

API Platform Conference 2025

Composer Best Practices 2025



Nils Adermann
@naderman

Private Packagist
<https://packagist.com>

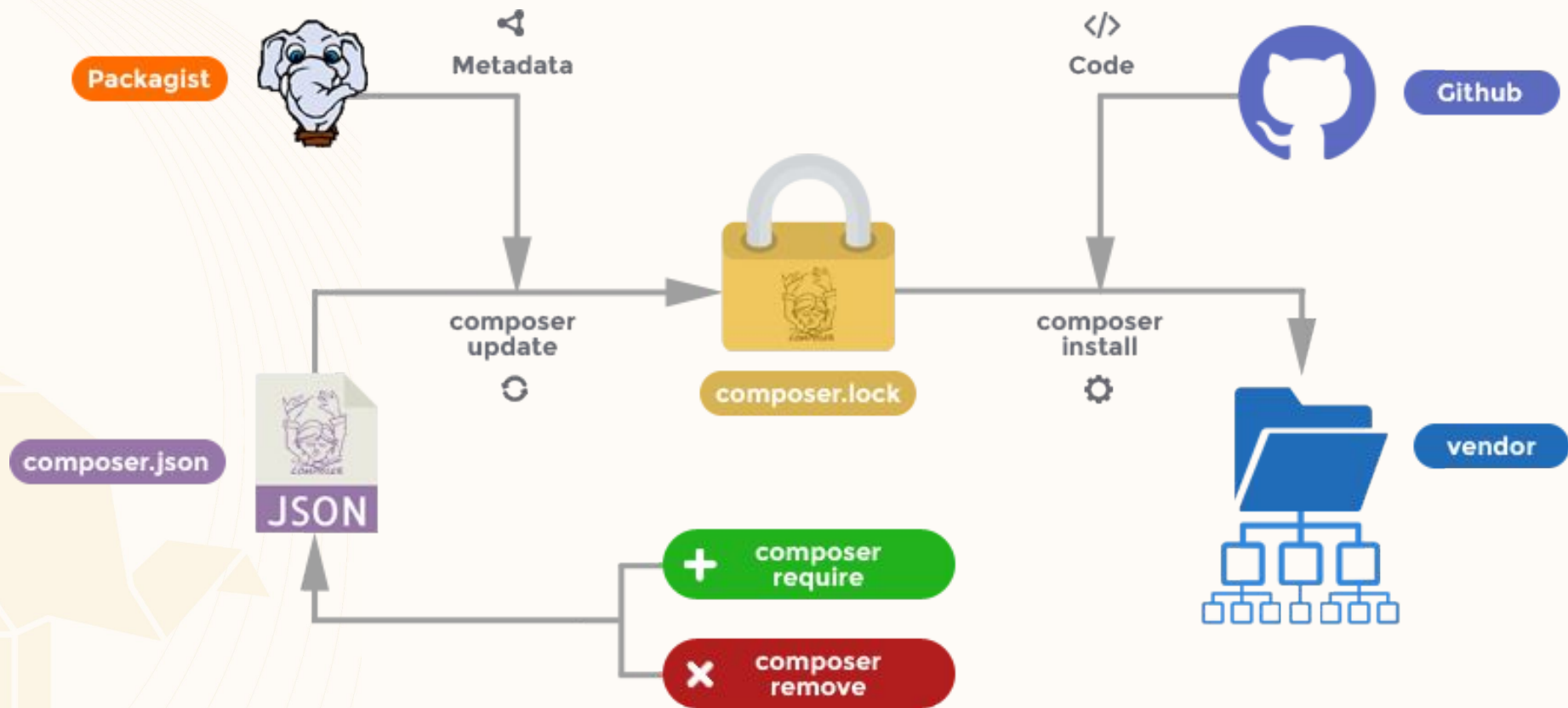


What happened so far

- February 5th, 2011 - first lines of code
- March 1st, 2012 - first alpha release
- November 14th, 2015 - first stable release
- October 24th, 2020 - second major release

What's actually new or different in 2025?

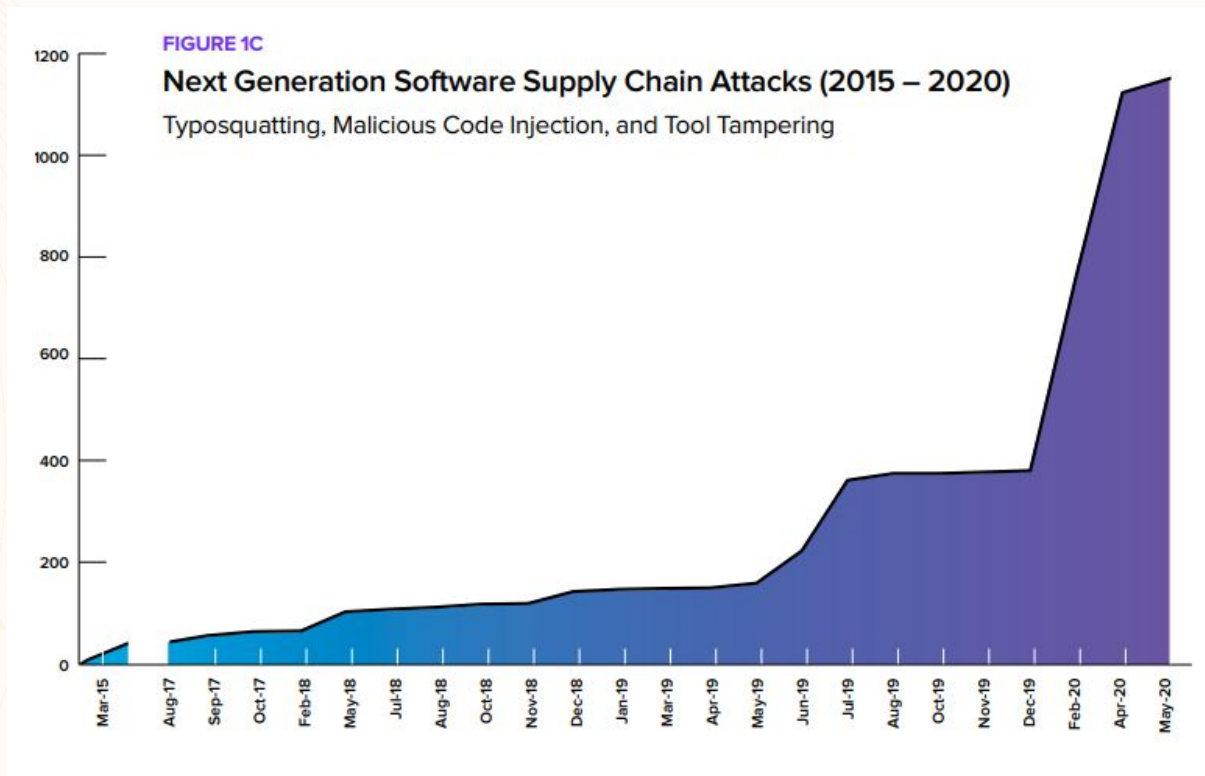
- packagist.org 1.x metadata shutdown
 - Still 2.7% of all install stats reported from Composer 1.x lock file installs
 - Switch to Composer 2! Stop using the v1 API!
 - Mission critical app? Private Packagist can proxy to Composer 1.x protocol
- New types of security attacks
 - First typo squatting
 - Now AI hallucinated package names
- Really trying not to change too much
- Occasionally new flags or commands
 - Today: composer audit, composer bump, composer update --minimal-changes, composer update --patch-only



Why should you care about supply chain security?

- Online crime is rampant
- **Criminals may attack your website** to steal your visitors/users/customers identities, payment info, or other personal data even if it's just for phishing or social engineering
 - Don't think your data isn't valuable!
- Still essentially fighting the same OWASP Top 10 as 20 years ago
 - But also in your dependencies!

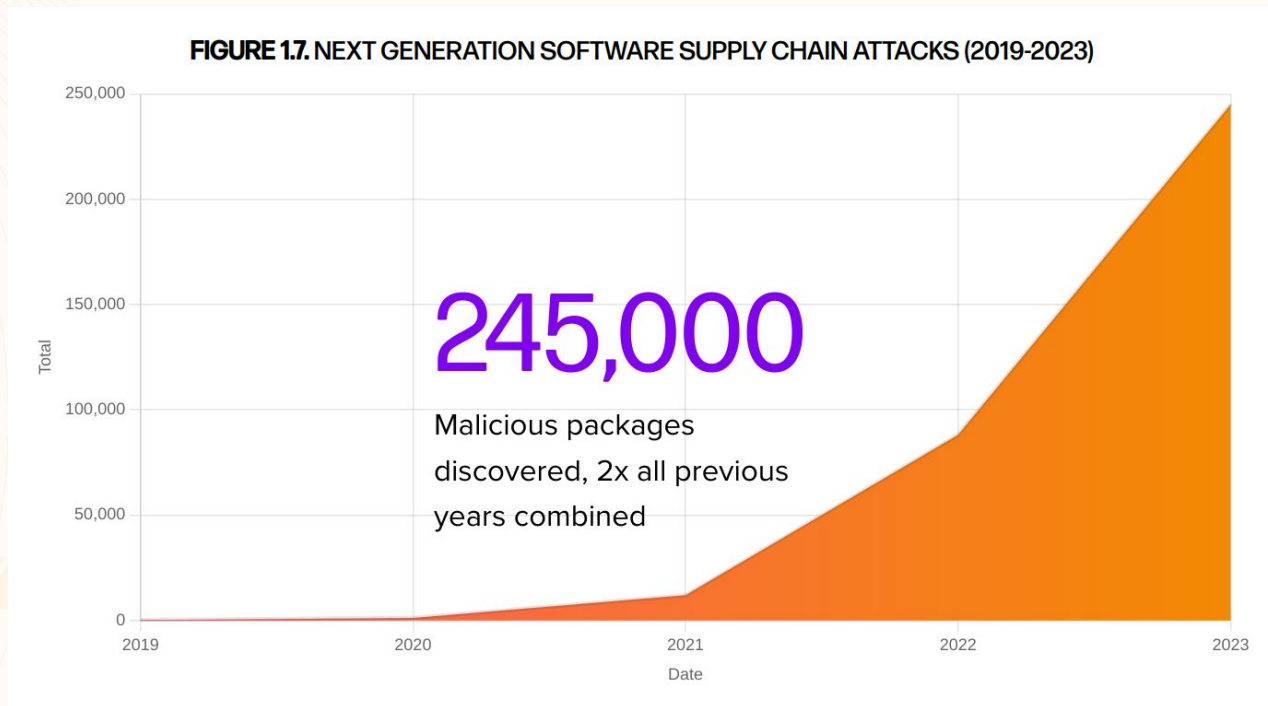
Supply Chain Attacks



“2020 State of the Software Supply Chain” by sonatype

https://www.sonatype.com/hubfs/Corporate/Software%20Supply%20Chain/2020/SON_SSSC-Report-2020_final_aug11.pdf

Supply Chain Attacks

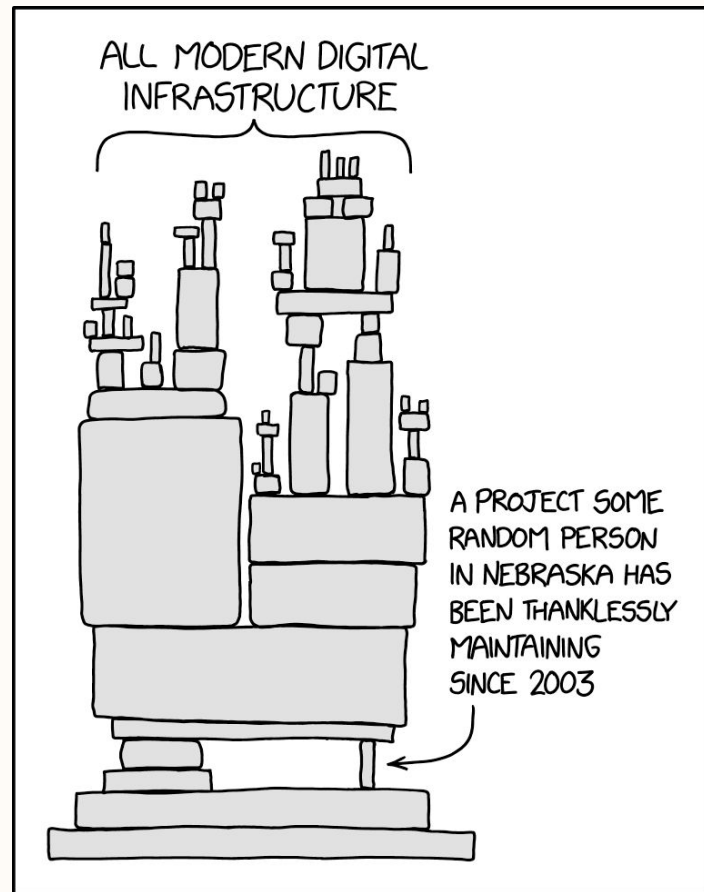


“9th Annual State of the Software Supply Chain” by sonatype

<https://www.sonatype.com/hubfs/2023%20Sonatype-%209th%20Annual%20State%20of%20the%20Software%20Supply%20Chain-%20Update.pdf>

Supply Chain Funding

- \$2,000 donations per year to OpenSSL
- \$841 in 3 days after Heartbleed 🤖
- Creation of Open Software Security Foundation (OpenSSF) at Linux Foundation
 - > \$10M raised by 2021
- German Government: Sovereign Tech Fund/Agency
 - <https://sovereigntechfund.de> since 2022
 - €17M budget in 2024, €11.5M in 2023
- Alpha-Omega
 - <https://alpha-omega.dev/> since 2022
 - \$5M granted in 2024



Supply Chain Funding

- **It's your supply chain, you need to help fund it!**
- composer fund will tell you which of your dependencies need financial help
- Sponsor the PHP Foundation
 - <https://thephp.foundation/sponsor/>
- Buy a Private Packagist subscription to help fund Composer development
 - <https://packagist.com>
- Join the Open Source Pledge
 - Commit to sponsoring open source for at least \$2000/year per FTE-equivalent developer
 - <https://opensourcepledge.com/>

Packagist.org

- Metadata only
 - No checksums for GitHub stored packages
 - <https://github.com/sansecio/composer-integrity-plugin>
 - No signatures
 - <https://www.drupal.org/project/infrastructure/issues/3325040> - TUF
 - No way to upload code
- Positively
 - Everything over TLS
 - Installation from GitHub source archive URLs improves trust in artifacts
 - Smaller attack surface on packagist.org

Composer Supply Chain Vulnerabilities

- Mar 11, 2021: Git Clone Security Vulnerability
 - <https://blog.packagist.com/git-clone-security-vulnerability/>
 - Git vulnerability on case insensitive filesystems can be exploited through Composer if you clone dependencies
- Apr 27, 2021: Composer Command Injection Vulnerability
 - <https://blog.packagist.com/composer-command-injection-vulnerability/>
 - Code execution through Mercurial repository URL injection
- Apr 13, 2022: Composer Command Injection Vulnerability
 - <https://blog.packagist.com/cve-2022-24828-composer-command-injection-vulnerability/>
 - Code execution through Git or Mercurial branch names

Composer Supply Chain Attacks

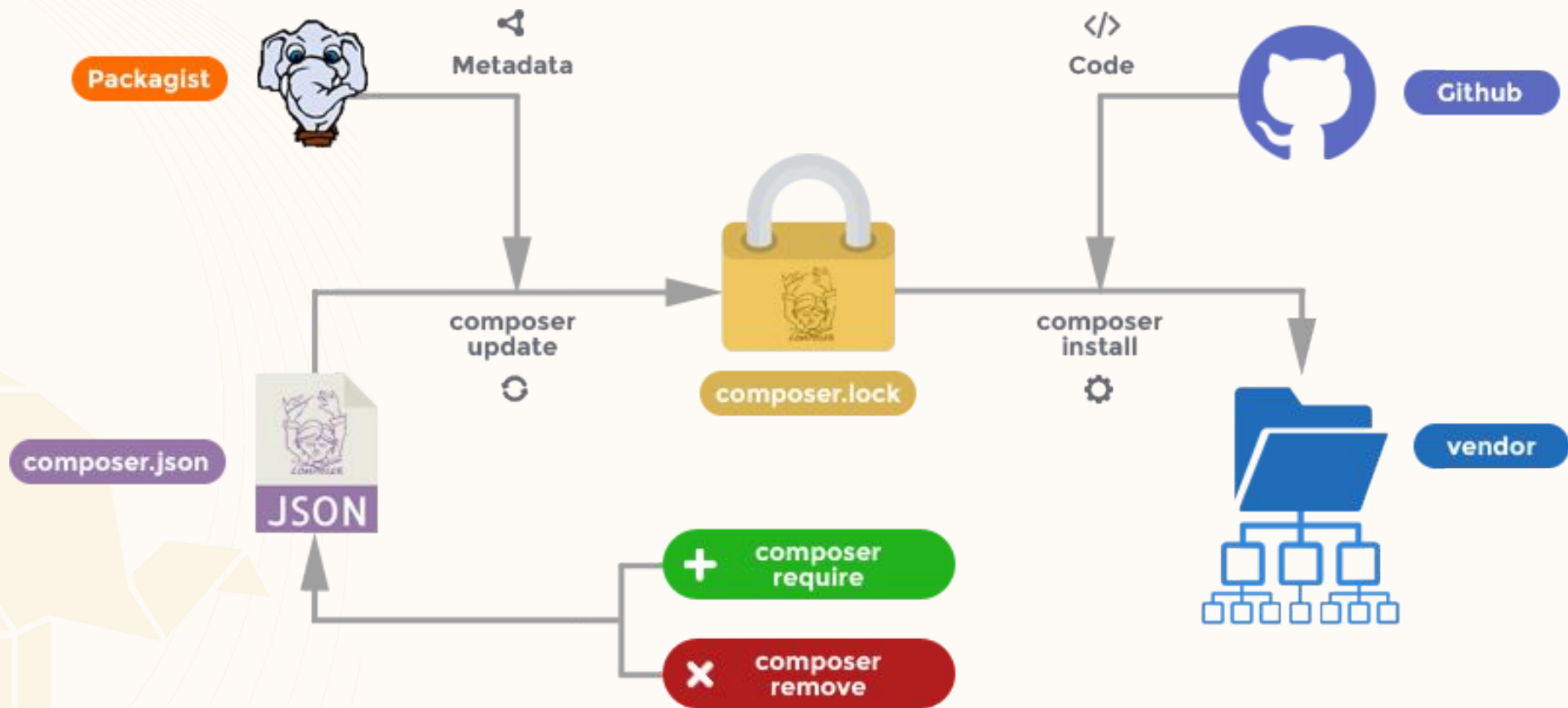
- May 19, 2022: GitHub Repo Jacking
 - Attacker registered GitHub username of former maintainer
 - Republished package with malicious code to steal AWS credentials
 - <https://thehackernews.com/2022/05/pypi-package-ctx-and-php-library-phpass.html>
 - <https://github.blog/2024-02-21-how-to-stay-safe-from-repo-jacking/>
 - Problematic with VCS repo URL references in composer.json too
 - Packagist.org uses GitHub repo ids: <https://github.com/composer/packagist/pull/1411>
- May 1, 2023: Packagist.org maintainer account takeover
 - <https://blog.packagist.com/packagist-org-maintainer-account-takeover/>
 - Editing of source URLs no longer allowed beyond 50k installs

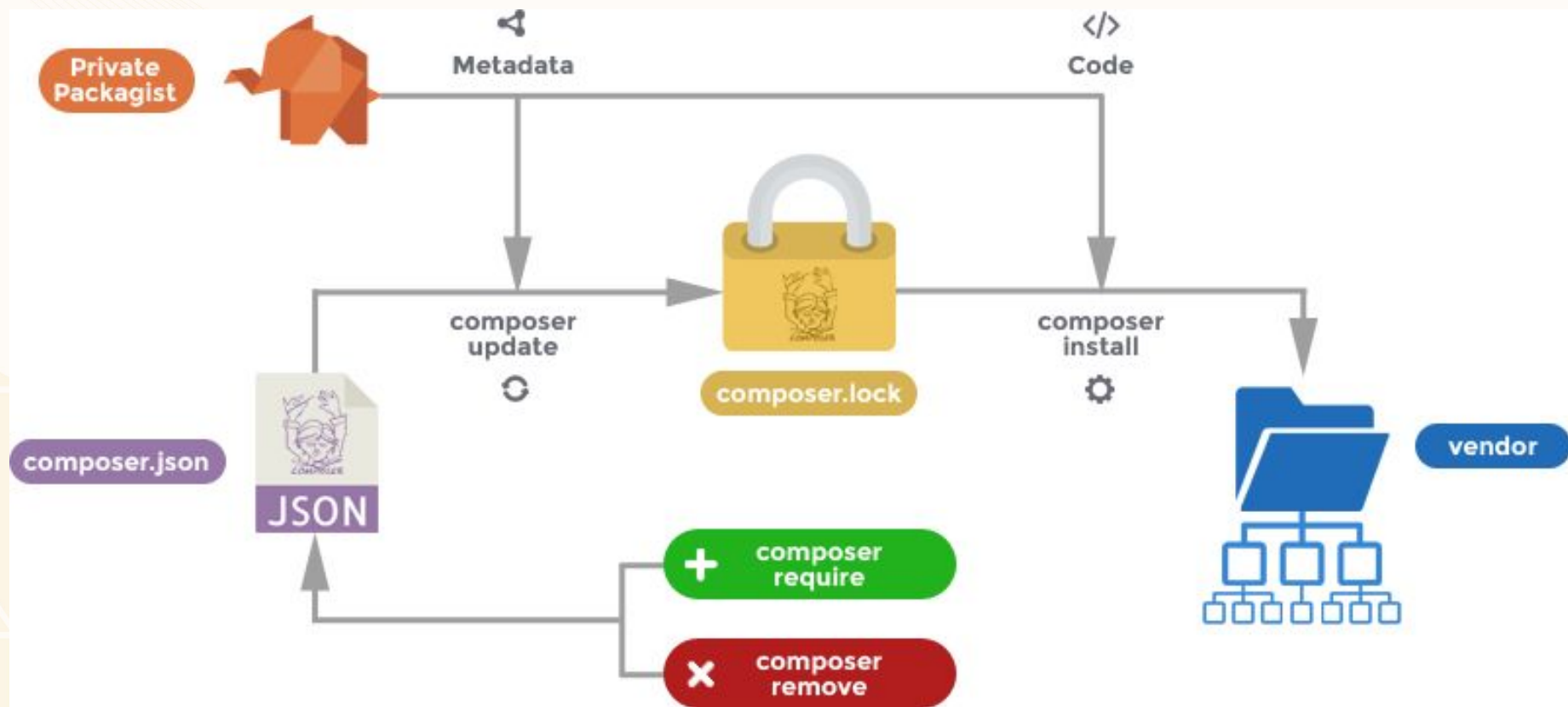
Use your own Composer repository

- Satis
- JFrog Artifactory
- Sonatype Nexus Repository
- Cloudsmith
- GitLab Package Registry
- ...
- **Private Packagist**

Private Packagist

- Stores a copy of all used versions of your dependencies
 - Safe from deletion
 - Safe from modification
- Serves package metadata **and** code
- Possible with some alternatives but usually with more effort and less convenience
 - e.g. copy all dependencies into git repositories, how do you keep those updated then?





Best Practice: Semantic Versioning

Promise of Compatibility

X.Y.Z

- Must be used consistently
 - Dare to increment **X**!
- Only valuable if BC/Compatibility promise formalized
 - See <https://symfony.com/doc/current/contributing/code/bc.html>
 - Document in Changelog

Versions Constraints

- | | | | |
|-------------------------------------|-------------------------|----------------------------------|----------------------------------|
| • Exact Match: | 1.0.0 | 1.2.3-beta2 | dev-main |
| • Wildcard Range: | 1.0.* | 2.* | |
| • Hyphen Range: | 1.0-2.0
>=1.0.0 <2.1 | 1.0.0 - 2.1.0
>=1.0.0 <=2.1.0 | |
| • <i>(Unbounded Range:
Bad!</i> | >= 1.0) | | |
| • Next Significant Release | ~1.2
>=1.2.0 <2.0.0 | ~1.2.3
>=1.2.3 <1.3.0 | |
| • Caret/Semver Operator | ^1.2
>=1.2.0 <2.0.0 | ^1.2.3
>=1.2.3 <2.0.0 | Best Choice for Libraries |

Operatoren: " " AND, "||" OR

Versions Constraints

- **Allowing multiple majors with minimum requirements**

`^1.1.4 || ^2.3.1 || ^3.0.2`

“Compatible with versions 1, 2 and 3, but only from the respective minimum versions”

- **Excluding broken versions**

`^1.1.3 !=1.4.1 !=1.7.2`

“^1.1.3 but not the broken versions 1.4.1 and 1.7.2”

Stabilities

- **Order**

dev -> alpha -> beta -> RC -> stable

- **Automatically from tags**

1.2.3

-> stable

1.3.0-beta3

-> beta

- **Automatically from branches**

Branch

-> Version (Stability)

2.0

-> 2.0.x-dev (dev)

Main

-> dev-main (dev)

myfeature

-> dev-myfeature (dev)

- **Stability selection**

"foo/bar": "1.3.*@beta"

"foo/bar": "2.0.x-dev"

"minimum-stability": "alpha"

Aliases

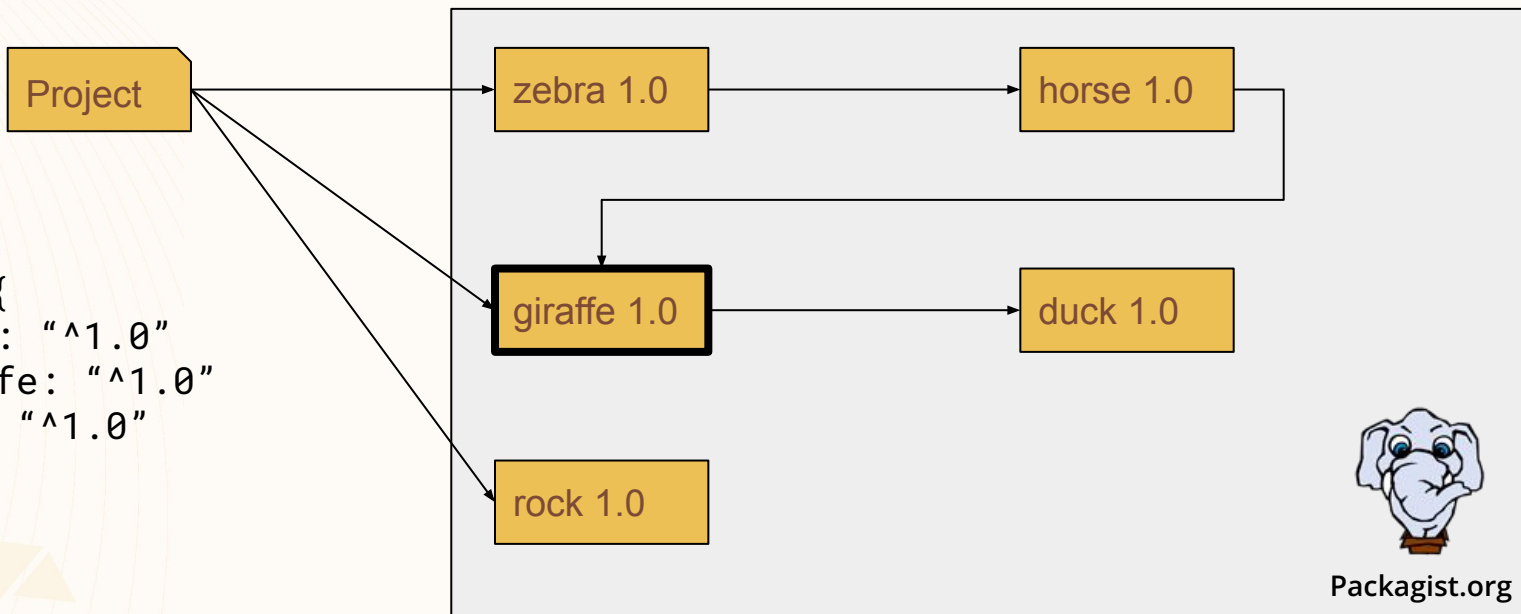
- Dependency
name: "dep/foo"
require: {
 "some/package": "^1.2.0"
}
- My project
require: {
 "dep/foo": "^1.0.0"
 "some/package": "dev-myfix as 1.2.3"
}

Forking

- Best option: Avoid forking
 - Send a PR upstream
 - Work with upstream maintainers
- Option 2: Temporary fork
 - Urgent fix
 - Can't wait for upstream merge
- Option 3: Permanent fork
 - Upstream no longer maintained
 - Upstream disagrees with your changes

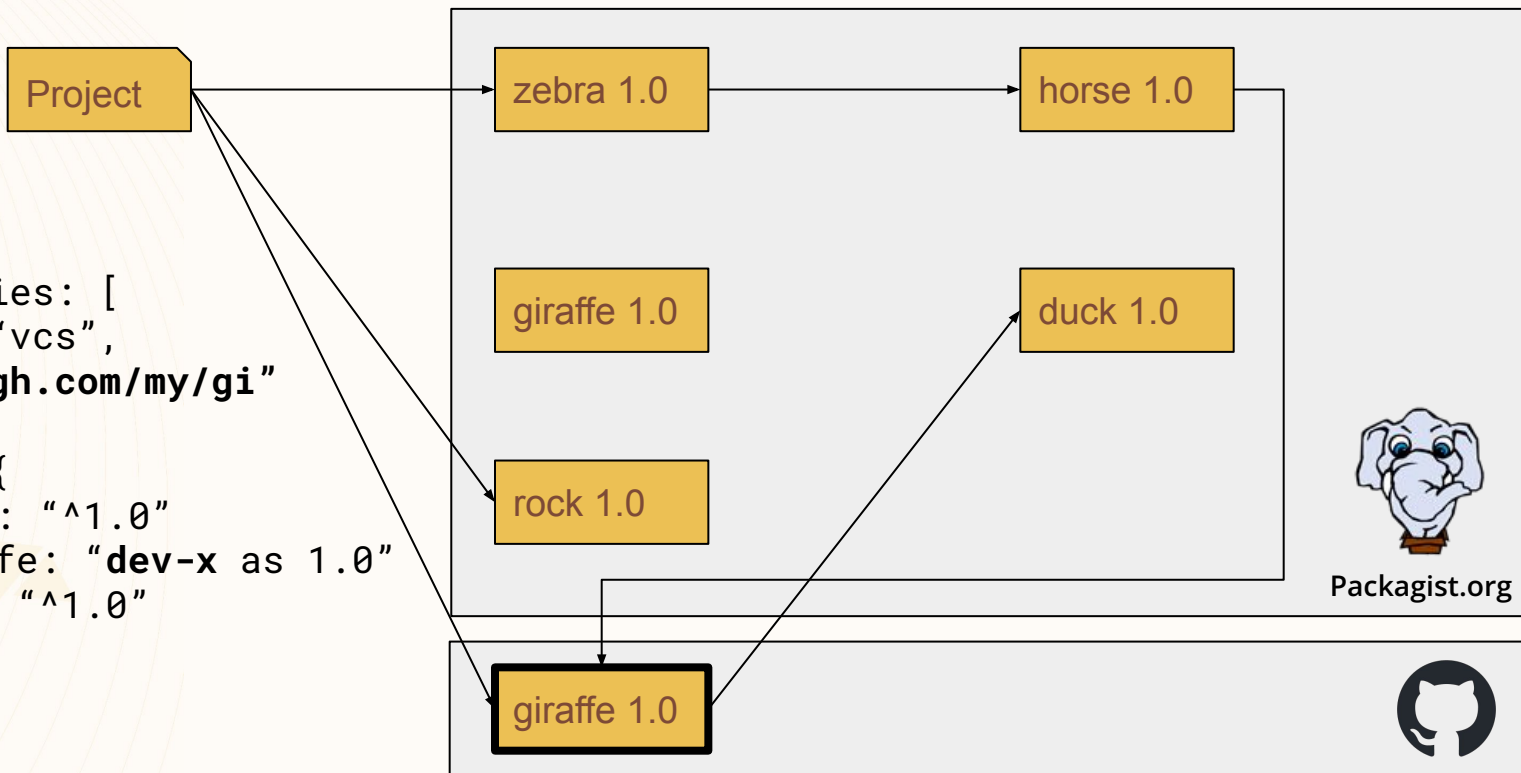
Forking: Temporary fork

```
require: {  
  zebra: "^1.0"  
  giraffe: "^1.0"  
  rock: "^1.0"  
}
```



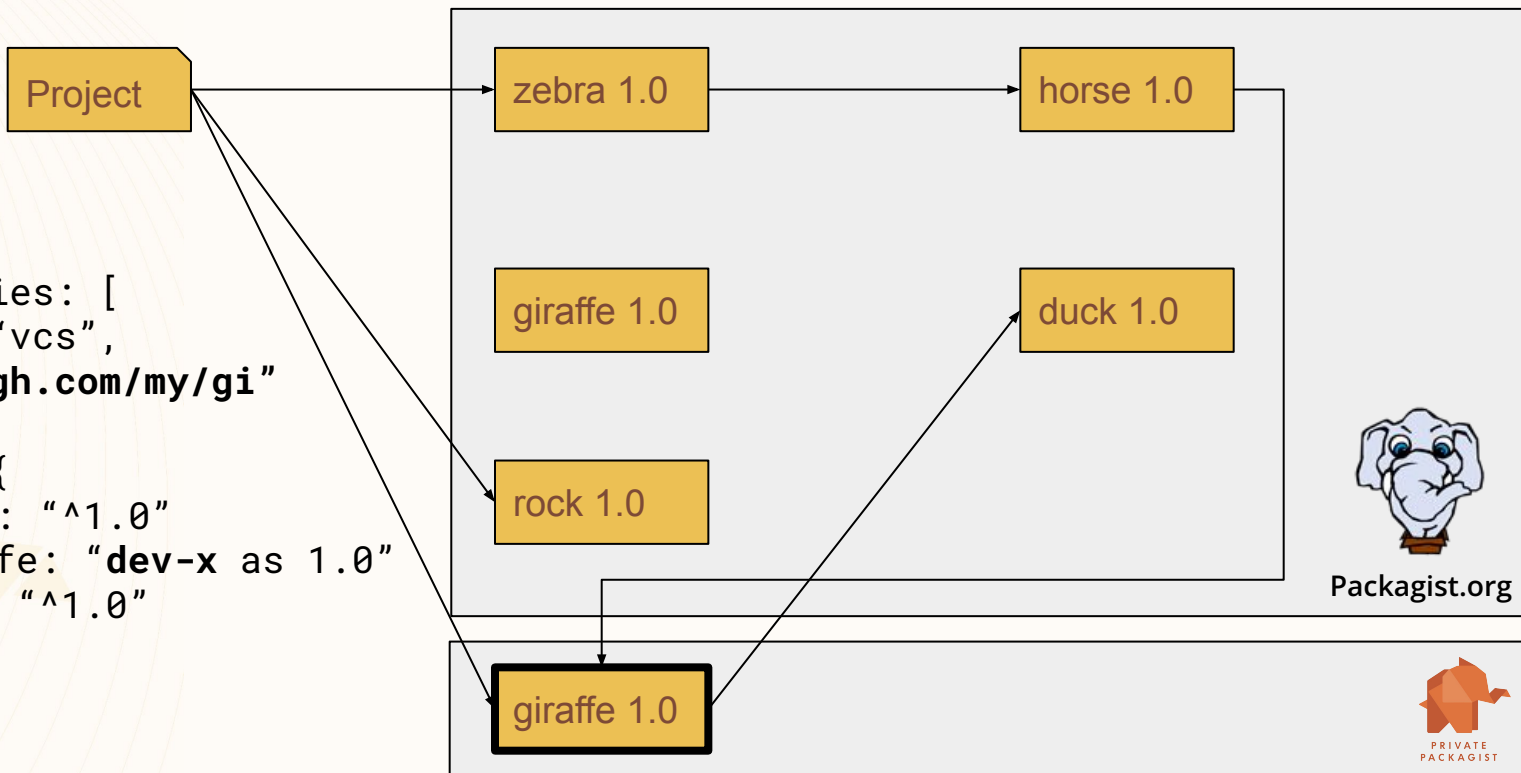
Forking: Temporary fork

```
repositories: [  
  type: "vcs",  
  url: "gh.com/my/gi"  
],  
require: {  
  zebra: "^1.0"  
  giraffe: "dev-x as 1.0"  
  rock: "^1.0"  
}
```

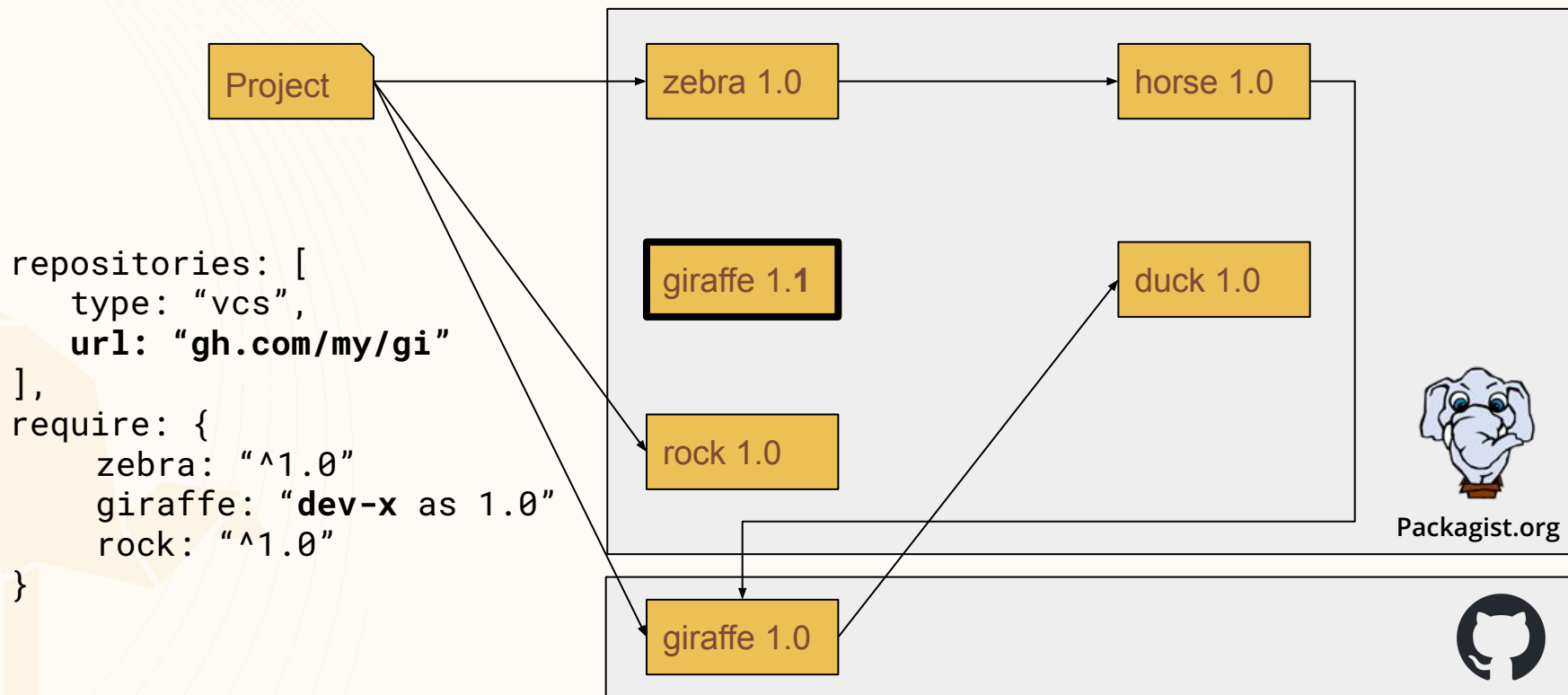


Forking: Temporary fork

```
repositories: [  
  type: "vcs",  
  url: "gh.com/my/gi"  
],  
require: {  
  zebra: "^1.0"  
  giraffe: "dev-x as 1.0"  
  rock: "^1.0"  
}
```



Forking: Temporary fork

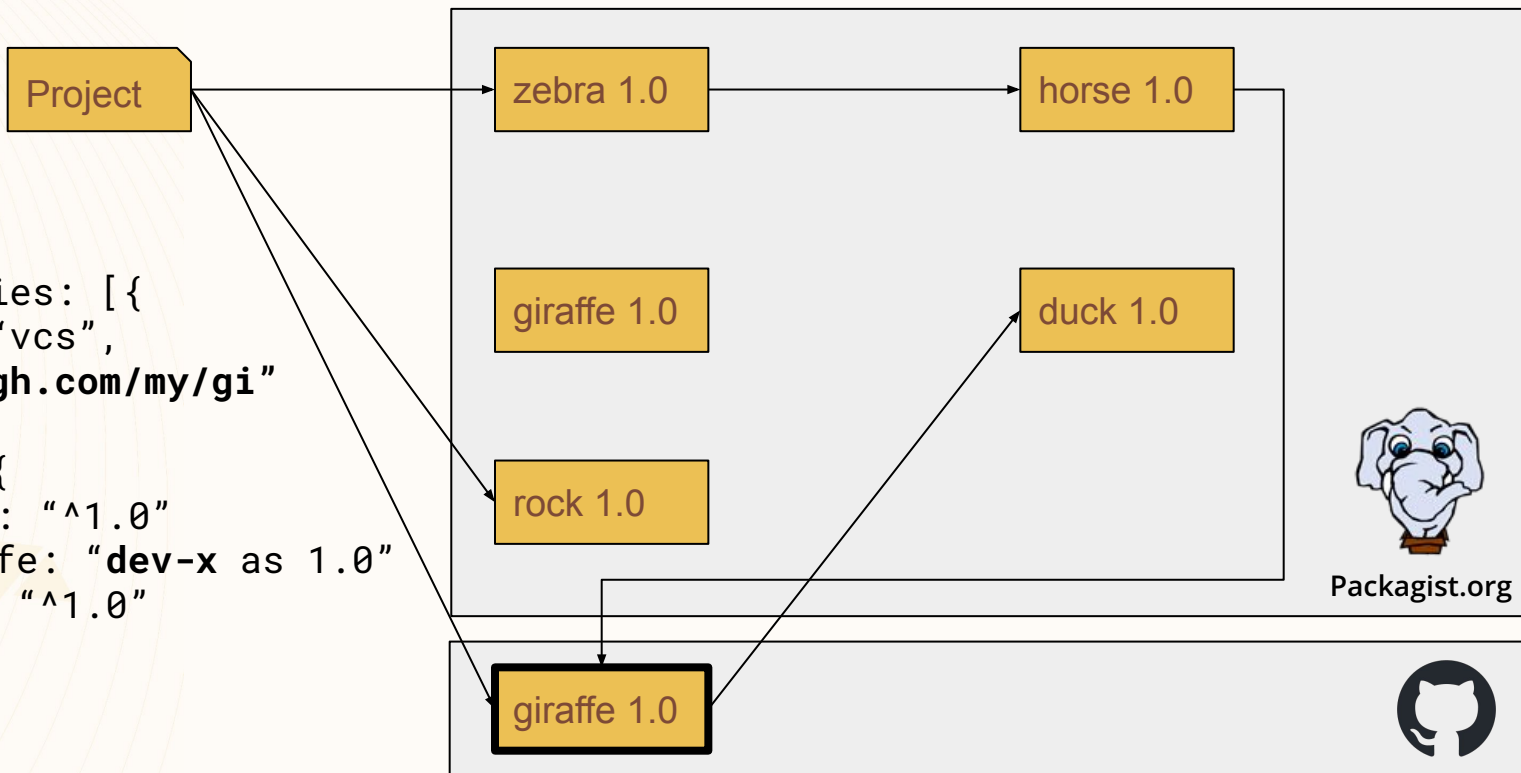


Forking: Temporary fork

- Good quick solution
- Problematic in the long run
 - Have to monitor upstream package for new releases
 - VCS repositories are slow
 - Can be solved by using your own Composer repository
 - **Private Packagist**
 - GitLab Packages
 - Sonatype Nexus Repository
 - JFrog Artifactory
 - etc.

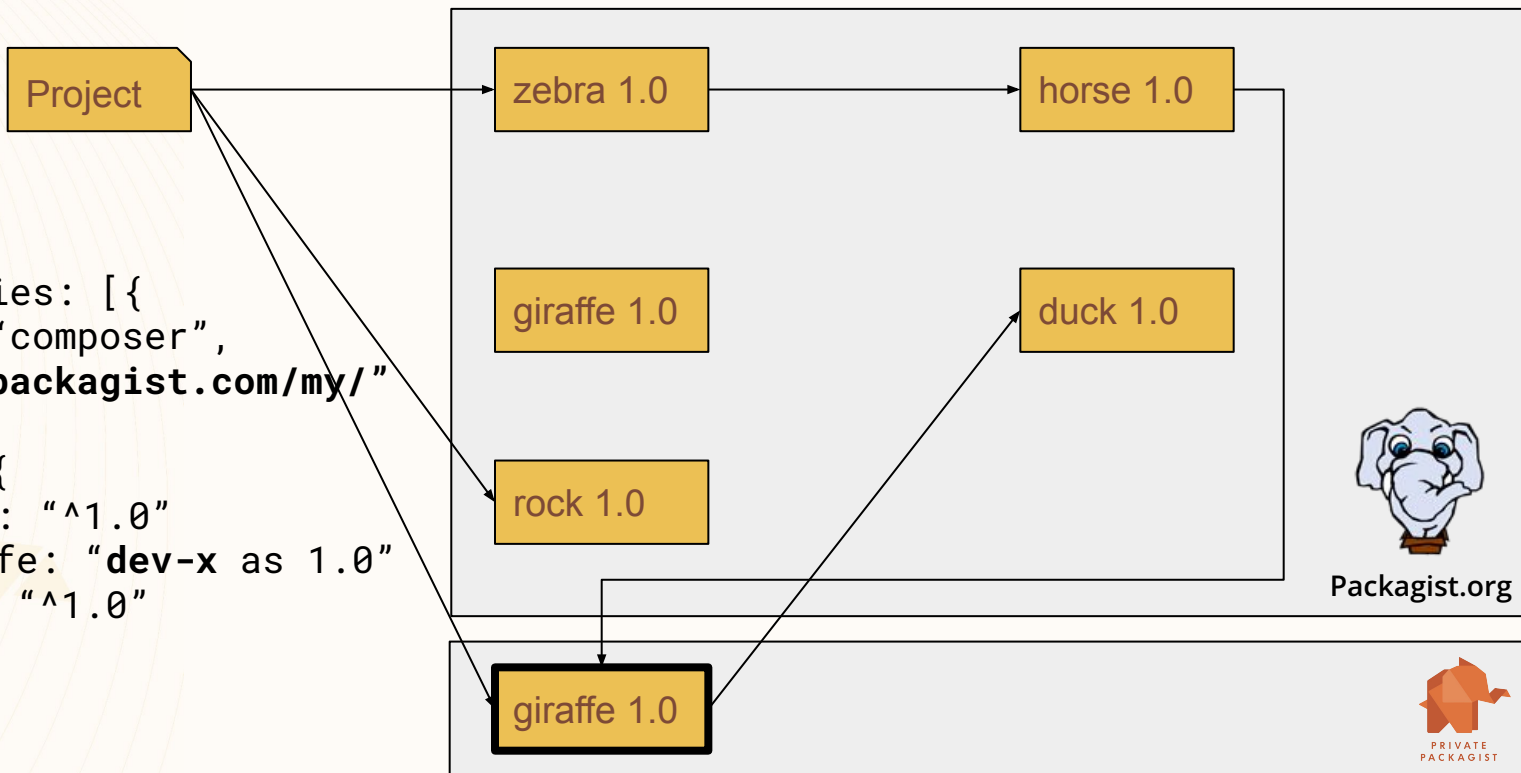
Forking: Temporary fork

```
repositories: [{  
  type: "vcs",  
  url: "gh.com/my/gi"  
}],  
require: {  
  zebra: "^1.0"  
  giraffe: "dev-x as 1.0"  
  rock: "^1.0"  
}
```



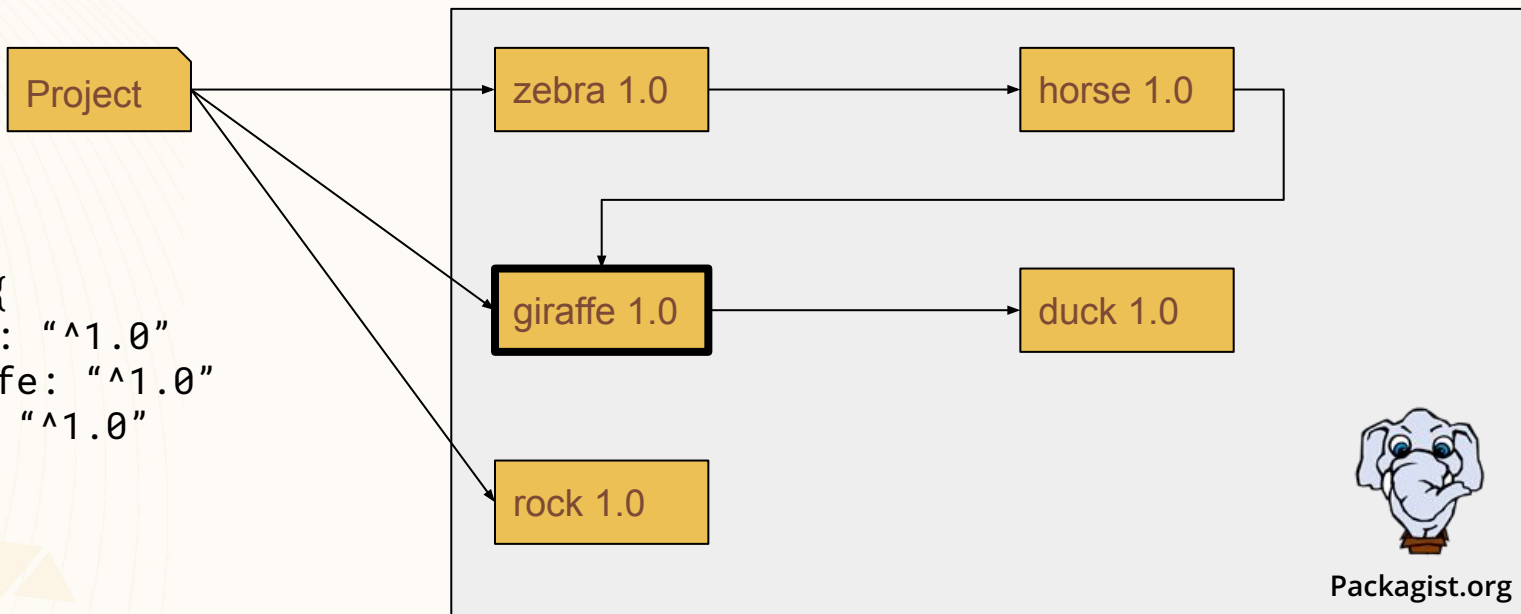
Forking: Temporary fork

```
repositories: [{  
  type: "composer",  
  url: "packagist.com/my/"  
}],  
require: {  
  zebra: "^1.0"  
  giraffe: "dev-x as 1.0"  
  rock: "^1.0"  
}
```



Forking: Permanent fork

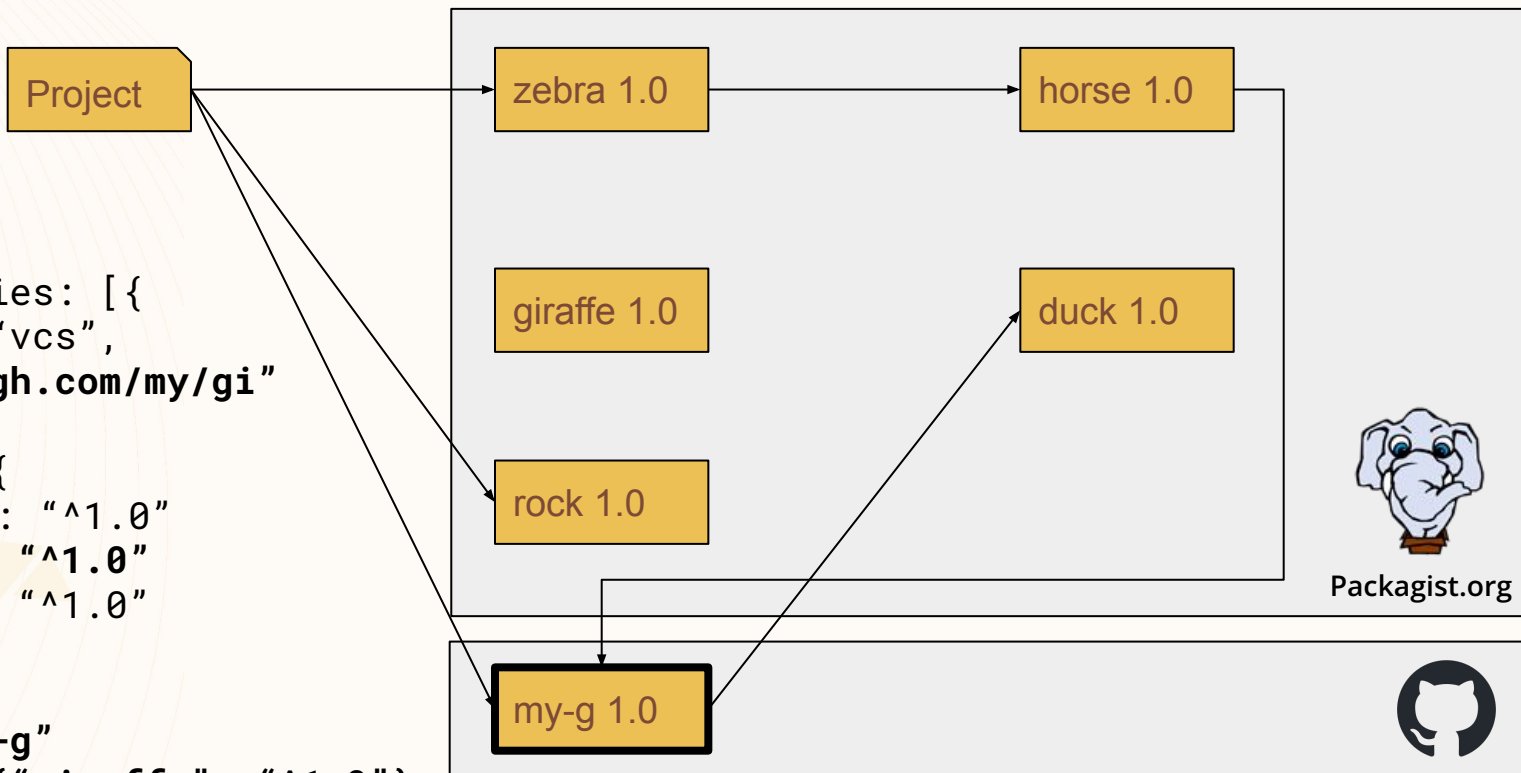
```
require: {  
  zebra: "^1.0"  
  giraffe: "^1.0"  
  rock: "^1.0"  
}
```



Forking: Temporary fork

```
repositories: [{  
  type: "vcs",  
  url: "gh.com/my/gi"  
}],  
require: {  
  zebra: "^1.0"  
  my-g: "^1.0"  
  rock: "^1.0"  
}
```

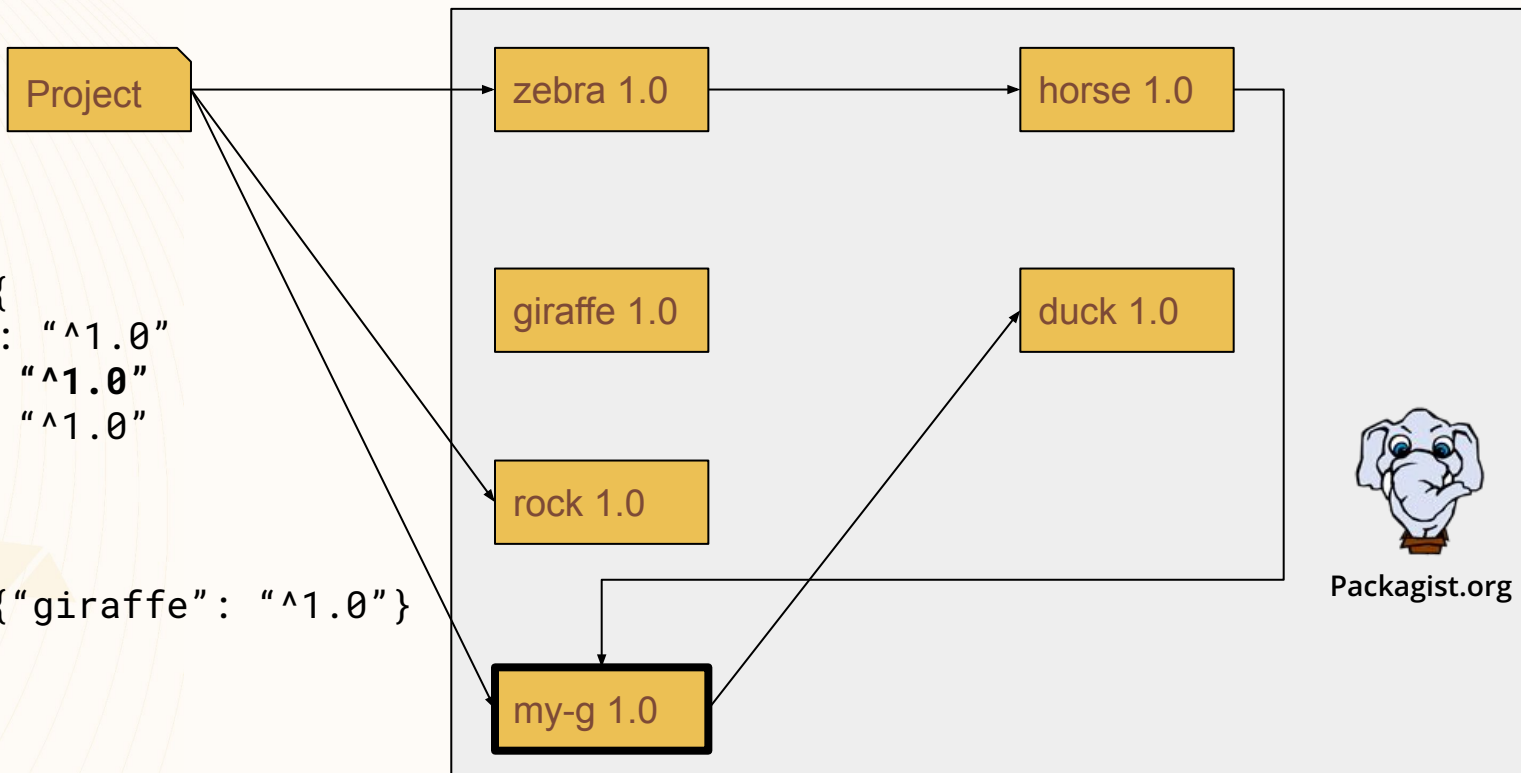
```
name: "my-g"  
replace: {"giraffe": "^1.0"}
```



Forking: Temporary fork

```
require: {  
  zebra: "^1.0"  
  my-g:  "^1.0"  
  rock:  "^1.0"  
}
```

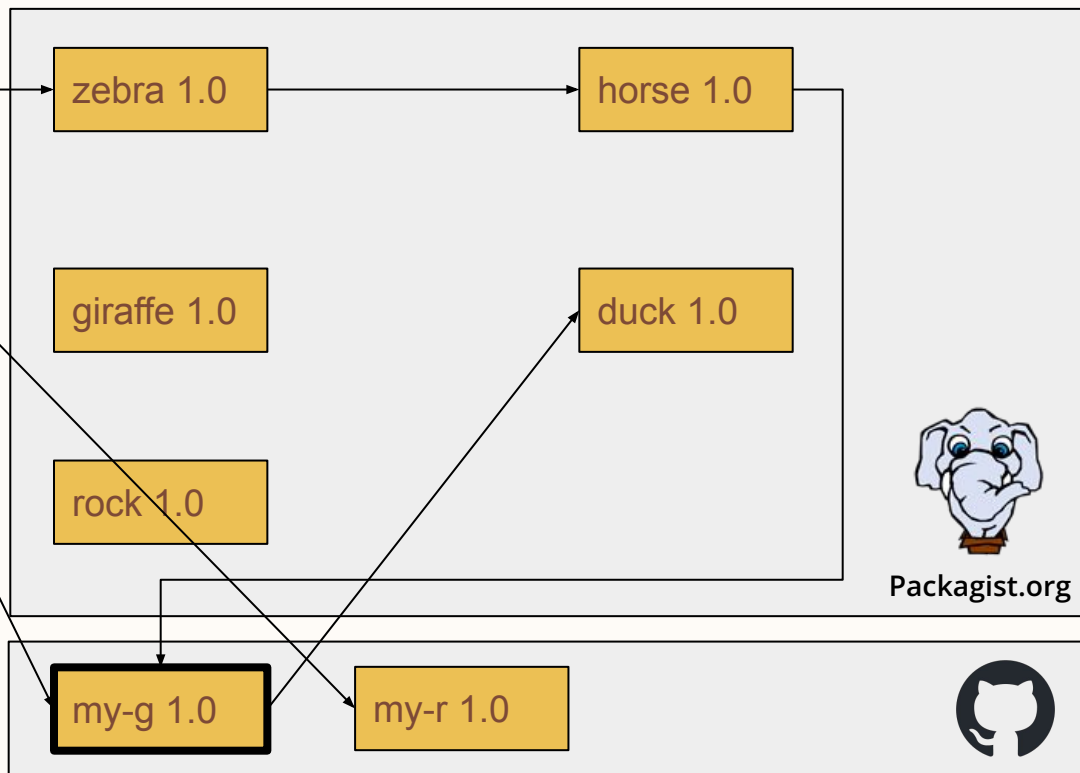
```
my-g:  
  replace: {"giraffe": "^1.0"}
```



Forking: Temporary fork

```
repositories: [{  
  type: "vcs",  
  url: "gh.com/my/gi"  
}, {  
  type: "vcs",  
  url: "gh.com/my/ro"  
}],  
require: {  
  zebra: "^1.0"  
  my-g: "^1.0"  
  my-ro: "^1.0"  
}
```

```
name: "my-g"  
replace: {"giraffe": "^1.0"}
```

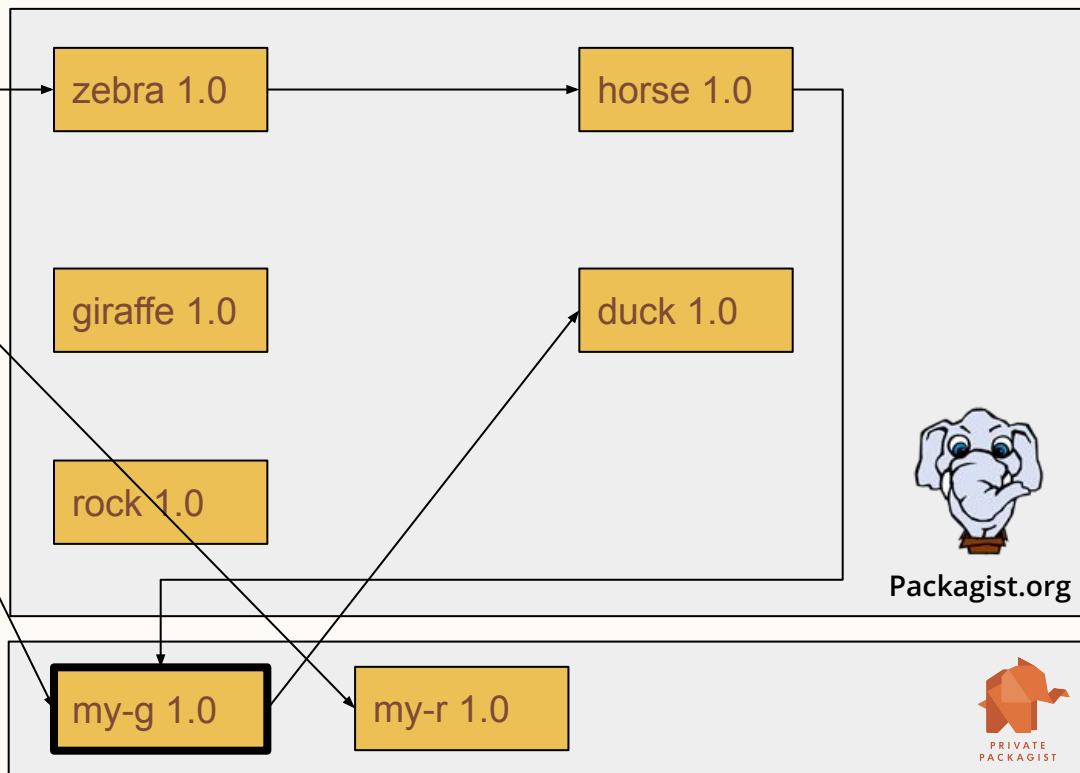


Forking: Temporary fork

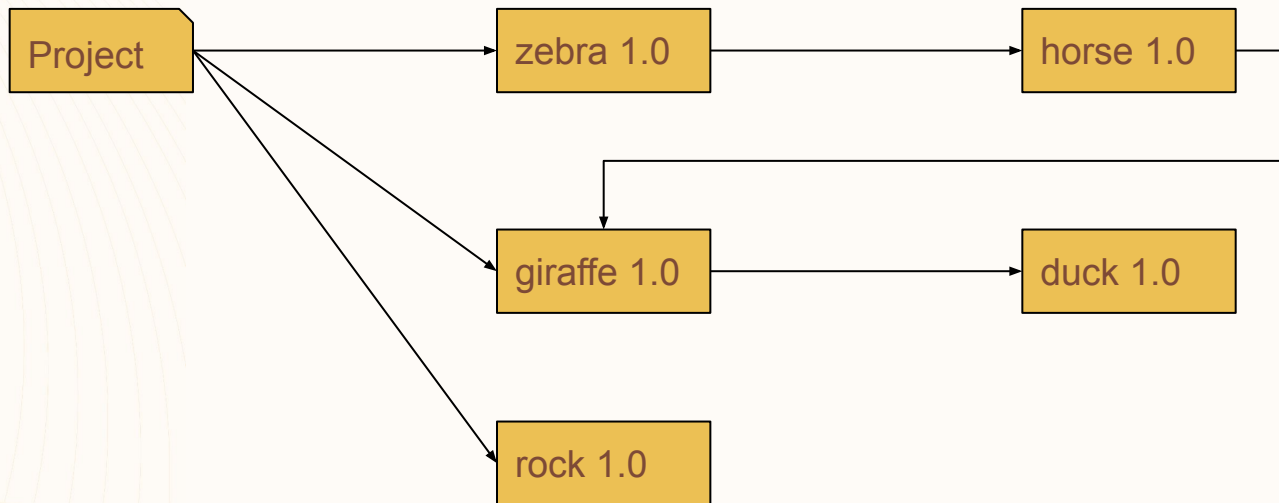
```
repositories: [{  
  type: "composer",  
  url: "packagist.com/my/"  
}],
```

```
require: {  
  zebra: "^1.0"  
  my-g: "^1.0"  
  my-ro: "^1.0"  
}
```

```
name: "my-g"  
replace: {"giraffe": "^1.0"}
```

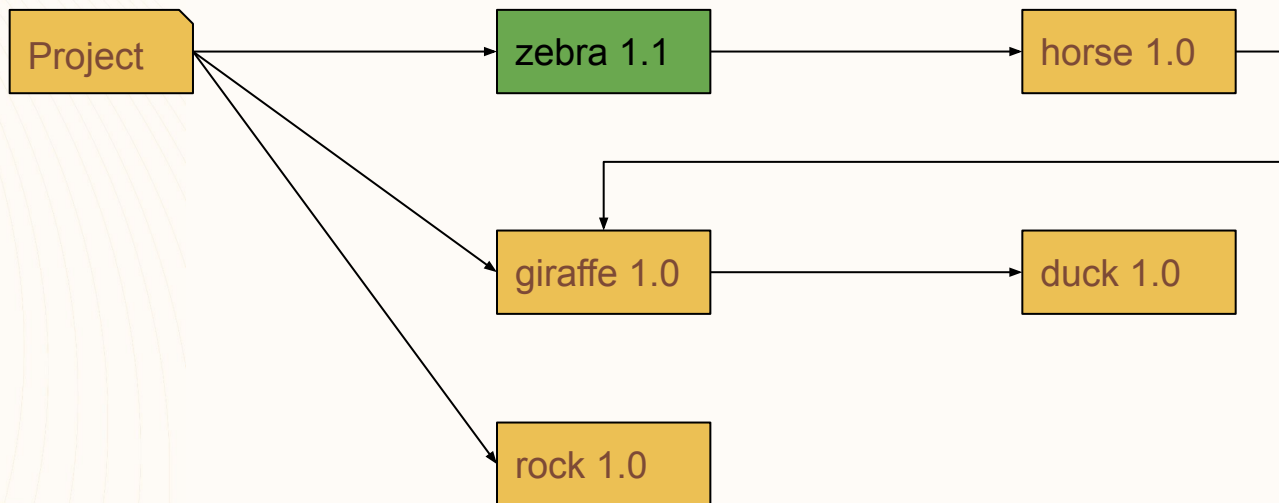


Partial Updates



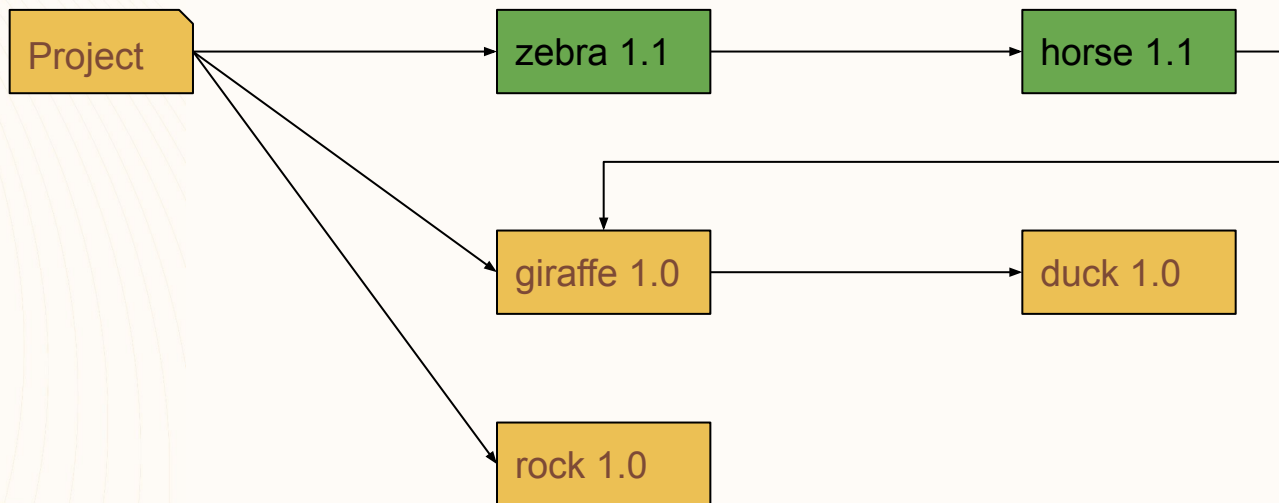
Now each package releases 1.1

Partial Updates



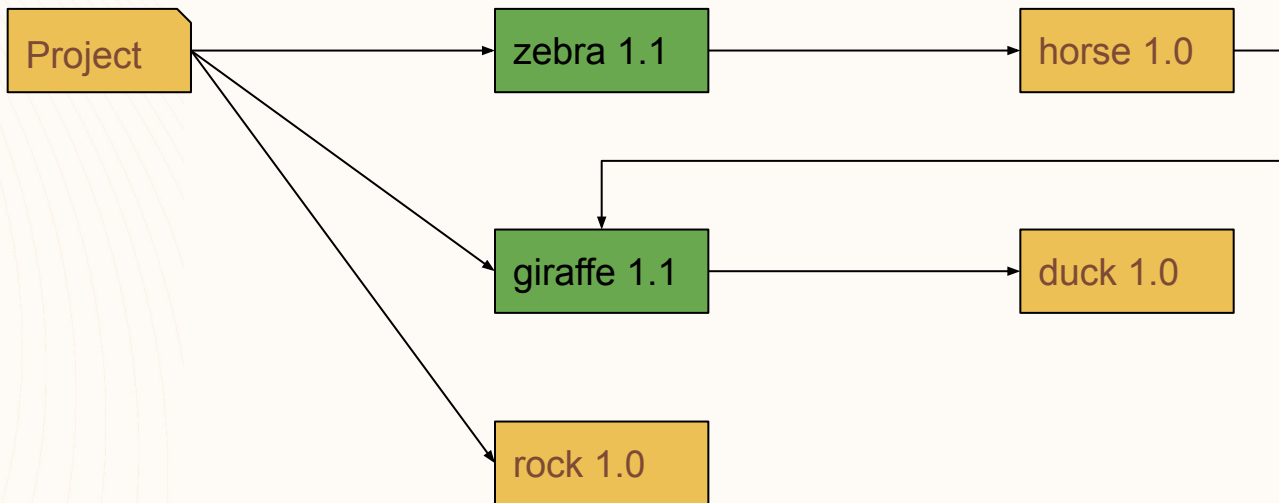
```
$ composer update --dry-run zebra/zebra  
Updating zebra/zebra (1.0 -> 1.1)
```

Partial Updates



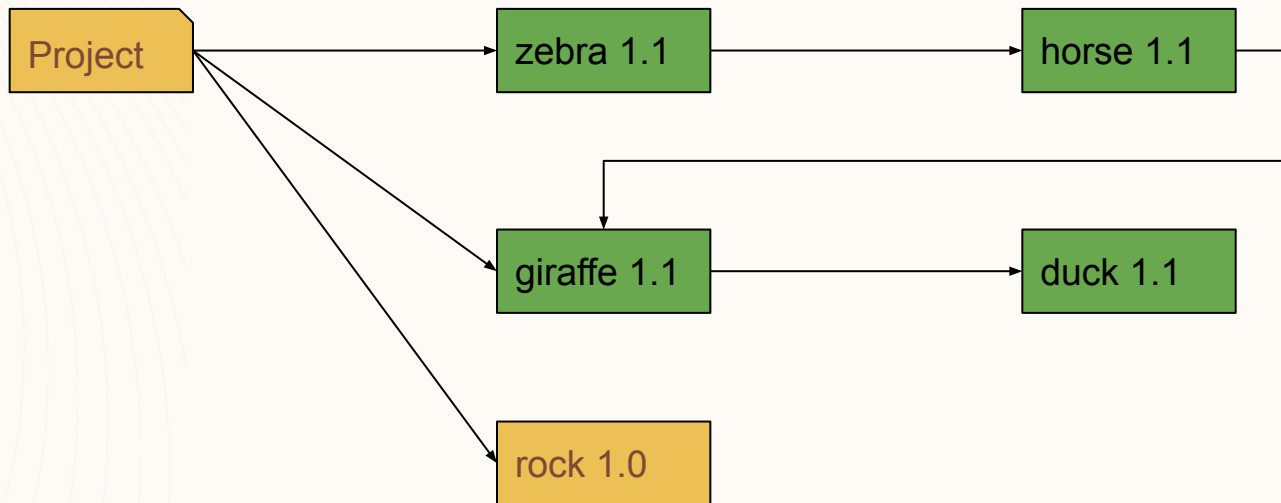
```
$ composer update --dry-run zebra/zebra --with-dependencies
Updating horse/horse (1.0 -> 1.1)
Updating zebra/zebra (1.0 -> 1.1)
```

Partial Updates



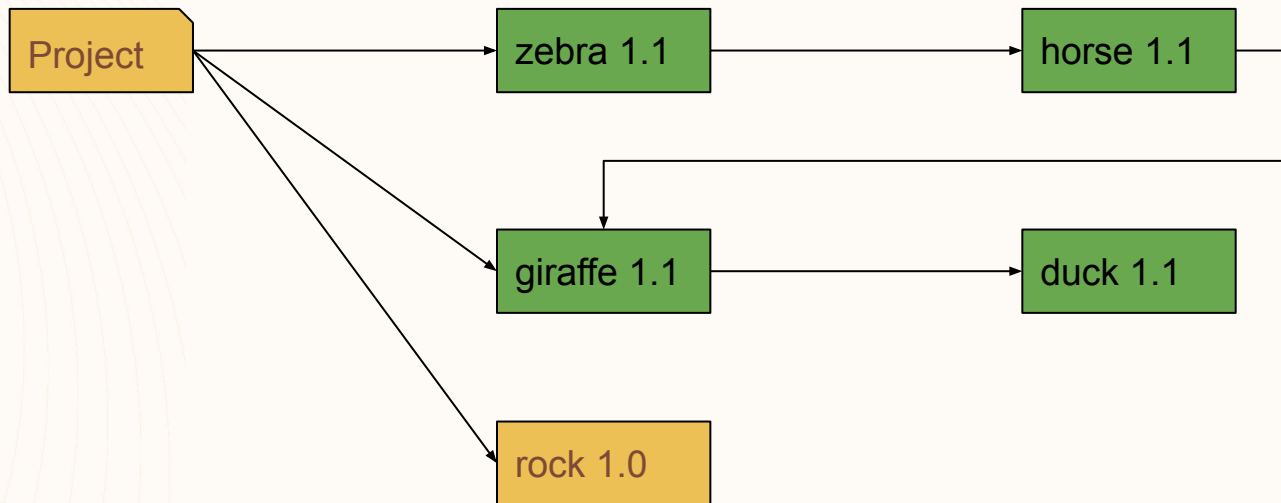
```
$ composer update --dry-run zebra/zebra giraffe/giraffe
Updating zebra/zebra (1.0 -> 1.1)
Updating giraffe/giraffe (1.0 -> 1.1)
```

Partial Updates



```
$ composer update zebra/zebra giraffe/giraffe --with-dependencies
Updating duck/duck (1.0 -> 1.1)
Updating giraffe/giraffe (1.0 -> 1.1)
Updating horse/horse (1.0 -> 1.1)
Updating zebra/zebra (1.0 -> 1.1)
```

Partial Updates



```
$ composer update zebra/zebra --with-all-dependencies
```

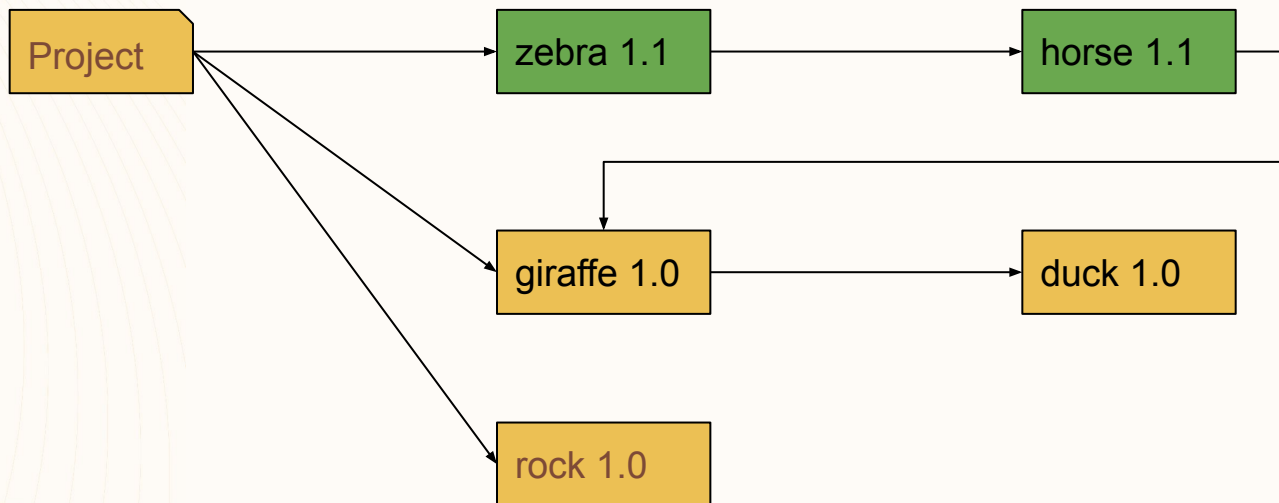
```
Updating duck/duck (1.0 -> 1.1)
```

```
Updating giraffe/giraffe (1.0 -> 1.1)
```

```
Updating horse/horse (1.0 -> 1.1)
```

```
Updating zebra/zebra (1.0 -> 1.1)
```


Partial Updates

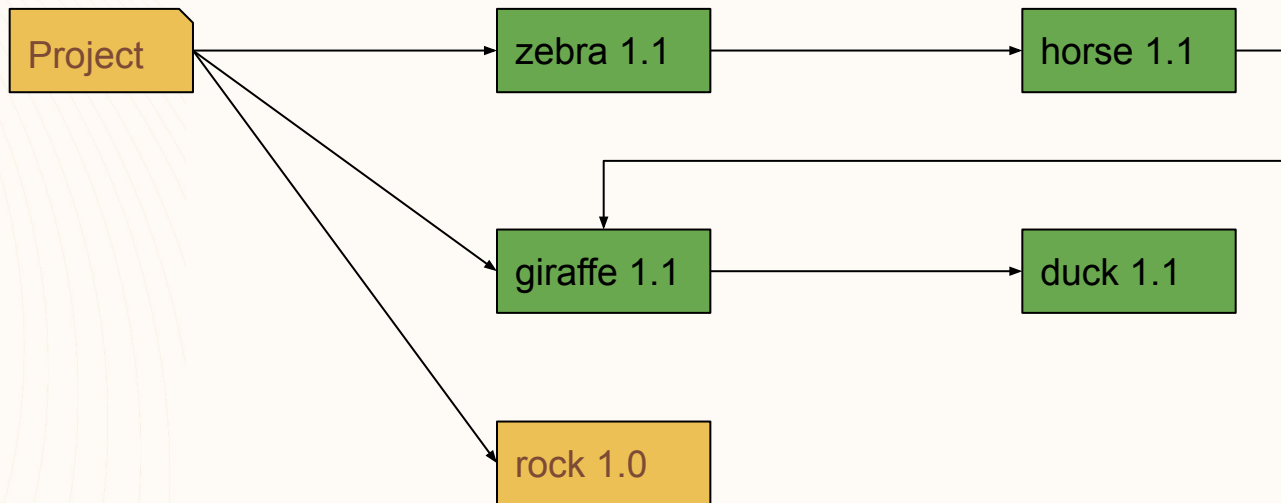


```
$ composer update zebra/zebra --with-dependencies
```

```
Updating horse/horse (1.0 -> 1.1)
```

```
Updating zebra/zebra (1.0 -> 1.1)
```

Partial Updates



```
$ composer update zebra/zebra --with-all-dependencies
```

```
Updating duck/duck (1.0 -> 1.1)
```

```
Updating giraffe/giraffe (1.0 -> 1.1)
```

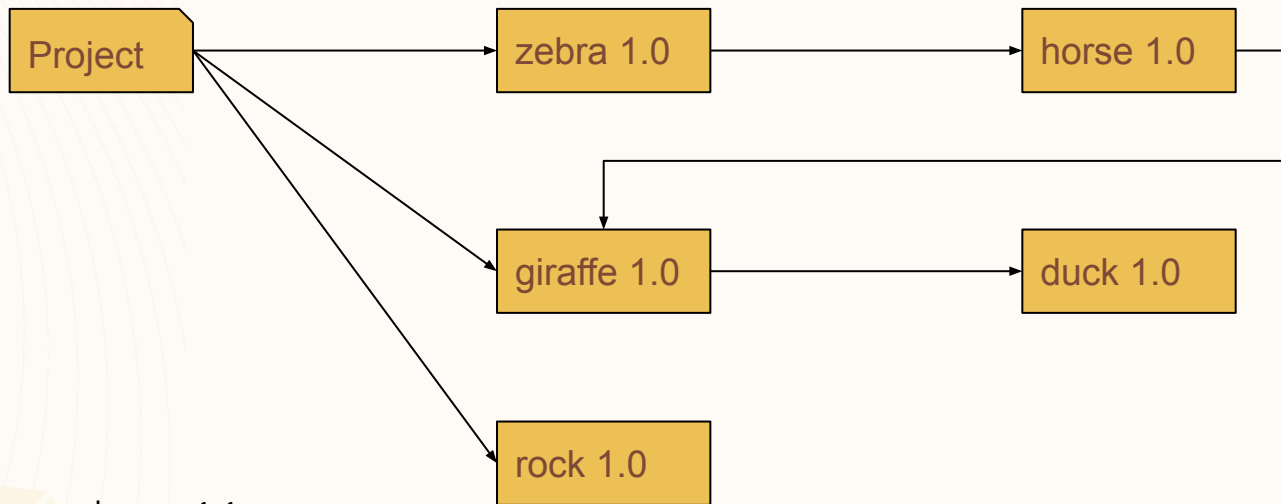
```
Updating horse/horse (1.0 -> 1.1)
```

```
Updating zebra/zebra (1.0 -> 1.1)
```

update --minimal-changes

- Since Composer 2.7
- Problem
 - I want to update one dependency
 - There's a conflict
 - I need to update more, **but** I don't want to update everything
- Solution
 - Partial updates with dependencies **but** keeping them at the same version as the lock file if possible

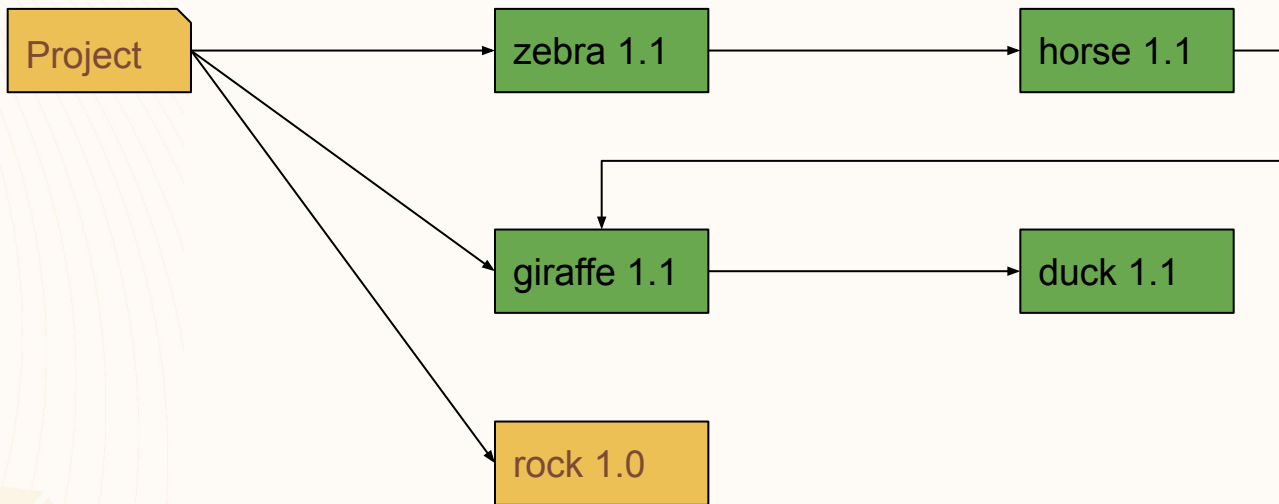
update --minimal-changes



Now each package releases 1.1

- zebra 1.1 requires horse ^1.1
- horse 1.1 requires giraffe ^1.1
- giraffe 1.1 still requires **duck ^1.0**

update --minimal-changes



```
$ composer update zebra/zebra --with-all-dependencies
```

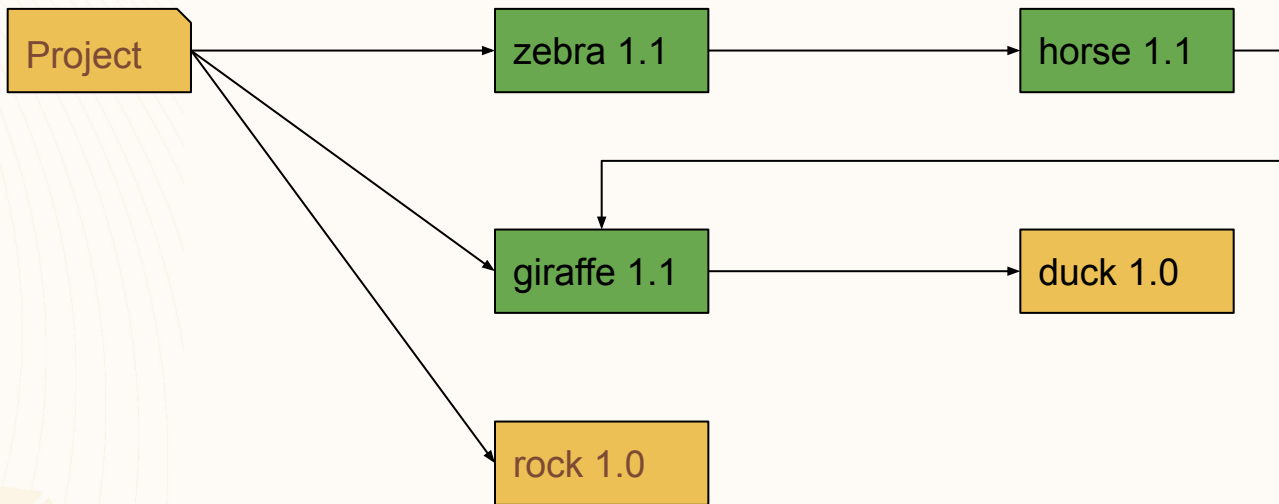
```
Updating duck/duck (1.0 -> 1.1)
```

```
Updating giraffe/giraffe (1.0 -> 1.1)
```

```
Updating horse/horse (1.0 -> 1.1)
```

```
Updating zebra/zebra (1.0 -> 1.1)
```

update --minimal-changes

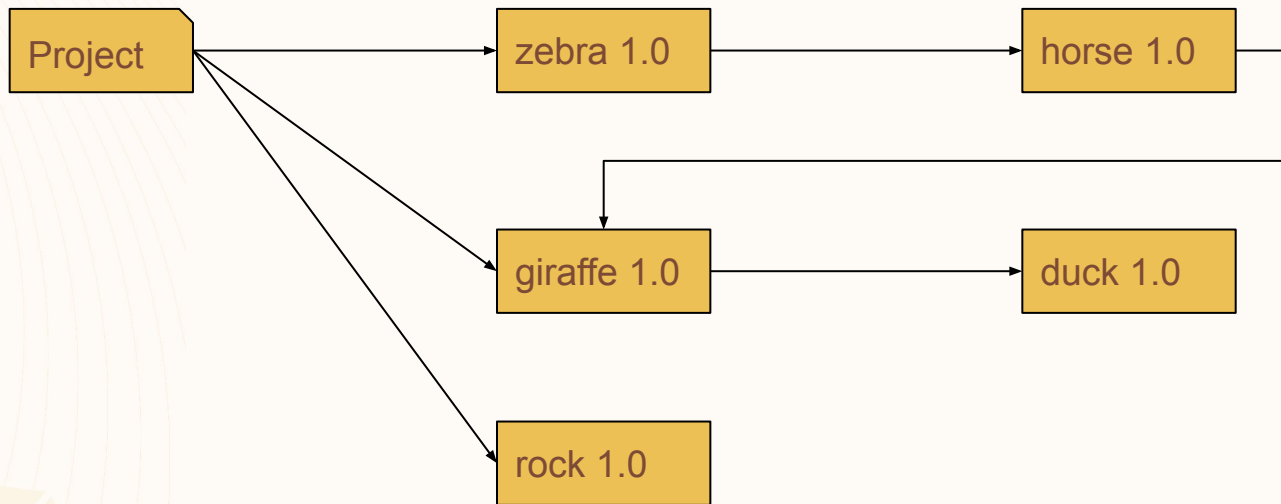


```
$ composer update zebra/zebra --with-all-dependencies --minimal-changes
Updating giraffe/giraffe (1.0 -> 1.1)
Updating horse/horse (1.0 -> 1.1)
Updating zebra/zebra (1.0 -> 1.1)
```

update --patch-only

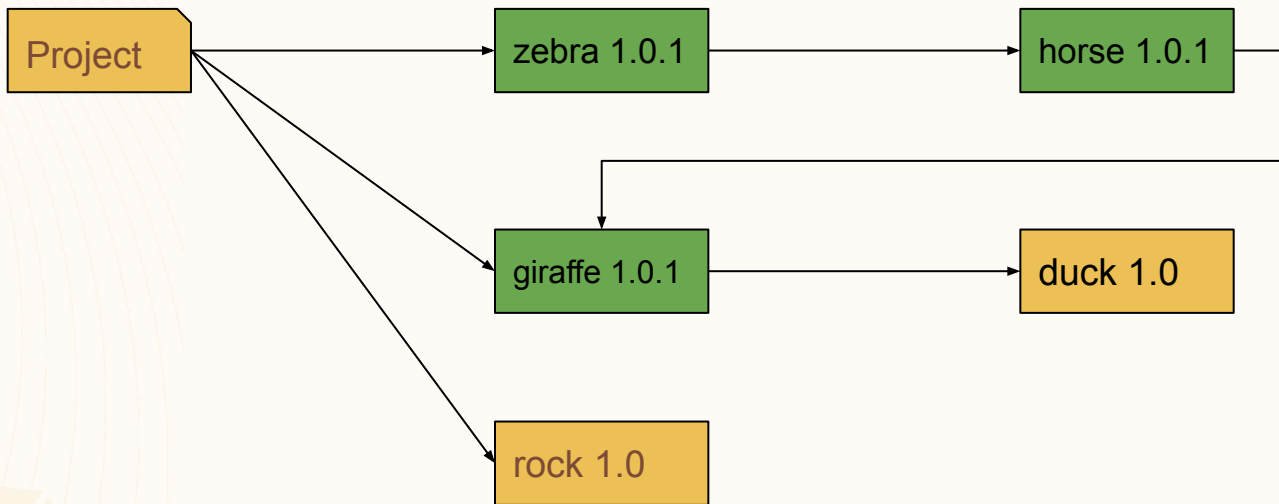
- Since Composer 2.8
- Only update **Z** in semver X.Y.**Z**
- Can be combined with other update options

update --patch-only



Now each package releases **1.1** and **1.0.1**

update --patch-only



```
$ composer update zebra/zebra --with-all-dependencies --minimal-changes --patch-only
Updating giraffe/giraffe (1.0 -> 1.0.1)
Updating horse/horse (1.0 -> 1.0.1)
Updating zebra/zebra (1.0 -> 1.0.1)
```

composer bump

- Since Composer 2.4
- `update --bump-after-update` since Composer 2.8
- Increases lower limits of `composer.json` constraints to current versions

```
"require": { "zebra/zebra": "^1.2.3" }
```

```
$ composer update --bump-after-update  
Updating zebra/zebra (1.3.5 -> 1.4.2)
```

```
"require": { "zebra/zebra": "^1.4.2" }
```

- Do not use on libraries for others, will limit compatibility

Composer 2.4: composer audit

- **composer audit** Command
 - Lists vulnerable versions in composer.lock
 - Uses packagist.org vulnerability db API
 - GitHub advisory database
 - FriendsOfPHP/security-advisories
 - Returns non-zero if vulnerabilities found -> can check in CI
- `composer update` implies `audit --format=summary`
- `composer require --dev roave/security-advisories:dev-latest`

Update Dependencies Frequently

- Smaller changes
 - Smaller likelihood of failures
 - Less fear to update
 - Less chance of falling behind a lot
- Set up a schedule or regular reminder to run dependency updates
- Set up alerting for discovered vulnerabilities in deps:

SCA tools (Software Composition Analysis)

- GitHub Dependabot
- Snyk
- Aikido
- Mend SCA
- **Private Packagist** Security Monitoring
- many more

Update Dependencies Frequently

Better yet: Automate your updates


- Mend Renovate <https://www.mend.io/renovate/>
- GitHub Dependabot <https://github.com/dependabot>


Get a pull request anytime an update is necessary

Caution!

Private Packagist Update Review

GitHub
BitBucket
GitLab


 Open


Update php-dev-dependencies #2794
renovate wants to merge 2 commits into [main](#) from [renovate/php-dev-depend...](#) 


This PR contains the following updates:

Package	Type	Update	Change
friendsofphp/php-cs-fixer	require-dev	minor	3.13.2 -> 3.16.0
friendsofphp/php-cs-fixer	require-dev	minor	3.14.1 -> 3.16.0
phpstan/phpstan-symfony	require-dev	patch	1.3.1 -> 1.3.2

This PR has been generated by [Mend Renovate](#). View repository job log [here](#).



 Update php-dev-dependencies Verified ✓ e6f84d0

 **private-packagist** bot commented 27 minutes ago • edited ▾

accounting/composer.lock

Dev Package changes

Package	Operation	From	To	Changes
friendsofphp/php-cs-fixer	upgrade	v3.13.2	v3.14.1	diff - changelog
phpstan/phpstan-symfony	upgrade	1.3.1	1.3.2	diff - changelog

core/composer.lock

Package changes **NOT DEV**

Package	Operation	From	To	Changes
psr/cache	upgrade	2.0.0	3.0.0	diff - changelog
symfony/cache-contracts	upgrade	v2.5.2	v3.2.1	diff - changelog

Update Dependencies Frequently

Better yet: Automate your updates

- Mend Renovate <https://www.mend.io/renovate/>
- GitHub Dependabot <https://github.com/dependabot>
- **Conductor by Private Packagist** <https://packagist.com/features/conductor>

Get a pull request anytime an update is necessary

Introducing



Conductor

By  PRIVATE PACKAGIST

**Automatic dependency
updates for Composer**

Sign up now for Early Access

Merged

[Conductor] [repo] Update symfony/flex to v2.5.0 #6611

glaubinix merged 1 commit into [main](#) from [conductor-symfony-flex-...](#)  last month

This PR was automatically generated by [Conductor](#).

The PR contains the changes generated by running the following command:

```
composer update --with-all-dependencies --minimal-changes symfony/flex:v2.5.0
```



Changelog

► symfony/flex (Source: [GitHub Releases](#))

► Task options

Powered by [Private Packagist](#)



  composer update --with-all-dependencies --minimal-changes symfony/fle... 

Verified

✓ 83602d4



private-packagist  commented on Mar 4

Author



repo/composer.lock

Package changes


Package	Operation	From	To	About
symfony/flex	upgrade	v2.4.7	v2.5.0	diff - changelog

[Settings](#) · [Docs](#) · Powered by [Private Packagist](#)

[Conductor] [core] Update all of symfony #6827

Merged stevenrombaults merged 1 commit into main from conductor-symfony-all-35501 3 days ago

Conversation 1 Commits 1 Checks 14 Files changed 1

 private-packagist bot commented last week

This PR was automatically generated by [Conductor](#).

The PR contains the changes generated by running the following command:

```
composer update --with-all-dependencies --minimal-changes symfony/cache:v7.2.5 symfony/console:v7.2.5 symfony/dependency-injection:v7.2.5 symfony/doctrine-bridge:v7.2.5 symfony/error-handler:v7.2.5 symfony/form:v7.2.5 symfony/framework-bundle:v7.2.5 symfony/http-foundation:v7.2.5 symfony/http-kernel:v7.2.5 symfony/messenger:v7.2.5 symfony/process:v7.2.5 symfony/property-info:v7.2.5 symfony/serializer:v7.2.5 symfony/twig-bridge:v7.2.5 symfony/type-info:v7.2.5 symfony/validator:v7.2.5 symfony/var-exporter:v7.2.5 symfony/yaml:v7.2.5
```

Changelog

Inline changelog information is available for pull requests updating up to three dependencies.


► Task options

Powered by [Private Packagist](#)

😊

composer update --with-all-dependencies --minimal-changes symfony/cache:v7.2.5 symfony/console:v7.2.5 symfony/dependency-injection:v7.2.5 symfony/doctrine-bridge:v7.2.5 symfony/error-handler:v7.2.5 symfony/form:v7.2.5 symfony/framework-bundle:v7.2.5 symfony/http-foundation:v7.2.5 symfony/http-kernel:v7.2.5 symfony/messenger:v7.2.5 symfony/process:v7.2.5 symfony/property-info:v7.2.5 symfony/serializer:v7.2.5 symfony/twig-bridge:v7.2.5 symfony/type-info:v7.2.5 symfony/validator:v7.2.5 symfony/var-exporter:v7.2.5 symfony/yaml:v7.2.5

Verified ✓ 205dd1b

 private-packagist bot commented last week

core/composer.lock

Merged [Conductor] [core] Update all of symfony #6827 stevenrombaults merged 1 commit into main from conductor-symfony-all-3... 3 days ago

Package changes

Package	Operation	From	To	About
symfony/cache	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/console	upgrade	v7.2.1	v7.2.5	diff - changelog
symfony/dependency-injection	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/doctrine-bridge	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/error-handler	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/form	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/framework-bundle	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/http-foundation	upgrade	v7.2.3	v7.2.5	diff - changelog
symfony/http-kernel	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/messenger	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/process	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/property-info	upgrade	v7.2.3	v7.2.5	diff - changelog
symfony/serializer	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/twig-bridge	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/type-info	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/validator	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/var-exporter	upgrade	v7.2.4	v7.2.5	diff - changelog
symfony/yaml	upgrade	v7.2.3	v7.2.5	diff - changelog

Differences from other solutions

- composer update runs in your CI
 - more control
 - better debugging options
 - full support for Composer plugins
 - run custom code before doing the update with access to your secrets
- Made for PHP
 - better default grouping behavior
 - no unexpected / unexplained updates
 - suitable use of composer update arguments like --minimal-changes
 - Care about high quality PHP support

Working with monorepos

- Path repositories

```
"repositories": [  
    {"type": "path", "url": "../internal/lib"}  
]
```

- Creates a symlink in vendor/
- Automatically infers version from git state
 - **But:** composer.json changes in deps require composer update

composer.lock

- Contents
 - all dependencies including transitive dependencies
 - all metadata (name, description, require, autoload, extra, ...)
 - Exact version for every package
 - Download URLs (source, dist, mirrors)
- Purpose
 - Reproducibility across teams, users and servers
 - Isolation of bug reports to code vs. potential dependency breaks
 - Transparency through explicit updating process

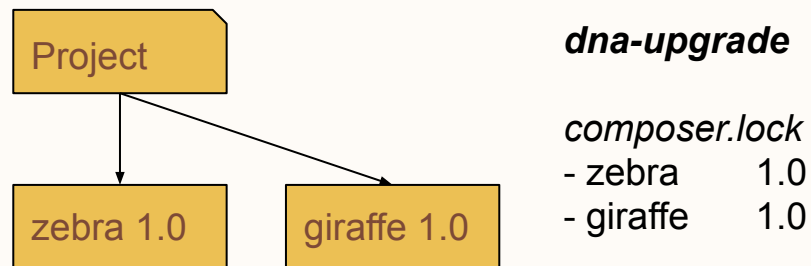
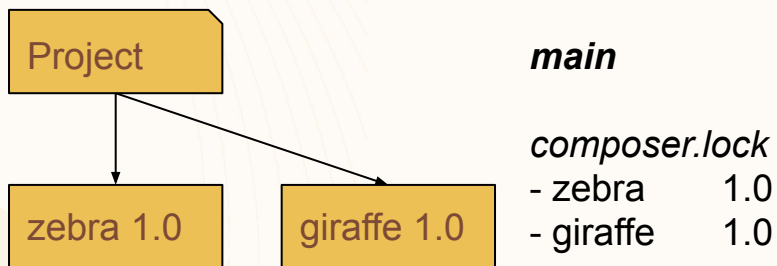
Commit The Lock File

- If you don't
 - composer install without a lock file is a composer update
 - You're not managing your dependencies, they're just doing whatever they want
 - Conflict can randomly occur on install
 - You may not get the same code
- The lock file exists to be committed!

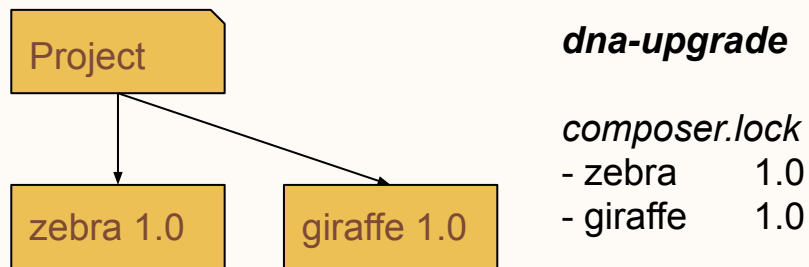
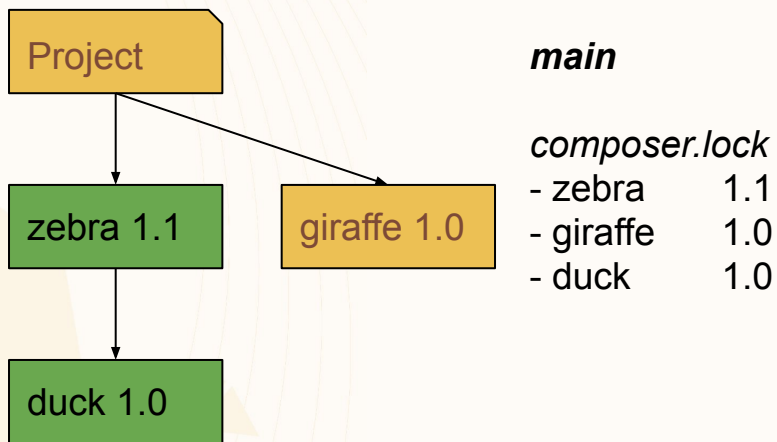
The Lock file will conflict



Day 0: "Initial Commit"



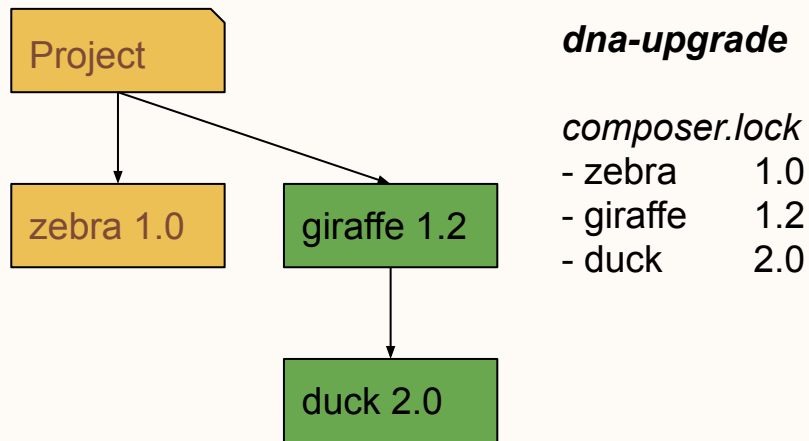
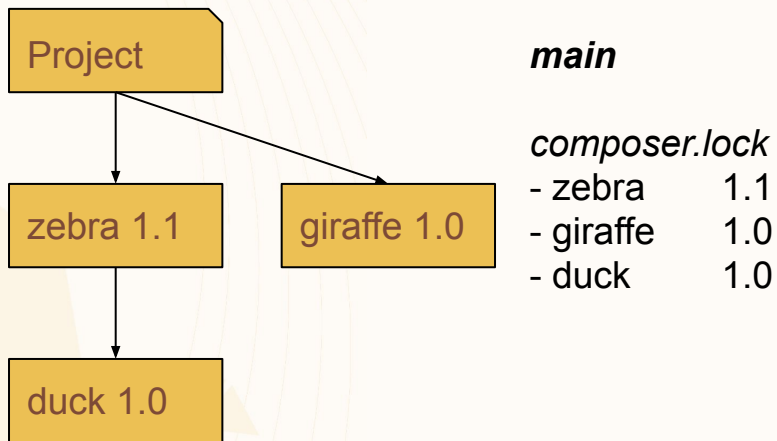
Week 2: Strange new zebras require duck



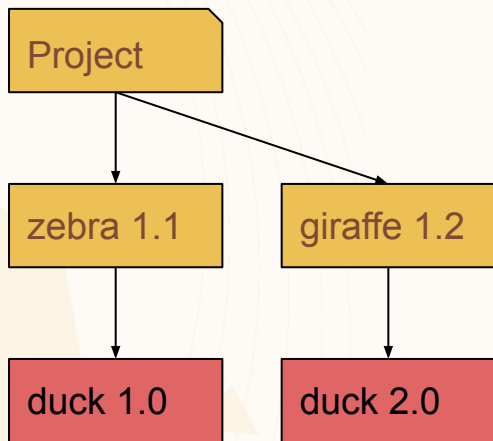


Week 3: Duck 2.0

Week 4: Giraffe evolves to require duck 2.0



Text-based Merge



main

composer.lock

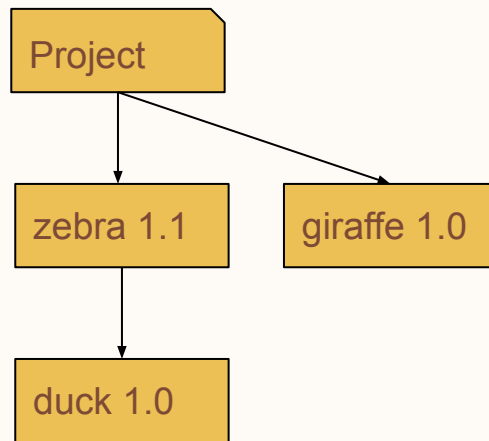
- zebra 1.1
- giraffe 1.2
- **duck 1.0**
- **duck 2.0**

Merge results in invalid dependencies



Reset composer.lock

```
git checkout <refspec> -- composer.lock  
git checkout main -- composer.lock
```



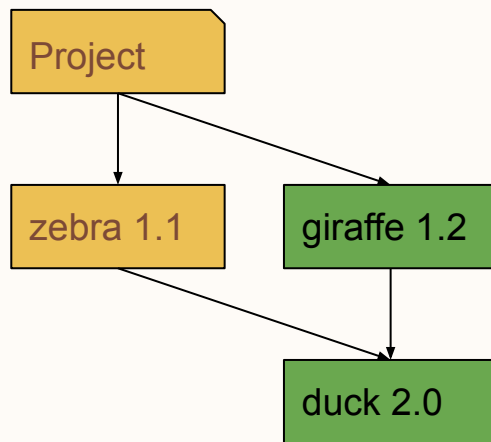
dna-upgrade

composer.lock

- zebra 1.1
- giraffe 1.0
- duck 1.0

Apply the update again

```
composer update giraffe  
--with-dependencies
```



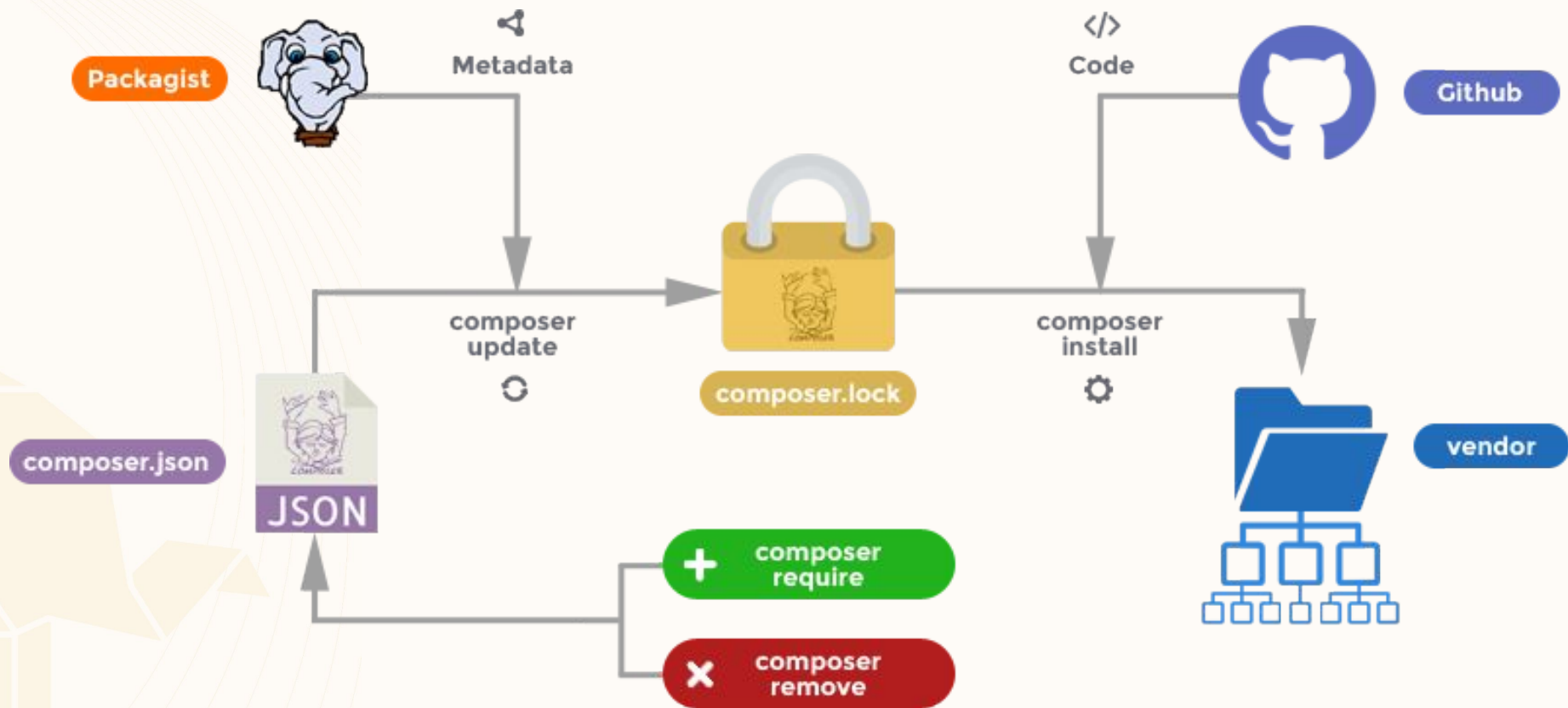
main

composer.lock

- zebra 1.1
- giraffe 1.2
- duck 2.0

How to resolve lock merge conflicts?

- composer.lock cannot be merged without conflicts
 - contains hash over relevant composer.json values
- `git checkout <refspec> -- composer.lock`
 - `git checkout main -- composer.lock`
- Reapply changes
 - `composer update <list of deps>`
 - put exact command in git commit message





Never Deploy without a Lock File

Do not run composer update during deployments

Recommended use of Composer in your Deployment Process

- commit composer.lock
- CI/CD
 - run composer install (not update!)
 - generate any potentially generated code
 - dump an optimized autoloader
 - package everything into an archive
- deployment
 - upload to production servers, move in place
 - (run composer check-platform-reqs)
 - switch webserver to use new code

Result

- no surprises in production
 - same dependency versions as tested
 - no risk of composer conflicts during deploy
 - code doesn't change at runtime
- deploying to multiple servers
 - exact same state everywhere
 - no unnecessarily repeated work

Caching

- First install
 - Download foo/bar 1.0.0 ref c0d2bc8b
 - Write to `$HOME/.composer/cache/files/foo/bar/c0d2bc8b.zip`
 - Unzip to `vendor/foo/bar/`
- Switch branches, next install
 - Download foo/bar 1.1.3 ref 3e708f6c
 - Write to `$HOME/.composer/cache/files/foo/bar/3e708f6c.zip`
 - Unzip to `vendor/foo/bar/`
- Switch back to original branch, install again
 - Unzip `$HOME/.composer/cache/files/foo/bar/c0d2bc8b.zip` to `vendor/foo/bar/`

Caching

- CI Considerations
 - Multiple developers
 - Multiple branches
 - Concurrent builds
 - CI specific caching properties, e.g. per-branch cache state?
- Caching vendor/
 - Builds on different composer.lock states with shared cache require frequent re-downloading of dependencies
 - Writing modified state can be slow
- Caching `$HOME/.composer/cache`
 - Can easily be shared across builds without having to delete or overwrite contents
 - Still frequent unzipping
 - **But:** Can be combined with vendor/ caching to avoid re-downloading and unnecessary unzipping
- Containers
 - Use filesystem layers to avoid re-installing entirely without lock file changes

Questions / Feedback?



<https://packagist.com>

E-Mail	<u>contact@packagist.com</u>
Bluesky	<u>@naderman.de</u>
Mastodon	<u>@naderman@phpc.social</u>
X	<u>@naderman</u>