Azure DevOps-3

✓

Points: 1/1

Azure Pipelines provides several types of triggers to configure how your pipeline starts.

- Scheduled triggers start your pipeline based on a schedule, such as a nightly build. This article provides guidance on using scheduled triggers to run your pipelines based on a schedule.
- Event-based triggers start your pipeline in response to events, such as creating a pull request or pushing to a branch. For information on using event-based triggers, see Triggers in Azure Pipelines.

[✓ Yes] ✓

No

✓

Points: 1/1

Pull request validation (PR) triggers also vary based on the type of repository.

- PR triggers in Azure Repos Git
- PR triggers in GitHub
- PR triggers in Bitbucket Cloud

[✓ Yes] ✓

No

✓

Points: 1/1

Continuous integration (CI) triggers vary based on the type of repository you build in your pipeline.

- CI triggers in Azure Repos Git
- CI triggers in GitHub
- CI triggers in Bitbucket Cloud
- CI triggers in TFVC

[✓ Yes] ✓

No

✓

Points: 1/1

When running automated tests in the CI/CD pipeline, you may need a special configuration in order to run UI tests such as Selenium, Appium or Coded UI tests. This topic describes the typical considerations for running UI tests.

No

[✓ Yes] ✓

✓

Points: 1/1

GitHub and Azure Pipelines are two independent services that integrate well together. Each of them have their own organization and user management.

[✓ Yes] ✓

No

✓

Points: 1/1

Azure Pipelines can automatically build and validate every pull request and commit to your GitHub repository.

[✓ yes] ✓

No

✓

Points: 1/1

You may want to label your source code files to enable your team to easily identify which version of each file is included in the completed build. You also have the option to specify whether the source code should be labeled for all builds or only for successful builds.

[✓ Yes] ✓

No

✓

Points: 1/1

You can perform different forms of cleaning the working directory of your self-hosted agent before a build runs.

[✓ **Correct** ✓]

Wrong

✓

Points: 1/1

This setting is not configurable in the classic editor. Your source code will be checked out into a directory called `s`, which is relative to `$(Agent.BuildDirectory)`. For example: if `$(Agent.BuildDirectory)` is `C:\agent\_work\1`, then the source code will be checked out into `C:\agent\_work\1\mycustompath`.

[✓ **Correct** ✓]

Wrong

✓

Points: 1/1

While creating a pipeline, to choose the repository to build, first select the project to which the repository belongs. Then, select the repository.

To clone additional repositories as part of your pipeline:

- If the repo is in the same project as your pipeline, or if the access token (explained below) has access to the repository in a different project, use the following command:

  `git clone -c http.extraheader="AUTHORIZATION: bearer $(System.AccessToken)" <clone URL>`

  In order to use `System.AccessToken` in a script, you must first make it available to the script. To do this, select the job under the **Tasks** tab in the editor, select **Additional Options** in the right panel, and check the option to **Allow scripts to access the OAuth token**.

- If the access token (explained below) does not have access to the repository:

  1. Get a personal access token (PAT) with `Code (read)` scope, and prefix it with `pat:`
  2. Base64-encode this string to create a basic auth token.
  3. Add a script in your pipeline with the following command to clone that repo `git clone -c http.extraheader="AUTHORIZATION: basic <BASIC_AUTH_TOKEN>" <clone URL>`

[✓ **Yes** ✓]

No
