**Overview**

This PowerShell script automates the migration of repositories from **TFS TFVC (Team Foundation Version Control)** to **GitHub**. It reads project and repository details from a JSON file, creates corresponding repositories on GitHub, clones the TFVC repositories, and pushes them to GitHub.

**Prerequisites**

1. **Personal Access Tokens (PATs):**

To authenticate programmatic access to TFS and GitHub, you need two PATs:

* + A TFS PAT with sufficient permissions to access and clone repositories.
  + A GitHub PAT with permissions to create repositories and push code. (scopes: repo, admin:repo\_hook, etc.).

⚠️ It’s a best practice to avoid hardcoding PATs. Instead, pull them securely from environment variables or a GitHub secrets store.

1. **Git-TFS Tool:**

Ensure git-tfs is installed and available in the system's PATH. git-tfs is an open-source tool that bridges TFVC and Git. It enables you to clone a TFVC repository, preserving history and converting it into a Git repository.

* Chocolatey (Windows): choco install gittfs
* Manual download: <https://github.com/git-tfs/git-tfs>

1. **JSON File:**

Prepare a JSON file containing project and repository details, such as: (e.g., projectRepoDetailsTFS2017.json).

Example

A screenshot of a computer program

AI-generated content may be incorrect.

This file drives the migration logic by listing all the repositories to process. You can generate this file using a helper script like **list-projects.ps1**

1. **Git Installation:**

Git must be installed and accessible from the command line (git --version). It is used to handle Git operations after converting TFVC repositories.

**Script Workflow**

**1. Configuration**

* **TFS and GitHub PATs:**
  + Set the $tfsPat and $githubPat variables with your respective access tokens. These are required for authenticating TFS and GitHub APIs.
* **Paths:**
  + Update $ProjectListJsonPath with the correct path to the JSON file containing project and repository details.
  + To get the latest projects list and repository details please run the below script **list-projects.ps1 ([tfs-migration/list-projects.ps1 at main · cigna-group-infrastructure-services/tfs-migration](https://github.com/cigna-group-infrastructure-services/tfs-migration/blob/main/list-projects.ps1))**
  + Update $downloadedReposFolder with the desired directory for cloning repositories.

Consider modifying the script to read these from environment variables for better security.

**2. Read and Parse JSON**

* The script reads the JSON file specified in $ProjectListJsonPath and parses it into $projectData.

**3. Environment Setup**

* The script sets the TFS PAT as an environment variable (GIT\_TFS\_PAT), which git-tfs uses for authentication.
* It ensures the $downloadedReposFolder exists. If not, it creates the directory so cloned repositories have a valid destination.

1. **Process Projects and Repositories**
   * The script loops through each project in the JSON file:
   * Skips invalid or non-TFVC entries (e.g., empty or incorrectly formatted repoPath).
   * Extracts the repository name and sanitizes it (e.g., removing invalid characters for GitHub).
   * Constructs URLs and folder names for use in the next steps.

**5. Create GitHub Repository**

* For each TFVC repository, the script uses the GitHub REST API to create a new repository
* It logs whether the repo creation was successful or if an error occurred (e.g., name already exists, invalid auth).

**6. Clone TFVC Repository**

Using git-tfs, the script clones the TFVC repo:

* git tfs clone "$tfsCollectionUrl" "$repoPath" "$localRepoFolder"

This step fetches all changesets and history from TFVC and builds a Git repository locally. Errors during this step are logged.

**Note**: Large TFVC repositories may take considerable time to clone, especially if history is deep.

**7. Push to GitHub and clean up**.

* After converting it to Git, the script pushes the repo to GitHub
* After successful migration “git tfs cleanup” is used to remove TFVC metadata from the local Git repo.