


Lessons learned from helping port the top contrib projects to Drupal 10.



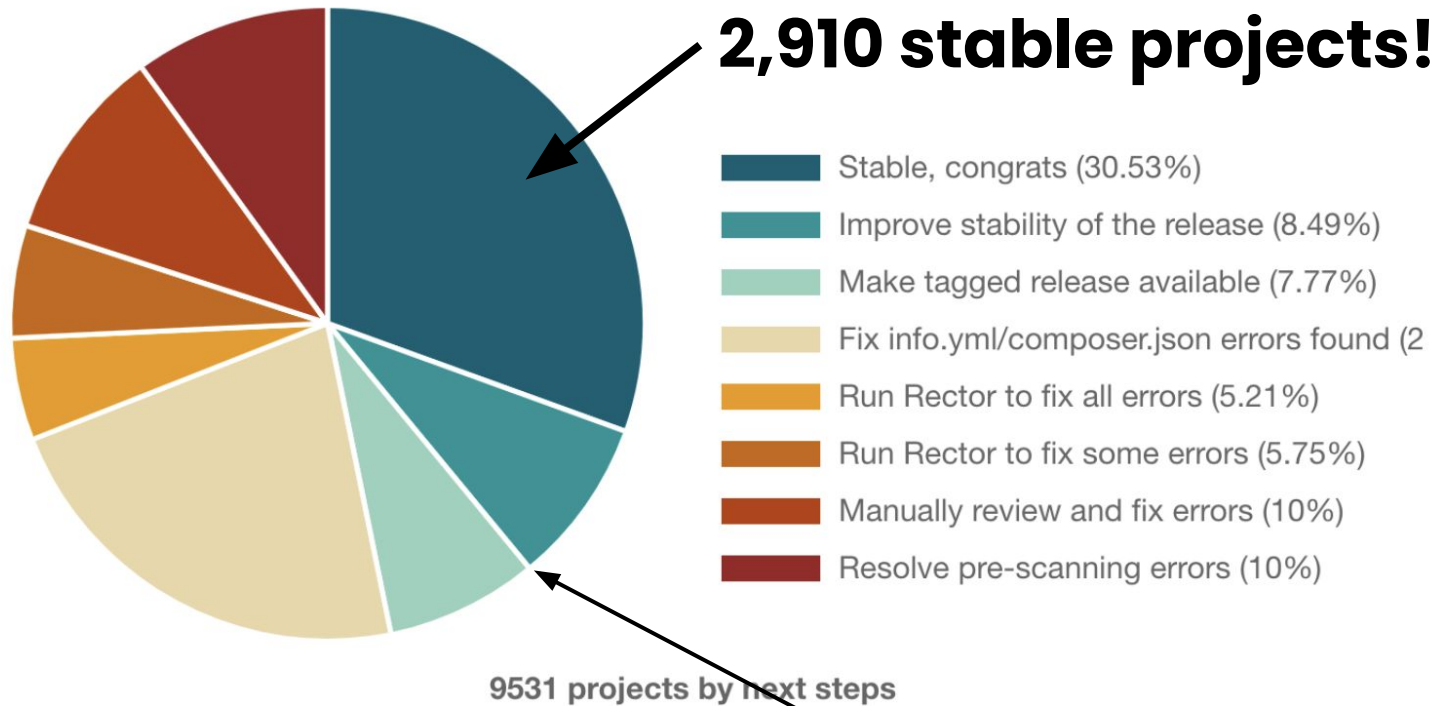
Matt Glaman

Principal Software Engineer @ Acquia
Drupal Integrations Team

Maintainer of `phpstan-drupal`
Contributor to `drupal-rector`

Who has made their
project sites
Drupal 10 ready?

Who has made
contributed modules
Drupal 10 ready?



1,550 just need stable releases

Results are from April 12th 2023

https://dev.acquia.com/drupal10/deprecation_status/projects

Our team tracked over
180 modules in our
Drupal 10 readiness
initiative.

Our team was credited
on over **90** modules and
3 Drupal core issues

Our goal

Ensure all of Acquia's product modules and dependencies were ready for the release of Drupal 10

(oh, and many of the top-used modules by Acquia's customers.)

Acquia's **Drupal Integration Team**

We maintain Acquia's product modules



japerry (Jakob Perry)



mglaman (Matt Glaman)



attilatilman (Attila Tilman)



Lal_ (Abhishek Lal)



balintpekker (Balint Pekker)



nkoporec (Necj Koporec)

#d10readiness Drupal Integration Team

Did you know...

Drupal 11

2024

<https://www.drupal.org/i/3330791>

Drupal 11 is
one year away.

What have we learned
from **Drupal 9** → **10**?

How can we make
upgrading to
Drupal 11 even easier?

Looking back:

Drupal 10 changes



DRUPAL 10 CHANGES



**ME READY TO FIX
CORE DEPRECATIONS**



WAIT THERE'S MORE?

It's just deprecations in Drupal core, right?

**Dependency
major versions**

Drupal

9 → 10

PHP

7.4 → 8.1

PHPUnit

8.5 → 9

Symfony

4 → 6

Guzzle

6 → 7

CKEditor

4 → 5

JavaScript changes and library removals

jQuery Once -> once

jQuery UI -> moved (contrib)

Backbone -> internal

Underscore -> internal

Pain from long Drupal 8 lifecycle

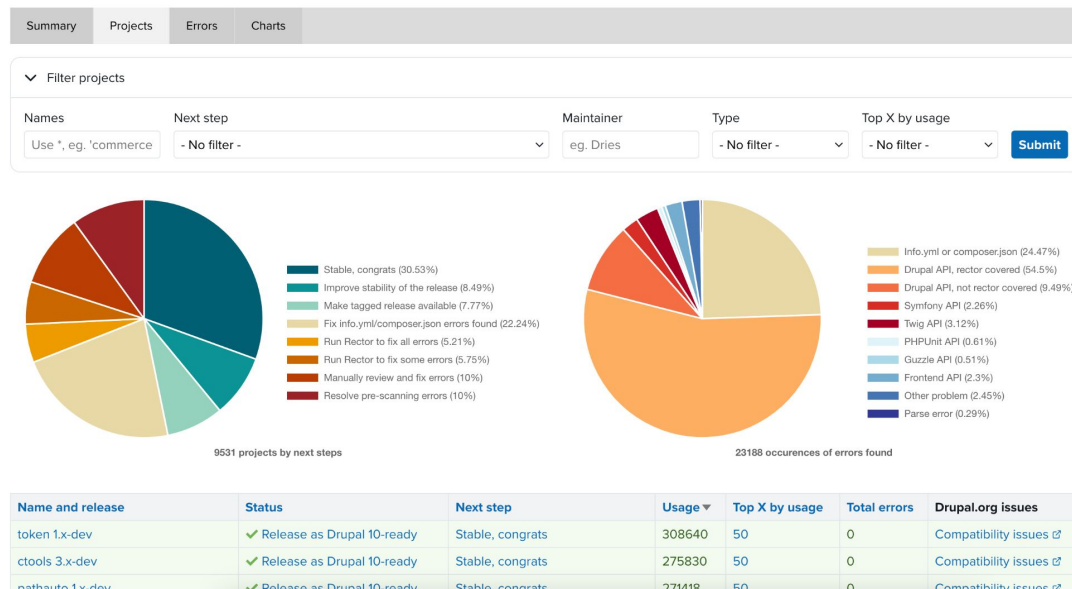
(Nov 2015 – Nov 2021)

Tracking progress

Readiness dashboard

- dev.acquia.com project readiness dashboard available
- Maintained by Gábor
- Requires results from a Jenkins job maintained by Drupal Association
- It was* fragile and not run often enough for active tracking
- [/project/deprecation_status](#)

Drupal 10 project deprecation status



* Was! Björn Brala (@bbra1a) stabilized as a project using GitLab CI on Drupal.org (project_analysis project)

How we did it

We decided up front that we needed to track progress for reporting in real time

The Canary

- **Private for our team at Acquia**
- Built at DrupalCon Portland at the Acquia booth with Jakob Perry
- Metapackages tracked dependencies as **stable**, **dev**, or **unstable**
- Allows collecting and verifying patches
- CI pipelines automated testing Drupal 10 and Guzzle 7 compatibility
- Upgrade Status results as a dashboard

Drupal Module Readiness Canary

[+ Export to CSV](#)[+ Jira sync](#)

Compatible

Jira	MoSCoW	Project	Local	Result	Dependencies	Patched	Drupal.org
DIT-907 Released	M - Must have	Acquia Connector acquia_connector	4.0.2	✓	None	No patches	Up to date (Compatible)
DIT-682 Released	M - Must have	Acquia ContentHub acquia_contenthub	3.0.3	✓	1 (0 problems)	No patches	Up to date (Compatible)
DIT-683 Released	M - Must have	Acquia Lift acquia_lift	8.x-4.5	✓	None	No patches	Up to date (Compatible)
DIT-684 Released	M - Must have	Acquia Purge acquia_purge	8.x-1.3	✓	1 (0 problems)	No patches	Up to date (Compatible)
DIT-698 Released	M - Must have	Acquia Search acquia_search	3.1.4	✓	3 (0 problems)	No patches	Up to date (Compatible)
DIT-699 Released	M - Must have	Address address	8.x-1.11	✓	None	No patches	Up to date (Compatible)
DIT-700 Released	M - Must have	Admin Toolbar admin_toolbar	3.3.0	✓	None	No patches	Up to date (Compatible)
DIT-702 Released	M - Must have	Automated Logout autologout	8.x-1.4	✓	None	No patches	Up to date (Compatible)
DIT-707 Needs release	M - Must have	Blazy blazy	8.x-2.x	✓	None	No patches	Unavailable (Unavailable)

The process of fixing a module

Scan!

Find deprecations, breaking changes, and required upgrades.

`upgrade_status`, `drupal-check`,
`phpstan-drupal`

Fix!

Once items have been found, they need to be fixed.

`drupal-rector, manually`

Commit!

Work with maintainers to commit
the fixes

Profit!

The module is now Drupal 10 compatible

WAIT!

Is the module *really* ready?

What about...

- Was it tested manually, checking for runtime deprecation logs?
- No longer compatible with previous versions of Drupal core
- There may be new deprecations added until the major version reached release candidate (RC) status
- A release needs to be made with the changes to avoid running -dev releases or series of patches

**What we learned
about this process**

Tooling

Improved tooling to help facilitate required changes to code.

Tooling can become out of date.

Cadence

Start earlier.

Not months before the major release,
but after each minor release.

Best practices

Define best practices for module maintainers & contributors to reference.

Coordination

This process requires *a lot* of coordination, most of which is currently ad-hoc

**Backwards
compatibility**

drupal/core support

Contributed projects *should* provide support for all supported versions of Drupal core.

3 versions supported at all times: 10.0.x, 9.5.x, 9.4.x

drupal/core support

Fixing deprecations uses **new code** that didn't exist and breaks support on previous versions.

9.5.x -> deprecations, new code

9.4.x -> new code does not exist

Modules must provide
backward compatibility
layers for deprecations

Example:

Drupal 9.1.0 and
EventDispatcher
breaking changes

<https://www.drupal.org/node/3159012>

```
if (version_compare(\Drupal::VERSION, '9.1', '≥')) {
  $this->dispatcher->dispatch(
    $event,
    AcquiaSearchEvents::GET_POSSIBLE_CORES
  );
}
else {
  // @phpstan-ignore-next-line
  $this->dispatcher->dispatch(
    AcquiaSearchEvents::GET_POSSIBLE_CORES,
    $event
  );
}
```

Fix to allow drupal/core: ^8 || ^9 without deprecations

Example:

Drupal 9.4.0 and
getImplementations
deprecated for
invokeAllWith

<https://www.drupal.org/node/3000490>

```
if (method_exists($this->moduleHandler, 'invokeAllWith')) {
    $implementations = [];
    $this->moduleHandler->invokeAllWith(
        'pathauto_is_alias_reserved',
        function (callable $hook, string $module) use (&$implementations) {
            $implementations[] = $module;
        }
    );
}
else {
    // Use the deprecated getImplementations() for Drupal < 9.4.
    $implementations = $this->moduleHandler->getImplementations(
        'pathauto_is_alias_reserved'
    );
}
```

Fix to allow drupal/core: ^9 || ^10 without deprecations

Change records *should*
include these patterns
for contributors &
maintainers to use

Releases & semantic versioning

We need **guidelines** and
recommendations for
semantic versioning

Major versions

Hard breaking changes from
Symfony or other dependencies.

8.x-2.8 -> 8.x-3.0

2.0.7 -> 3.0.0

Minor versions

Adding support for new Drupal core majors, dropping unsupported Drupal core minors

8.x-2.8 -> 8.x-2.9

2.0.7 -> 2.1.0

Patch versions

Adding support for new Drupal core majors

8.x-2.8 -> 8.x-2.9

2.0.7 -> 2.0.8

Create releases that
bridge versions of Drupal

2.0.x -> drupal/core:^9
2.1.x -> drupal/core:^9.3
2.2.x -> drupal/core:^9.4 || ^10
3.0.x -> drupal/core:^10

Supporting a new
PHP version isn't
a breaking change

(just don't use its features.)

Adding dependencies to **fix deprecated libraries**



Example: jQuery UI removal to contrib

“Don’t go making major
version changes”

by Jakob Perry

<https://medium.com/jakob-on-drupal/dont-go-making-major-version-changes-474293dda1d7>

Challenge of getting maintainers to commit fixes & make releases

(Me. I am one of them. I am guilty.)

Drupal 10 readiness project adoption

https://www.drupal.org/project/contribution_events/issues/3342443

JavaScript fixes

There are **no tools!**

jscodeshift

“A JavaScript codemod toolkit.” by Facebook.

```
function replacejQueryOnce(ast) {
  const method = ast.node.callee.property.name;
  j(ast).replaceWith(
    wrapWithjQuery(
      j.callExpression(
        newMethod(method),
        [
          ast.node.arguments[0],
          ... processSelector(ast.node.callee.object),
        ],
      ),
    ),
  );
}
```

[mg1aman/jquery-once-jscodeshift](#)

nod_ used jscodeshift on
Drupal core for
jQuery.once

<https://www.drupal.org/project/drupal/issues/3183149#comment-14011264>

Should Drupal core
provide
jscodeshift scripts?

Or another standalone tool

PHPStan

PHPStan levels

`upgrade_status` runs PHPStan at level 0 – deprecation only reporting.

We need to run **at least** level 2 to improve usage. But...

```
function mymodule_node_insert(EntityInterface $node) {  
  
    if ($node→getRevisionAuthor()) {  
        // Missing deprecation warning!  
    }  
  
}
```

8.9.x example: PHPStan level 0 silently fails on unknown method of EntityInterface::getRevisionAuthor (deprecated on NodeInterface.)

PHPStan levels

drupal-check >1.4.0 runs PHPStan at level 2.

It resulted in unrelated changes Drupal 10 patches.

<https://mglaman.dev/blog/drupal-check-140-enforcing-phpstan-level-2>

Custom rules

phpstan-drupal has to provide rules to find issues only uncovered at runtime

phpstan-drupal rules for breaking changes

- Deprecated global constants (`FILE_STATUS_PERMANENT`)
- 9.2.x Access checking must be explicitly specified on content entity queries
- 9.1.x Usage of `symfony-cmf/routing` dependency deprecated
- 9.3.x Forward compatibility shim added for Symfony 5 `RequestStack`
- 9.3.x `#date_time_callbacks` and `#date_date_callbacks` must implement `TrustedCallbackInterface`

Upgrade Status

Dependencies

Detect new dependencies required to replace deprecated libraries.

Suggest adding contrib modules to `composer.json`

Allow customizing the
default PHPStan level

Rector

Great for end-users

Rough for contrib

Why?

Backward compatibility

A one-way street

Rector automates code fixes, but it does not provide backward compatibility for its fixes.

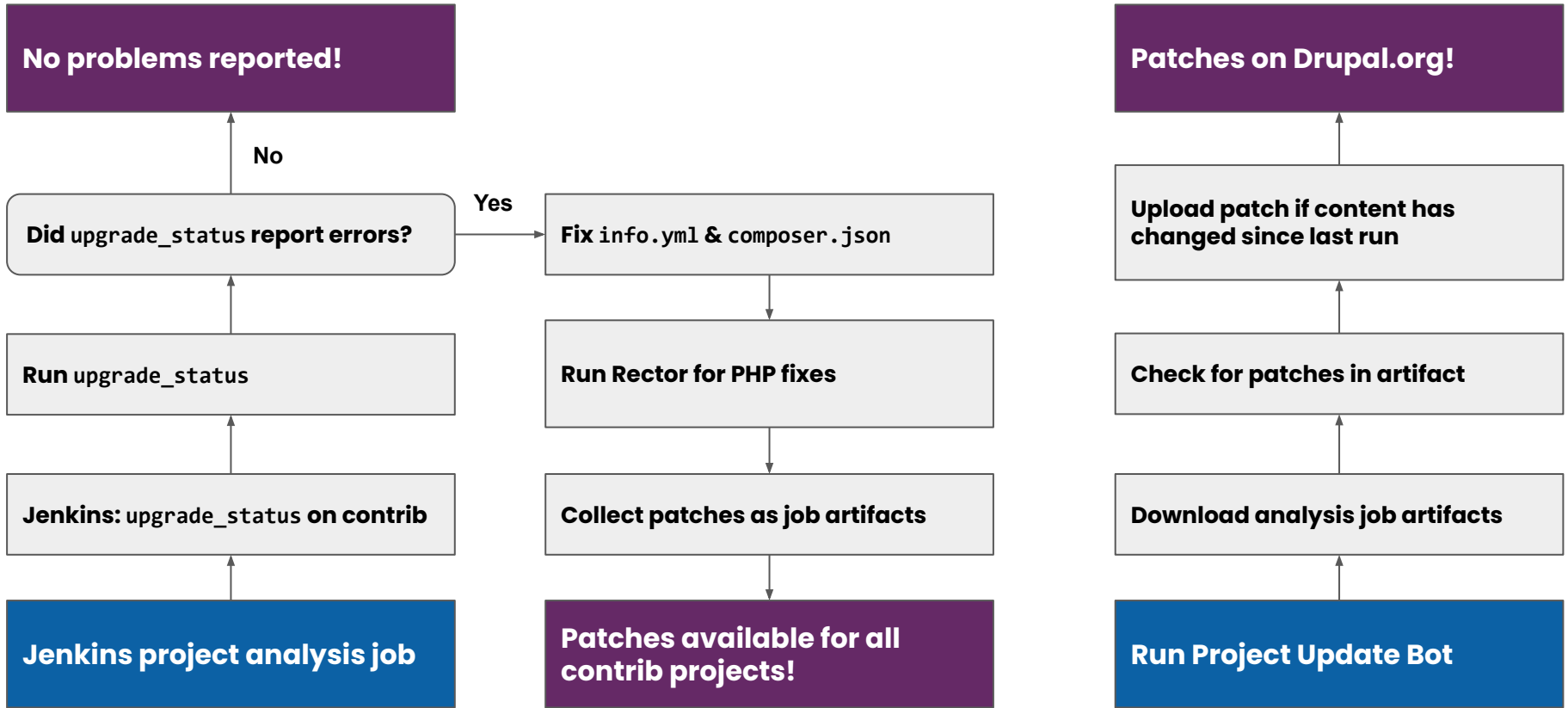
Requires manual intervention to apply backwards compatibility fixes

Can we **automate**
backward compatibility
with Rector?

<https://www.drupal.org/project/rector/issues/3350886>

Project Update Bot

Providing patches at
scale for **all projects**
on Drupal.org

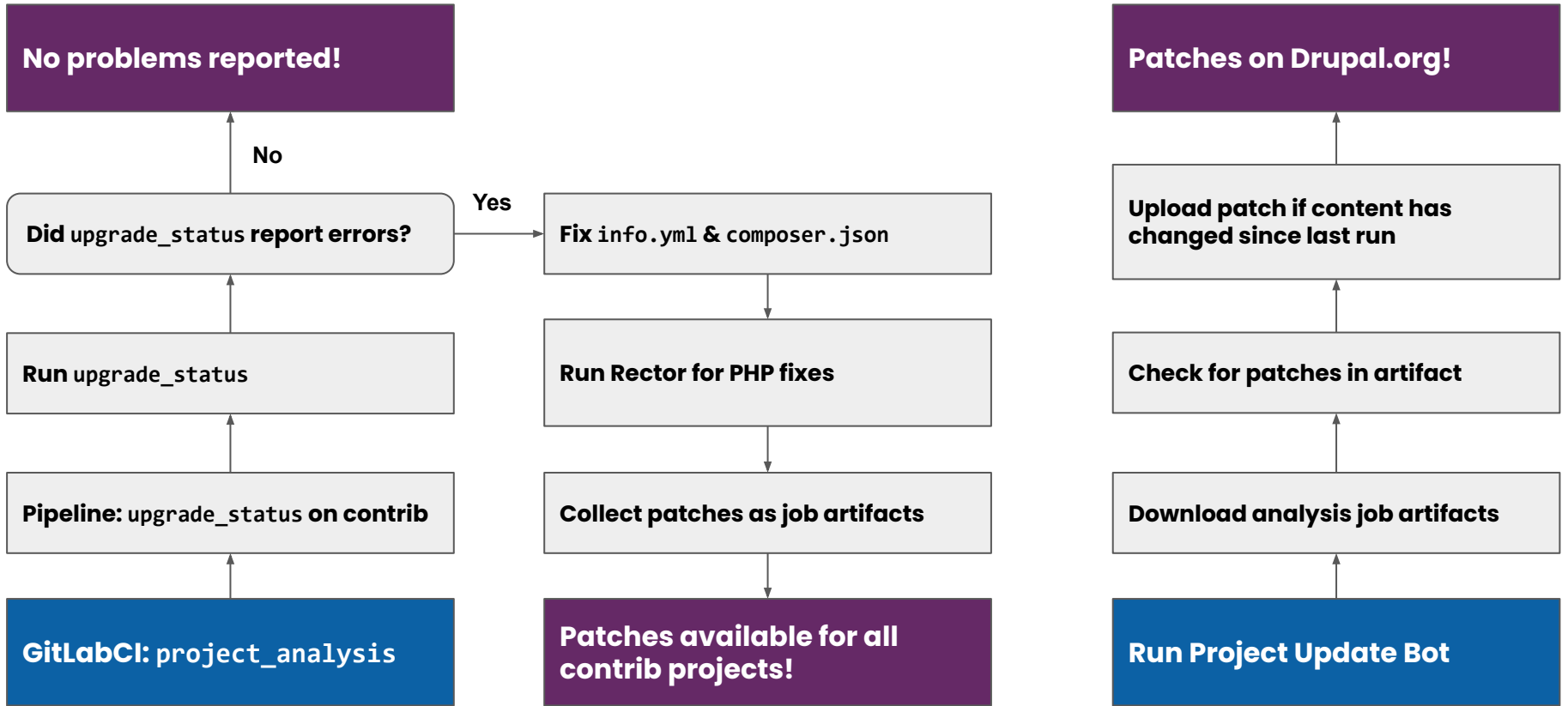


Before: Diagram of the Project Update Bot

project_analysis

Björn Brala (@bbra1a) created a general project on Drupal.org that runs on GitLab CI to replace the Jenkins job.

https://git.drupalcode.org/project/project_analysis/



After: Diagram of the Project Update Bot

Learn more at
DrupalCon

“Project update bot: The road to Drupal 11”

by Björn Brala (bbrala)

<https://events.drupal.org/pittsburgh2023/session/project-update-bot-road-drupal-11>

**This process is only as
good as our tooling**

Rector patches **need**
backwards compatibility to
be useful at scale.

<https://mglaman.dev/blog/adding-backward-compatibility-rector-rules>

Can we automate
JavaScript fixes?

Can we automate
dependency changes?

https://www.drupal.org/project/project_analysis/issues/3356371

**Let's keeping
improving**

Onwards, to Drupal 11

Thank you!

Questions?

Please provide your feedback!

mid.camp/6884