specsolutions
given.when.then.

Self-service quality

Everything I learned from open-source development that is applicable to enterprise development as well

HUSTEF

6th October, 2022

HUSTEF
HUNGARIAN SOFTWARE TESTING FORUM

Gáspár Nagy
coach • trainer • bdd addict • creator of specflow
"The BDD Books" series • http://bddbooks.com
@gasparnagy • gaspar@specsolutions.eu

This talk is…

…not about that you should use open-source stuff – although that's a good advice

…not about open-source business economics – although that is a great topic

This talk is about
stealing the secret of successful open-source projects for our own enterprise projects

bddaddict.com

CAUTION!

bdd addict

on the stage

given.when.then

Gáspár Nagy

coach, trainer and bdd addict
creator of SpecFlow

gaspar@specsolutions.eu

https://specsolutions.eu

@gasparnagy

specsolutions

HUSTEF

Open-source vs Enterprise

- Random people
- No accountability
- Super-distributed
- Asynchronous communication
- Voluntary work
- Self-improvement
- Can produce quality work

- Employees in an org. hierarchy
- Accountability governed by contracts and law
- Co-located and distributed
- Synchronous and asynchronous communication
- Tasks assigned
- Coordinated learning & training
- Achieving expected quality is hard

How is this possible?

the key is…

Self-service

Approach

- Show differences and application of self-servicing by reviewing an open-source development process, by focusing on the quality
  - Making changes (pull requests & co)
  - Reach quality expectation of different quality aspects (CI/CD, review, etc.)
  - Handling support cases
  - Dealing with dependencies and releases
- Highlight ideas to steal

Imagine an open-source project

Where you are both user and contributor…

And keep comparing it with the project you work on,

where you are both user and author of the components & tools you develop

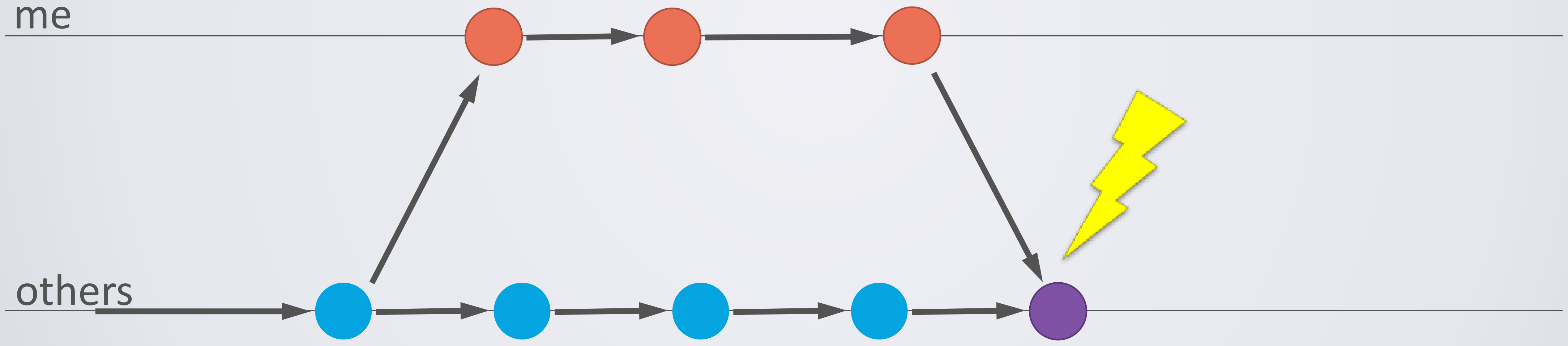
Making changes through pull requests

What is a pull request (PR)?

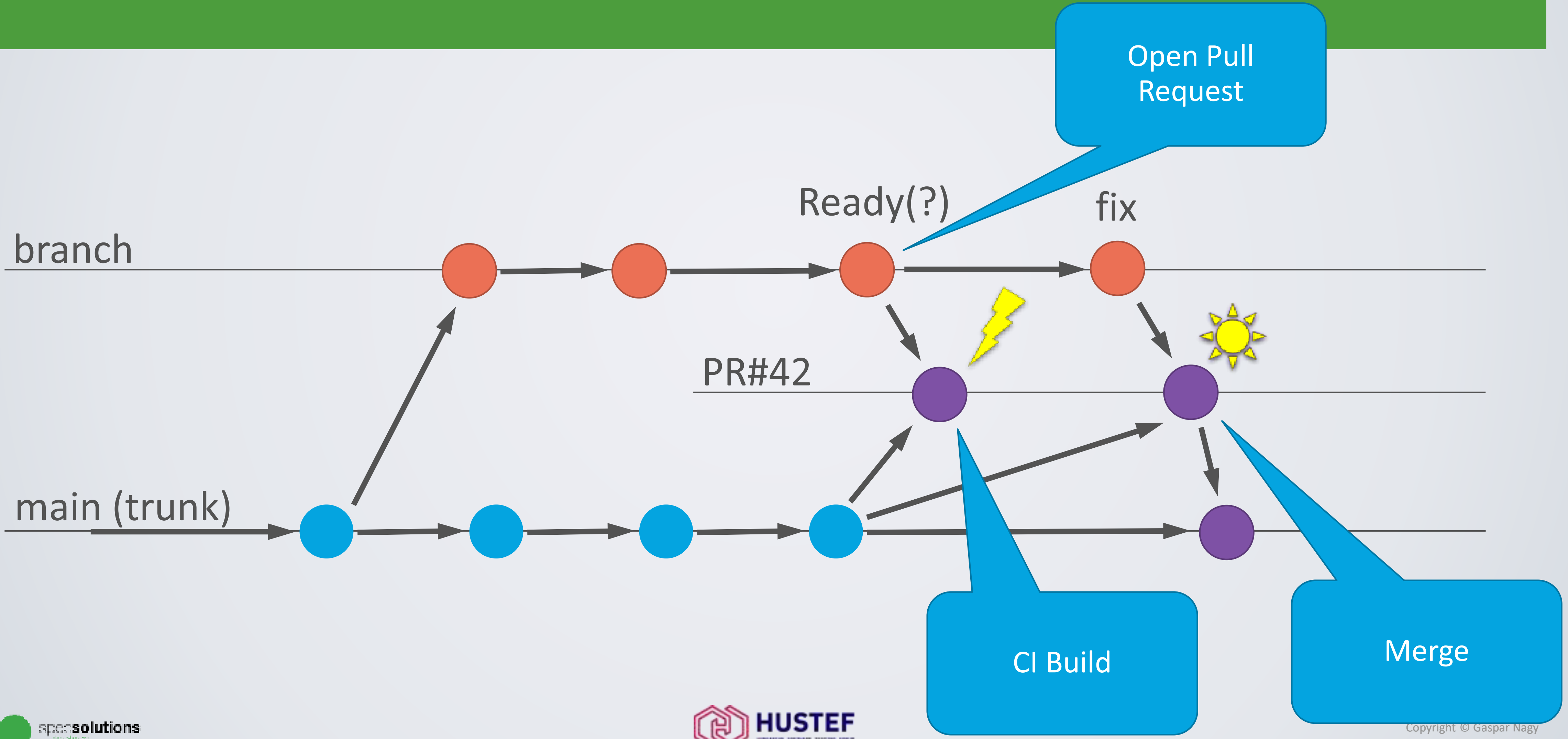
- “**proposed** changes” (GitHub)
- “a mechanism for a developer to **notify** team members that they have *completed* a feature” (Atlassian)

- A feature to support **integration** of independent changes with the main development line
- PR evolved from a feature to a **process** over the years
- This process has many **quality-related** aspects

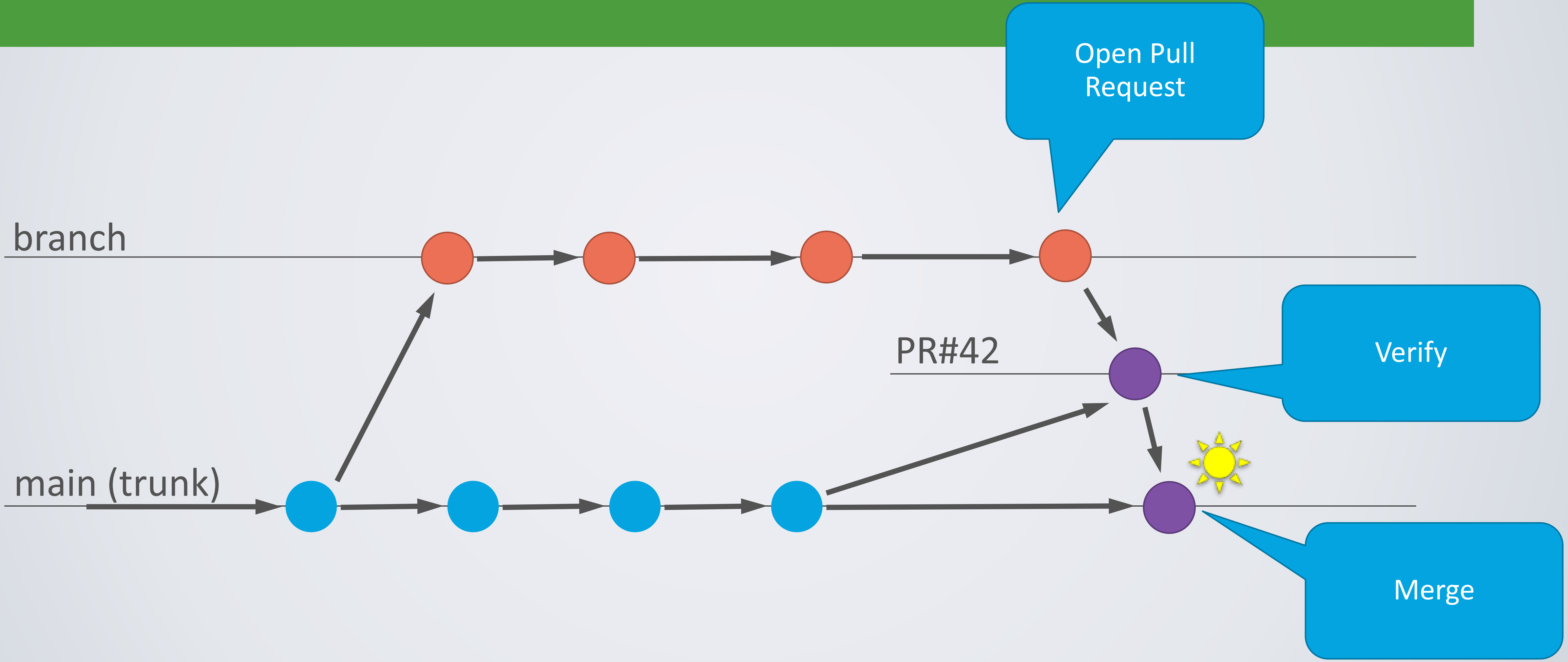
Integration – Merging parallel work



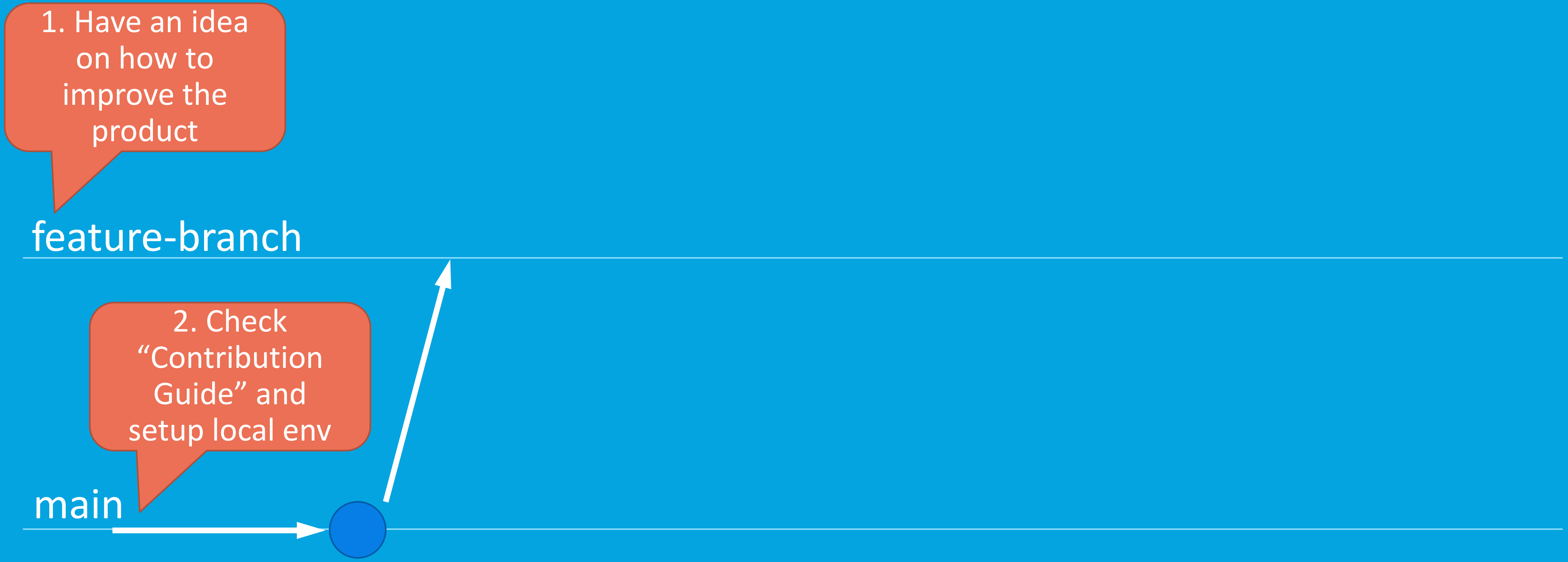
Pull Requests



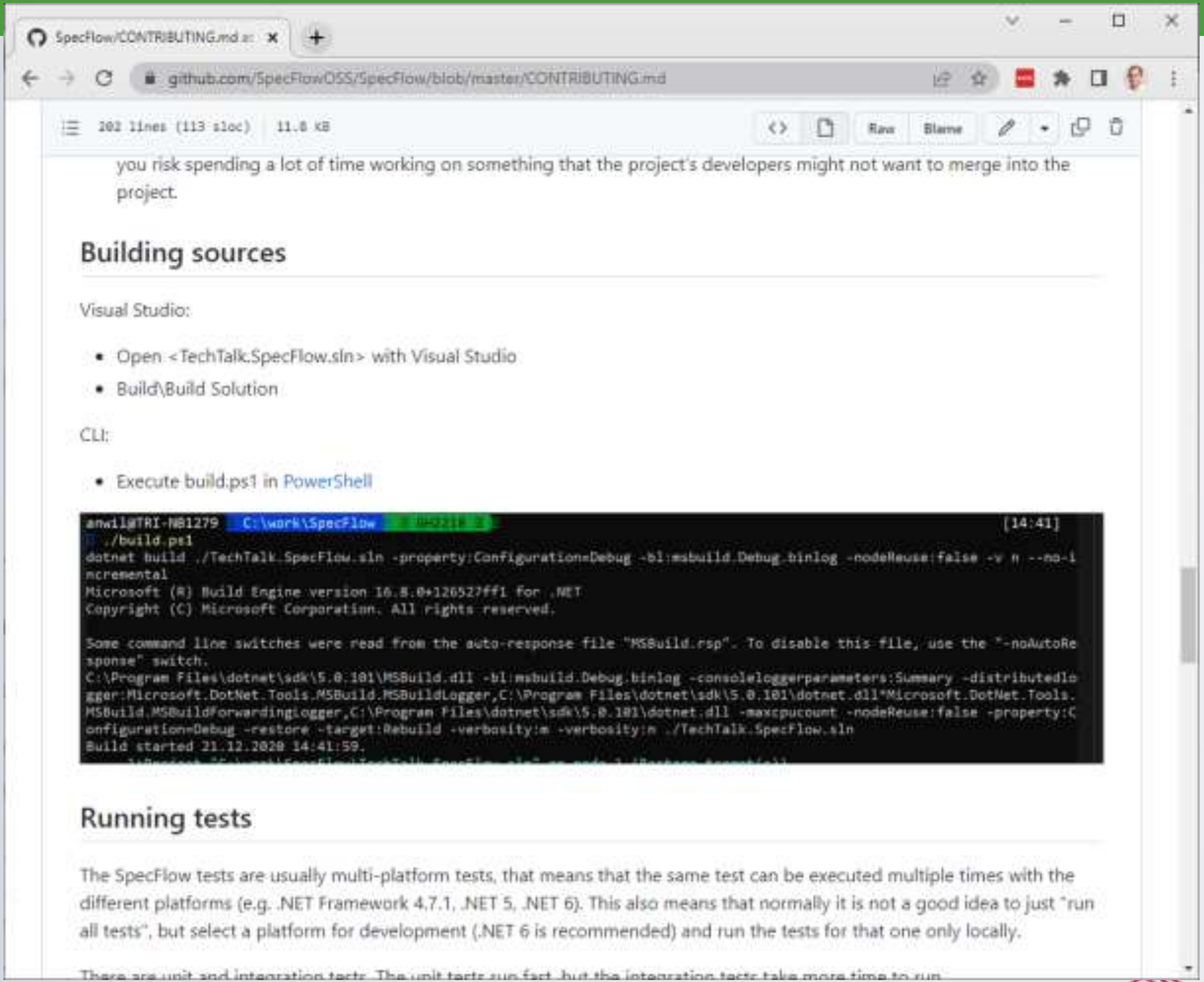
Pull Requests



Open-source pull request process


1. Have an idea on how to improve the product
feature-branch
2. Check "Contribution Guide" and setup local env
main

Idea: Contribution guide


SpecFlow/CONTRIBUTING.md
github.com/SpecFlowOSS/SpecFlow/blob/master/CONTRIBUTING.md
202 lines (113 sloc) 11.8 KB
Raw
Blame

you risk spending a lot of time working on something that the project's developers might not want to merge into the project.

Building sources

Visual Studio:

- Open <TechTalk.SpecFlow.sln> with Visual Studio
- Build\Build Solution

CLI:

- Execute build.ps1 in PowerShell

```
anwil@TRI-NB1279  C:\work\SpecFlow                                                    [14:41]
 ./build.ps1
dotnet build ./TechTalk.SpecFlow.sln -property:Configuration=Debug -bl:msbuild.Debug.binlog -nodeReuse:false -v n --no-i
ncremental
Microsoft (R) Build Engine version 16.8.0+126527ff1 for .NET
Copyright (C) Microsoft Corporation. All rights reserved.

Some command line switches were read from the auto-response file "MSBuild.rsp". To disable this file, use the "-noAutoRe
sponse" switch.
C:\Program Files\dotnet\sdk\5.0.101\MSBuild.dll -bl:msbuild.Debug.binlog -consoleloggerparameters:Summary -distributedlo
gger:Microsoft.DotNet.Tools.MSBuild.MSBuildLogger,C:\Program Files\dotnet\sdk\5.0.101\dotnet.dll*Microsoft.DotNet.Tools.
MSBuild.MSBuildForwardingLogger,C:\Program Files\dotnet\sdk\5.0.101\dotnet.dll -maxcpucount -nodeReuse:false -property:C
onfiguration=Debug -restore -target:Rebuild -verbosity:m -verbosity:n ./TechTalk.SpecFlow.sln
Build started 21.12.2020 14:41:59.
```

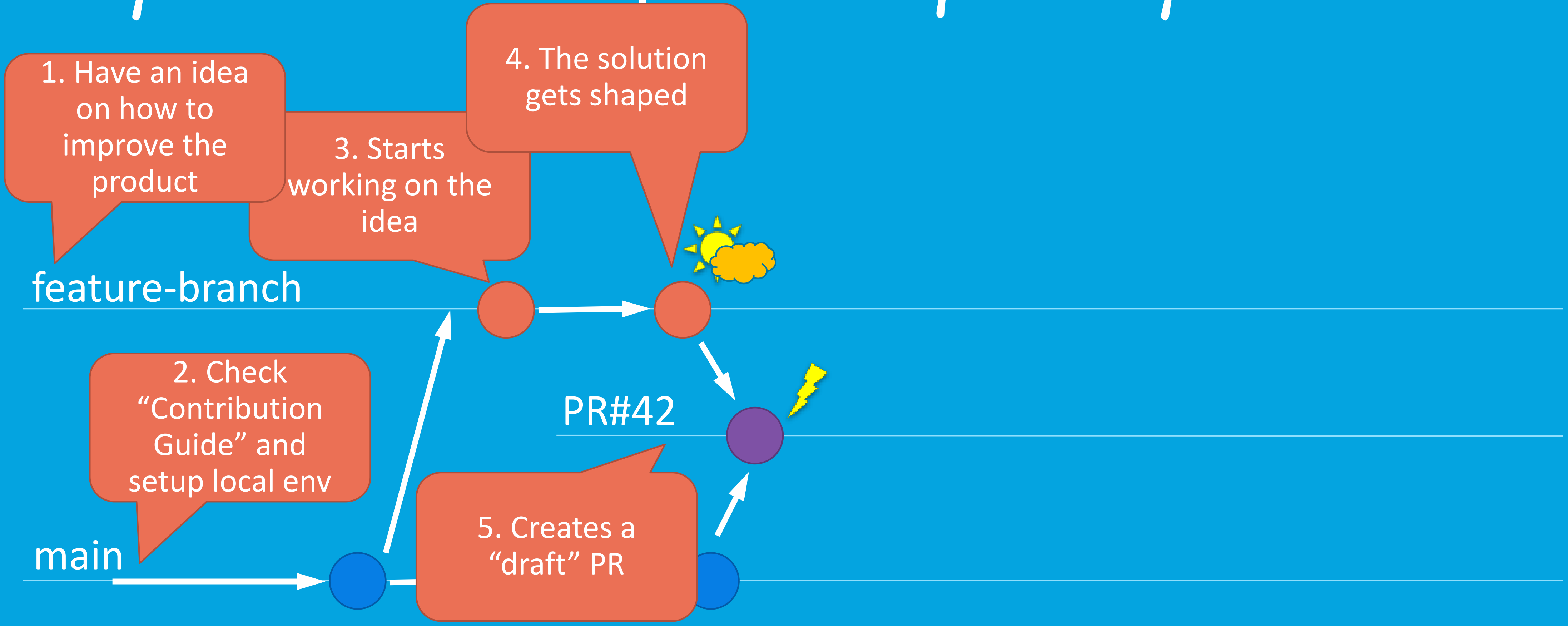
Running tests

The SpecFlow tests are usually multi-platform tests, that means that the same test can be executed multiple times with the different platforms (e.g. .NET Framework 4.7.1, .NET 5, .NET 6). This also means that normally it is not a good idea to just "run all tests", but select a platform for development (.NET 6 is recommended) and run the tests for that one only locally.

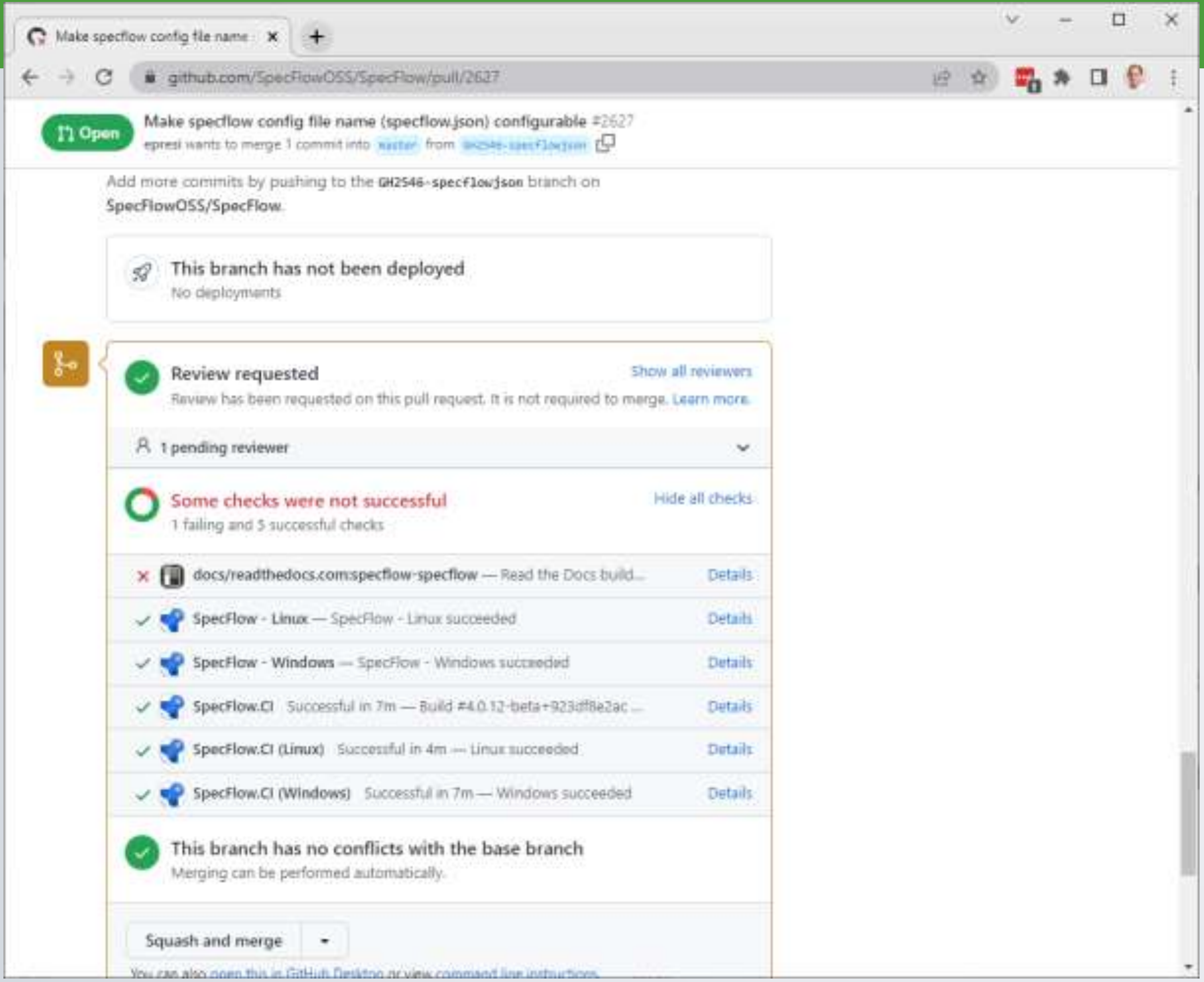
There are unit and integration tests. The unit tests run fast, but the integration tests take more time to run.

- An all-in-one documentation to get up to speed
- Easy to find: Standard location, included in the repository
- Also contains information about how to run the tests

Open-source pull request process


1. Have an idea on how to improve the product
3. Starts working on the idea
4. The solution gets shaped
feature-branch
2. Check "Contribution Guide" and setup local env
PR#42
5. Creates a "draft" PR
main

Idea: Pull Request concept


Make specflow config file name

github.com/SpecFlowOSS/SpecFlow/pull/2627

Open **Make specflow config file name (specflow.json) configurable** #2627

epresi wants to merge 1 commit into master from GH2546-specflowjson

Add more commits by pushing to the `GH2546-specflowjson` branch on SpecFlowOSS/SpecFlow.

This branch has not been deployed
No deployments

Review requested — Show all reviewers
Review has been requested on this pull request. It is not required to merge. Learn more.

1 pending reviewer

Some checks were not successful — Hide all checks
1 failing and 5 successful checks

✗	**docs/readthedocs.com:specflow-specflow** — Read the Docs build...	Details
✓	**SpecFlow - Linux** — SpecFlow - Linux succeeded	Details
✓	**SpecFlow - Windows** — SpecFlow - Windows succeeded	Details
✓	**SpecFlow.CI** Successful in 7m — Build #4.0.12-beta+923df8e2ac ...	Details
✓	**SpecFlow.CI (Linux)** Successful in 4m — Linux succeeded	Details
✓	**SpecFlow.CI (Windows)** Successful in 7m — Windows succeeded	Details

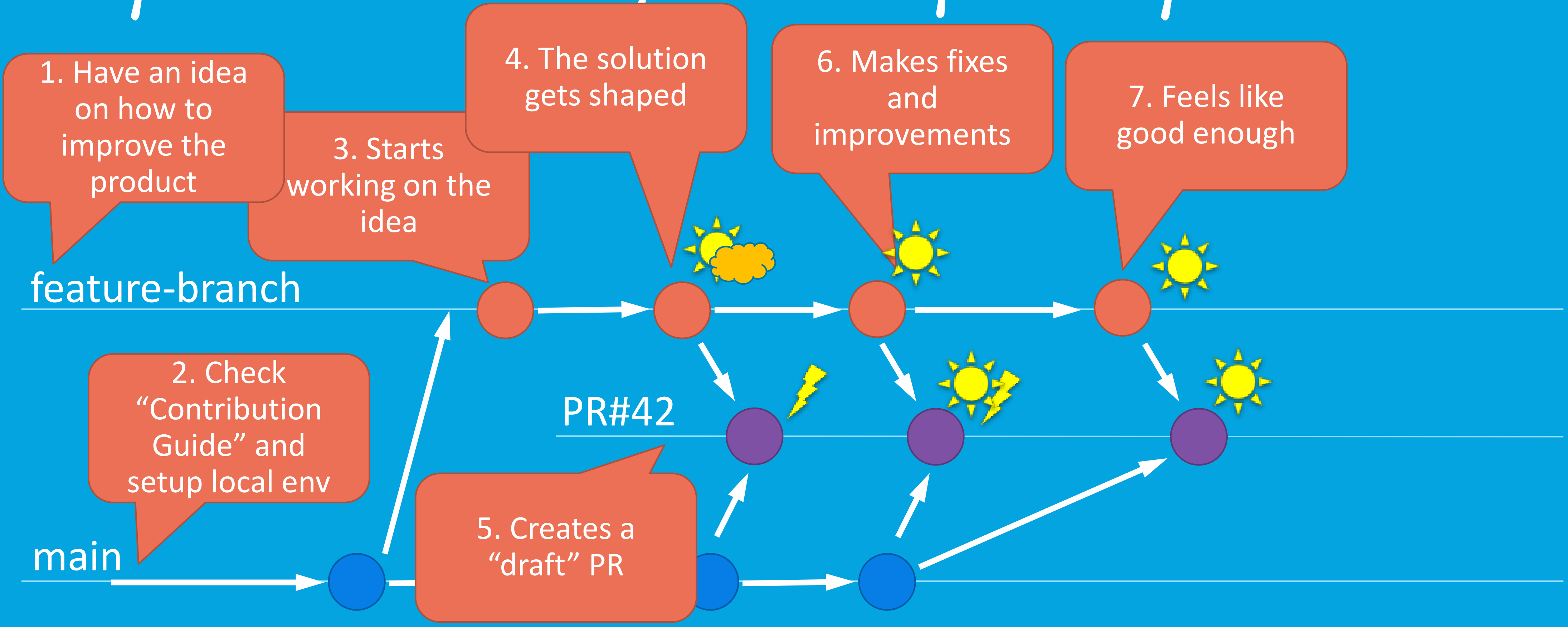
This branch has no conflicts with the base branch
Merging can be performed automatically.

Squash and merge

You can also open this in GitHub Desktop or view command line instructions.

- Pull requests can be used to pre-validate the solution (without being ashamed of a “public” failed build)
- They can be initiated (as “draft”) as soon as you have something to check – encourages early verification, small steps (commits)
- The ALM tool configures a private temporary branch and a private temporary CI pipeline to verify the PR – no configuration efforts needed
- The project admins can control how the PR pipeline should behave, but by default it just uses the CI pipeline as a template

Open-source pull request process


1. Have an idea on how to improve the product
2. Check "Contribution Guide" and setup local env
3. Starts working on the idea
4. The solution gets shaped
5. Creates a "draft" PR
6. Makes fixes and improvements
7. Feels like good enough
feature-branch
PR#42
main

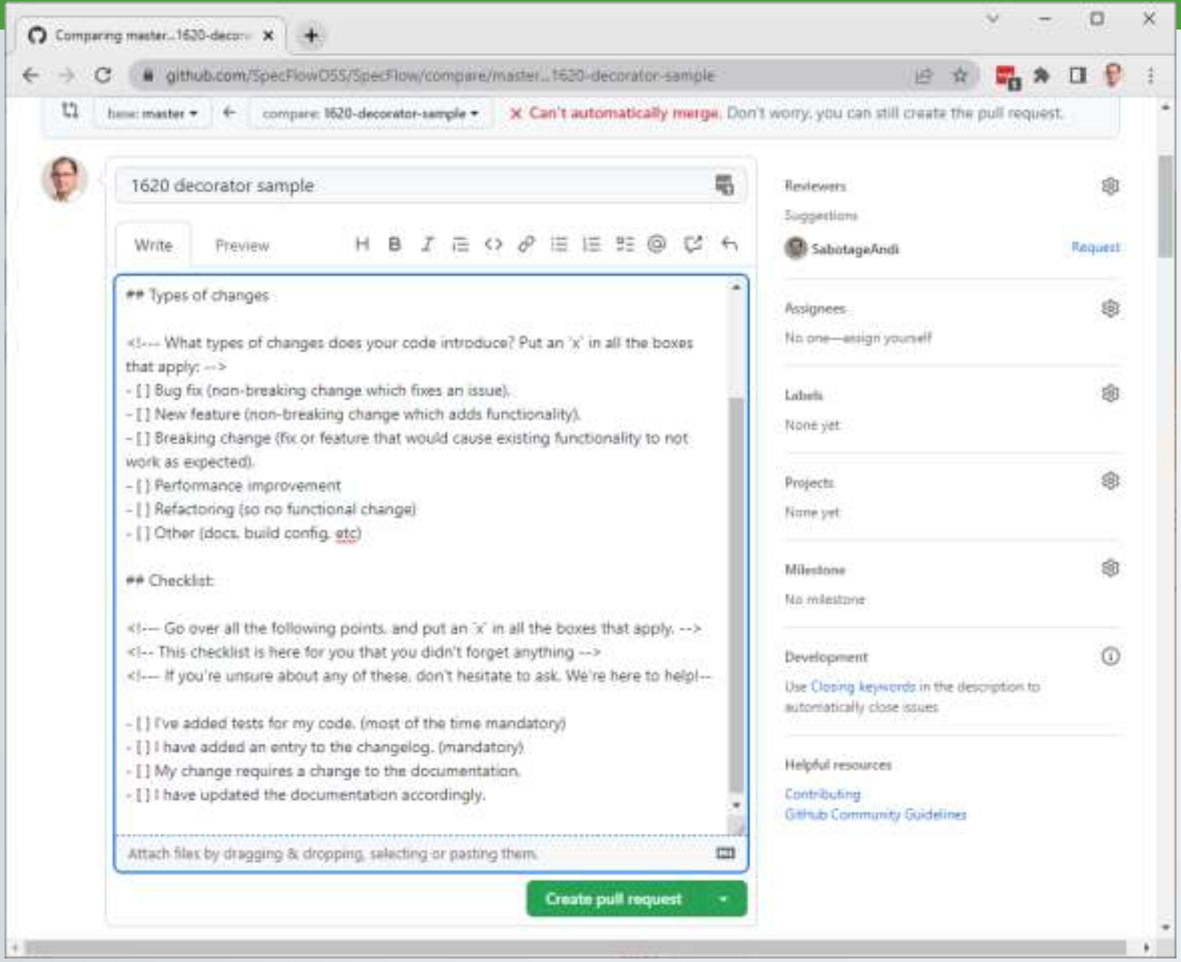
Many team members cannot contribute to quality
simply because they are hopeless with figuring out what is good enough

Breaking down the undefinable quality: quality aspects (sample)

Functional	Operational	Strategic
• Works as expected • Expectations are good • Expectations are documented	• Secure • Fast • Convenient • Pretty • Consistent • Predictable	• Maintainable • Architecture • Code quality • Easy to integrate • Flexible

Source: Kevlin Henney

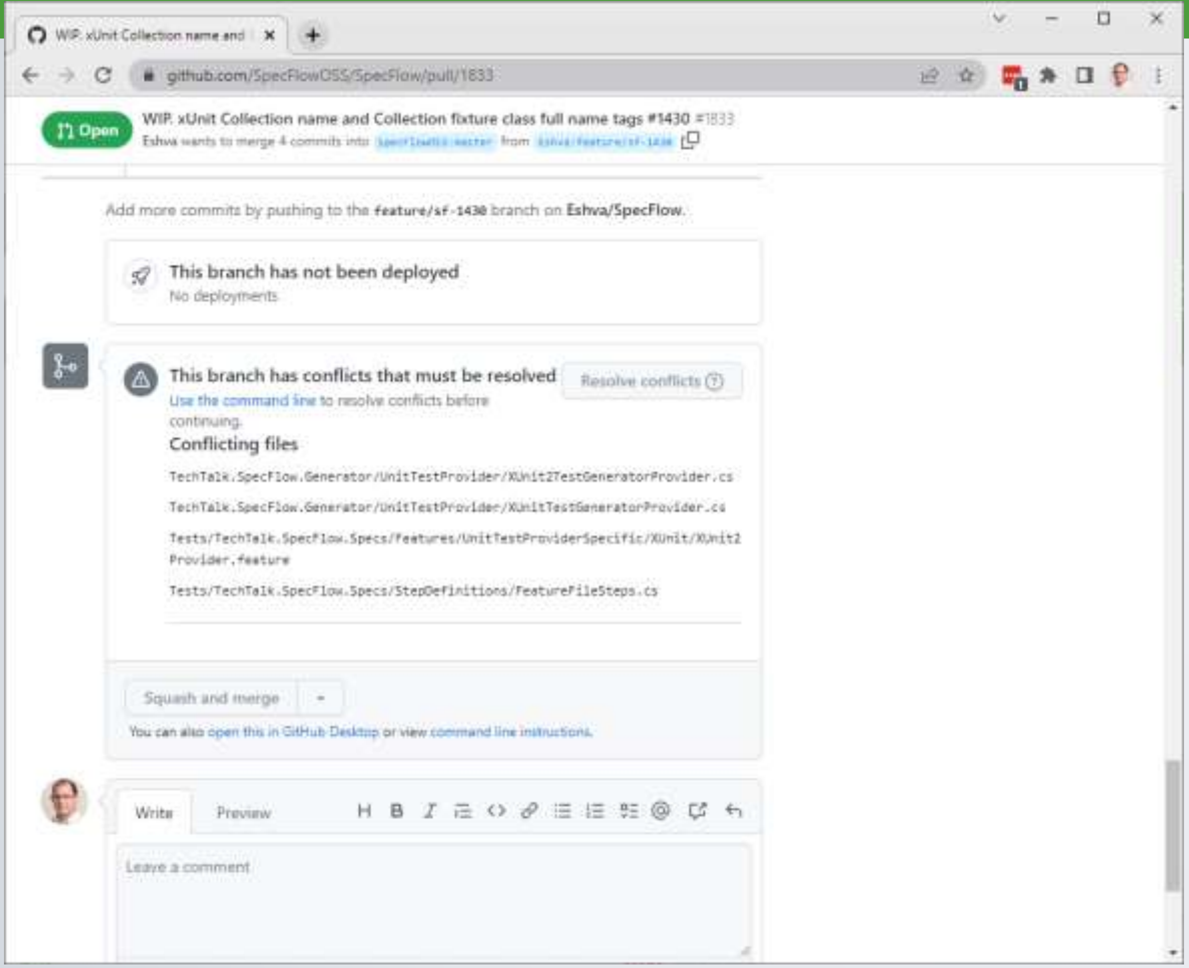
Idea: Quality checklist in PR template


Comparing master...1620-decor
github.com/SpecFlowOSS/SpecFlow/compare/master...1620-decorator-sample
base: master
compare: 1620-decorator-sample
Can't automatically merge. Don't worry, you can still create the pull request.
1620 decorator sample
Write
Preview
Reviewers
Suggestions
SabotageAndi
Request
Assignees
No one—assign yourself
Labels
None yet
Projects
None yet
Milestone
No milestone
Development
Use Closing keywords in the description to automatically close issues
Helpful resources
Contributing
Github Community Guidelines
Attach files by dragging & dropping, selecting or pasting them.
Create pull request

```
## Types of changes

<!--- What types of changes does your code introduce? Put an `x` in all the boxes
that apply: -->
- [ ] Bug fix (non-breaking change which fixes an issue).
- [ ] New feature (non-breaking change which adds functionality).
- [ ] Breaking change (fix or feature that would cause existing functionality to not
work as expected).
- [ ] Performance improvement
- [ ] Refactoring (so no functional change)
- [ ] Other (docs, build config, etc)

## Checklist:

<!--- Go over all the following points, and put an `x` in all the boxes that apply. -->
<!-- This checklist is here for you that you didn't forget anything -->
<!--- If you're unsure about any of these, don't hesitate to ask. We're here to help!--

- [ ] I've added tests for my code. (most of the time mandatory)
- [ ] I have added an entry to the changelog. (mandatory)
- [ ] My change requires a change to the documentation.
- [ ] I have updated the documentation accordingly.
```

- Some of the quality aspects can be automatically verified by the PR
- Some need human attention – the things that are typically got forgotten
- The PR template can be configured in a way that it asks the contributor to go over the checklist and reminds them to complete all necessary quality steps
- Other information, like the type of change, related issues or its potential impact can also be collected in the same way

Idea: Check in integration, not in isolation



github.com/SpecFlowOSS/SpecFlow/pull/1833

Open WIP: xUnit Collection name and Collection fixture class full name tags #1430 #1833
Eshva wants to merge 4 commits into SpecFlowOSS:master from Eshva:feature/sf-1430

Add more commits by pushing to the `feature/sf-1430` branch on Eshva/SpecFlow.

This branch has not been deployed
No deployments

This branch has conflicts that must be resolved Resolve conflicts
Use the command line to resolve conflicts before continuing.

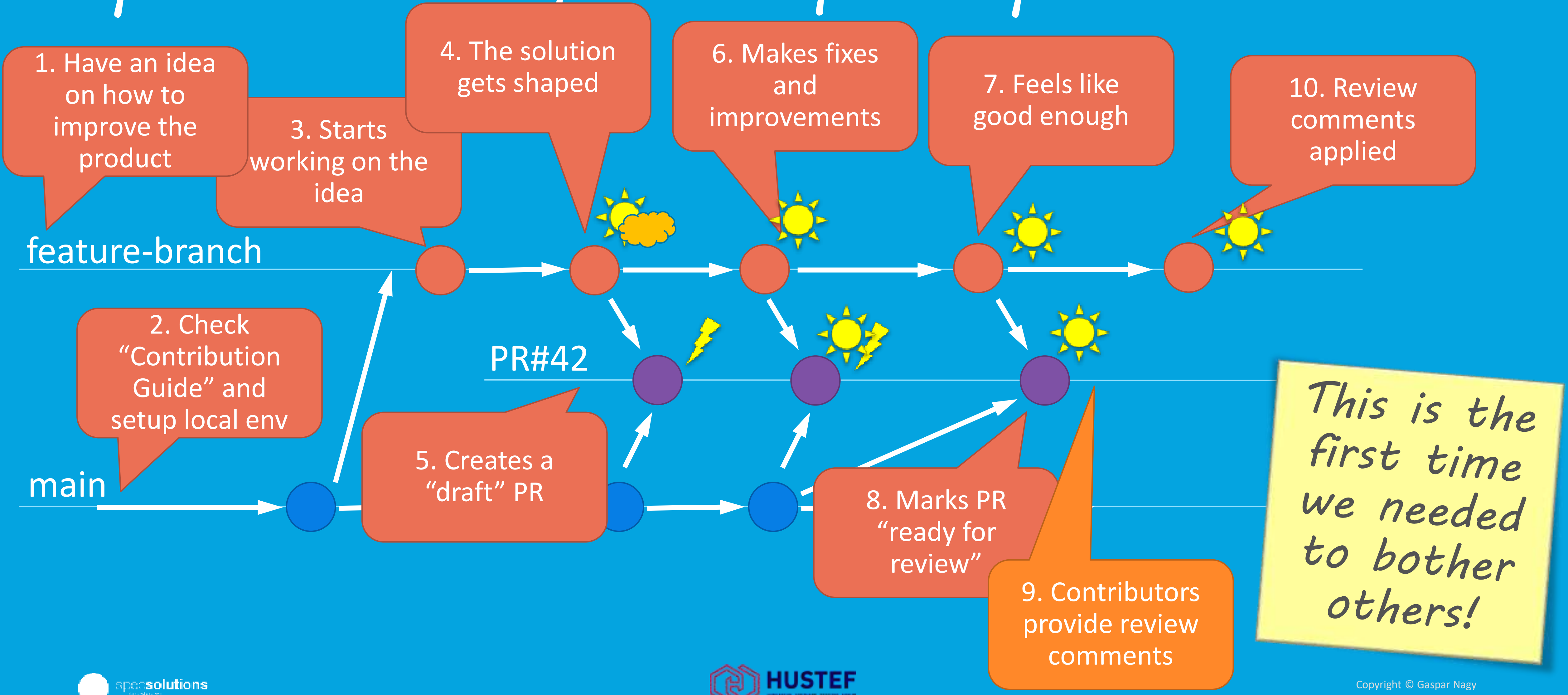
Conflicting files

```
TechTalk.SpecFlow.Generator/UnitTestProvider/XUnit2TestGeneratorProvider.cs
TechTalk.SpecFlow.Generator/UnitTestProvider/XUnitTestGeneratorProvider.cs
Tests/TechTalk.SpecFlow.Specs/Features/UnitTestProviderSpecific/XUnit/XUnit2Provider.feature
Tests/TechTalk.SpecFlow.Specs/StepDefinitions/FeatureFileSteps.cs
```

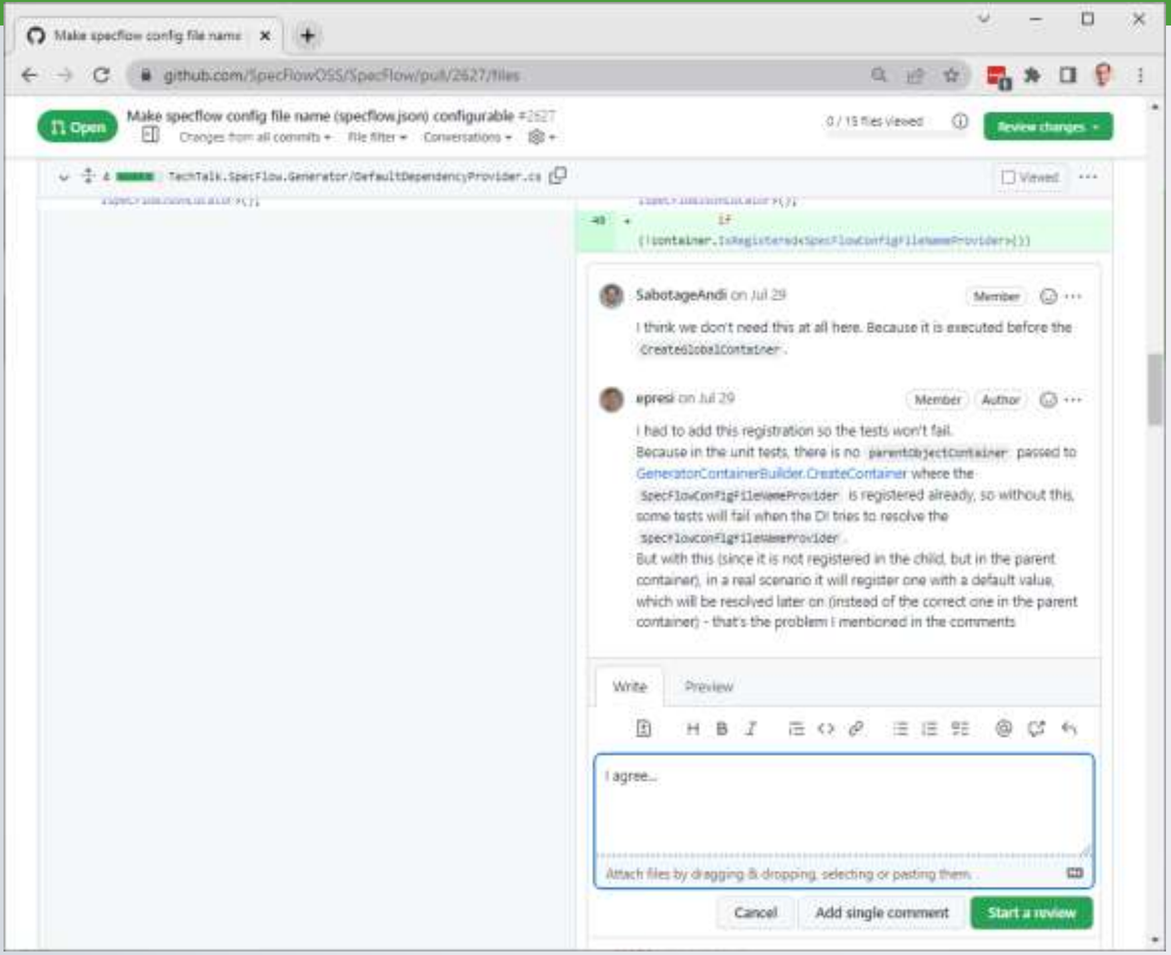
Squash and merge
You can also open this in GitHub Desktop or view command line instructions.

- Pull requests check the changes in integration with the base branch (main)
- They do not only check if your proposed changes are “good”, but also whether they are compatible with the ongoing changes on the main
- And they keep re-checking, always with the latest main
- This way they reduce the effort required for merging at the end

Open-source pull request process


1. Have an idea on how to improve the product
3. Starts working on the idea
4. The solution gets shaped
6. Makes fixes and improvements
7. Feels like good enough
10. Review comments applied
feature-branch
2. Check "Contribution Guide" and setup local env
PR#42
main
5. Creates a "draft" PR
8. Marks PR "ready for review"
9. Contributors provide review comments
This is the first time we needed to bother others!

Idea: Asynchronous change review



Make specflow config file name (specflow.json) configurable #2627

TechTalk.SpecFlow.Generator/DefaultDependencyProvider.cs

```
+            if (!container.IsRegistered<SpecFlowConfigFileNameProvider>())
```

SabotageAndi on Jul 29

I think we don't need this at all here. Because it is executed before the `CreateGlobalContainer`.

epresi on Jul 29

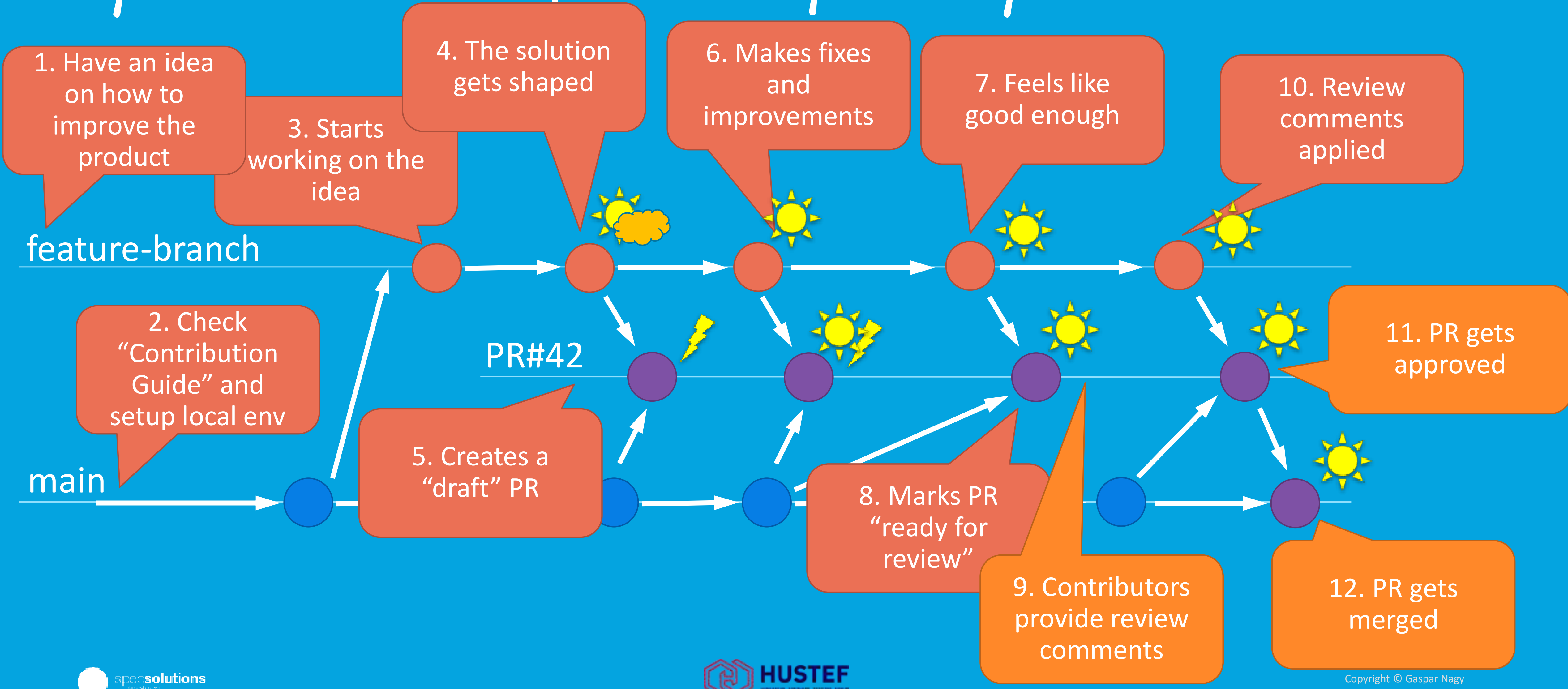
I had to add this registration so the tests won't fail.
Because in the unit tests, there is no `parentObjectContainer` passed to GeneratorContainerBuilder.CreateContainer where the `SpecFlowConfigFileNameProvider` is registered already, so without this, some tests will fail when the DI tries to resolve the `SpecFlowConfigFileNameProvider`.
But with this (since it is not registered in the child, but in the parent container), in a real scenario it will register one with a default value, which will be resolved later on (instead of the correct one in the parent container) - that's the problem I mentioned in the comments

I agree...

Attach files by dragging & dropping, selecting or pasting them.

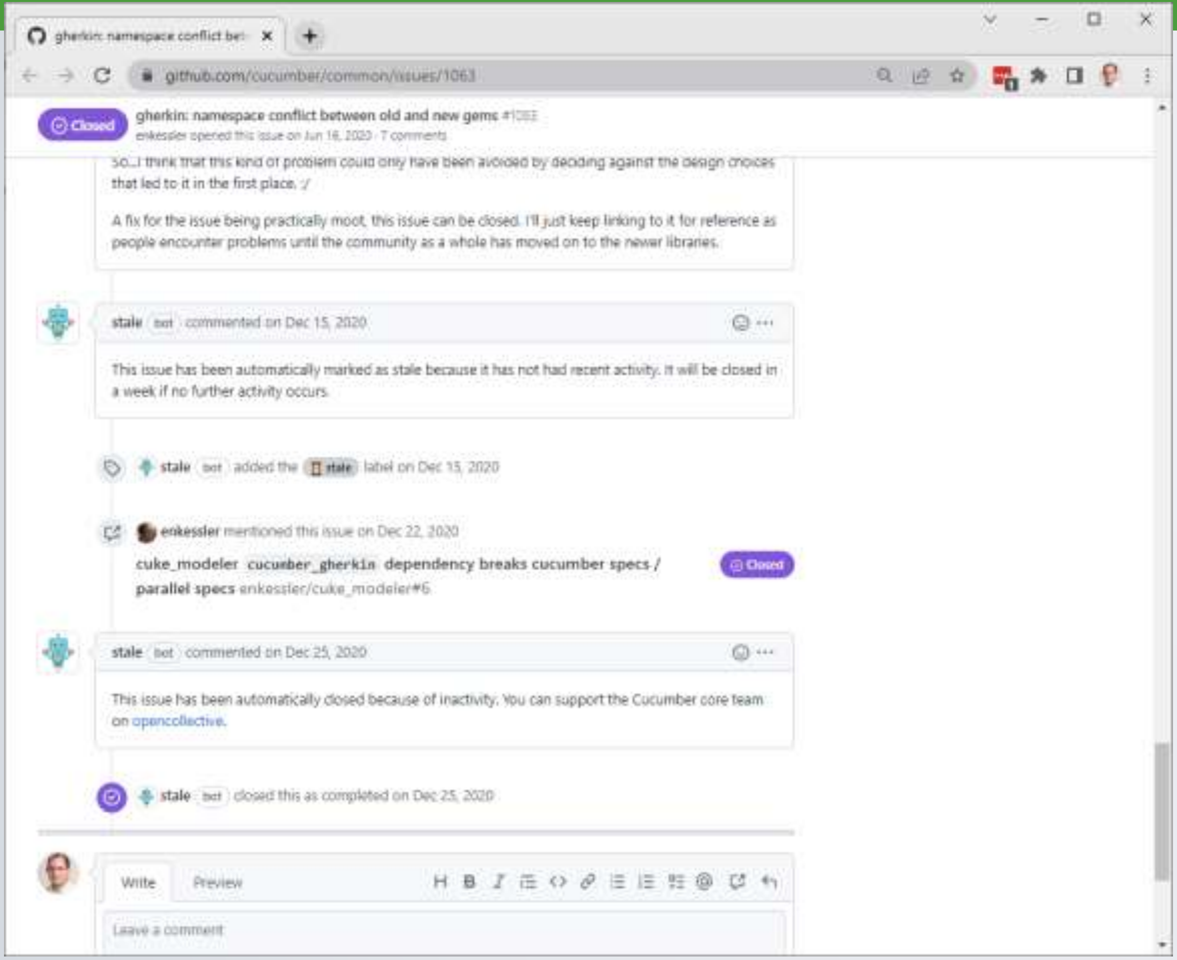
- Conducting a review meeting where the changes are discussed is uncomfortable for many of us
- Public, asynchronous review discussions leave enough time for everyone to consider the problem and respond accordingly
- Publicity might also help to avoid bullying or other non-appropriate behavior
- Asynchronous feedback is easier to schedule
- PR let's you review the change against the *latest* main!

Open-source pull request process


1. Have an idea on how to improve the product
2. Check "Contribution Guide" and setup local env
3. Starts working on the idea
4. The solution gets shaped
5. Creates a "draft" PR
6. Makes fixes and improvements
7. Feels like good enough
8. Marks PR "ready for review"
9. Contributors provide review comments
10. Review comments applied
11. PR gets approved
12. PR gets merged
feature-branch
PR#42
main

When an issue comes

Idea: Let the issue creator do the tracking


github.com/cucumber/common/issues/1063

Closed **gherkin: namespace conflict between old and new gems** #1063
enkessler opened this issue on Jun 16, 2020 · 7 comments

So...I think that this kind of problem could only have been avoided by deciding against the design choices that led to it in the first place. :/

A fix for the issue being practically moot, this issue can be closed. I'll just keep linking to it for reference as people encounter problems until the community as a whole has moved on to the newer libraries.

stale bot commented on Dec 15, 2020

This issue has been automatically marked as stale because it has not had recent activity. It will be closed in a week if no further activity occurs.

stale bot added the stale label on Dec 15, 2020

enkessler mentioned this issue on Dec 22, 2020

cuke_modeler `cucumber_gherkin` dependency breaks cucumber specs / parallel specs enkessler/cuke_modeler#6 Closed

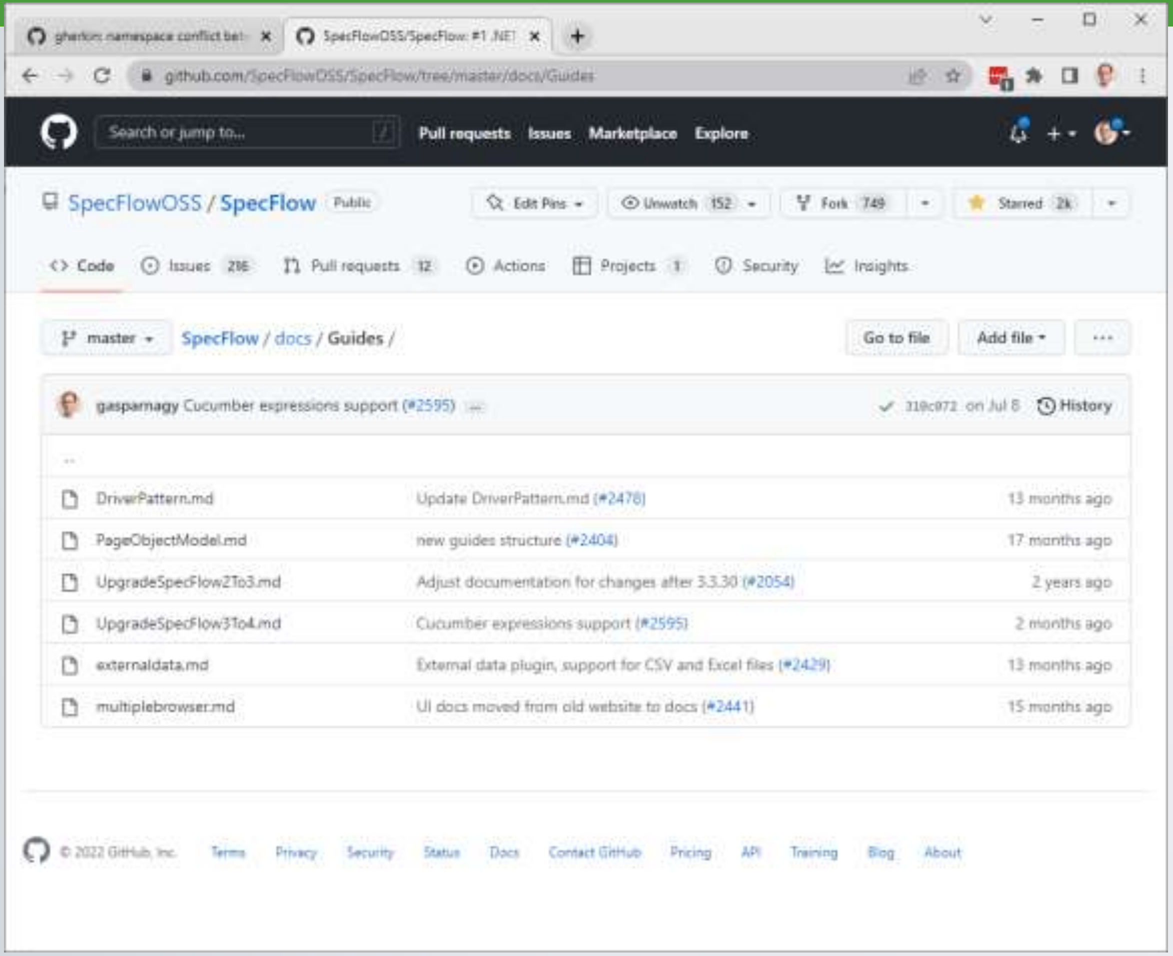
stale bot commented on Dec 25, 2020

This issue has been automatically closed because of inactivity. You can support the Cucumber core team on opencollective.

stale bot closed this as completed on Dec 25, 2020

- When an issue comes with unclear circumstances it is hard to classify the issue and decide on the severity
- Many open-source projects use some “auto-close” model – they close the issue ticket once they cannot progress with it. It becomes the responsibility of the issue creator to re-open if more information is available
- Some projects auto-close the issues automatically after some idle time (e.g. 60 days) – if we could not solve it in 60 days, probably we will not solve it.
- These strategies might sound rude, but with good communication they help a lot

Idea: Extend documentation instead of answering the issue



github.com/SpecFlowOSS/SpecFlow/tree/master/docs/Guides

SpecFlowOSS / SpecFlow

master · SpecFlow / docs / Guides /

gasparnagy Cucumber expressions support (#2595) … ✓ 318c072 on Jul 8 History

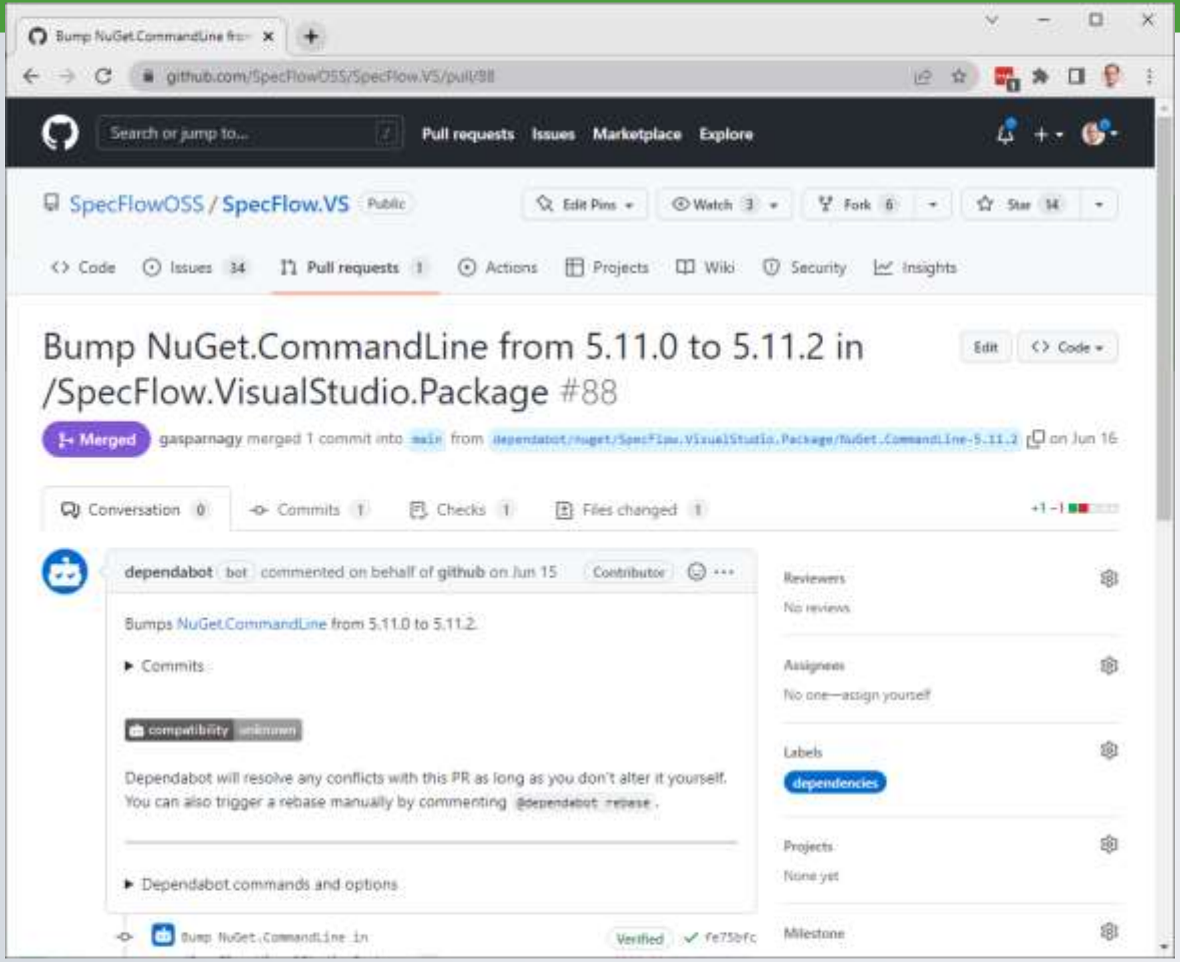
DriverPattern.md	Update DriverPattern.md (#2478)	13 months ago
PageObjectModel.md	new guides structure (#2404)	17 months ago
UpgradeSpecFlow2To3.md	Adjust documentation for changes after 3.3.30 (#2054)	2 years ago
UpgradeSpecFlow3To4.md	Cucumber expressions support (#2595)	2 months ago
externaldata.md	External data plugin, support for CSV and Excel files (#2429)	13 months ago
multiplebrowser.md	UI docs moved from old website to docs (#2441)	15 months ago

© 2022 GitHub, Inc.

- Many open-source project keeps the documentation in source control editable with simple tools (e.g. using Markdown format)
- This might encourage contributors to extend the documentation (and respond with a link) instead of responding with long details (if the question is of general interest)
- You don't need to wait for the second issue of the same topic to create a separate ticket for extending the documentation.

When it's time to release

Idea: Make dependencies trackable



Bump NuGet.CommandLine from 5.11.0 to 5.11.2 in /SpecFlow.VisualStudio.Package #88

Merged gasparnagy merged 1 commit into `main` from `dependabot/nuget/SpecFlow.VisualStudio.Package/NuGet.CommandLine-5.11.2` on Jun 16

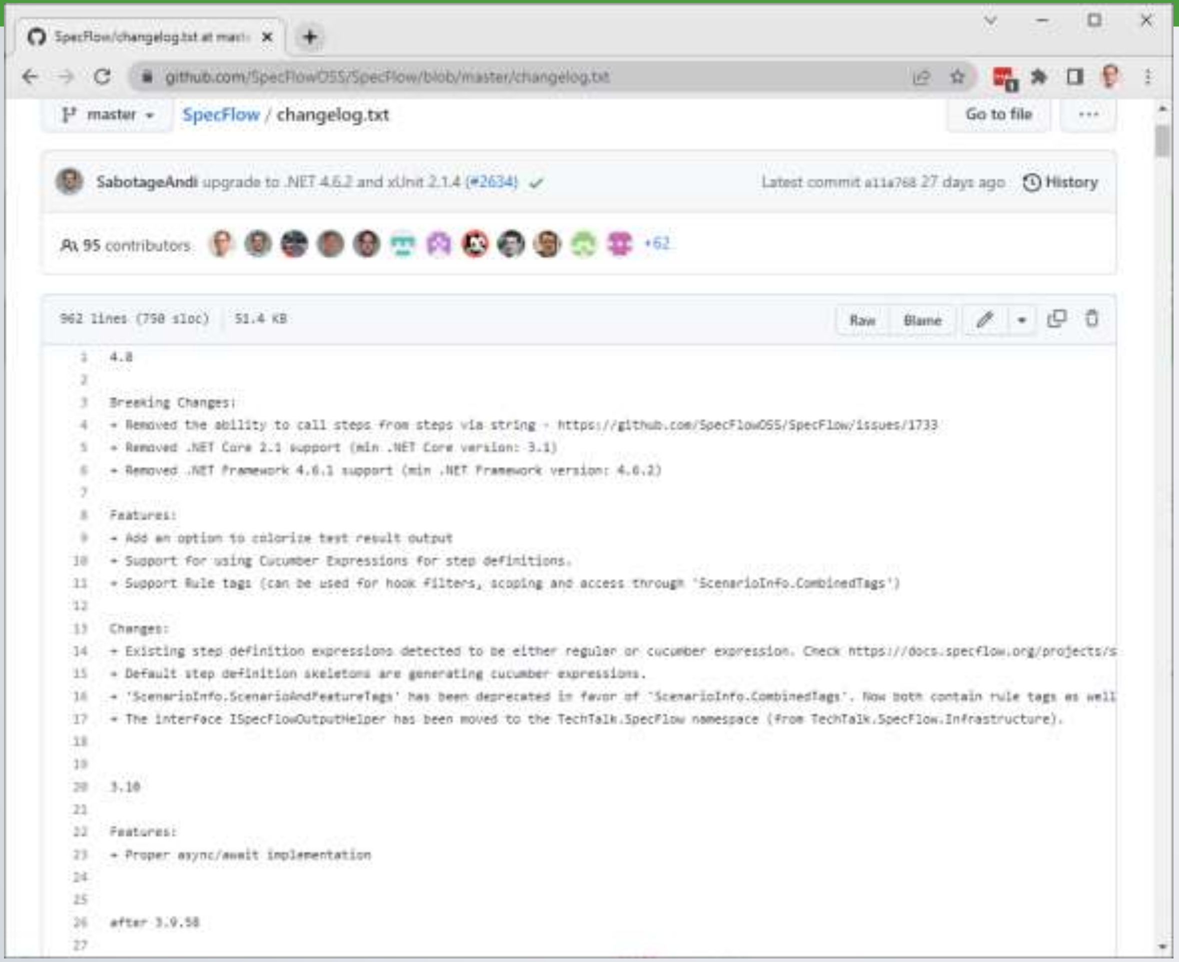
dependabot bot commented on behalf of github on Jun 15

Bumps NuGet.CommandLine from 5.11.0 to 5.11.2.

Dependabot will resolve any conflicts with this PR as long as you don't alter it yourself. You can also trigger a rebase manually by commenting `@dependabot rebase`.

- Open-source project use standardized package management tools to track their dependencies (e.g. npm, NuGet, Maven, etc.)
- As their dependencies are trackable it is easier to discover (or even visualize) dependencies – that is also very useful for impact analysis of a bug
- In some cases there are even some automatic tooling that can fix dependency problems (e.g. security alerts)
- These package management tools can also be hosted on premises, so you can also track your internal components in the same way

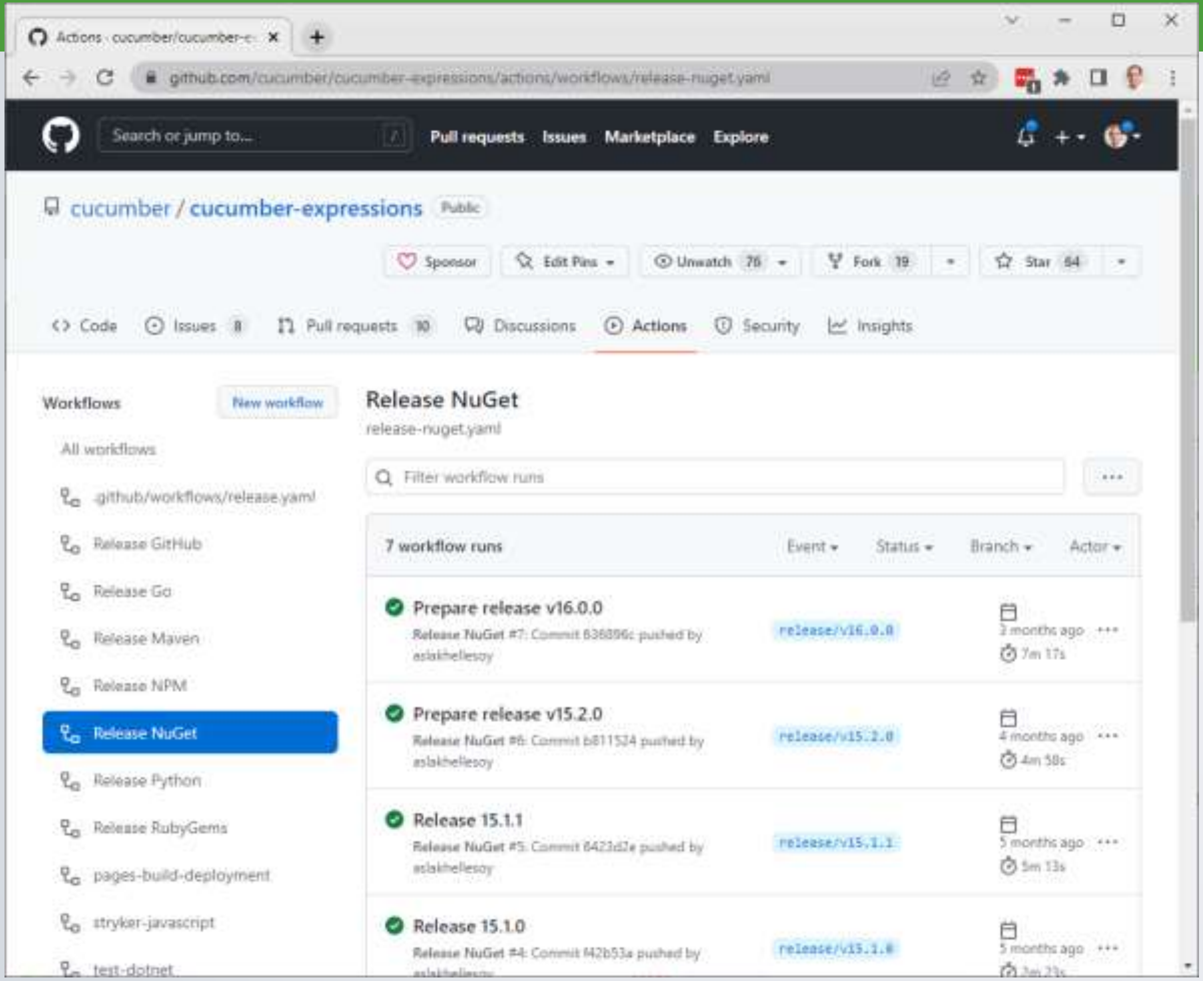
Idea: Use changelog & semantic versioning


SpecFlow/changelog.txt at mast
github.com/SpecFlowOSS/SpecFlow/blob/master/changelog.txt
master
SpecFlow / changelog.txt
Go to file
SabotageAndi upgrade to .NET 4.6.2 and xUnit 2.1.4 (#2634)
Latest commit a11a768 27 days ago
History
95 contributors
+62
962 lines (750 sloc) | 51.4 KB
Raw
Blame

```
4.8

Breaking Changes:
+ Removed the ability to call steps from steps via string - https://github.com/SpecFlowOSS/SpecFlow/issues/1733
+ Removed .NET Core 2.1 support (min .NET Core version: 3.1)
+ Removed .NET Framework 4.6.1 support (min .NET Framework version: 4.6.2)

Features:
+ Add an option to colorize test result output
+ Support for using Cucumber Expressions for step definitions.
+ Support Rule tags (can be used for hook filters, scoping and access through 'ScenarioInfo.CombinedTags')

Changes:
+ Existing step definition expressions detected to be either regular or cucumber expression. Check https://docs.specflow.org/projects/s
+ Default step definition skeletons are generating cucumber expressions.
+ 'ScenarioInfo.ScenarioAndFeatureTags' has been deprecated in favor of 'ScenarioInfo.CombinedTags'. Now both contain rule tags as well
+ The interface ISpecFlowOutputHelper has been moved to the TechTalk.SpecFlow namespace (from TechTalk.SpecFlow.Infrastructure).


3.10

Features:
+ Proper async/await implementation


after 3.9.58

```

- The users of open-source libraries have to make decisions about upgrading on their own
- Detailed changelogs can help and avoid unnecessary issues being created
- By using semantic versioning (https://semver.org/) the users (or even tools) can make decisions when it is safe to update a particular dependency
  - "Given a version number MAJOR.MINOR.PATCH, increment the:
    - MAJOR version when you make incompatible API changes
    - MINOR version when you add functionality in a backwards compatible manner
    - PATCH version when you make backwards compatible bug fixes"

Idea: Automate release process


Actions · cucumber/cucumber-e

github.com/cucumber/cucumber-expressions/actions/workflows/release-nuget.yaml

Search or jump to... Pull requests Issues Marketplace Explore

cucumber / cucumber-expressions Public

Sponsor · Edit Pins · Unwatch 76 · Fork 19 · Star 64

Code · Issues 8 · Pull requests 10 · Discussions · Actions · Security · Insights

Workflows · New workflow

- All workflows
- .github/workflows/release.yaml
- Release GitHub
- Release Go
- Release Maven
- Release NPM
- Release NuGet
- Release Python
- Release RubyGems
- pages-build-deployment
- stryker-javascript
- test-dotnet

Release NuGet
release-nuget.yaml

Filter workflow runs

7 workflow runs · Event · Status · Branch · Actor

Run	Branch	Time
✓ Prepare release v16.0.0 Release NuGet #7: Commit 636896c pushed by aslakhellesoy	release/v16.0.0	3 months ago 7m 17s
✓ Prepare release v15.2.0 Release NuGet #6: Commit b811524 pushed by aslakhellesoy	release/v15.2.0	4 months ago 4m 58s
✓ Release 15.1.1 Release NuGet #5: Commit 6423d2e pushed by aslakhellesoy	release/v15.1.1	5 months ago 5m 13s
✓ Release 15.1.0 Release NuGet #4: Commit f42b53a pushed by aslakhellesoy	release/v15.1.0	5 months ago 2m 23s

- Automating the release process is very important for open-source projects, because
  - Multiple people might need to be able to release
  - There might be calm periods when there is no release – manual processes are forgotten
  - There might be a need for fast reaction (e.g. hotfixes)
- Automating the release might also improve security, because
  - Individuals don't need to have the publish keys on their laptops
  - Repeatable releases can be used to protect against injection attacks

Wrap up

What can we learn from open-source development?

Verification → Feedback

Avoiding conflicts → Embracing conflicts

Rollback → Roll forward

Ideas to steal…

- Contribution guide
- Pull Request concept
- Quality checklist in PR template
- Check in integration, not in isolation
- Asynchronous change review
- Let the issue creator do the tracking
- Extend documentation instead of answering the issue
- Make dependencies trackable
- Use changelog & semantic versioning
- Automate release process

Learn how to self-service quality!

•

specsolutions
given.when.then.

Thank you!

Gáspár Nagy
coach • trainer • bdd addict • creator of specflow
“The BDD Books” series • http://bddbooks.com
@gasparnagy • gaspar@specsolutions.eu

specsolutions

given.when.then.