

# Creating starter workflows for your organization

## In this article

Overview

Creating a starter workflow

Next steps

Learn how you can create starter workflows to help people in your team add new workflows more easily.

## Overview

Starter workflows allow everyone in your organization who has permission to create workflows to do so more quickly and easily. When you create a new workflow, you can choose a starter workflow and some or all of the work of writing the workflow will be done for you. You can use starter workflows as a starting place to build your custom workflow or use them as-is. This not only saves time, it promotes consistency and best practice across your organization.

GitHub provides ready-to-use starter workflows for the following high level categories:

- **Deployment (CD)**. For more information, see "[About continuous deployment](#)."
- **Security**. For more information, see "[Configuring advanced setup for code scanning](#)."
- **Continuous Integration (CI)**. For more information, see "[About continuous integration](#)."
- **Automation**. Automation starter workflows offer solutions for automating workflows, such as triaging pull requests and applying a label based on the paths that are modified in the pull request, or greeting users who are first time contributors to the repository.

## Creating a starter workflow

Starter workflows can be created by users with write access to the organization's `.github` repository. These can then be used by organization members who have permission to create workflows.

**Note:** To avoid duplication among starter workflows you can call reusable workflows from within a workflow. This can help make your workflows easier to maintain. For more information, see "[Reusing workflows](#)."

This procedure demonstrates how to create a starter workflow and metadata file. The metadata file describes how the starter workflows will be presented to users when they are creating a new workflow.

- 1 If it doesn't already exist, create a new public repository named `.github` in your

organization.

- 2 Create a directory named `workflow-templates`.
- 3 Create your new workflow file inside the `workflow-templates` directory.

If you need to refer to a repository's default branch, you can use the `$default-branch` placeholder. When a workflow is created the placeholder will be automatically replaced with the name of the repository's default branch.

For example, this file named `octo-organization-ci.yml` demonstrates a basic workflow.

YAML

```
name: Octo Organization CI

on:
  push:
    branches: [ $default-branch ]
  pull_request:
    branches: [ $default-branch ]

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v4

      - name: Run a one-line script
        run: echo Hello from Octo Organization
```

- 4 Create a metadata file inside the `workflow-templates` directory. The metadata file must have the same name as the workflow file, but instead of the `.yml` extension, it must be appended with `.properties.json`. For example, this file named `octo-organization-ci.properties.json` contains the metadata for a workflow file named `octo-organization-ci.yml`:

JSON

```
{
  "name": "Octo Organization Workflow",
  "description": "Octo Organization CI starter workflow.",
  "iconName": "example-icon",
  "categories": [
    "Go"
  ],
  "filePatterns": [
    "package.json$",
    "^Dockerfile",
    ".*\\.md$"
  ]
}
```

- `name` - **Required.** The name of the workflow. This is displayed in the list of available workflows.
- `description` - **Required.** The description of the workflow. This is displayed in the list of available workflows.
- `iconName` - **Optional.** Specifies an icon for the workflow that is displayed in the list of workflows. `iconName` can one of the following types:

- An SVG file that is stored in the `workflow-templates` directory. To reference a file, the value must be the file name without the file extension. For example, an SVG file named `example-icon.svg` is referenced as `example-icon`.
- An icon from GitHub's set of [Octicons](#). To reference an octicon, the value must be `octicon <icon name>`. For example, `octicon smiley`.
- `categories` - **Optional.** Defines the categories that the workflow is shown under. You can use category names from the following lists:
  - General category names from the [starter-workflows](#) repository.
  - Linguist languages from the list in the [linguist](#) repository.
  - Supported tech stacks from the list in the [starter-workflows](#) repository.
- `filePatterns` - **Optional.** Allows the workflow to be used if the user's repository has a file in its root directory that matches a defined regular expression.

To add another starter workflow, add your files to the same `workflow-templates` directory.

## Next steps [↗](#)

---

To continue learning about GitHub Actions, see "[Using starter workflows](#)."

### Legal

© 2023 GitHub, Inc. [Terms](#) [Privacy](#) [Status](#) [Pricing](#) [Expert services](#) [Blog](#)