


Troubleshoot GitHub & Azure Boards connection

04/03/2019 • 4 minutos para ler • Colaboradores 

Neste artigo

[Unexpected results when linking to projects defined in two or more Azure DevOps organizations](#)

[Resolve connection issues](#)

[Update XML definitions for select work item types](#)

Azure Boards | Azure DevOps Server 2019

When you create a GitHub connection, you are granted access to GitHub as an OAuth app or by using a Personal Access Token (PAT).

The access by Azure Boards to the GitHub repo can be revoked in one or more ways. If the user who created the connection PAT is revoked or the permission scope changes, then the Azure Boards access is revoked. Or the OAuth app's authorization can be revoked entirely for a given repo.

Observação

Azure Boards and Azure DevOps Services support integration with GitHub.com and GitHub Enterprise Server repositories.

Azure DevOps Server 2019 supports integration with GitHub Enterprise Server repositories.

Unexpected results when linking to projects defined in two or more Azure DevOps organizations

If you connect your GitHub repository to two or more projects that are defined in more than one Azure DevOps organization, such as dev.azure.com/Contoso and dev.azure.com/Fabrikam, you may get unexpected results when using **AB#** mentions to link to work items. This problem occurs because work item IDs are not unique across Azure DevOps organizations, so **AB#12** can refer to a work item in either the Contoso or Fabrikam organization. So, when a work item is mentioned in a commit message or pull request, both organizations will attempt to create a link to a work item with a matching ID (if one exists).

In general, a user intends an **AB#** mention to link to a single work item in one of the projects. However, if a work item of the same ID exists in both accounts, then links are created for both work items, likely causing confusion.

Currently, there is no way to work around this issue, so we recommend that you connect a single GitHub repository only to a single Azure DevOps organization.

Observação

When making the connection using the Azure Boards app for GitHub, the app prevents you from connecting to two different organizations. If a GitHub repository is incorrectly connected to the wrong Azure DevOps organization, you'll need to contact the owner of that organization to remove the

connection before you'll be able to add the repository to the correct Azure DevOps organization.

Resolve connection issues

When the Azure Boards connection to GitHub no longer has access, it shows an alert status in the user interface with a red-X that has a tooltip such as, *Unable to connect to GitHub*.

To resolve the problem, consider the following:

- If the connection is using OAuth:**
 - The Azure Boards application had it's access denied for one of the repositories.
 - GitHub might be unavailable/unreachable. This could be due to an outage in either service or an infrastructure/network issue on-prem. You can check service status from the following links:
 - [GitHub](#)
 - [Azure Devops](#)

To resolve the first issue, delete and recreate the connection to the GitHub repository. This will cause GitHub to prompt to reauthorize Azure Boards.

- If the connection is using a PAT:**
 - The PAT may have been revoked or the required permission scopes changed and are insufficient.
 - The user may have lost admin permissions on the GitHub repo.

To resolve, recreate the PAT and ensure the scope for the token includes the required permissions:


```
repo, read:user, user:email, admin:repo_hook
```

Update XML definitions for select work item types

If your organization uses the Hosted XML or On-premises XML process model to customize the work tracking experience and you want to link to and view the GitHub link types from the Development section in the work item forms, you'll need to update the XML definitions for the work item types.

For example, if you want to link user stories and bugs to GitHub commits and pull requests from the Development section, then you need to update the XML definitions for user stories and bugs.

Follow the sequence of tasks provided in [Hosted XML process model](#) to update the XML definitions. For each work item type, find the `<Group Label="Development">` section, and add the following two lines in the following code syntax to support the external links types: **GitHub Commit** and **GitHub Pull Request**.

XML	 Copiar
<pre><ExternalLinkFilter Type="GitHub Pull Request" /> <ExternalLinkFilter Type="GitHub Commit" /></pre>	

When updated, the section should appear as shown.

XML	 Copiar
<pre><Group Label="Development"> <Control Type="LinksControl" Name="Development"> <LinksControlOptions ViewMode="Dynamic" ZeroDataExperience="Development" ShowCallToAction="true"></pre>	

```

<ListViewOptions GroupLinks="false">
</ListViewOptions>
<LinkFilters>
  <Externallinkfilter Type="Build" />
  <Externallinkfilter Type="Integrated in build" />
  <Externallinkfilter Type="Pull Request" />
  <Externallinkfilter Type="Branch" />
  <Externallinkfilter Type="Fixed in Commit" />
  <Externallinkfilter Type="Fixed in Changeset" />
  <Externallinkfilter Type="Source Code File" />
  <Externallinkfilter Type="Found in build" />
  <Externallinkfilter Type="GitHub Pull Request" />
  <Externallinkfilter Type="GitHub Commit" />
</LinkFilters>
</LinksControlOptions>
</Control>
</Group>

```