

The Gotchas: GitHub Actions Edition

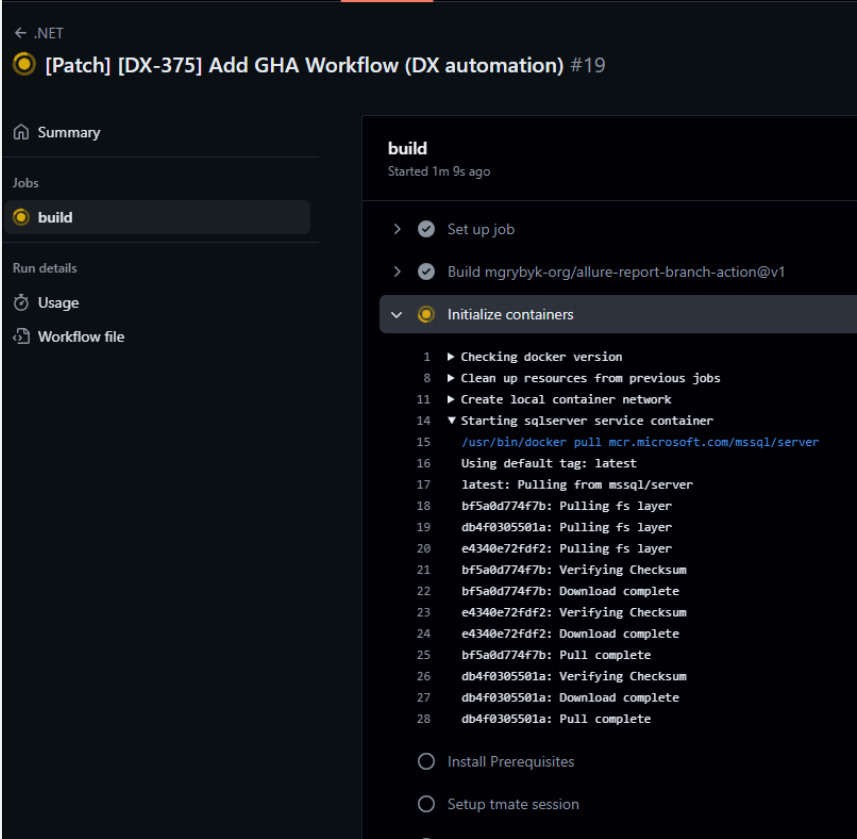
From Bede Gaming Ltd

Introduction

Here at Bede, we've been working on a GitHub Actions migration project for all of our microservices (okay, *most* of our microservices, I'm rounding up). In doing so we've obviously hit a bunch of snags which we will now share the solutions for in a handy table.

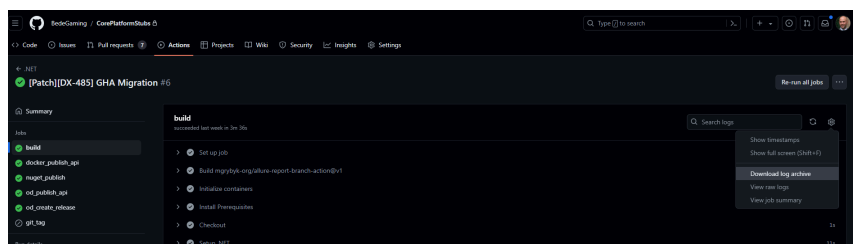
While it may feel personally to the author as if we have hit all the possible snags, it's likely that we've avoided some, but there should still be enough here that a Ctrl+F could help you out of a tricky situation.

Common Debugging & Troubleshooting

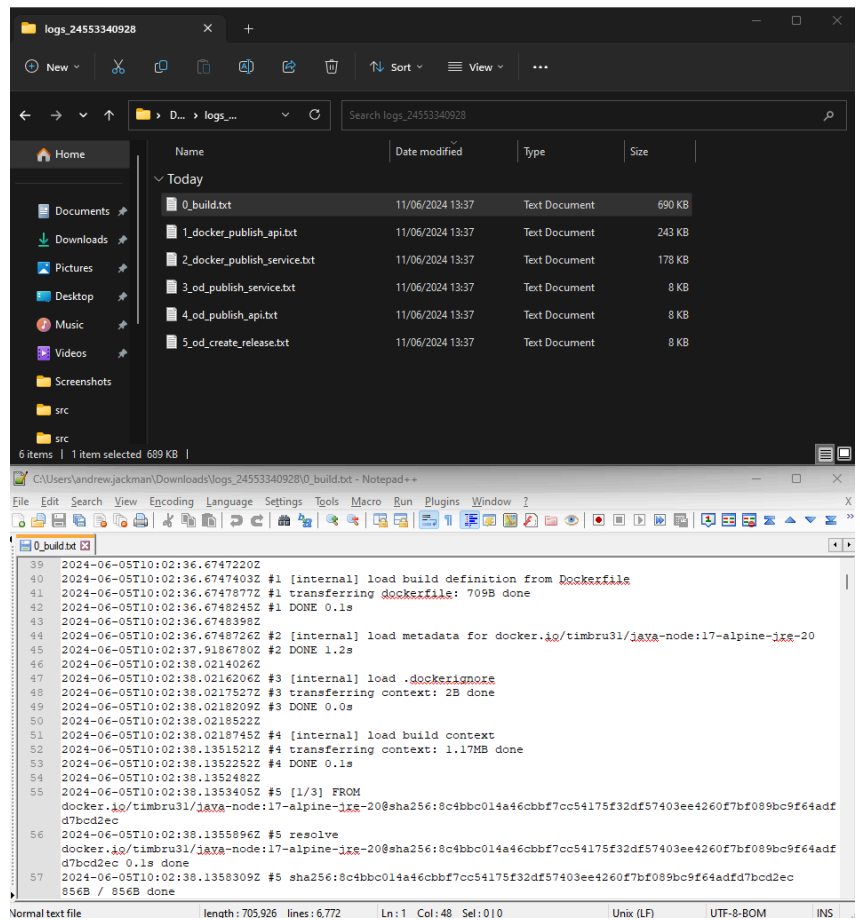
SUMMARY	EXAMPLE
<p>Logs can be accessed via the GitHub Actions UI, and the rendering is live and informative.</p> <p>You can see current progress with searching & downloading (see below) the raw log.</p>	<p>These can be accessed from the PR or the Actions tab.</p> 

These can also be revisited once the build has completed

Log Download - you can download the full log archive if the log is too big to be rendered.



Select job from build summary & download



Zip Archive by Build Step

SSH Login to Build Runner - normally when logs don't give you enough info to find the issue we can get BASH shell access to the arc-runner.

This will give you access to the OS file system, processes, build directory & you can make temporary changes to packages, permissions etc.

These won't be persisted so you'll need to add any required changes to the workflow steps.

This can be done by adding this temporary step to the workflow.yml normally before or after the problem you are experiencing:

```
# DELETE - get a ssh session at this point in the workflow
- name: Setup tmate session
  uses: mxschmitt/action-tmate@v3
```

Pushing this will trigger another GHA build but when it reaches the SSH step it will output the SSH command you need to use to connect to the logs. This will continue to loop until you connect then exit or cancel the build.

```
Setup tmate session

63 update-alternatives: warning: skip creation of /usr/share/man/man1/lzfgrep.1.gz
64 ssh rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
65
66 SSH: ssh rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
67 or: ssh -i <path-to-private-SSH-key> rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
68 SSH: ssh rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
69 or: ssh -i <path-to-private-SSH-key> rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
70 SSH: ssh rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
71 or: ssh -i <path-to-private-SSH-key> rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
72 SSH: ssh rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
73 or: ssh -i <path-to-private-SSH-key> rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
74 SSH: ssh rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
75 or: ssh -i <path-to-private-SSH-key> rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
76 SSH: ssh rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
77 or: ssh -i <path-to-private-SSH-key> rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
78 SSH: ssh rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
79 or: ssh -i <path-to-private-SSH-key> rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
80 SSH: ssh rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
81 or: ssh -i <path-to-private-SSH-key> rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
82 SSH: ssh rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
83 or: ssh -i <path-to-private-SSH-key> rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
84 SSH: ssh rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
85 or: ssh -i <path-to-private-SSH-key> rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
86 SSH: ssh rL4bntmebXF3z348mnLfJrmvh@lon1.tmate.io
```

Here's an example of an issue we were resolving with failing time zone tests where we checked currently installed time zones & installing tzdata temporarily to see if it fixed the issue:

```
MINGW64/c:/source/repos
bash: ll: command not found
runner@arc-runner-rgd7n-runner-9dzkc:/usr/share$ ls zoneinfo
ls: cannot access 'zoneinfo': No such file or directory
runner@arc-runner-rgd7n-runner-9dzkc:/usr/share$ sudo apt-get install tzdata
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  tzdata
0 upgraded, 1 newly installed, 0 to remove and 2 not upgraded.
Need to get 348 kB of archives.
After this operation, 3994 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 tzdata all 2024a-0ubuntu0.22.04
[348 kB]
Fetched 348 kB in 0s (2548 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package tzdata.
(Reading database ... 12707 files and directories currently installed.)
Preparing to unpack .../tzdata_2024a-0ubuntu0.22.04_all.deb ...
Unpacking tzdata (2024a-0ubuntu0.22.04) ...
Setting up tzdata (2024a-0ubuntu0.22.04) ...

Current default time zone: 'Etc/UTC'
Local time is now:    Wed Jun 12 07:41:01 UTC 2024.
Universal Time is now: Wed Jun 12 07:41:01 UTC 2024.
Run 'dpkg-reconfigure tzdata' if you wish to change it.

runner@arc-runner-rgd7n-runner-9dzkc:/usr/share$ ls zoneinfo
Africa      CST6CDT  Etc       Greenwich Kwajalein PRC       UCT       leapseconds
America     Canada  Europe    HST        Libya      PST8PDT  US        localtime
Antarctica  Chile  Factory   Hongkong  MET        Pacific  UTC       posix
Arctic      Cuba   GB         Iceland   MST        Poland   Universal posixrules
Asia        EET    GB-Eire    Indian     MST7MDT    Portugal W-SU      right
Atlantic    EST    GMT        Iran       Mexico     ROC      WET       tzdata.zi
Australia   ESTSEDT GMT+0      Israel     NZ         ROK      Zulu      zone.tab
Brazil      Egypt  GMT-0      Jamaica   NZ-CHAT    Singapore iso3166.tab zone1970.tab
CET         Eire    GMT0       Japan      Navajo     Turkey   leap-seconds.list zonenow.tab
0:bash*
```

Exiting the SSH session will then allow the GHA workflow to continue onto the next step.

You can also Cancel workflow to kill the build otherwise it'll continue to loop.

Add additional containers to support integration tests

Commonly we use a SQL Server container instance to support our integration tests. Some integration tests also require a Redis instance and there might be more prerequisites in the future.

These are fairly simple to add and with a corresponding change to transform the `appsettings.json` so that the component points to the container version of the service.

 [Creating Redis service containers - GitHub Docs](#)

 [Creating PostgreSQL service containers - GitHub Docs](#)

Install Linux packages - the build runner is a lightweight Linux distribution & might be missing some of the critical OS dependencies to build & test the component.

We have seen failing tests related to time zone related tests, and this was because .NET uses the time zones installed on the OS.

This was resolved by installing the tzdata package via `sudo apt-get install tzdata` in the step Install Prerequisites



Don't assume that these will be automatically installed onto K8 pods, this should be tested separately once the component has been deployed.

Build Steps - The `workflow.yml` is built up of common steps from [GitHub Actions Marketplace](#) and custom Bede actions.

The source code for all these steps is readily available if the logging isn't sufficient. Normally contained in an `action.yml` you can take a look at what the step is doing in more detail e.g. [action.yml](#)

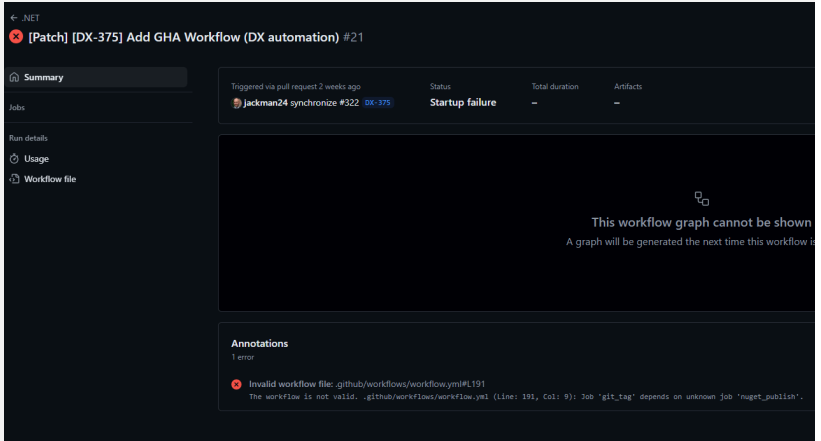
Similar to API versioning we can support change by releasing a new version of the action.

Common issues with steps from the marketplace steps should bring back search results in GitHub or Stack Overflow.

Failing & unstable tests should be a lot easier to debug in GHA 🙌

1. As mentioned above, logging is improved
2. Run the test command locally outside of the VS test runner e.g. `dotnet test --no-build -c Release --logger "trx;LogFileName=test-results.trx;verbosity=detailed" --collect:"XPlat Code Coverage;Exclude=[Test]%2c[]Swagger" -m:1`
3. Add additional temporary debug to the console so that it appears in GHA logs - `Debug.WriteLine("");`
4. Test & code coverage reporting should allow trends overtime to spot slow performing or flakey tests

Common Setup Issues

SUMMARY	Fix
Missing build steps on the PR is normally due to invalid <code>workflow.yml</code> file.	<p>You can still access the failed build via the Actions tab & it'll point you to the syntax error in the <code>workflow.yml</code></p>  <p>Invalid Build Configuration</p>

Incorrect .NET Version?

"The current .NET SDK does not support targeting .NET 7.0. Either target .NET 6.0 or lower, or use a version of the .NET SDK that supports .NET 7.0."

Update to the correct version required for this solution:

```
- name: Setup .NET
  uses: actions/setup-dotnet@v4
  with:
    dotnet-version: |
      7.0.x
  env:
    DOTNET_INSTALL_DIR: ../
```

Failing Integration Tests - SQL connection issues connecting to the SQL Container.

```
# Transform the settings and run tests after creating the package
# so that packages contain the original appsettings
- name: Transform appsettings
  uses: BedeGaming/gha-rewrite-json@v1
  with:
    input-file-pattern: './src/**/*.appsettings.json'
    json-data: '{
      "Bede.Configuration.Mode": "AppConfig",
      "Db.ConnectionStringTemplate": "Data Source=localhost,1433;
    }'
```

Transformation Step

- Ensure the integration tests are working locally
- Ensure that the transformation step is updating the `appsettings.json`

You can use the SSH debug following the transformation step to inspect the current values of the `appsettings.json` inside the integration test(s) project.


```
runner@arc-runner: /gd/n-runner-ws/j:/_work/Bede.Claims/Bede.Claims/src/Bede.Claims.Tests.Integration$ cat appsettings.json
{
  "Logging": {
    "LogLevel": {
      "Default": "Warning"
    }
  },
  "AllowedHosts": "*",
  "min:appstartUp": "Bede.Claims.Host-Startup",
  "Deployment.BuildVersion": "0.0.0.0",
  "CurrentEnvironment": "Local",
  "ServiceUrlClaims": "localhost:1433",
  "Db.ConnectionStringTemplate": "Data Source=localhost,1433;Initial Catalog=Test;User=sa;Password=Testing123;TrustServerCertificate=True;MultipleActiveResultSets=True",
  "Db.Name": "",
  "Db.User": "",
  "Db.Password": "",
  "Bede.Claims.Features.LoggingEnabled": "true",
  "Bede.Claims.Features.Patterns.Type": "A[\\w_~][{1,00}]$",
  "Bede.Configuration.Mode": "AppConfig",
  "Bede.Configuration.ConsulToken": "anonymous",
  "Bede.Configuration.ConsulAddress": "",
  "Bede.Configuration.ApplyKey": "Claims",
  "Bede.Configuration.HealthCheckEndpoint": "/api/ping",
  "Bede.Metrics.Enabled": "false",
}
```

SSH BASH example

This example uses cat but you can use less on larger files for easier navigation.

You can also install an editor if you wanted to update files in situ, for example with vim:

```
sudo apt-get update
sudo apt-get install vim -y
```

Failed or faulty transformation of the connection string is the most common cause of “Test host crashed”.

Make sure your connection string includes the parameter `TrustServerCertificate=True` if your service is .NET8 or above.

Ensure that the correct settings are being transformed - some services might have a `UseLocalDatabase` option that is intended for supplying a direct connection string for test runs.

No Code Coverage or Failing Code Coverage step.

.NET test projects require the package `coverlet.collector` to capture coverage data during the test build step with the following options:-

```
<PackageReference Include="coverlet.collector"
Version="6.0.2">
  <PrivateAssets>all</PrivateAssets>
  <IncludeAssets>runtime; build; native;
contentfiles; analyzers;
buildtransitive</IncludeAssets>
</PackageReference>
```

At time of writing, it is also recommended to upgrade the package `Microsoft.NET.Test.Sdk` to at least 17.9.0:

```
<PackageReference Include="Microsoft.NET.Test.Sdk"
Version="17.9.0" />
```

Failing “upload NuGet packages” step.

Not all components publish a contracts/clients/etc package. If you see this error, delete the step or change to a soft fail when no *.nupkg files exist for libraries.

Failing `dotnet restore --configfile "Nuget.Config"` step.

Sometimes there’s a mix of cases used for this file or sometimes the working directory is not set up correctly.

This file provides base config for custom NuGet sources. Importantly, this is case sensitive so any variation across components will cause issues.

You can either update the step or try to make the file consistent.

This file is often referenced in the Dockerfile of each application e.g. API & Service

Add a temporary SSH debug & run the command manually if it's not obvious what the issue is.

```
.NET - Restore
1 Run dotnet restore --configfile "NuGet.Config"
105 Determining projects to restore...
106 /home/runner/work/Bede.IdentityServer4.BosModule/Bede.IdentityServer4.BosModule/sdk/6.9.423/NuGet.targets(649,5): error : File
'/home/runner/work/Bede.IdentityServer4.BosModule/Bede.IdentityServer4.BosModule/src/Bede.IdentityServer4.BosModule/NuGet.Config' does not exist.
107 Error: Process completed with exit code 1.
```

Inconsistent *.csproj and Docker app names causing issues on the docker publish steps:

```
Run azure/docker-login@v1
DOCKER_CONFIG environment variable is set
Run IMAGE_TAG=***.azurecr.io/bede-claims-api:4.29.2-pr8269-008
ERROR: resolve : lstat Bede.Claims.Api: no such file or directory
Error: Process completed with exit code 1.

docker_publish_api:
  needs: build
  runs-on: arc-runner
  steps:
    - name: docker build and publish
      uses: BedeGaming/gha-docker-build-and-publish@v1
      with:
        username: ${ secrets.ACR_USERNAME }
        password: ${ secrets.ACR_PASSWORD }
        dockerfile: Bede.Claims.Host/Dockerfile
        image-name: bede-claims-api
        image-version: ${ needs.build.outputs.build-version }
```

Ensure the image and project are defined correctly and consistently throughout the workflow file and re-run.

Docker might use -api whereas projects might have a mix of .Host and .Api in the project names.

In some scenarios, pointing to the wrong project could result in empty, undeployable packages that give no further information.

Long-running code coverage step & errors based on reaching size limits.

WCF references in the solution can throw code coverage into meltdown and so exclusions need to be added for WCF namespaces on the test command:

```
dotnet test --no-build -c Release --logger
"trx;LogFileName=test-results.trx;verbosity=detailed" --collect:"XPlat Code
Coverage;Exclude=[*Test*]*%2c[*]*Swagger*%2c[*]Bede
.Providers.WcfProvider.*%2c[*]" -m:1 -p:Version=${{
steps.semver.outputs.version }}
```

Filter Swagger Code to Improve Code Coverage

There are several different filters that have been used previously but for coverlet.collector we must use this exclusion filter to exclude certain code from coverage analysis.

The test step should include Exclude option within collect switch e.g.

```
dotnet test Bede.Cashier.sln --no-build -c Release --logger
"trx;LogFileName=test-results.trx;verbosity=detailed"
--collect:"XPlat Code
Coverage;Exclude=[*Test*]*%2c[*]*Swagger*" -m:1
-p:Version=${{ steps.semver.outputs.version }}
```

Missing gh-pages branch - this branch is used for publishing test & code coverage reports.

This lives in the repository but shouldn't be tracking develop code & the only files present should be test & code coverage reports.

This will remove any report history as well:

```
git checkout --orphan gh-pages
git rm -rf .
git commit --allow-empty -m "gh-pages initial
commit"
git push --force --set-upstream origin gh-pages
```