choose the location for your data during initial sign-up and creation of your organization.

Data locations

Azure DevOps data is available in the following geographical locations:

- Australia
- Brazil
- Canada
- Asia Pacific
- Europe (EU)
- India
- United Kingdom (UK)
- United States (US)

By default, your organization uses the closest location. However, you can choose a different location when you create your organization. If you change your mind later, you can migrate your organization to a different location.

Customer data

Except as noted later in this article, Azure DevOps maintains all customer data within your selected geographical location. Customer data includes the following data types:

- Source code
- Work items
- Test results
- Geo-redundant mirrors and offsite backups

Azure DevOps works with and uses many Microsoft Azure services. For more information on customer data retention by location, see Data residency in Azure ☐ .

Profile data

Azure DevOps stores information that's global in nature, such as user identities and profile data, as follows:

- For US-based users: in the US datacenter
- For EU-based users: in the EU datacenter
- For UK-based users: in the UK datacenter
- For users from all other countries and regions: in the US datacenter

Token data

Azure DevOps stores token data, such as personal access tokens and Secure Shell (SSH) keys, in a US datacenter.

Allowlist data for tenant policies

We recommend using groups with your tenant policy allowlists. If you use a named user, be aware that a reference to the named user's identity resides in the US, EU, and Southeast Asia (Singapore).

Transferring your data

We don't transfer customer data outside your selected location. However, we transfer your data if we need to take any of the following actions:

- Provide customer support
- Troubleshoot the service
- Comply with legal requirements

If necessary, you can transfer your data by using preview, beta, or other prerelease services. These services typically store your data in the United States, but they might store it globally.

> ⊙ **Note**
>
> For builds and releases that run on Microsoft-provided macOS agents, your data is transferred to a GitHub datacenter in the United States. GitHub owns and manages this datacenter location with compliance certifications, such as **SOC 1 Type 2 and SOC 2 Type 2** ⧉ .

Microsoft doesn't control or limit the locations from which you or your users can access your data.

> ⓘ **Note**
>
> Because there's only one region in Brazil, customer data in Brazil is replicated to the South Central US region for disaster recovery and load balancing. For more information, see **Data residency in Azure** ↗ .

Related articles

- [Get started with Azure DevOps](#) ↗
- [Data protection overview](#)

How we store your credentials for Azure DevOps Services

Article • 10/04/2022

Azure DevOps Services

> ⓘ **Important**
>
> Azure DevOps no longer supports Alternate Credentials authentication since the beginning of March 2, 2020. If you're still using Alternate Credentials, we strongly encourage you to switch to a more secure authentication method (for example, personal access tokens). **Learn more** ⬈ .

Credential security

Microsoft is committed to ensuring that your projects remain safe and secure, without exception. In Azure DevOps, your projects benefit from multiple layers of security and governance technologies, operational practices, and compliance policies. We enforce data privacy and integrity both at rest and in transit. In addition, we adhere to the following practices with respect to the credentials or secrets that Azure DevOps stores. To learn more about how to choose the right authentication mechanism, see Guidance for authentication.

Personal access tokens (PATs)

- We store a hash of the PAT
- Raw PAT is generated in-memory on the server side as 32 bytes randomly generated through RNGCryptoServiceProvider then shared with the caller as a base-32-encoded string. This value is NOT stored
- PAT hash is generated in-memory on the server side as an *HMACSHA256Hash* of the raw PAT using a 64-byte symmetric signing key stored in our key vault
- Hash is stored in our database

Secure shell (SSH) keys

- We store a hash of the enclosing organization ID and the SSH public key
- Raw public key is provided directly by the caller over SSL

- SSH hash is generated in-memory on the server side as an *HMACSHA256Hash* of the organization ID and raw public key using a 64-byte symmetric signing key stored in our key vault
- Hash is stored in our database

OAuth credentials (JWTs)

- These are issued as fully self-describing JSON web tokens (JWTs) and are NOT stored in our service
- The claims in JWTs issued and presented to our service are validated using a certificate stored in our key vault

Launch Visual Studio via Azure DevOps Services

Article • 10/04/2022

Azure DevOps Services

When you first open Visual Studio 2015 ⧉, you can sign in and connect to Azure DevOps Services ⧉.

If you're already signed in to Visual Studio or using Visual Studio 2017, connect to Azure DevOps Services.

Once you're connected, you can store or share code in free, unlimited, private, cloud-based Git repositories or Team Foundation Version Control (TFVC). Organize and manage your work with Agile tools for DevOps, continuous integration, and continuous delivery. Your team can build often, test early, and ship faster.

> To set up Visual Studio without Azure DevOps Services, learn how to get started. To host your own server, learn how to install and set up Azure DevOps Server.

Azure DevOps Services is free for up to five users with access to Basic features ⧉ and for unlimited Visual Studio subscribers ⧉ and Stakeholders who can access limited features ⧉. Learn what else you get with Azure DevOps Services ⧉. If you want, you can also use Azure DevOps Services with any IDE or code editor, like the following examples:

- Eclipse, Android Studio, or IntelliJ
- Xcode (see Git or TFVC)
- Visual Studio Code ⧉
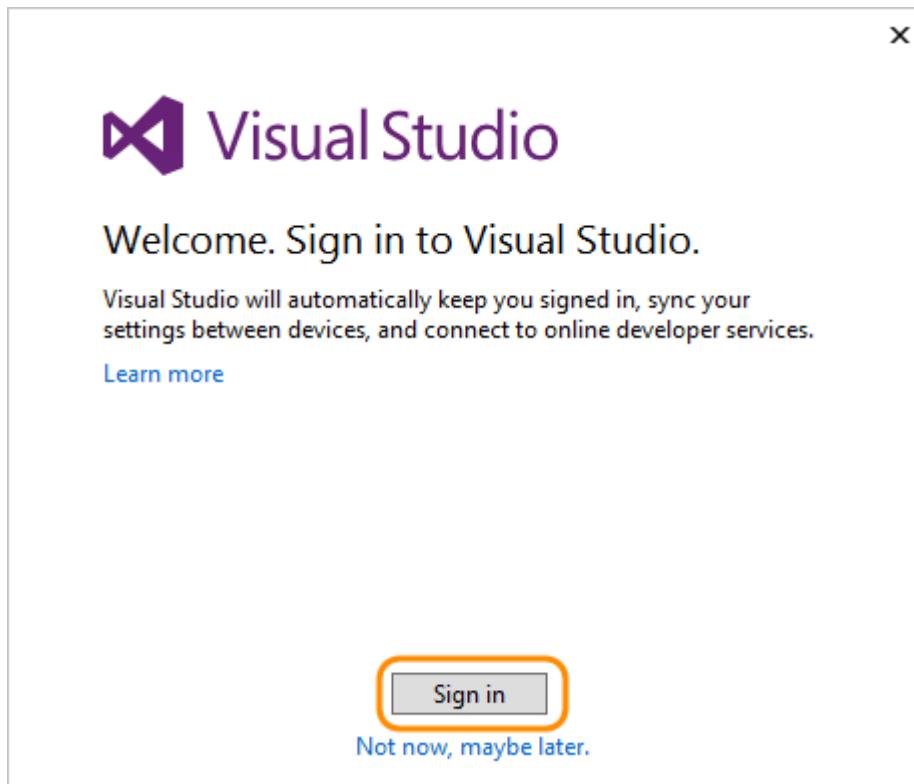
How do I set up Visual Studio 2015 for Azure DevOps Services when I sign in?

1. Download and install Visual Studio ⧉, if you don't have the version you want already. Which versions can I use with Azure DevOps Services?

   If you have a Visual Studio subscription that includes the Visual Studio IDE, get the version that's available with your subscription.

2. Start Visual Studio, and then sign in to create your profile.

This profile saves your settings and roams with you when you sign in to Visual Studio on any computer. Why else should I sign in? If you're a Visual Studio subscriber, use the sign in address for your subscription.

Can't sign in?

3. Enter your sign in address, and then enter your password.

4. Add your Visual Studio profile details. You only need to add these details once.

5. Give your organization a name, and confirm its location.

6. Create your first project to store your code, work items, backlog, builds, tests, and other assets. Name your project, select a process to organize your work, and choose the version control to manage your code.
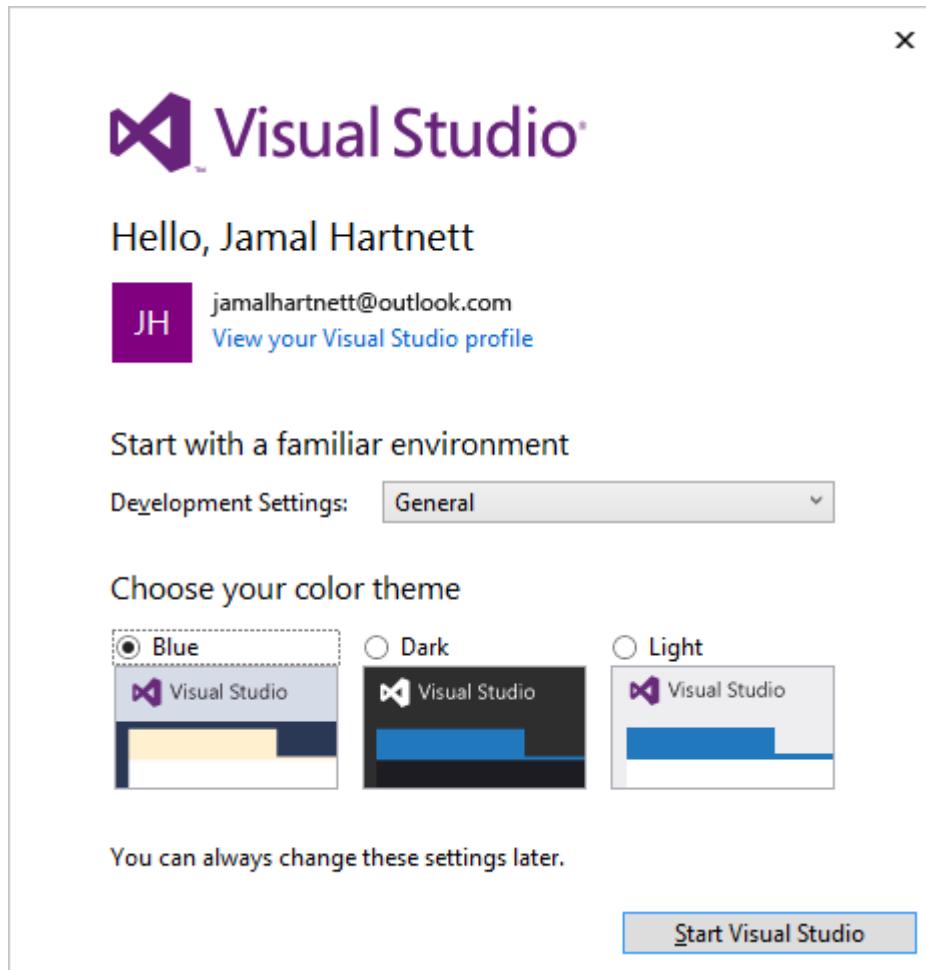
Not sure which to choose? Learn which process and version control (Git or TFVC) work best for you.

7. If you're a new Visual Studio user, you can change your settings here, or change them later in Visual Studio options.

These changes are saved with your profile, and your settings roam with you wherever you sign in.

8. To view your new organization, sign in to `https://dev.azure.com/{yourorganization}`.

Next steps

Add users to your organization

Related articles

- Add code to Git or TFVC.

- Create your backlog to organize your work, manage your process, or customize your process.

Visual Studio for Mac documentation

Important! Visual Studio for Mac is scheduled for retirement on August 31st, 2024 in accordance with Microsoft's Modern Lifecycle Policy. While you can continue to work with Visual Studio for Mac, there are several other options for developers on Mac such as the preview version of the new C# Dev Kit extension for VS Code.

DOWNLOAD
Install Visual Studio for Mac

OVERVIEW
Tour Visual Studio for Mac

WHAT'S NEW
Visual Studio for Mac Release Notes

TRAINING
Improve your development skills

What can I build?

ASP.NET Core

.NET Core Support

Getting Started with .NET Core

ASP.NET Documentation

Azure

Introduction to Azure Functions

Azure Functions Tutorial

Unity

Unity Overview

Set up Unity and Visual Studio for Mac

Xamarin

Xamarin mobile app development in Visual Studio for Mac

What is Xamarin?

Getting started building games with Unity

Get Started with Xamarin

Get Started

Get Started with Visual Studio for Mac

- Tour the IDE
- Open code from a repo
- Work with Projects and Solutions
- Write code
- Debug your code
- Compile and build Projects
- Install a preview release
- Update the IDE

Make it your own

- Sign in
- Customize the IDE
- Use keyboard shortcuts
- Set an EditorConfig

More information

- Visual Studio for Mac support – FAQs
- Pricing and subscriptions
- Compare editions
- Subscriber access

Languages

With Visual Studio for Mac you can write in C#, F#, Razor, HTML5, CSS, Javascript and Typescript, XAML, and XML.

C# **C#**	F# **F#**	JS **JavaScript**
TS **TypeScript**	HTML/Razor	

Tasks

Develop

Intellisense, refactoring, and syntax highlighting for your favorite language.

Build

Configure your compiler and build your app.

Debug

Run your app under the debugger to investigate problems.

Version Control

Share code using version control technologies such as Git and Subversion.

Blogs ⧉ - Twitter ⧉ - Issue Reporting - Developer Community ⧉

Settings & Usage documentation

Configure resources and manage settings for an organization, project, team, or user.

Get started

OVERVIEW

About settings

Get started as an administrator

QUICKSTART

Set user preferences

HOW-TO GUIDE

Enable preview features

Manage your organization (cloud)

OVERVIEW

About managing your organization

Plan your organizational structure

CONCEPT

Manage access with Microsoft Entra ID

QUICKSTART

Create an organization

HOW-TO GUIDE

Add users to your organization

Connect your organization to Microsoft Entra ID

Add & manage projects

⊟ CONCEPT

About projects & scaling up

Customize your project

▣ OVERVIEW

Manage your project

📖 HOW-TO GUIDE

Create a project

Connect to GitHub

Add & manage teams

⊟ CONCEPT

About teams & Agile tools

📖 HOW-TO GUIDE

Add a team

Configure team tools

Define team area paths

Configure team iterations

Add a team administrator

Set alerts or notifications

⊟ CONCEPT

About notifications

HOW-TO GUIDE

Set personal notifications

Set team notifications

Audit & usage (cloud)

📑 CONCEPT

Rate limits & usage

🚀 QUICKSTART

Access, export, and filter audit logs

Configure Pipelines resources

📖 HOW-TO GUIDE

Create & manage agent pools

Create & manage deployment groups

Set build & release retention policies

Create & manage service connections

Configure Repos resources

📖 OVERVIEW

About branches and branch policies

📖 HOW-TO GUIDE

Create & manage Git repositories

Set Git repository settings and policies

Manage Git branch policies

Add TFVC check-in policies

Configure Test resources

HOW-TO GUIDE

Set test retention policies

About projects and scaling your organization

Article • 12/05/2023

Azure DevOps Services | Azure DevOps Server 2022 - Azure DevOps Server 2019
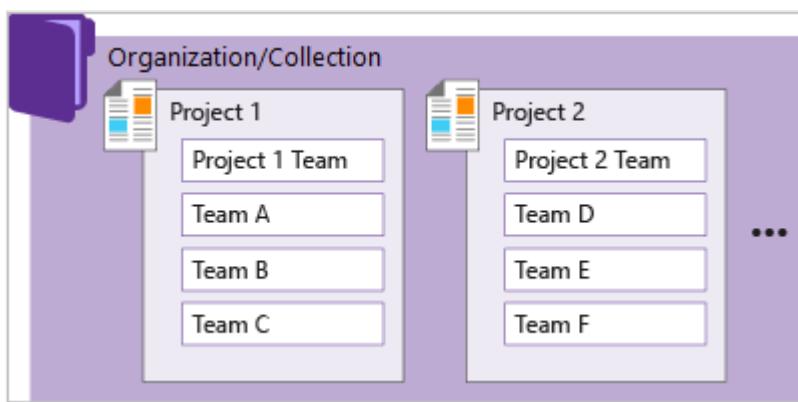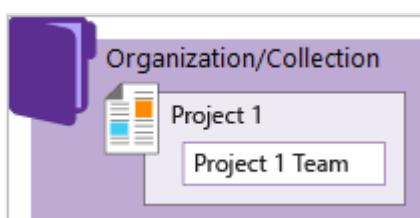
A project in Azure DevOps provides a place for users to plan, track progress, and collaborate on building software solutions. A project represents a fundamental container where you can store data and source code.

When you create your project, Azure DevOps automatically creates a team of the same name, which is sufficient for small organizations. For enterprise-level organizations, you might need to scale up and create more teams and projects. You can have up to 1000 projects within an organization in Azure DevOps.

The following diagram shows one project and team versus multiple projects and teams in an organization or collection.

One project + team

Multiple projects + teams

This structure allows teams to configure the tools in ways that work for them and complete administrative tasks at the appropriate levels. As your organization grows, your tools can grow to support a culture of team autonomy and organizational alignment.

For more information, see [Work tracking, process, and project limits](#) and [Plan your organizational structure](#).

Manage work across your organization

When you connect to Azure DevOps, you connect to an organization. Within that container, you can define one or more projects. At least one project must be created to use the system.
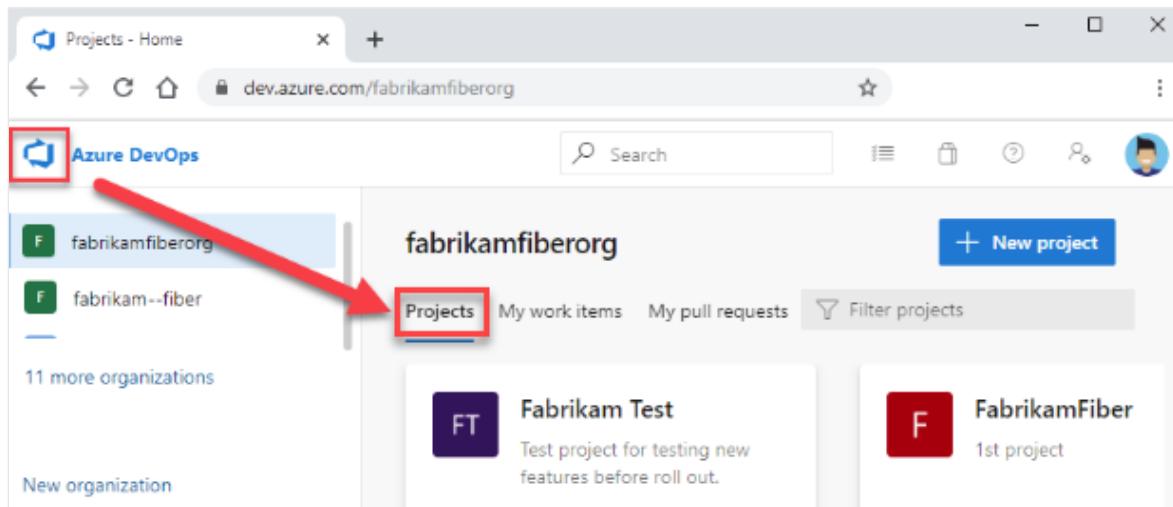
You can scale your organization in the following ways:

- Add projects to support different business units
- Add teams within a project
- Add repositories and branches
- Add agents, agent pools, and deployment pools to support continuous integration and deployment
- Manage access through Microsoft Entra ID to manage a large number of users

View projects in your organization

View the projects defined for your organization by opening the **Projects** page.

1. Select **Azure DevOps** to open **Projects**.

2. Choose a project from the list of projects.

For more information, see [Create a project](#).

Limit project visibility

By default, users added to an organization can view all organization and project information and settings.

For more information, see Limit user visibility for projects and more and Change project visibility to public or private.

View historical data

All project members can view identities that were added to a comment, discussion, or assignment. For example, everyone in the project (even users with the new restriction) can still see a user's name assigned to a work item when the user's no longer part of the project. The same is true for @mentions in PRs, comments, discussions, and more.

Use a single project

One recommended approach is to use a single project to support your organization or enterprise. A single project can help minimize the maintenance of administrative tasks and supports the most optimized and full-flexibility cross-link object experience.

Even if you have many teams working on hundreds of different applications and software projects, you can easily manage them within a single project. A project serves to isolate data stored within it and you can't easily move data from one project to another. When you move data from one project to another, you typically lose the history associated with that data.

For more information, see How many projects do you need?.

Add another project

Another feasible approach is to have multiple projects, which is a recommend approach if your organization is looking to accommodate the following scenarios:

- To prohibit or manage access to the information contained within a project to select groups
- To support custom work tracking processes for specific business units within your organization
- To support entirely separate business units that have their own administrative policies and administrators
- To support testing customization activities or adding extensions before rolling out changes to the working project
- To support an open-source software (OSS) project

Use private and public projects

You can have both private and public projects. You can also change the visibility of a project from either one to the other.

Private projects require that you add and manage user access. Users must sign in to gain access to a project, even if it's read-only access. All project members have access to the project and organization information. For more information, see Resources granted to project members.

Public projects don't require users to sign in to gain read-only access to many of the following services. Public projects provide support to share code with others and to support continuous integration/continuous deployment (CI/CD) of open-source software.

For more information, see Change visibility of a project.

Version control support

Git repositories can be browsed and cloned, but only via HTTPS. SSH and GVFS endpoints are unavailable. Clients like Visual Studio and IntelliJ work with the HTTPS clone URL but don't offer the connected experience linking to work items and other collateral.

Dashboard widget support

The following dashboard widgets don't display any useful information for nonmembers.

- Assigned to me
- Code tile
- New work item
- Pull request
- Query results
- Requirements quality
- Sprint burndown
- Sprint capacity
- Sprint overview
- Team members
- Welcome
- Work links
- Other links

Structure your project

Use the following elements to structure your project to support your business needs.

- Create a Git repository for each subproject or application, or create root folders within a TFVC repository for each subproject. If you're using TFVC and heading toward a combined project model, create root folders for different teams and projects, just as you would create separate repos in Git. Secure folders as needed and control which segments of the repo you're actively using with workplace mappings.
- Define area paths to support different subprojects, products, features, or teams.
- Define iteration paths (also known as sprints) that can be shared across teams.
- Add a team for each product team that develops a set of features for a product. Each team you create automatically creates a security group for that team, which you can use to manage permissions for a team. For more information, see Portfolio management.
- Grant or restrict access to select features and functions using custom security groups.
- Create query folders to organize queries for teams or product areas into folders.
- Define or modify notifications set at the project level.

Customize and configure your project

You can configure and customize most services and applications to support your business needs or the way your teams work. Within each project, you can do the following tasks. For a comprehensive view of which resources can be configured, see About team, project, and organizational-level settings.

- **Dashboards**: Each team can configure their set of dashboards to share information and monitor progress.
- **Source control**: For each Git repository, you can apply branch policies and define branch permissions. For TFVC repositories, you can set check-in policies.
- **Work tracking**: You can add fields, change the workflow, add custom rules, and add custom pages to the work item form of most work item types. You can also add custom work item types. For more information, see Customize an inheritance process.
- **Azure Pipelines**: You can fully customize your build and release pipelines, and define build steps, release environments, and deployment schedule. For more information, see Build and release.
- **Azure Test Plans**: You can define and configure test plans, test suites, test cases, and test environments. You can also add test steps within your build pipelines. For

more information, see Exploratory and manual testing and continuous testing for your builds.

Add a team

As your organization grows, you can add teams equipped with configurable Agile tools to meet each team's workflow. For more information, see the following articles.

- Scale Agile to large teams
- About teams and Agile tools
- Manage a portfolio of backlogs and see progress.
- Use delivery plans to scheduled work items by sprint (iteration path) of selected teams against a calendar view.
- Incrementally adopt practices that scale to create greater rhythm and flow within your organization, engage customers, improve project visibility, and develop a productive workforce.
- Structure projects to gain visibility across teams or to support epics, release trains, and multiple backlogs to support the Scaled Agile Framework.

Connect to a project with other clients

Aside from connecting via a web browser, you can connect to a project from the following clients:

- Visual Studio (Professional, Enterprise, Test Professional) ⧉
- Visual Studio Code ⧉
- Visual Studio Community ⧉
- Office Excel
- Test & Feedback extension
- Microsoft Feedback Client

For more information, see Compatibility with Azure DevOps Server versions.

Key concepts

Use the following index to quickly access concepts and tasks related to managing projects and teams.

- About projects
- About teams
- Access levels

- [Area paths](#)
- [Dashboards](#)
- [Notifications and subscriptions](#)
- [GitHub connections](#)
- [Iteration paths](#)
- [Permissions](#)
- [Process (Inherited)](#)
- [Project resources viewable by members](#)
- [Project Wiki](#)
- [Project-level permissions](#)
- [Project-level security groups](#)
- [Project and process object limits](#)
- [Projects page](#)
- [Public vs private projects](#)
- [Security groups](#)
- [Service hooks](#)
- [Service visibility](#)
- [Summary page](#)

User and administrative tasks

Several of the following tasks require permissions granted to a member of the Project Administrators group or a team administrator.

- [Add Git repository](#)
- [Add project administrators](#)
- [Add project dashboard](#)
- [Add project members](#)
- [Add security groups](#)
- [Add team administrators](#)
- [Add team members](#)
- [Add/manage service hooks](#)
- [Connect to a project](#)
- [Connect to GitHub](#)
- [Create project](#)
- [Delete project](#)
- [Edit project Summary](#)
- [Enable/disable project services](#)
- [Export list of projects](#)
- [Export list of teams](#)
- [Manage notifications](#)

- Manage your project
- Navigate the Web portal
- Remove team
- Rename project
- Rename team
- Restore project
- Change user access levels
- Search across project(s)
- Set area paths
- Set favorites
- Set iteration paths
- Set project-level permissions
- Set project visibility
- Switch project, repository, team

Frequently asked questions (FAQs)

Q: Can I move or transfer a project to another organization or collection?

A: Yes, but not without losing data. You can manually copy resources and leave some behind, or use a third-party tool, such as OpsHub Visual Studio Migration Utility ↗, which copies data using the REST APIs.

Q: What programmatic tools support projects?

A. See Projects REST API.

You can also use the **az devops project** CLI.

Related articles

- Get started as an administrator
- Web portal navigation
- What do I get with a project?
- Understand differences between Azure DevOps

Marketplace & Extensibility documentation

Discover, manage, develop extensions and widgets for integration with Azure DevOps.

Discover & manage Marketplace extensions

⊟ CONCEPT

About Azure DevOps Marketplace

📖 HOW-TO GUIDE

Request extensions

Install extensions

Manage extension permissions

Develop extensions

🚀 QUICKSTART

Develop a web extension

Extension samples

Extensibility points

📑 TUTORIAL

Create a custom pipelines task

Integrate applications

🗺 OVERVIEW

Integration overview

Authorize access to REST APIs with OAuth 2.0

Authorize access with service principals and managed identities

Authenticate access with personal access tokens

REFERENCE

.NET client libraries

Integrate with Slack or Microsoft Teams

HOW-TO GUIDE

Azure Boards with Slack

Azure Pipelines with Slack

Azure Pipelines with Microsoft Teams

Integrate with Service hooks

OVERVIEW

Integrate with service hooks

TUTORIAL

Create a service hook with Microsoft Teams

Create a service hook with WebHooks

What is DevOps?

Article • 01/25/2023
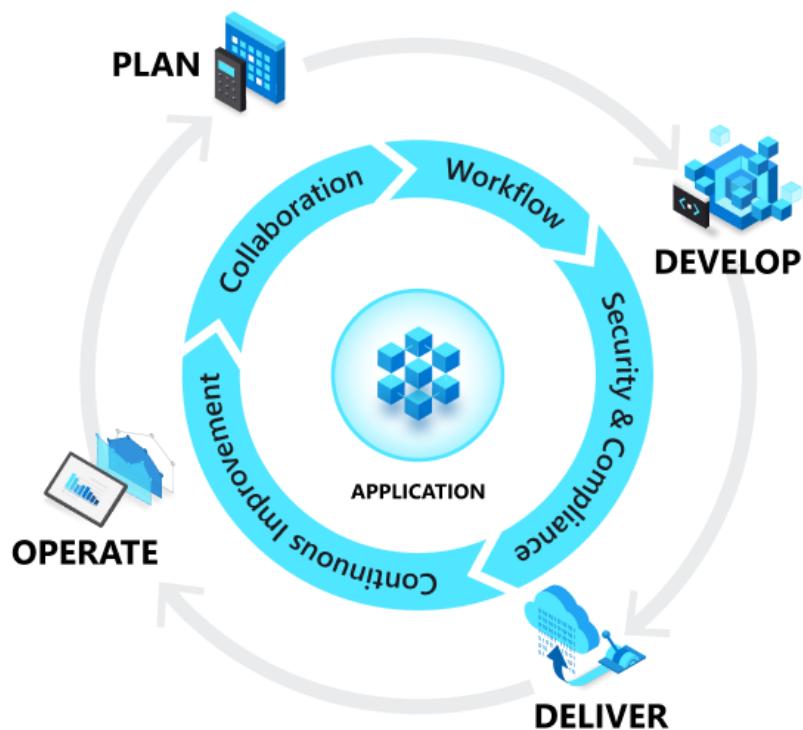
DevOps combines development (Dev) and operations (Ops) to unite people, process, and technology in application planning, development, delivery, and operations. DevOps enables coordination and collaboration between formerly siloed roles like development, IT operations, quality engineering, and security.

Teams adopt DevOps culture, practices, and tools to increase confidence in the applications they build, respond better to customer needs, and achieve business goals faster. DevOps helps teams continually provide value to customers by producing better, more reliable products.

DevOps and the application lifecycle

DevOps influences the application lifecycle ⤢ throughout its planning, development, delivery, and operations phases. Each phase relies on the other phases, and the phases aren't role-specific. A DevOps culture involves all roles in each phase to some extent.

The following diagram illustrates the phases of the DevOps application lifestyle:
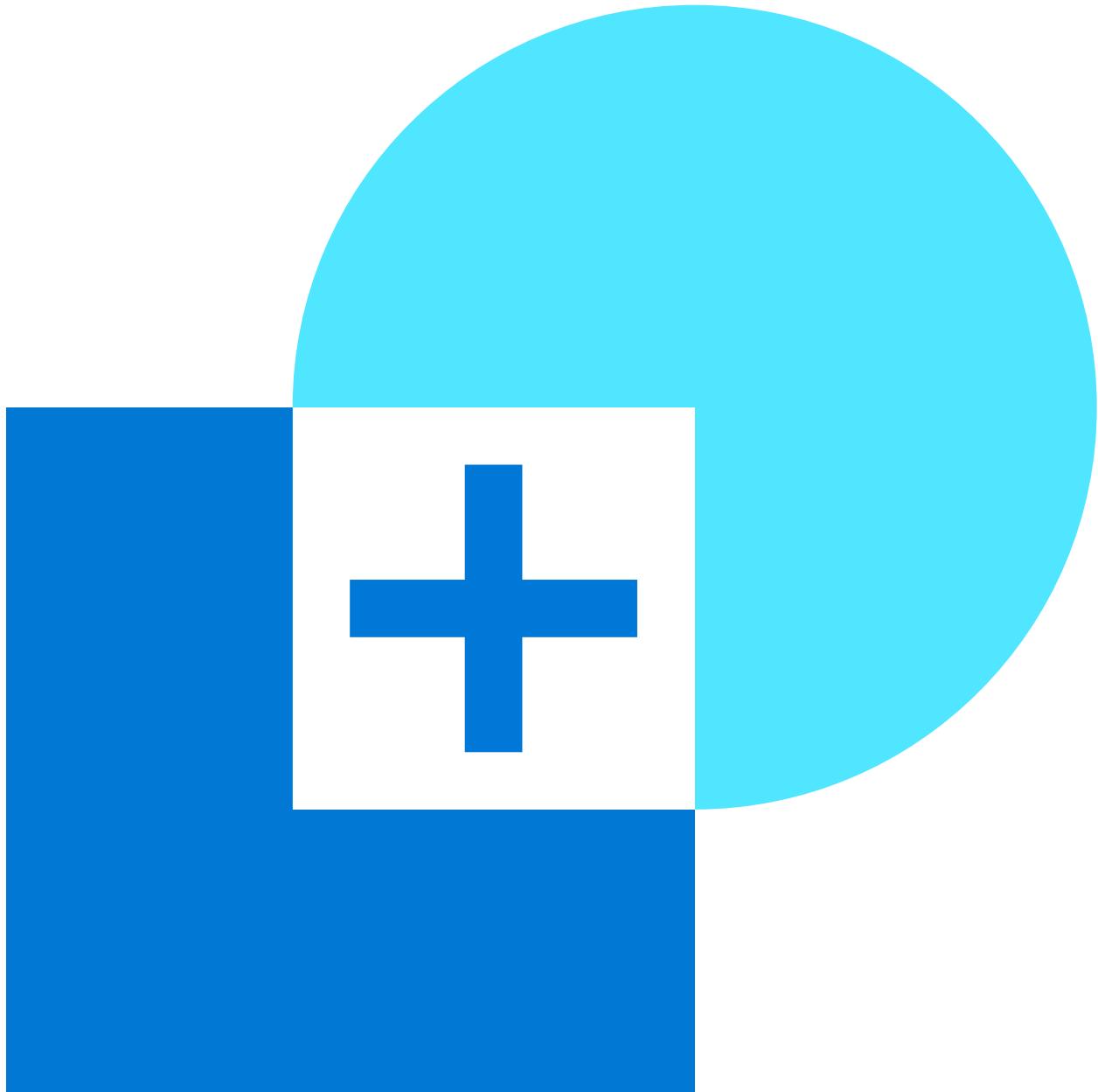
DevOps goals and benefits

When a team adopts DevOps culture, practices, and tools, they can achieve amazing things:
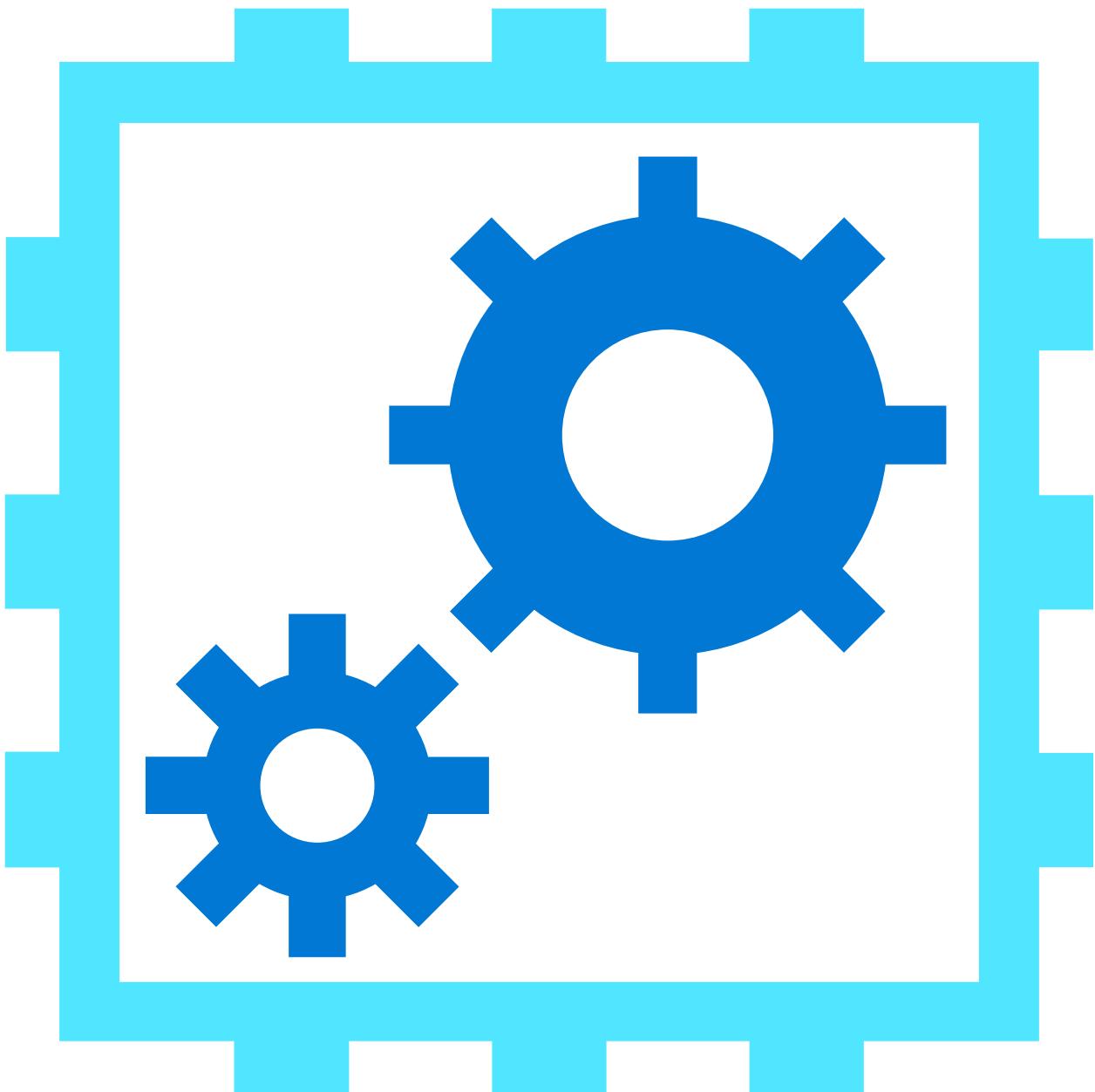
Accelerate time to market

Through increased efficiencies, improved team collaboration, automation tools, and continuous deployment--teams are able to rapidly reduce the time from product inception to market launch.
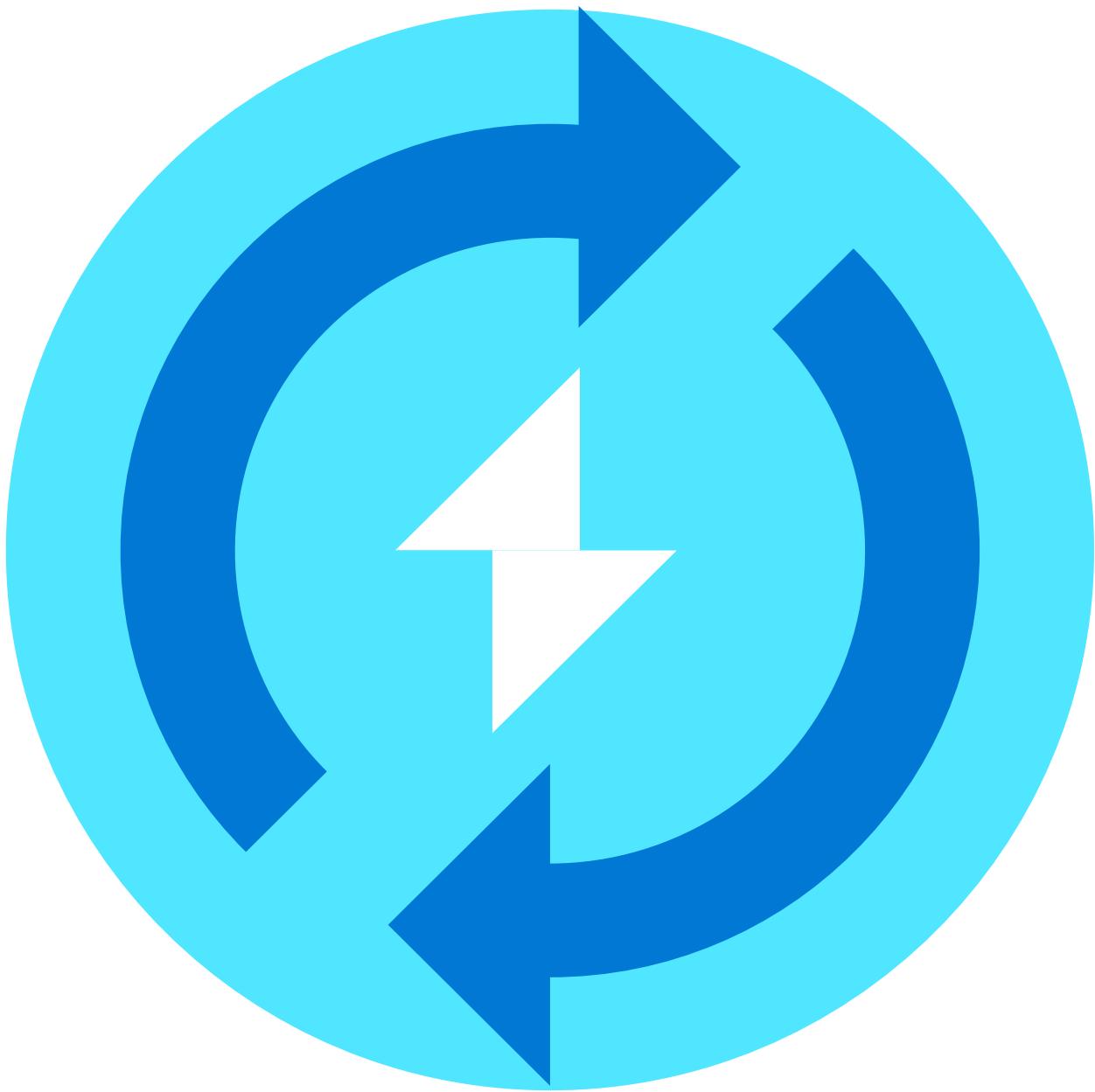
Adapt to the market and competition

A DevOps culture demands teams have a customer-first focus. By marrying agility, team collaboration, and focus on the customer experience, teams can continuously deliver value to their customers and increase their competitiveness in the marketplace.

Maintain system stability and reliability

By adopting continuous improvement practices, teams are able to build in increased stability and reliability of the products and services they deploy. These practices help reduce failures and risk.
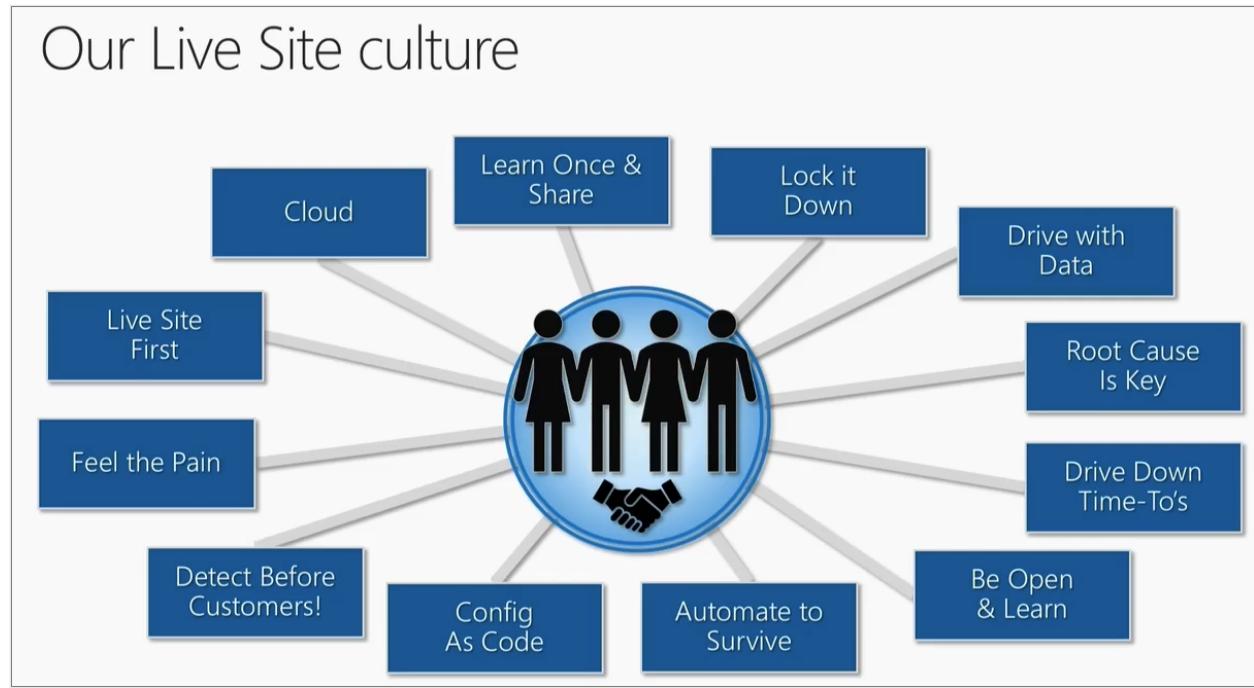
Improve the mean time to recovery

The *mean time to recovery* metric indicates how long it takes to to recover from a failure or breach. To manage software failures, security breaches, and continuous improvement plans, teams should measure and work to improve this metric.

Adopt a DevOps culture

To fully implement DevOps, you must adopt a DevOps culture. Cultivating a DevOps culture requires deep changes in the way people work and collaborate. When organizations commit to a DevOps culture, they create an environment for high-performing teams to evolve. While adopting DevOps practices automates and optimizes processes through technology, without a shift to a DevOps culture within the organization and its people, you won't gain the full benefits of DevOps.

The following image captures key aspects of Microsoft's live site culture.

The following practices are key components of a DevOps culture:

- **Collaboration, visibility, and alignment**: A hallmark of a healthy DevOps culture is collaboration between teams. Collaboration starts with visibility. Development, IT, and other teams should share their DevOps processes, priorities, and concerns with each other. By planning their work together, they are better positioned to align on goals and measures of success as they relate to the business.
- **Shifts in scope and accountability**: As teams align, they take ownership and become involved in other lifecycle phases—not just the ones central to their roles. For example, developers become accountable not only to the innovation and quality established in the develop phase, but also to the performance and stability their changes bring in the operate phase. At the same time, IT operators are sure to include governance, security, and compliance in the plan and develop phase.
- **Shorter release cycles**: DevOps teams remain agile by releasing software in short cycles. Shorter release cycles make planning and risk management easier since progress is incremental, which also reduces the impact on system stability. Shortening the release cycle also allows organizations to adapt and react to evolving customer needs and competitive pressure.
- **Continuous learning**: High-performing DevOps teams establish a growth mindset. They fail fast and incorporate learnings into their processes. They strive to continually improve, increase customer satisfaction, and accelerate innovation and market adaptability.

Implement DevOps practices

You implement DevOps by following DevOps practices (described in the sections that follow) throughout the application lifecycle. Some of these practices help accelerate, automate, and improve a specific phase. Others span several phases, helping teams create seamless processes that help improve productivity.

Continuous integration and continuous delivery (CI/CD)

Continuous Integration (CI) is the practice used by development teams to automate, merge, and test code. CI helps to catch bugs early in the development cycle, which makes them less expensive to fix. Automated tests execute as part of the CI process to ensure quality. CI systems produce artifacts and feed them to release processes to drive frequent deployments.

Continuous Delivery (CD) is a process by which code is built, tested, and deployed to one or more test and production environments. Deploying and testing in multiple environments increases quality. CD systems produce deployable artifacts, including infrastructure and apps. Automated release processes consume these artifacts to release new versions and fixes to existing systems. Systems that monitor and send alerts run continually to drive visibility into the entire CD process.

Version Control

Version control is the practice of managing code in versions—tracking revisions and change history to make code easy to review and recover. This practice is usually implemented using version control systems such as Git, which allow multiple developers to collaborate in authoring code. These systems provide a clear process to merge code changes that happen in the same files, handle conflicts, and roll back changes to earlier states.

The use of version control is a fundamental DevOps practice, helping development teams work together, divide coding tasks between team members, and store all code for easy recovery if needed. Version control is also a necessary element in other practices such as continuous integration and infrastructure as code.

Agile software development

Agile is a software development approach that emphasizes team collaboration, customer and user feedback, and high adaptability to change through short release cycles. Teams that practice Agile provide continual changes and improvements to customers, collect their feedback, then learn and adjust based on customer wants and needs. Agile is substantially different from other more traditional frameworks such as

waterfall, which includes long release cycles defined by sequential phases. Kanban and Scrum are two popular frameworks associated with Agile.

Infrastructure as code

Infrastructure as code defines system resources and topologies in a descriptive manner that allows teams to manage those resources as they would code. Those definitions can also be stored and versioned in version control systems, where they can be reviewed and reverted—again like code.

Practicing infrastructure as code helps teams deploy system resources in a reliable, repeatable, and controlled way. Infrastructure as code also helps automate deployment and reduces the risk of human error, especially for complex large environments. This repeatable, reliable solution for environment deployment lets teams maintain development and testing environments that are identical to production. Duplicating environments to different data centers and cloud platforms likewise becomes simpler and more efficient.

Configuration management

Configuration management refers to managing the state of resources in a system including servers, virtual machines, and databases. Using configuration management tools, teams can roll out changes in a controlled, systematic way, reducing the risks of modifying system configuration. Teams use configuration management tools to track system state and help avoid configuration drift, which is how a system resource's configuration deviates over time from the desired state defined for it.

Along with infrastructure as code, it's easy to templatize and automate system definition and configuration, which help teams operate complex environments at scale.

Continuous monitoring

Continuous monitoring means having full, real-time visibility into the performance and health of the entire application stack. This visibility ranges from the underlying infrastructure running the application to higher-level software components. Visibility is accomplished through the collection of telemetry and metadata and setting of alerts for predefined conditions that warrant attention from an operator. Telemetry comprises event data and logs collected from various parts of the system, which are stored where they can be analyzed and queried.

High-performing DevOps teams ensure they set actionable, meaningful alerts and collect rich telemetry so they can draw insights from vast amounts of data. These insights help the team mitigate issues in real time and see how to improve the application in future development cycles.

Planning

In the planning phase, DevOps teams ideate, define, and describe the features and capabilities of the applications and systems they plan to build. Teams track task progress at low and high levels of granularity, from single products to multiple product portfolios. Teams use the following DevOps practices to plan with agility and visibility:

- Create backlogs.
- Track bugs.
- Manage Agile software development with Scrum.
- Use Kanban boards.
- Visualize progress with dashboards.

For an overview of the several lessons learned and practices Microsoft adopted to support DevOps planning across the company's software teams, see How Microsoft plans with DevOps.

Development

The development phase includes all aspects of developing software code. In this phase, DevOps teams do the following tasks:

- Select a development environment.
- Write, test, review, and integrate the code.
- Build the code into artifacts to deploy into various environments.
- Use version control, usually Git, to collaborate on code and work in parallel.

To innovate rapidly without sacrificing quality, stability, and productivity, DevOps teams:

- Use highly productive tools.
- Automate mundane and manual steps.
- Iterate in small increments through automated testing and continuous integration (CI).

For an overview of the development practices Microsoft adopted to support their shift to DevOps, see How Microsoft develops with DevOps.

Deliver

Delivery is the process of consistently and reliably deploying applications into production environments, ideally via continuous delivery (CD).

In the delivery phase, DevOps teams:

- Define a release management process with clear manual approval stages.
- Set automated gates to move applications between stages until final release to customers.
- Automate delivery processes to make them scalable, repeatable, controlled, and well-tested.

Delivery also includes deploying and configuring the delivery environment's foundational infrastructure. DevOps teams use technologies like infrastructure as code (IaC), containers ⧉, and microservices to deliver fully governed infrastructure environments.

Safe deployment practices can identify issues before they affect the customer experience. These practices help DevOps teams deliver frequently with ease, confidence, and peace of mind.

Core DevOps principles and processes Microsoft evolved to provide efficient delivery systems are described in How Microsoft delivers software with DevOps.

Operations

The operations phase involves maintaining, monitoring, and troubleshooting applications in production environments, including hybrid or public clouds like Azure ⧉. DevOps teams aim for system reliability, high availability, strong security, and zero downtime.

Automated delivery and safe deployment practices help teams identify and mitigate issues quickly when they occur. Maintaining vigilance requires rich telemetry, actionable alerting, and full visibility into applications and underlying systems.

Practices Microsoft uses to operate complex online platforms are described in How Microsoft operates reliable systems with DevOps.

Next steps

- Plan efficient workloads with DevOps

- Develop modern software with DevOps
- Deliver quality services with DevOps
- Operate reliable systems with DevOps

Other resources

- DevOps solutions on Azure ⧉
- The DevOps journey at Microsoft ⧉
- Start doing DevOps with Azure ⧉
- Security in DevOps (DevSecOps)

Training and Certifications

- Get started with Azure DevOps
- Introduce DevOps Dojo: Create efficiencies that support your business
- AZ-400: Get started on a DevOps transformation journey
- Facilitate communication and collaboration
- Exam AZ-400: Designing and Implementing Microsoft DevOps Solutions
- AZ-400: Implement security and validate code bases for compliance

What is Agile?

Article • 11/28/2022

Agile is a term that describes approaches to software development that emphasize incremental delivery, team collaboration, continual planning, and continual learning. The term *Agile* was coined in 2001 in the Agile Manifesto⬈ . The manifesto set out to establish principles to guide a better approach to software development. At its core, the manifesto declares four value statements that represent the foundation of the Agile movement. As written, the manifesto states:

We have come to value:

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

The manifesto doesn't imply that the items on the right side of these statements aren't important or needed. Rather, items on the left are simply more valued.

Agile methods and practices

It's important to understand that Agile isn't a *thing*. You don't *do Agile*. Rather, Agile is a mindset that drives an approach to software development. Because there's no single approach that works for all situations, the term *Agile* has come to represent various methods and practices that align with the value statements in the manifesto.

Agile methods, which are often called frameworks, are comprehensive approaches to phases of the DevOps lifecycle: planning, development, delivery, and operations. They prescribe a method for accomplishing work, with clear guidance and principles.

Scrum is the most common Agile framework, and the one that most people start with. Agile practices, on the other hand, are techniques that are applied during phases of the software development lifecycle.

- Planning Poker☑ is a collaborative estimation practice that's designed to encourage team members to share their understanding of what *done* means. Many people find the process fun, and it has proven to help foster teamwork and better estimates.
- Continuous integration (CI) is a common Agile engineering practice that involves integrating code changes into the main branch frequently. An automated build verifies changes. As a result, there's a reduction in integration debt and a continually shippable main branch.

These practices, like all Agile practices, carry the *Agile* label, because they're consistent with the principles in the Agile manifesto.

What Agile isn't

As Agile has gained popularity, many stereotypes and misinterpretations have cast a negative shadow on its effectiveness. It's easy to say "*Yes, we're doing Agile*," without any accountability. Keeping this point in mind, consider a few things that Agile isn't.

- Agile isn't cowboy coding☑. Agile shouldn't be confused with a "we'll figure it out as we go" approach to software development. Such an idea couldn't be further from the truth. Agile requires both a definition of done☑ and explicit value that's delivered to customers in every sprint. While Agile values autonomy for individuals and teams, it emphasizes aligned autonomy to ensure that the increased autonomy produces increased value.
- Agile isn't without rigor and planning. On the contrary, Agile methodologies and practices typically emphasize discipline in planning. The key is continual planning throughout the project, not just planning up front. Continual planning ensures that the team can learn from the work that they execute. Through this approach, they maximize the return on investment (ROI) of planning.

> "Plans are worthless, but planning is everything." — Dwight D. Eisenhower

- Agile isn't an excuse for the lack of a roadmap. This misconception has probably done the most harm to the Agile movement overall. Organizations and teams that

follow an Agile approach absolutely know where they're going and the results that they want to achieve. Recognizing change as part of the process is different from pivoting in a new direction every week, sprint, or month.

- Agile isn't development without specifications. It's necessary in any project to keep your team aligned on *why* and *how* work happens. An Agile approach to specs includes ensuring that specs are *right-sized*, and that they reflect appropriately how the team sequences and delivers work.

- Agile isn't incapable of accommodating unplanned work and other interruptions. It's important to complete sprints on schedule. But just because an issue comes up that sidetracks development doesn't mean that a sprint has to fail. Teams can plan for interruptions by designating resources ahead of time for problems and unexpected issues. Then they can address those issues but stay on track with development.

- Agile isn't inappropriate for large organizations. A common complaint is that collaboration, a key component of Agile methodologies, is difficult in large teams. Another gripe is that scalable approaches to Agile introduce structure and methods that compromise flexibility. In spite of these misconceptions, it's possible to scale Agile principles successfully. For information about overcoming these difficulties, see Scaling Agile to large teams.

- Agile isn't inefficient. To adapt to customers' changing needs, developers invest time each iteration to demonstrate a working product and collect feedback. It's true that these efforts reduce the time that they spend on development. But incorporating customer requests early on saves significant time later. When features stay aligned with the customer's vision, developers avoid major overhauls down the line.

- Agile isn't a poor fit for today's applications, which often center on data streaming. Such projects typically involve more data modeling and extract-transform-load (ETL) workloads than user interfaces. This fact makes it hard to demonstrate usable software on a consistent, tight schedule. But by adjusting goals, developers can still use an Agile approach. Instead of working to accomplish tasks each iteration, developers can focus on running data experiments. Instead of presenting a working product every few weeks, they can aim to better understand the data.

Why Agile?

So why would anyone consider an Agile approach? It's clear that the rules of engagement around building software have fundamentally changed in the last 10-15 years. Many of the activities look similar, but the landscape and environments where we apply them are noticeably different.

- Compare what it's like to purchase software today with the early 2000s. How often do people drive to the store to buy business software?
- Consider how feedback is collected from customers about products. How did a team understand what people thought about their software before social media?
- Consider how often a team desires to update and improve the software that they deliver. Annual updates are no longer feasible against modern competition.

Forrester's Diego Lo Guidice says it best in his blog, *Transforming Application Delivery* (October, 2020).

> "Everything has dramatically changed. Sustainability, besides green and clean, means that what we build today has to be easily and quickly changed tomorrow. Strategic plans are short-term, and planning and change are continuous." — Diego Lo Guidice, Forrester

The rules have changed, and organizations around the world now adapt their approach to software development accordingly. Agile methods and practices don't promise to solve every problem. But they do promise to establish a culture and environment where solutions emerge through collaboration, continual planning and learning, and a desire to ship high-quality software more often.

Next steps

Deciding to take the Agile route to software development can introduce some interesting opportunities for enhancing your DevOps process. One key set of considerations focuses on how Agile development compares and contrasts with an organization's current approach.

What is Git?

Article • 11/28/2022

Git has become the worldwide standard for version control. So what exactly is it?

Git is a distributed version control system, which means that a local clone of the project is a complete version control repository. These fully functional local repositories make it easy to work offline or remotely. Developers commit their work locally, and then sync their copy of the repository with the copy on the server. This paradigm differs from centralized version control where clients must synchronize code with a server before creating new versions of code.

Git's flexibility and popularity make it a great choice for any team. Many developers and college graduates already know how to use Git. Git's user community has created resources to train developers and Git's popularity make it easy to get help when needed. Nearly every development environment has Git support and Git command line tools implemented on every major operating system.
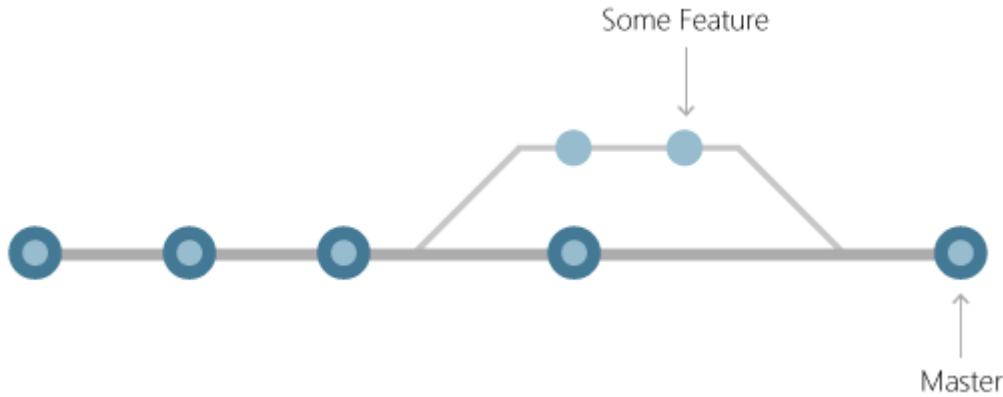
Git basics

Every time work is saved, Git creates a commit. A commit is a snapshot of all files at a point in time. If a file hasn't changed from one commit to the next, Git uses the previously stored file. This design differs from other systems that store an initial version of a file and keep a record of deltas over time.

Commits create links to other commits, forming a graph of the development history. It's possible to revert code to a previous commit, inspect how files changed from one commit to the next, and review information such as where and when changes were made. Commits are identified in Git by a unique cryptographic hash of the contents of the commit. Because everything is hashed, it's impossible to make changes, lose information, or corrupt files without Git detecting it.
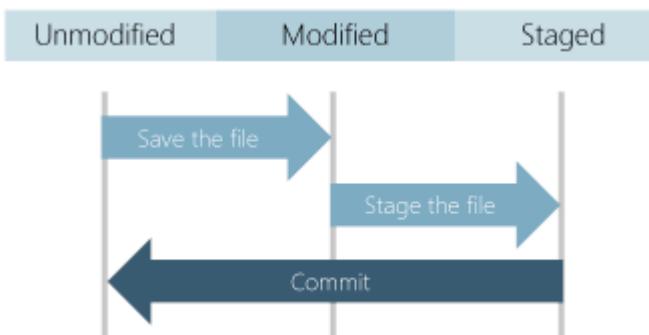
Branches

Each developer saves changes to their own local code repository. As a result, there can be many different changes based off the same commit. Git provides tools for isolating changes and later merging them back together. Branches, which are lightweight pointers to work in progress, manage this separation. Once work created in a branch is finished, it can be merged back into the team's main (or trunk) branch.

Files and commits

Files in Git are in one of three states: modified, staged, or committed. When a file is first modified, the changes exist only in the working directory. They aren't yet part of a commit or the development history. The developer must *stage* the changed files to be included in the commit. The staging area contains all changes to include in the next commit. Once the developer is happy with the staged files, the files are packaged as a *commit* with a message describing what changed. This commit becomes part of the development history.

Staging lets developers pick which file changes to save in a commit in order to break down large changes into a series of smaller commits. By reducing the scope of commits, it's easier to review the commit history to find specific file changes.

Benefits of Git

The benefits of Git are many.

Simultaneous development

Everyone has their own local copy of code and can work simultaneously on their own branches. Git works offline since almost every operation is local.

Faster releases

Branches allow for flexible and simultaneous development. The main branch contains stable, high-quality code from which you release. Feature branches contain work in progress, which are merged into the main branch upon completion. By separating the release branch from development in progress, it's easier to manage stable code and ship updates more quickly.

Built-in integration

Due to its popularity, Git integrates into most tools and products. Every major IDE has built-in Git support, and many tools support continuous integration, continuous deployment, automated testing, work item tracking, metrics, and reporting feature integration with Git. This integration simplifies the day-to-day workflow.

Strong community support

Git is open-source and has become the de facto standard for version control. There is no shortage of tools and resources available for teams to leverage. The volume of community support for Git compared to other version control systems makes it easy to get help when needed.

Git works with any team

Using Git with a source code management tool increases a team's productivity by encouraging collaboration, enforcing policies, automating processes, and improving visibility and traceability of work. The team can settle on individual tools for version control, work item tracking, and continuous integration and deployment. Or, they can choose a solution like GitHub ⧉ or Azure DevOps ⧉ that supports all of these tasks in one place.

Pull requests

Use pull requests to discuss code changes with the team before merging them into the main branch. The discussions in pull requests are invaluable to ensuring code quality

and increase knowledge across your team. Platforms like GitHub and Azure DevOps offer a rich pull request experience where developers can browse file changes, leave comments, inspect commits, view builds, and vote to approve the code.

Branch policies

Teams can configure GitHub and Azure DevOps to enforce consistent workflows and process across the team. They can set up branch policies to ensure that pull requests meet requirements before completion. Branch policies protect important branches by preventing direct pushes, requiring reviewers, and ensuring clean builds.

Next steps

Install and set up Git