

Automating Monthly .NET Patching with GitHub Actions and dotnet-outdated



Martin Costello

Principal Engineer @ Just Eat Takeway.com

@martin_costello



@martincostello 🖸









- .NET Patching and Support Policy 📃
- The Problem 😩
- Automate the humans away! 🤖
 - GitHub Actions
 - Update .NET SDK Action and .NET release notes
 - dotnet outdated Global Tool
 - GitHub apps
- Q&A

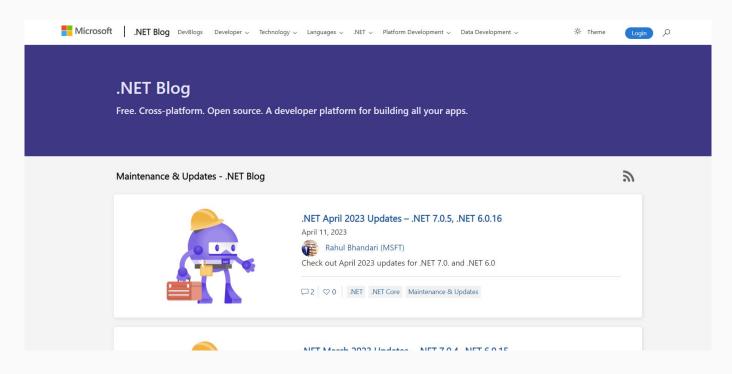


.NET Patching and Support Policy



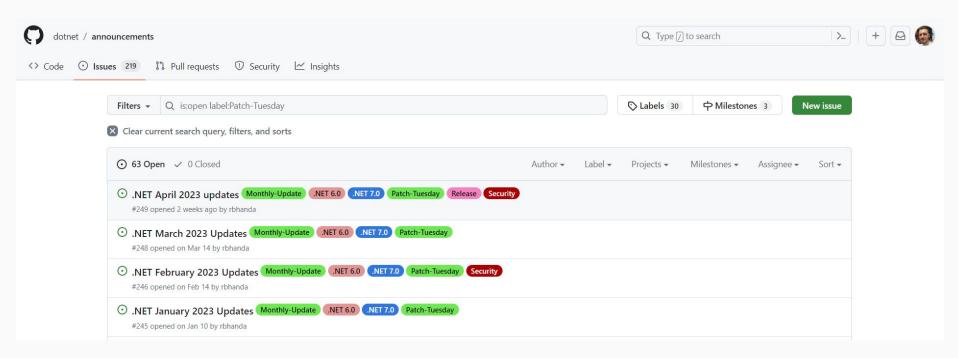
- Microsoft publishes a new .NET patch release (almost) every month
- Patches are released on the second Tuesday of the month
 - "Patch Tuesday"





devblogs.microsoft.com/dotnet





github.com/dotnet/announcements



- ~80% of monthly patches include security fixes
 - 21 of 27 .NET 6 releases to March 2024
 - 2 of 2 .NET 8 releases to March 2024
- Recent patched vulnerabilities include:
 - CVE-2024-21386 Denial of Service
 - CVE-2024-0057 Security Feature bypass
 - CVE-2023-36796 Remote Code Execution
 - CVE-2023-36049 Elevation of Privilege



You need to run the latest patch for support from Microsoft.

"Updates are cumulative and released as patches, with each update built upon all of the updates that preceded it. **A device** needs to install the latest update to remain supported. Updates may include new features, fixes (security and/or non-security), or a combination of both."

.NET and .NET Core Support Policy



The problem 😩



- Every application needs patching every month
 - Keep ahead of bad actors
 - Stay supported
- We only have so much engineering time
- Manual patching doesn't scale
- <u>Automated Updates</u> only apply to a subset of use cases
- Patching is repetitive
- Patching is boring



- Every application needs patching every month
 - Keep ahead of bad actors
 - Stay supported
- We only have so much engineering time
- Manual patching doesn't scale
- Automated Updates only apply to a subset of use cases
- Patching is repetitive
- Patching is boring







Homer Simpson - "Trash of the Titans" S9E22



Automate the humans away!



Solution

- Create a scheduled, automated GitHub Actions workflow
- Use open source tools to patch our repository
 - update-dotnet-sdk GitHub Action
 - dotnet outdated global tool
- Use GitHub apps to approve and merge the changes
- No extra code needs to be deployed to manage the process

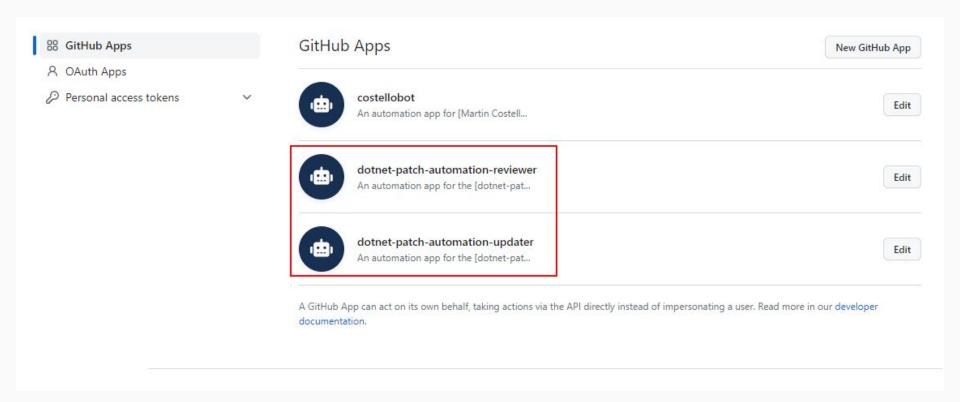


GitHub Actions Workflow

- Just 17 lines of YAML
- Runs weekly on a Cron schedule
- Can be run on-demand
- Uses a <u>reusable workflow</u>
- Minimal configuration

```
name: update-dotnet-sdk
on:
  schedule:
    - cron: '00 21 * * TUE'
 workflow_dispatch:
permissions:
  contents: read
jobs:
  update-sdk:
   uses: martincostello/update-dotnet-sdk/.github/workflows/update-dotnet-sdk.yml@v2
   with:
      labels: "dependencies,.NET"
      user-email: ${{ vars.GIT_COMMIT_USER_EMAIL }}
      user-name: ${{ vars.GIT_COMMIT_USER_NAME }}
    secrets:
      application-id: ${{ secrets.UPDATER_APPLICATION_ID }}
      application-private-key: ${{ secrets.UPDATER_APPLICATION_PRIVATE_KEY }}
```







Permissions

User permissions are granted on an individual user basis as part of the User authorization flow. Read our permissions documentation for information about specific permissions.

Changes to permissions will be applied to all future installations. Current users will be prompted to accept any changes and enable the new permissions on their installation.

Repository permissions 3 Selected

Repository permissions permit access to repositories and related resources.



Contents (i)

Repository contents, commits, branches, downloads, releases, and merges.

Access: Read and write ▼

Pull requests (i)

Pull requests and related comments, assignees, labels, milestones, and merges.

Access: Read and write ▼

Creating a GitHub app



Private keys

Generate a private key

You need a private key to sign access token requests. Learn more about private keys.



Private key

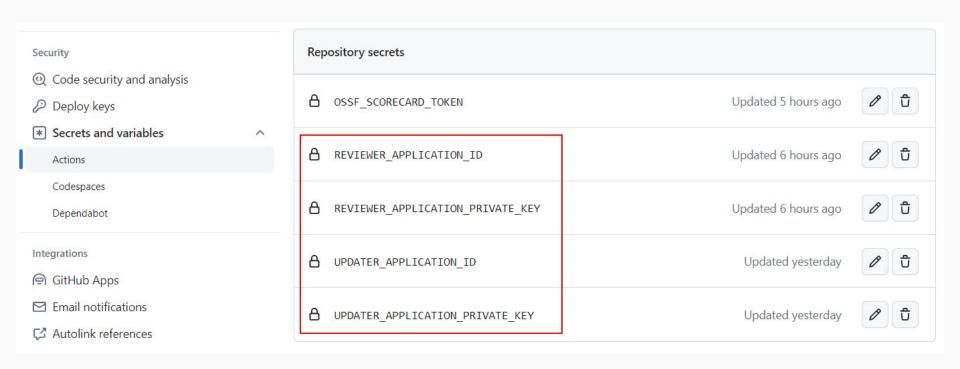
SHA256:I6B311hYl8BuDxIV+F5ugpJbor8Qb5yEcdaLQynZGv8=

Private Added 6 hours ago by martincostello

Delete

Creating a GitHub app

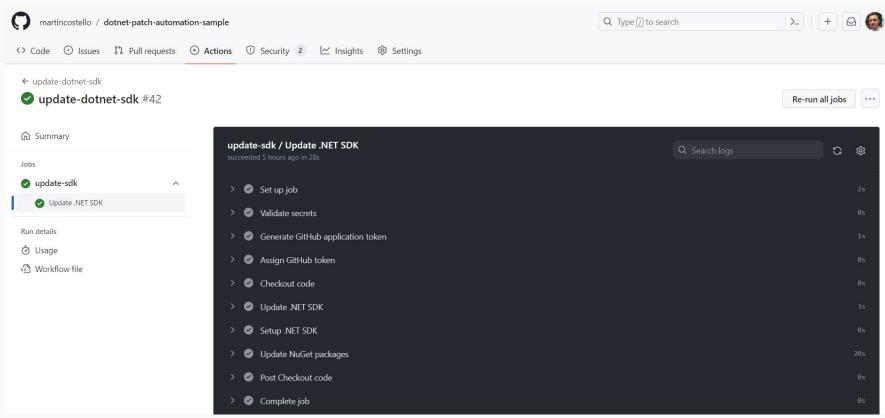




Encrypted secrets

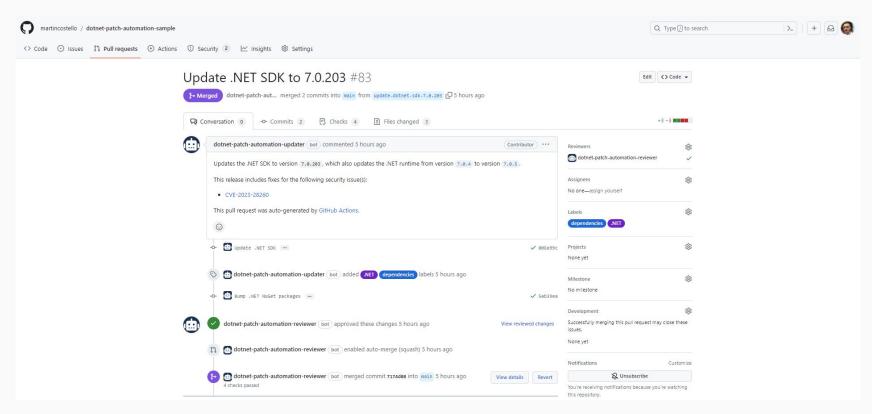
GitHub Actions





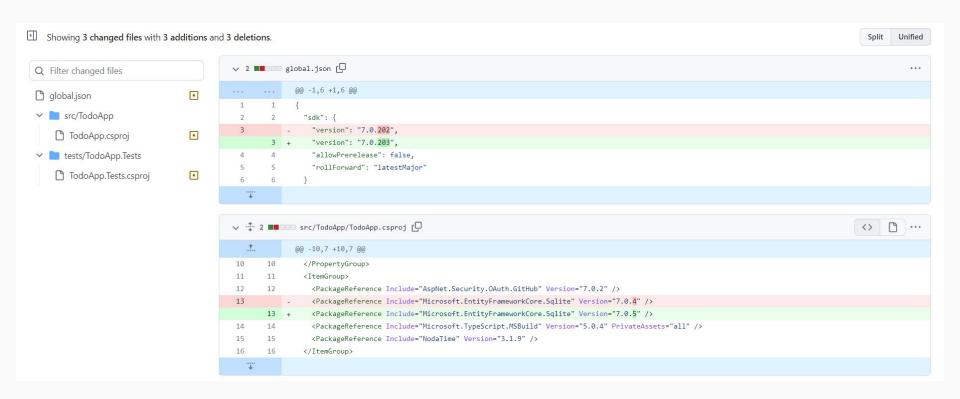
github.com/martincostello/dotnet-patch-automation-sample





github.com/martincostello/dotnet-patch-automation-sample



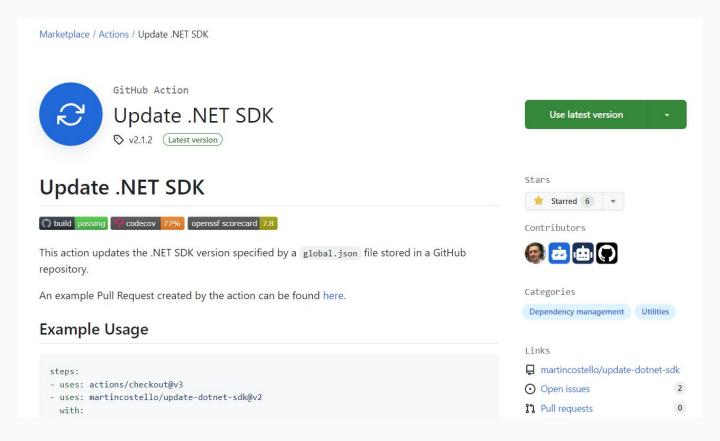


github.com/martincostello/dotnet-patch-automation-sample



How does it work?





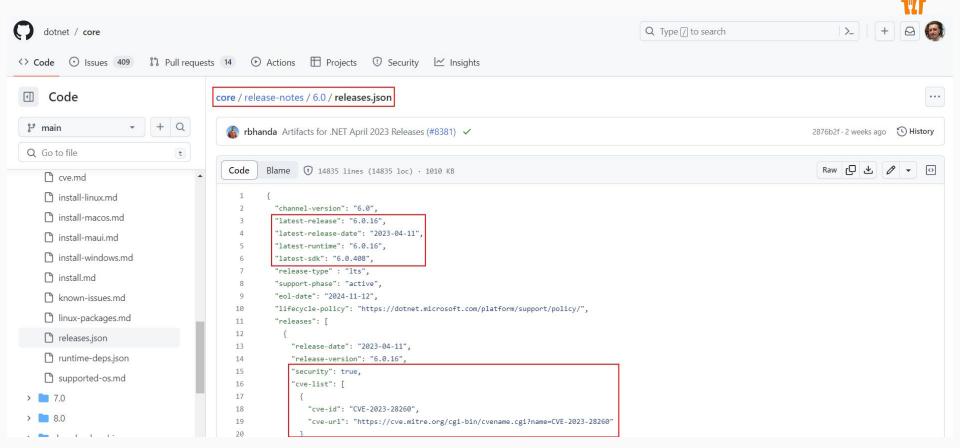


Update .NET SDK Action

- JavaScript action
- Checks the <u>.NET release notes</u>
- Updates global.json
- Uses the GitHub API to create (and label) a pull request

```
- name: Update .NET SDK
 id: update-dotnet-sdk
 uses: martincostello/update-dotnet-sdk@v2
 with:
    branch-name: ${{ inputs.branch-name }}
    channel: ${{ inputs.channel }}
    commit-message: ${{ inputs.commit-message }}
    dry-run: ${{ inputs.dry-run }}
    global-json-file: ${{ inputs.global-json-file }}
    labels: ${{ inputs.labels }}
    repo-token: ${{ steps.assign-token.outputs.access-token }}
    user-email: ${{ inputs.user-email }}
    user-name: ${{ inputs.user-name }}
```

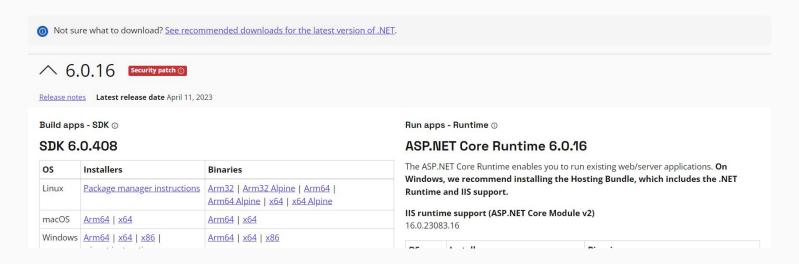
Update .NET SDK Action







Download .NET 6.0







- .NET global tool CLI application
- Published via NuGet package
- Checks NuGet for package updates
- Supports filtering by package and SemVer tags
- Updates <PackageReference>
 and <PackageVersion> tags

```
PowerShell Core
dotnet-patch-automation-sample (ca97a99) [!] via .NET v7.0.202
 dotnet outdated
» TodoApp
  [net7.0]
  Microsoft.EntityFrameworkCore.Sqlite 7.0.4 -> 7.0.5
» TodoApp.Tests
  [net7.0]
  Shouldly 4.2.0 -> 4.2.1
Version color legend:
<red> : Major version update or pre-release version. Possible breaking changes.
<yellow>: Minor version update. Backwards-compatible features added.
<green> : Patch version update. Backwards-compatible bug fixes.
You can upgrade packages to the latest version by passing the -u or -u:prompt option.
Elapsed: 00:00:02.9052013
dotnet-patch-automation-sample (ca97a99) [!] via .NET v7.0.202 took 3s
```



- .NET global tool CLI application
- Published via NuGet package
- Checks NuGet for package updates
- Supports filtering by package and SemVer tags
- Updates < Package Reference > and <PackageVersion> tags

```
PowerShell Core
dotnet-patch-automation-sample (ca97a99) via .NET v7.0.202
 dotnet outdated --upgrade --version-lock Major --include "Microsoft.AspNetCore."
  [net7.0]
  Microsoft.AspNetCore.Mvc.Testing 7.0.4 -> 7.0.5
Version color legend:
<red> : Major version update or pre-release version. Possible breaking changes.
<yellow>: Minor version update. Backwards-compatible features added.
 <green> : Patch version update. Backwards-compatible bug fixes.
Upgrading package Microsoft.AspNetCore.Mvc.Testing...
Project TodoApp.Tests [net7.0] upgraded successfully
Elapsed: 00:00:08.3794555
dotnet-patch-automation-sample (ca97a99) [!] via .NET v7.0.202 took 9s
```



- .NET global tool CLI application
- Published via NuGet <u>package</u>
- Checks NuGet for package updates
- Supports filtering by package and SemVer tags
- Updates <PackageReference>
 and <PackageVersion> tags

```
PowerShell Core
dotnet-patch-automation-sample (ca97a99) [!] via .NET v7.0.202
 git diff
diff --qit a/tests/TodoApp.Tests/TodoApp.Tests.csproj b/tests/TodoApp.Tests/TodoApp.Tests.csproj
index 1974f40..3d994ca 100644
--- a/tests/TodoApp.Tests/TodoApp.Tests.csproj
+++ b/tests/TodoApp.Tests/TodoApp.Tests.csproj
@@ -10,7 +10,7 @@
     <PackageReference Include="GitHubActionsTestLogger" Version="2.0.1" />
     <PackageReference Include="JustEat.HttpClientInterception" Version="4.0.0" />
     <PackageReference Include="MartinCostello.Logging.XUnit" Version="0.3.0" />
     <PackageReference Include="Microsoft.AspNetCore.Mvc.Testing" Version="7.0.5" />
     <PackageReference Include="Microsoft.NET.Test.Sdk" Version="17.5.0" />
     <PackageReference Include="Microsoft.Playwright" Version="1.32.0" />
     <PackageReference Include="ReportGenerator" Version="5.1.19" />
dotnet-patch-automation-sample (ca97a99) [!] via .NET v7.0.202
```



- .NET global tool CLI application
- Published via NuGet package
- Checks NuGet for package updates
- Supports filtering by package and SemVer tags
- Updates <PackageReference> and <PackageVersion> tags

```
PowerShell Core
dotnet-patch-automation-sample on / main via .NET v7.0.203
> dotnet outdated --upgrade --version-lock Major --include "Microsoft.AspNetCore."
No outdated dependencies were detected
Elapsed: 00:00:03.5705699
dotnet-patch-automation-sample on | main via .NET v7.0.203 took 4s
```



- .NET global tool CLI application
- Published via NuGet package
- Checks NuGet for package updates
- Supports filtering by package and SemVer tags
- Updates <PackageReference>
 and <PackageVersion> tags

```
$eligiblePackages = "${{ inputs.include-nuget-packages }}".Split(',')
$includePackages = @()
foreach ($package in $eligiblePackages) {
  $includePackages += "--include"
  $includePackages += $package
dotnet outdated `
  --upgrade
  --version-lock Major
  --output $updatesPath
  $includePackages
$dependencies = @()
if (Test-Path $updatesPath) {
  $dependencies = `
    Get-Content -Path $updatesPath |
    ConvertFrom-Json |
    Select-Object -ExpandProperty projects |
    Select-Object -ExpandProperty TargetFrameworks |
    Select-Object -ExpandProperty Dependencies | `
    Sort-Object -Property Name -Unique
```

- .NET global tool CLI application
- Published via NuGet package
- Checks NuGet for package updates
- Supports filtering by package and SemVer tags
- Updates <PackageReference>
 and <PackageVersion> tags

```
$commitLines = @()
$sdkVersion = "${{ steps.update-dotnet-sdk.outputs.sdk-version }}"
if ($deps.Count -eq 1) {
 $commitLines += "Bump $($deps[0].Name) from $($deps[0].ResolvedVersion) to $($deps[0].LatestVersion)"
  $commitLines += ""
  $commitLines += "Bumps $($deps[0].Name) from $($deps[0].ResolvedVersion) to $($deps[0].LatestVersion)."
} else {
  $commitLines += "Bump .NET NuGet packages"
  $commitLines += ""
  $commitLines += "Bumps .NET dependencies to their latest versions for the .NET $sdkVersion SDK."
  $commitLines += ""
  foreach ($dep in $deps) {
   $commitLines += "Bumps $($dep.Name) from $($dep.ResolvedVersion) to $($dep.LatestVersion)."
$commitLines += ""
$commitLines += "---"
$commitLines += "updated-dependencies:"
foreach ($dep in $deps) {
 $commitLines += "- dependency-name: $($dep.Name)"
 $commitLines += " dependency-type: direct:production"
  $commitLines += " update-type: version-update:semver-$($dep.UpgradeSeverity.ToLowerInvariant())"
$commitLines += "..."
$commitLines += ""
$commitLines += ""
$commitMessage = $commitLines -join "`n"
```



- .NET global tool CLI application
- Published via NuGet package
- Checks NuGet for package updates
- Supports filtering by package and SemVer tags
- Updates <PackageReference> and <PackageVersion> tags

```
✓ Bump Shouldly from 4.2.0 to 4.2.1

Bumps [Shouldly](https://github.com/shouldly/shouldly) from 4.2.0 to 4.2.1.

    [Release notes](https://github.com/shouldly/shouldly/releases)

- [Changelog](https://github.com/shouldly/shouldly/blob/master/BREAKING%20CHANGES.txt)
- [Commits](shouldly/shouldly@4.2.0...4.2.1)
updated-dependencies:
- dependency-name: Shouldly
  dependency-type: direct:production
 update-type: version-update:semver-patch
Signed-off-by: dependabot[bot] <support@github.com>
```

dependabot[bot] committed yesterday (Verified)



dotnet outdated global tool

- .NET global tool CLI application
- Published via NuGet package
- Checks NuGet for package updates
- Supports filtering by package and SemVer tags
- Updates < Package Reference > and <PackageVersion> tags

```
Bump .NET NuGet packages
Bumps .NET dependencies to their latest versions for the .NET 7.0.203 SDK.
Bumps Microsoft.AspNetCore.Mvc.Testing from 7.0.0 to 7.0.5.
```

Bumps Microsoft. EntityFrameworkCore. Sqlite from 7.0.0 to 7.0.5.

updated-dependencies:

dependency-name: Microsoft.AspNetCore.Mvc.Testing dependency-type: direct:production

update-type: version-update:semver-patch - dependency-name: Microsoft.EntityFrameworkCore.Sqlite

dependency-type: direct:production

update-type: version-update:semver-patch



- Uses another GitHub app to approve and auto-merge
- Only runs for PRs from the expected updater GitHub app
- Parses the commit messages to check only the expected changes have been made
- Copies approach from Dependabot's <u>fetch-metadata</u> action
- Leverages the <u>GitHub CLI</u> to perform API requests

```
name: approve-and-merge

on:
    pull_request:
        branches: [ main ]

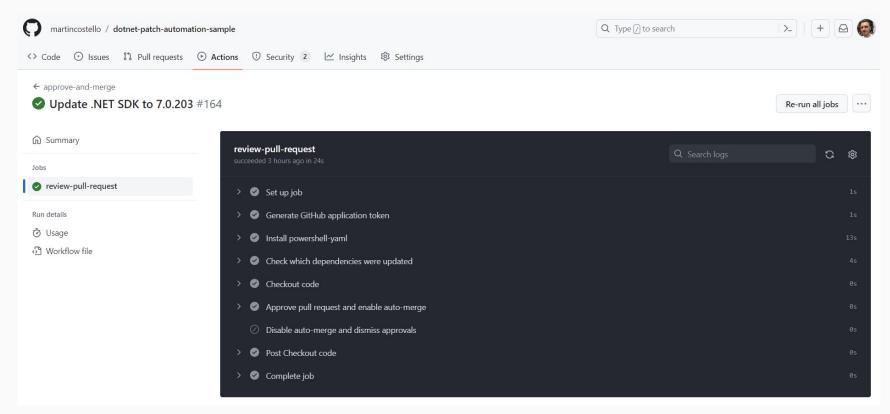
permissions:
    contents: read

jobs:
    review-pull-request:
        runs-on: ubuntu-latest

    if: ${{ github.event.pull_request.user.login == vars.GIT_COMMIT_USER_NAME }}
```

Approve and Merge with a GitHub app







- Uses another GitHub app to approve and auto-merge
- Only runs for PRs from the expected updater GitHub app
- Parses the commit messages to check only the expected changes have been made
- Copies approach from Dependabot's <u>fetch-metadata</u> action
- Leverages the <u>GitHub CLI</u> to perform API requests

```
$commits = qh api `
 --jq '.[] | { author: .author.login, message: .commit.message }' | ConvertFrom-Json
$expectedUser = "${{ vars.GIT_COMMIT_USER_NAME }}"
$onlyDependencyUpdates = $True
$onlyChangesFromUser = $True
dependencies = @()
foreach ($commit in $commits) {
 if ($commit.Author -ne $expectedUser) {
   $onlyChangesFromUser = $False
  match = [Regex]::Match(scommit.Message, '(?m)^-{3}\s(?<dependencies>[\S|\s]*?)\s^\.{3}\s')
  if ($match.Success -eq $True) {
   $metadata = ($match.Value | ConvertFrom-Yaml -Ordered)
   $updates = $metadata["updated-dependencies"]
   if ($updates) {
     foreach ($update in $updates) {
       $dependencies += @{
         Name = $update['dependency-name'];
         Type = $update['update-type'];
  else {
   $onlyDependencyUpdates = $False
$isPatch = $dependencies.Length -gt 0
$onlyTrusted = $dependencies.Length -gt 0
$trustedPackages = $env:INCLUDE_NUGET_PACKAGES.Split(',')
foreach ($dependency in $dependencies) {
 $isPatch = $isPatch -And $dependency.Type -eq "version-update:semver-patch"
 $onlyTrusted = $onlyTrusted -And
     ($dependency.Name -eq "Microsoft.NET.Sdk") -Or
     (($trustedPackages | Where-Object { $dependency.Name.StartsWith($_) }).Count -gt 0)
$isTrusted = (($onlyTrusted -And $isPatch) -And $onlyChangesFromUser) -And $onlyDependencyUpdates
"is-trusted-update=$isTrusted" >> $env:GITHUB_OUTPUT
```

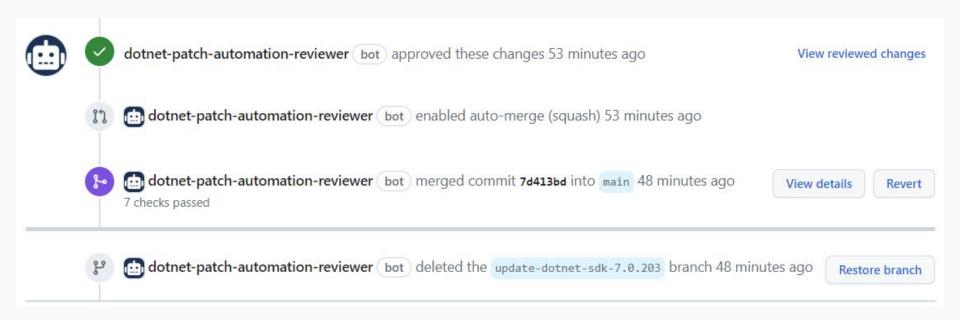


- Uses another GitHub app to approve and auto-merge
- Only runs for PRs from the expected updater GitHub app
- Parses the commit messages to check only the expected changes have been made
- Copies approach from Dependabot's <u>fetch-metadata</u> action
- Leverages the <u>GitHub CLI</u> to perform API requests

```
$approvals = gh api /repos/${{ github.repository }}/pulls/${{ github.event.pull_request.number }}/reviews | ConvertFrom-Json
$approvals = $approvals | Where-Object { $_.user.login -eq $env:REVIEWER_LOGIN }
$approvals = $approvals | Where-Object { $_.state -eq "APPROVED" }

if ($approvals.Length -eq 0) {
    gh pr checkout "$env:PR_URL"
    gh pr review --approve "$env:PR_URL"
    gh pr merge --auto --squash "$env:PR_URL"
}
else {
    Write-Host "PR already approved.";
}
```

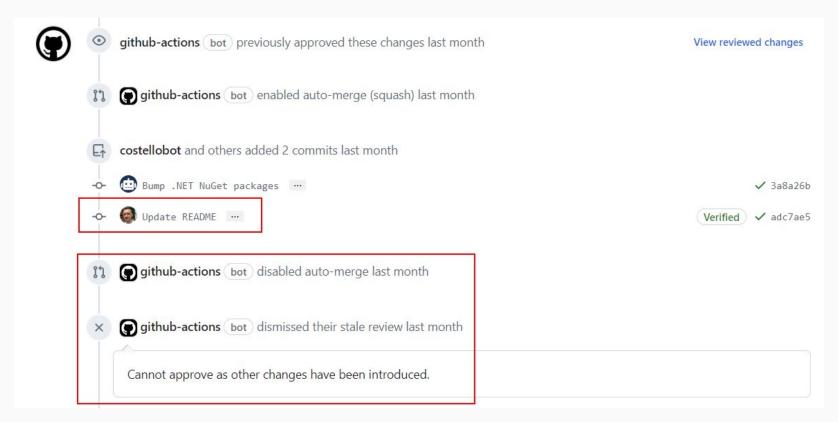






- Uses another GitHub app to approve and auto-merge
- Only runs for PRs from the expected updater GitHub app
- Parses the commit messages to check only the expected changes have been made
- Copies approach from Dependabot's <u>fetch-metadata</u> action
- Leverages the <u>GitHub CLI</u> to perform API requests







In Summary



- Create two GitHub apps:
 - Updater
 - Reviewer
- Add two workflow files:
 - o <u>Update</u>
 - Approve and merge
- Wait until Patch Tuesday



- Create two GitHub apps:
 - Updater
 - Reviewer
- Add two workflow files:
 - o <u>Update</u>
 - Approve and merge
- Wait until Patch Tuesday





Links

- Sample code <u>martincostello/dotnet-patch-automation-sample</u>
- Update .NET SDK Action <u>martincostello/update-dotnet-sdk</u>
- .NET Outdated <u>dotnet-outdated/dotnet-outdated</u>

Pull requests welcome!

- @martin_costello
- @martincostello







@martin_costello



@martincostello







