



Drupal & Composer

Matthew Grasmick & Jeff Geerling



Speakers

Matthew Grasmick

@grasmash

Acquian

BLT maintainer

10+ years of Drupal

Jeff Geerling

@geerlingguy

Acquian

Drupal VM maintainer



Agenda

- Composer Overview ~40 min
- Hands-on exercises ~30 min
- Advanced Topics ~20 min
- Hands-on free-for-all ~30 min

Total ~2 hrs.

What is Composer?



Composer is a **dependency management** tool for PHP.

It allows you to **install**, **update**, and **load** the PHP libraries that your PHP application depends on.

What does that mean?

**Let's look at the type of
problem Composer solves**



Say you have a **Drupal 7** application.

It **requires jCarousel**.

A third party, external **dependency**.

You download the tarball, decompress, move it into place. Voila!

Easy, right?

Except when it isn't.



Versions matter.

Your hypothetical Drupal 7 site requires:

- Drupal Core, which requires jQuery **1.2.0**.
- jCarousel, which requires jQuery **1.3.0**.

1.2.0 != 1.3.0

Uh oh!

—



What do you do?

In **Drupal 7**, we used

- Various contributed modules
- Hacky workarounds to load multiple versions of jQuery.

That worked for dealing with a single library incompatibility.

Enter Drupal 8





Drupal 8

In Drupal 8, we use lots of third-party, external dependencies, like

- Symfony
- Doctrine
- Twig
- Etc.



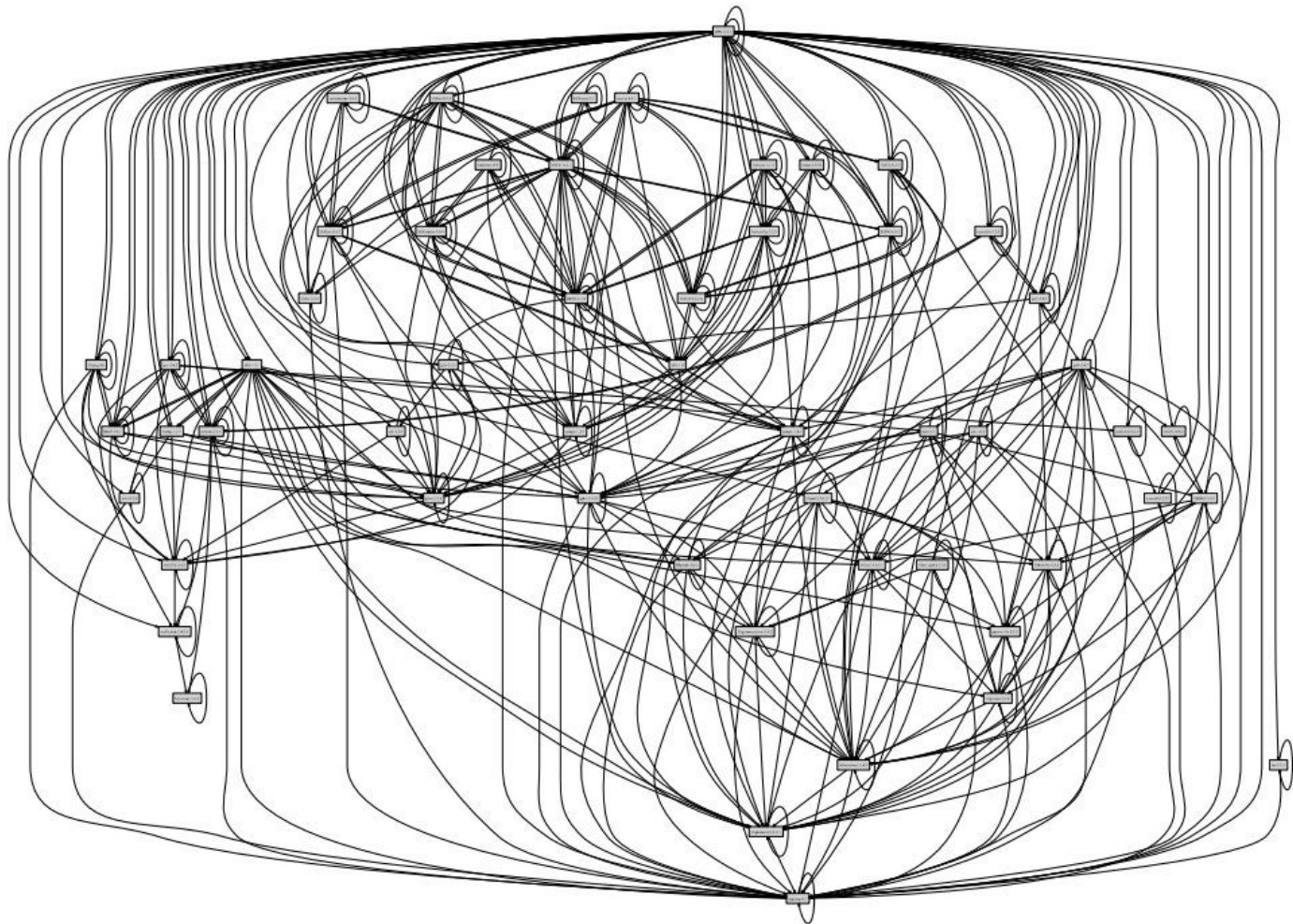
This is good.

- We're **getting of the island** and using libraries used by the rest of the PHP community!
- We're using software that is **Proudly Found Elsewhere** (and tested / supported elsewhere)
- We're **not re-inventing the wheel!**

But it gets complicated fast.

Say you have a **Drupal 8** site that requires...

- Drupal Core, which requires ...
 - Symfony components
 - **And dozens of other things**
 - **Which in turn require more things**
- CSV Serialization module, which requires League CSV
- Views Slideshow module, which requires ...
 - jQuery Cycle, which requires jQuery
- Bootstrap theme, which requires jQuery.
- Guzzle (for a custom module), which requires PSR HTTP message.
- Drush, which requires ...
 - Symfony Yaml
 - And dozens of other things
 - Which in turn require more things



**Imagine resolving an
incompatibility in that tree.**

A green road sign with a white border, mounted on two wooden posts. The sign is tilted and features the text "Rock Hard Place" in white, bold, sans-serif font. A white arrow points left above the word "Rock", and another white arrow points right below the word "Place". The background is a bright blue sky with scattered white clouds.

← Rock
Hard Place →

This is dependency hell.

We need a dependency manager.



This is not a novel need.

Dependency management is a fundamental part of software engineering.

- Javascript:
 - NPM
 - Yarn
 - Bower
- Ruby
 - Bundler
- PHP
 - **Composer!**

How does it work? Demo time.



Install composer

1. Download Composer @ getcomposer.org/download
2. Execute command ...

```
$ php composer-setup.php
--install-dir=bin
--filename=composer
```


OSX Only!



Install composer

If you are a Homebrew user on OSX, execute...

```
$ brew install composer
```



Create a new PHP project from scratch.

Not Drupal. Just a directory and an empty PHP file.

```
# Make a directory.  
$ mkdir my-new-project  
$ cd my-new-project  
  
# Create an empty file.  
$ touch index.php
```



directory tree

- `my-new-project`
 - `index.php`



Initialize Composer for the project

This creates composer.json.

```
$ composer init
```





directory tree

- my-new-project
 - composer.json
 - index.php





What should our app do?

Log stuff. We want a PHP script that logs messages to a log file.



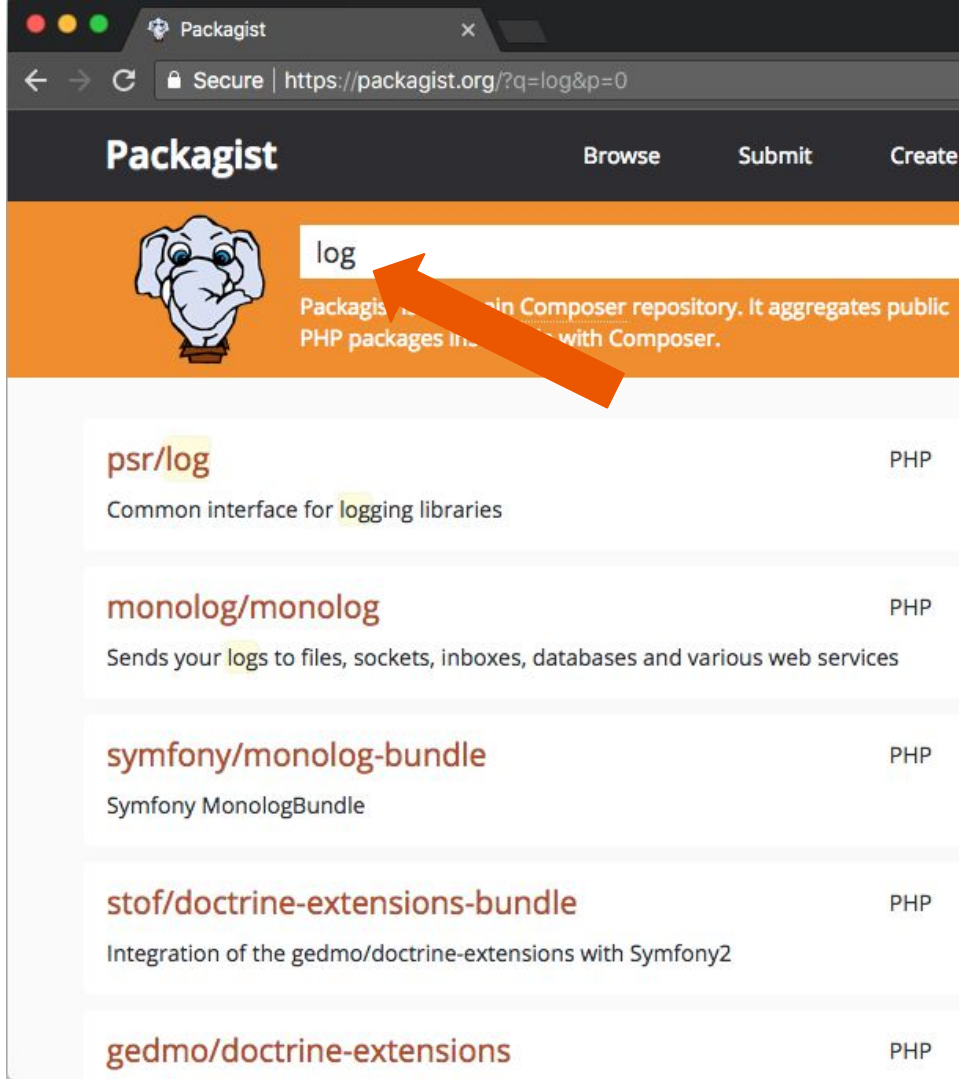


Let's see if a PHP library already exists

That **handles logging stuff**. Like writing files, defining level (info, warning, error), timestamps, etc.

Find a dependency

On packagist.org.



The screenshot shows the Packagist website interface. At the top, there is a navigation bar with the Packagist logo and links for 'Browse', 'Submit', and 'Create'. Below the navigation bar is a search bar containing the text 'log'. An orange arrow points to the search bar. The search results are displayed in a list format, showing the package name, a brief description, and the language (PHP). The results include:

- psr/log**: Common interface for logging libraries
- monolog/monolog**: Sends your logs to files, sockets, inboxes, databases and various web services
- symfony/monolog-bundle**: Symfony MonologBundle
- stof/doctrine-extensions-bundle**: Integration of the gedmo/doctrine-extensions with Symfony2
- gedmo/doctrine-extensions**

Find a dependency

On packagist.org.



The screenshot shows the Packagist website interface. At the top, there's a navigation bar with 'Packagist' logo and links for 'Browse', 'Submit', and 'Create'. Below the navigation bar is a search bar containing the text 'log'. To the left of the search bar is a cartoon elephant logo. Below the search bar, there's a description: 'Packagist is the main Composer repository. It aggregates public PHP packages installable with Composer.' Below this, there's a list of search results for 'log':

Package Name	Description	Language
psr/log	Common interface for logging libraries	PHP
monolog/monolog	Sends your logs to files, sockets, inboxes, databases and various web services	PHP
symfony/monolog-bundle	Symfony MonologBundle	PHP
stof/doctrine-extensions-bundle	Integration of the gedmo/doctrine-extensions with Symfony2	PHP
gedmo/doctrine-extensions		PHP

Search packages...

monolog/monolog

composer require monolog/monolog

Sends your logs to files, sockets, inboxes, databases and various web services



github.com/Seldaek/monolog

[Homepage](#)

[Source](#)

[Issues](#)

Installs: 90 314 221
 Dependents: 3 421
 Suggesters: 328
 Stars: 9 423
 Watchers: 336
 Forks: 1 304
 Open Issues: 131

1.23.0

2017-06-19 01:22 UTC

requires

- php: >=5.3.0
- psr/log: ~1.0

requires (dev)

- phpunit/phpunit: ~4.5
- graylog2/gelf-php: ~1.0
- sentry/sentry: ^0.13
- rufin/elastica: >=0.90 <3.0
- doctrine/couchdb: ~1.0@dev
- aws/aws-sdk-php: ^2.4.9 || ^3.0
- php-amqplib/php-amqplib: ~2.4

suggests

- graylog2/gelf-php: Allow sending log messages to a GrayLog2 server
- sentry/sentry: Allow sending log messages to a Sentry server
- doctrine/couchdb: Allow sending log messages to a CouchDB server
- rufin/elastica: Allow sending log messages to an ElasticSearch server

dev-master / 2.0.x-dev

1.x-dev

1.23.0

1.22.1

1.22.0

1.21.0

1.20.0

1.19.0

1.18.2

Search packages...

monolog/monolog

composer require monolog/monolog

Sends your logs to files, sockets, inboxes, databases and various web services



github.com/Seldaek/monolog

[Homepage](#)

[Source](#)

[Issues](#)

Installs: 90 314 221

Dependents: 3 421

Suggesters: 328

Stars: 9 423

Watchers: 336

Forks: 1 304

Open Issues: 131



1.23.0

2017-06-19 01:22 UTC

requires

- php: >=5.3.0
- psr/log: ~1.0

requires (dev)

- phpunit/phpunit: ~4.5
- graylog2/gelf-php: ~1.0
- sentry/sentry: ^0.13
- rufin/elastica: >=0.90 <3.0
- doctrine/couchdb: ~1.0@dev
- aws/aws-sdk-php: ^2.4.9 || ^3.0
- php-amqplib/php-amqplib: ~2.4

suggests

- graylog2/gelf-php: Allow sending log messages to a GrayLog2 server
- sentry/sentry: Allow sending log messages to a Sentry server
- doctrine/couchdb: Allow sending log messages to a CouchDB server
- rufin/elastica: Allow sending log messages to an ElasticSearch server

dev-master / 2.0.x-dev

1.x-dev

1.23.0

1.22.1

1.22.0

1.21.0

1.20.0

1.19.0

1.18.2

Search packages...

monolog/monolog

↓ composer require monolog/monolog

Sends your logs to files, sockets, inboxes, databases and various web services



github.com/Seldaek/monolog

[Homepage](#)

[Source](#)

[Issues](#)

Installs: 90 314 221
Dependents: 3 421
Suggesters: 328
Stars: 9 423
Watchers: 336
Forks: 1 304
Open Issues: 131

1.23.0

2017-06-19 01:22 UTC

requires

- php: >=5.3.0
- psr/log: ~1.0

requires (dev)

- phpunit/phpunit: ~4.5
- graylog2/gelf-php: ~1.0
- sentry/sentry: ^0.13
- rufin/elastica: >=0.90 <3.0
- doctrine/couchdb: ~1.0@dev
- aws/aws-sdk-php: ^2.4.9 || ^3.0
- php-amqplib/php-amqplib: ~2.4

suggests

- graylog2/gelf-php: Allow sending log messages to a GrayLog2 server
- sentry/sentry: Allow sending log messages to a Sentry server
- doctrine/couchdb: Allow sending log messages to a CouchDB server
- rufin/elastica: Allow sending log messages to an ElasticSearch server

dev-master / 2.0.x-dev

1.x-dev

1.23.0

1.22.1

1.22.0

1.21.0

1.20.0

1.19.0

1.18.2



Require the package

This command discovers the library, determines correct version, and **downloads** it.

```
$ composer require  
monolog/monolog
```

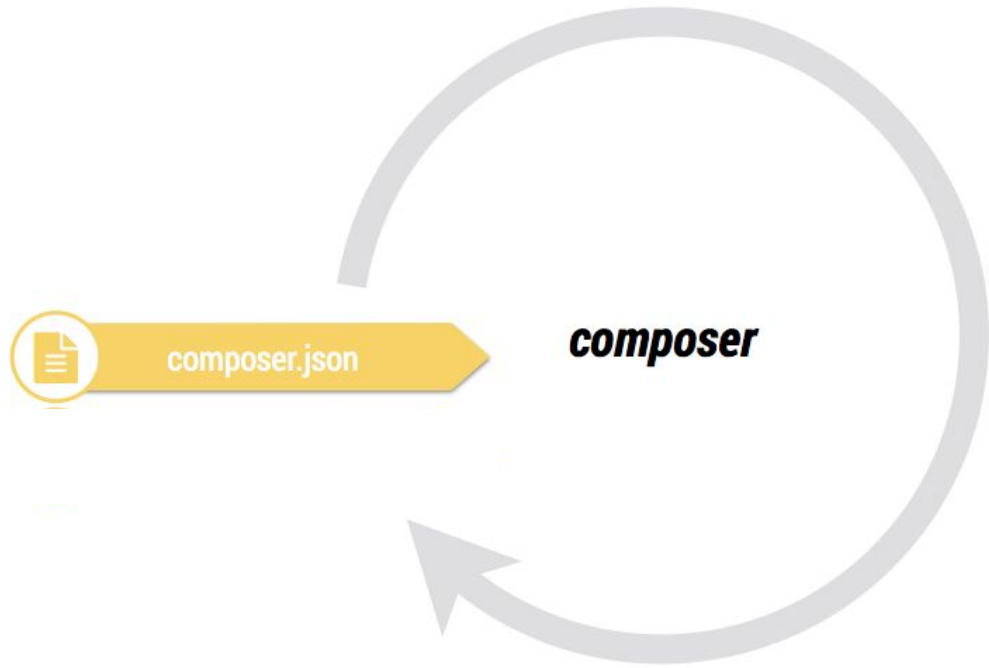


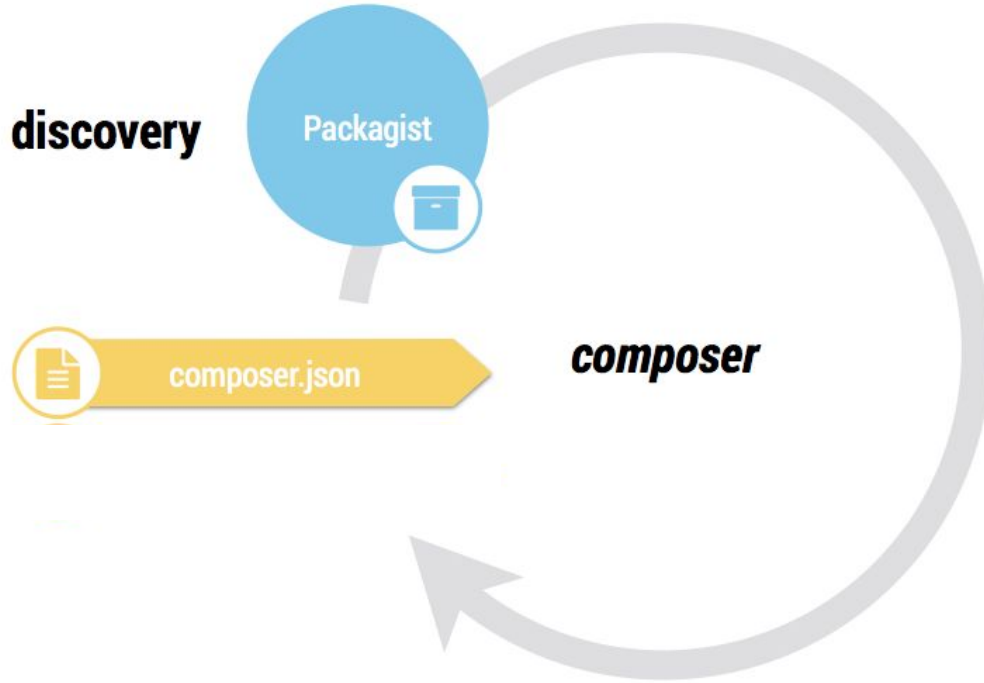
Under the hood.

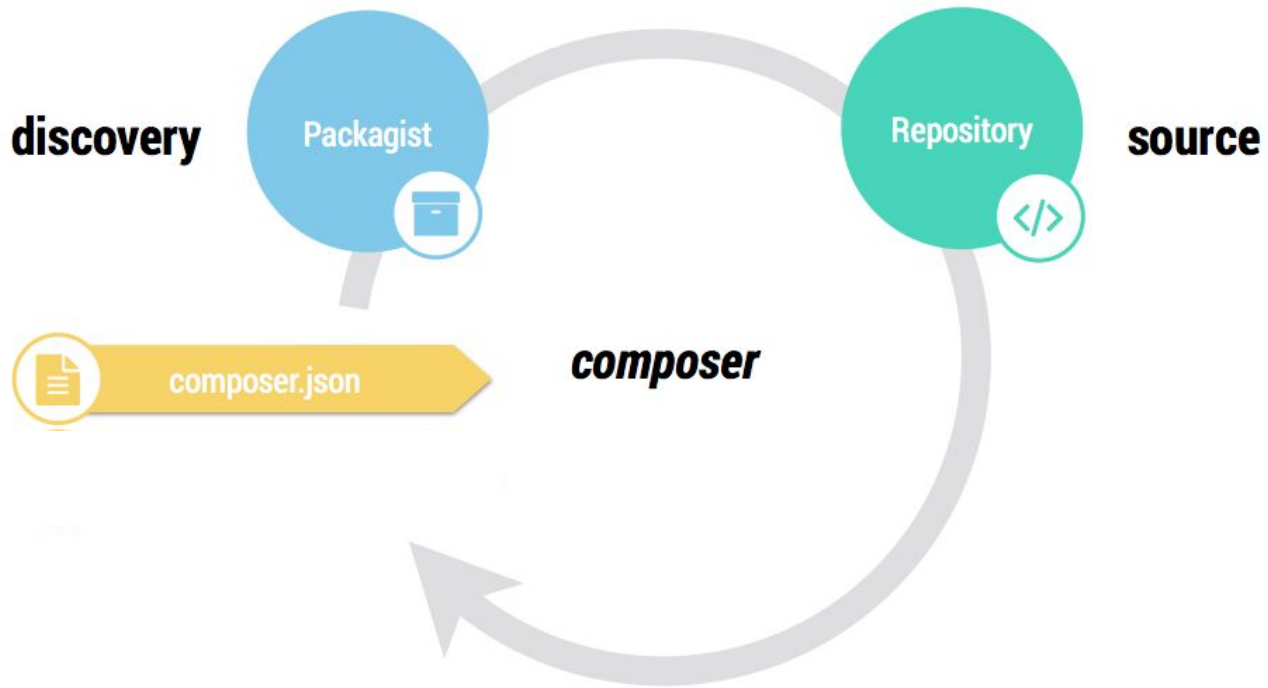


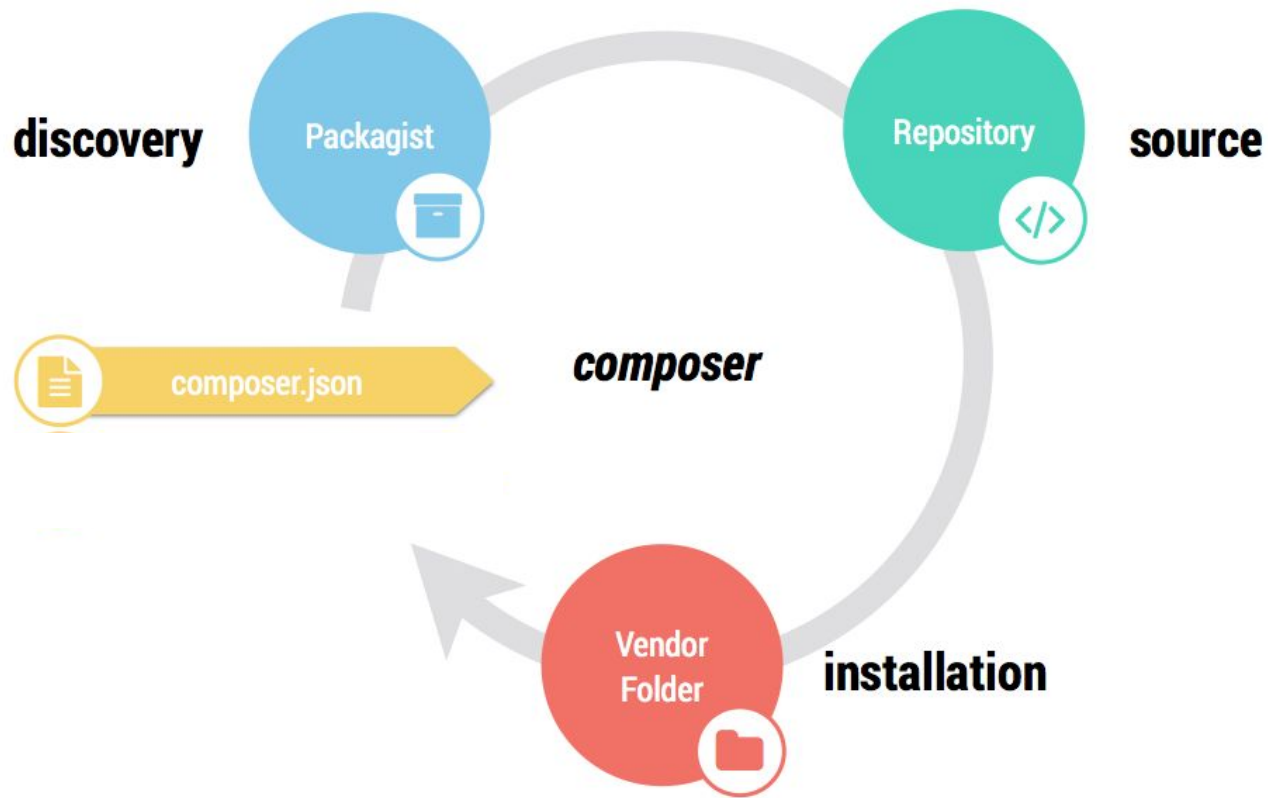


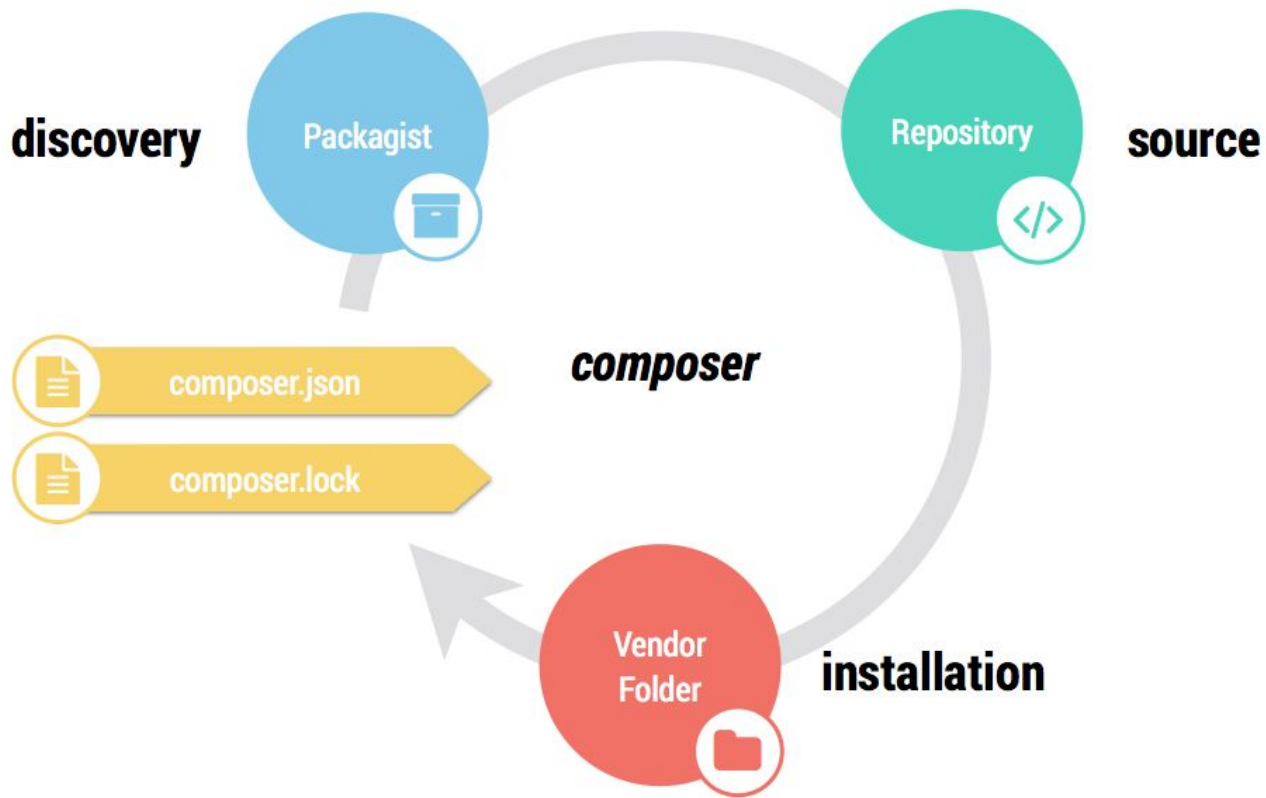
composer













directory-tree

- my-new-project
 - composer.json
 - composer.lock
 - index.php
 - vendor
 - monolog
 - monolog
 - Src
 - autoload.php



directory-tree

- my-new-project
 - composer.json
 - composer.lock
 - index.php
 - vendor
 - monolog
 - monolog
 - Src
 - autoload.php





directory-tree

- my-new-project
 - composer.json
 - composer.lock
 - index.php
 - vendor
 - monolog
 - monolog
 - Src
 - autoload.php





directory-tree

- my-new-project
 - composer.json
 - composer.lock
 - index.php
 - vendor
 - monolog
 - monolog
 - Src
 - autoload.php





directory-tree

- my-new-project
 - composer.json
 - composer.lock
 - index.php
 - vendor
 - monolog
 - monolog
 - Src
 - autoload.php



Let's implement it.

index.php



```
<?php


// Require Composer's autoloader.
require __DIR__ . "/vendor/autoload.php";

use Monolog\Logger;
use Monolog\Handler\StreamHandler;

// Create a logger
$log = new Logger('my-log');
$log->pushHandler(new StreamHandler(__DIR__ . "/my.log", Logger::WARNING));

// Log a message!
$log->error('I did it!');
```


index.php



```
<?php

// Require Composer's autoloader.
require __DIR__ . "/vendor/autoload.php";

use Monolog\Logger;
use Monolog\Handler\StreamHandler;

// Create a logger
$log = new Logger('my-log');
$log->pushHandler(new StreamHandler(__DIR__ . "/my.log", Logger::WARNING));

// Log a message!
$log->error('I did it!');
```

index.php



```
<?php

// Require Composer's autoloader.
require __DIR__ . "/vendor/autoload.php";

use Monolog\Logger;
use Monolog\Handler\StreamHandler;

// Create a logger
$log = new Logger('my-log');
$log->pushHandler(new StreamHandler(__DIR__ . "/my.log", Logger::WARNING));

// Log a message!
$log->error('I did it!');
```



index.php



```
<?php

// Require Composer's autoloader.
require __DIR__ . "/vendor/autoload.php";

use Monolog\Logger;
use Monolog\Handler\StreamHandler;

// Create a logger
$log = new Logger('my-log');
$log->pushHandler(new StreamHandler(__DIR__ . "/my.log", Logger::WARNING));

// Log a message!
$log->error('I did it!');
```





Execute it

```
# Execute the script.  
$ php -f index.php
```



It works!

```
# Execute the script.  
$ php -f index.php  
  
# Look in the log file.  
$ cat my.log  
[2018-03-25 12:05:00] my-log.ERROR: I did it! [] []
```

Let's add some dev tools.






Like PHP Codesniffer.

We **use it during development only** to make sure our code is nicely formatted.
We don't need it to run our logger.



Require a dev dependency.

```
$ composer require squizlabs/php_codesniffer --dev
```




Indicates this is a dev-only
dependency



New executables

In vendor/bin

- my-new-project
 - composer.json
 - composer.lock
 - index.php
 - vendor
 - bin
 - phpcs
 - monolog
 - monolog
 - src
 - autoload.php
 - squizlabs
 - php_codesniffer
- 




Execute the binary

```
$ ./vendor/bin/phpcs --standard=PSR2 index.php
```



Executes new phpcs binary



The dev tool is part of the project repo.

If you **clone the repo, you have phpcs**. No need to install separately. **Neat.**

Recap.





We built a logging app from scratch.

In *less than 10 commands*.

Created project

Added libraries

Ran the app

Added dev tools

Sniffed code

```
$ mkdir my-new-project
$ cd my-new-project
$ touch index.php
$ composer init
$ composer require monolog/monolog
# Wrote a few PHP lines.
$ php -f index.php
$ cat my.log
$ composer require
squizlabs/php_codesniffer --dev
$ ./vendor/bin/phpcs
--standard=PSR2 index.php
```



Let's look in more depth.





Library consumer



Packagist

Library consumer



Library consumer

Packagist

Library

```
{  
  "require": {  
    "monolog/monolog": "^1.0"  
  }  
}
```



```
{  
  "require": {  
    "monolog/monolog": "^1.0"  
  }  
}
```

```
{  
  "name": "monolog/monolog"  
  ...  
}
```



composer.json

```
{  
  "require": {  
    "monolog/monolog": "^1.0"  
  }  
}
```

```
{  
  "name": "monolog/monolog"  
  ...  
}
```



composer.json

```
{  
  "require": {  
    "monolog/monolog": "^1.0"  
  }  
}
```

```
{  
  "name": "monolog/monolog"  
  ...  
}
```

composer.json



composer.json schema





metadata for consumers

```
{  
  "name": "grasmash/my-new-project",  
  "description": "My nifty thingee.",  
  "authors": [ { "name": "Matthew Grasmick" } ],  
}
```



composer init



metadata for vendors

```
{
  "name": "monolog/monolog",
  "description": "Sends your logs to files, sockets, inboxes, databases and
various web services",
  "keywords": ["log", "logging", "psr-3"],
  "homepage": "http://github.com/Seldaek/monolog",
  "type": "library",
  "license": "MIT",
  "authors": [ { "name": "Jordi Boggiano" } ],
}
```

Search packages...

monolog/monolog

composer require monolog/monolog

Sends your logs to files, sockets, inboxes, databases and various web services



github.com/Seldaek/monolog

[Homepage](#)

[Source](#)

[Issues](#)

Installs: 90 314 221
Dependents: 3 421
Suggesters: 328
Stars: 9 423
Watchers: 336
Forks: 1 304
Open Issues: 131

1.23.0

2017-06-19 01:22 UTC

requires

- php: >=5.3.0
- psr/log: ~1.0

requires (dev)

- phpunit/phpunit: ~4.5
- graylog2/gelf-php: ~1.0
- sentry/sentry: ^0.13
- rufliin/elastica: >=0.90 <3.0
- doctrine/couchdb: ~1.0@dev
- aws/aws-sdk-php: ^2.4.9 || ^3.0
- php-amqplib/php-amqplib: ~2.4

suggests

- graylog2/gelf-php: Allow sending log messages to a GrayLog2 server
- sentry/sentry: Allow sending log messages to a Sentry server
- doctrine/couchdb: Allow sending log messages to a CouchDB server
- rufliin/elastica: Allow sending log messages to an ElasticSearch server

dev-master / 2.0.x-dev

1.x-dev

1.23.0

1.22.1

1.22.0

1.21.0

1.20.0

1.19.0

1.18.2

Everything is a package



require

```
{
  "name": "grasmash/my-new-project",
  "require": {
    "monolog/monolog": "^1.23.0",
  },

  "require-dev": {
    "squizlabs/php_codesniffer": "^3.2.3",
  },
}
```



composer require



require

```
{
  "name": "monolog/monolog",
  "description": "Sends your logs to files, sockets, inboxes, databases and
various web services",
  "require": {
    "php": "^7.0",
    "psr/log": "^1.0.1"
  },
}
```

Search packages...

monolog/monolog

composer require monolog/monolog

Sends your logs to files, sockets, inboxes, databases and various web services



github.com/Seldaek/monolog

[Homepage](#)

[Source](#)

[Issues](#)

Installs: 90 314 221

Dependents: 3 421

Suggesters: 328

Stars: 9 423

Watchers: 336

Forks: 1 304

Open Issues: 131

1.23.0

2017-06-19 01:22 UTC

requires

- php: >=5.3.0
- psr/log: ~1.0

requires (dev)

- phpunit/phpunit: ~4.5
- graylog2/gelf-php: ~1.0
- sentry/sentry: ^0.13
- rufin/elastica: >=0.90 <3.0
- doctrine/couchdb: ~1.0@dev
- aws/aws-sdk-php: ^2.4.9 || ^3.0
- php-amqplib/php-amqplib: ~2.4

suggests

- graylog2/gelf-php: Allow sending log messages to a GrayLog2 server
- sentry/sentry: Allow sending log messages to a Sentry server
- doctrine/couchdb: Allow sending log messages to a CouchDB server
- rufin/elastica: Allow sending log messages to an ElasticSearch server

dev-master / 2.0.x-dev

1.x-dev

1.23.0

1.22.1

1.22.0

1.21.0

1.20.0

1.19.0

1.18.2





require-dev

```
{  
  "name": "grasmash/my-new-project",  
  "require": {  
    "monolog/monolog": "^1.23.0",  
  },  
  
  "require-dev": {  
    "squizlabs/php_codesniffer": "^3.2.3",  
  },  
}
```



`composer require --dev`



require-dev

```
{
  "name": "monolog/monolog",
  "require-dev": {
    "phpunit/phpunit": "^5.7",
    "graylog2/gelf-php": "^1.4.2",
    "sentry/sentry": "^0.13",
    "jakub-onderka/php-parallel-lint": "^0.9",
    ...
  },
}
```


Search packages...

monolog/monolog

composer require monolog/monolog

Sends your logs to files, sockets, inboxes, databases and various web services



github.com/Seldaek/monolog

[Homepage](#)

[Source](#)

[Issues](#)

Installs: 90 314 221
Dependents: 3 421
Suggesters: 328
Stars: 9 423
Watchers: 336
Forks: 1 304
Open Issues: 131

1.23.0

2017-06-19 01:22 UTC

requires

- php: >=7.0
- psr/log: ~1.0



requires (dev)

- phpunit/phpunit: ~4.5
- graylog2/gelf-php: ~1.0
- sentry/sentry: ^0.13
- rufliin/elastica: >=0.90 <3.0
- doctrine/couchdb: ~1.0@dev
- aws/aws-sdk-php: ^2.4.9 || ^3.0
- php-amqplib/php-amqplib: ~2.4

suggests

- graylog2/gelf-php: Allow sending log messages to a GrayLog2 server
- sentry/sentry: Allow sending log messages to a Sentry server
- doctrine/couchdb: Allow sending log messages to a CouchDB server
- rufliin/elastica: Allow sending log messages to an ElasticSearch server

dev-master / 2.0.x-dev

1.x-dev

1.23.0

- 1.22.1
- 1.22.0
- 1.21.0
- 1.20.0
- 1.19.0
- 1.18.2



misc

```
{  
  "minimum-stability": "dev",  
  "prefer-stable": true  
}
```



```
composer config prefer-stable true
```



Lots more!

- repositories
- config
- autoload
- autoload-dev
- bin
- extra

Check out website @ <https://getcomposer.org/doc/04-schema.md>.

Version best practices.



Versions matter.

For each dependency, we typically want the **latest version that won't break our site**.



Version promiscuity

If we just get the latest version of everything,
an upstream **change will break something** in our site.

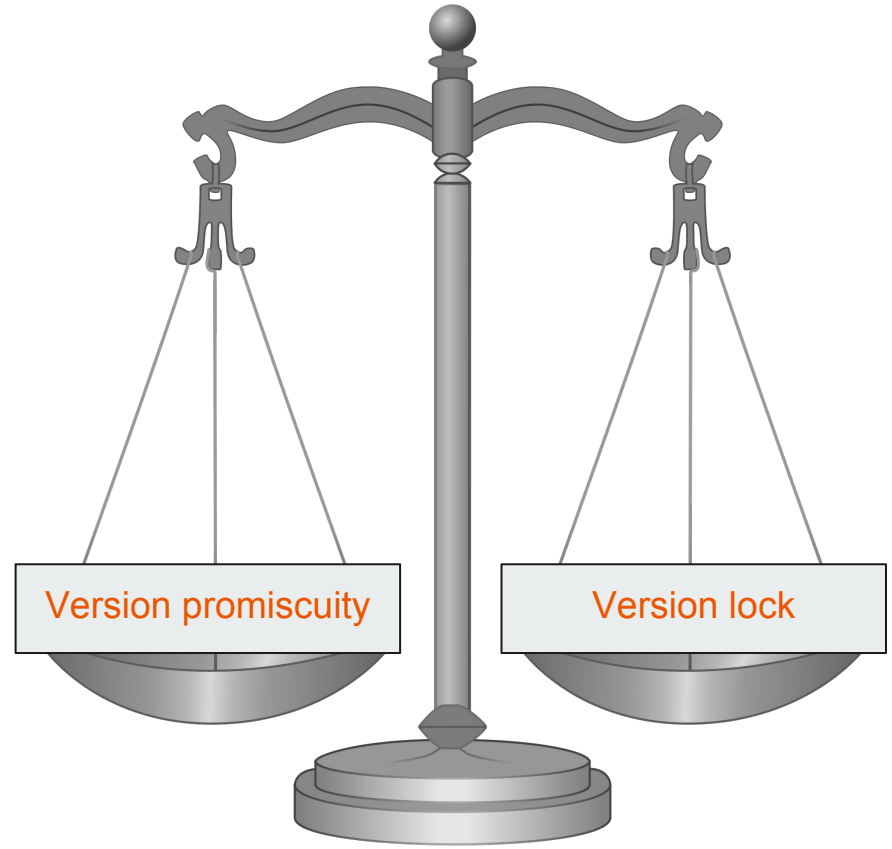


Version lock

If we just stick to the exact versions we use now, we get no **new features or bug fixes**.

How do we find a balance?

Between the two...



Version constraints.

—



Version constraint

```
{  
  "require": {  
    "monolog/monolog": "^1.23.0",  
  },  
}
```



Version constraint

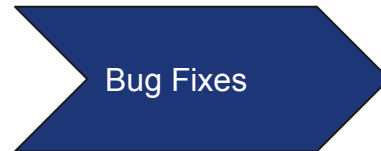
```
{  
  "require": {  
    "monolog/monolog": "^1.23.0",  
  },  
}
```

Semantic Versioning



1 . **2** . **3**
major minor patch

1 . **2** . **3**
major minor patch





1

major

.

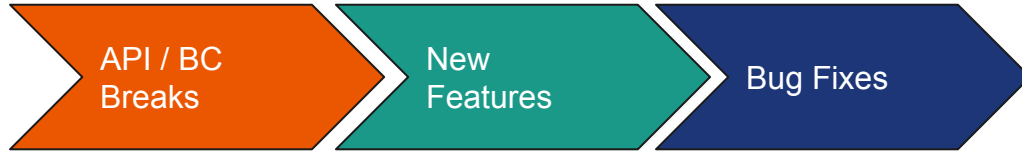
2

minor

.

3

patch



1 . **2** . **3**
major minor patch

Version constraint best practices





Version constraint

```
{  
  "require": {  
    "monolog/monolog": "^1.23.0",  
  },  
}
```

Version constraint operators.

caret, ^

`^1.2.3` is equivalent to `>=1.2.3,<2.0.0`.

Downloads latest minor or patch version above specified version. Allows last 2 digits to go up.

Preferred!

tilde, ~

~1.2.3 is equivalent to $\geq 1.2.3 < 1.3.0$.

Specifies a minimum version, but allows the last digit specified to go up.



range, \geq X, \leq

≥ 1.0

$\geq 1.0, \leq 1.5$

≤ 1.5

Specify a range of valid versions and combine multiple ranges with AND and OR operands.

asterisk, *

1.0.* is equivalent to $\geq 1.0, < 1.1$.

Specify a pattern with a * wildcard.



exact version

1.0.0 will download exactly 1.0.0.



git branch

dev-master uses the master branch.

Using the prefix dev- followed by a git branch name like master will checkout that branch.



more

<https://getcomposer.org/doc/articles/versions.md>



Use the caret as your default.

```
{  
  "require": {  
    "monolog/monolog": "~2.0"  
  }  
}
```



Image from "Composer the right way," Rafael Dohms

```
{  
  "require": {  
    "monolog/monolog": "2.0"  
  }  
}
```

There is no monolog/monolog 2.0.



Image from "Composer the right way," Rafael Dohms

```
{  
  "require": {  
    "monolog/monolog": "2.0"  
    "^1.0"  
  }  
}
```



Image from "Composer the right way," Rafael Dohms

```
{  
  "require": {  
    "monolog/monolog": "2.0"  
    "^1.0"  
  }  
}
```

Take 1.23.0, on the house.



Image from "Composer the right way," Rafael Dohms

composer.lock

—

**Records which specific versions
were actually installed.**

```
{  
  "require": {  
    "monolog/monolog": "^1.0"  
  }  
}
```



composer.json

composer

```
{  
  "require": {  
    "monolog/monolog": "^1.0"  
  }  
}
```



composer.json

composer

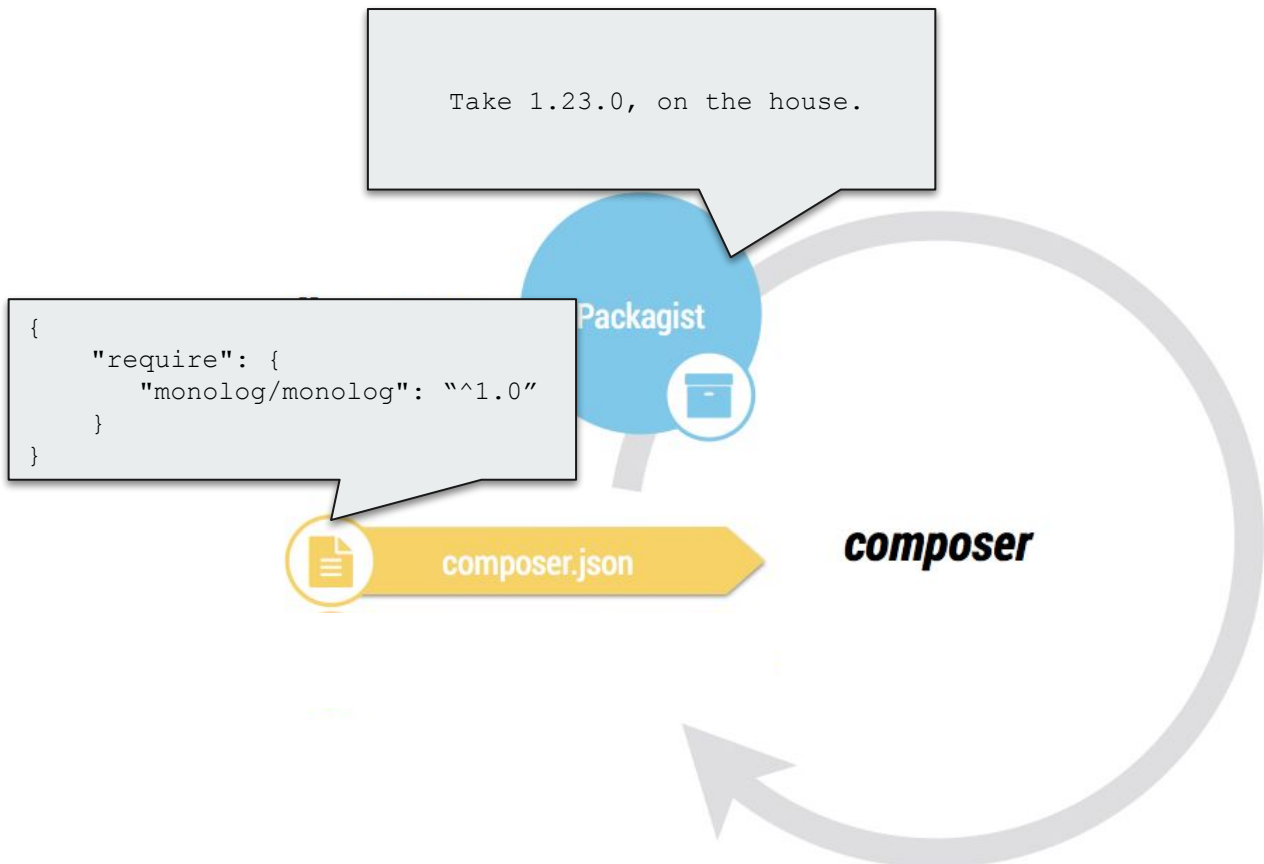
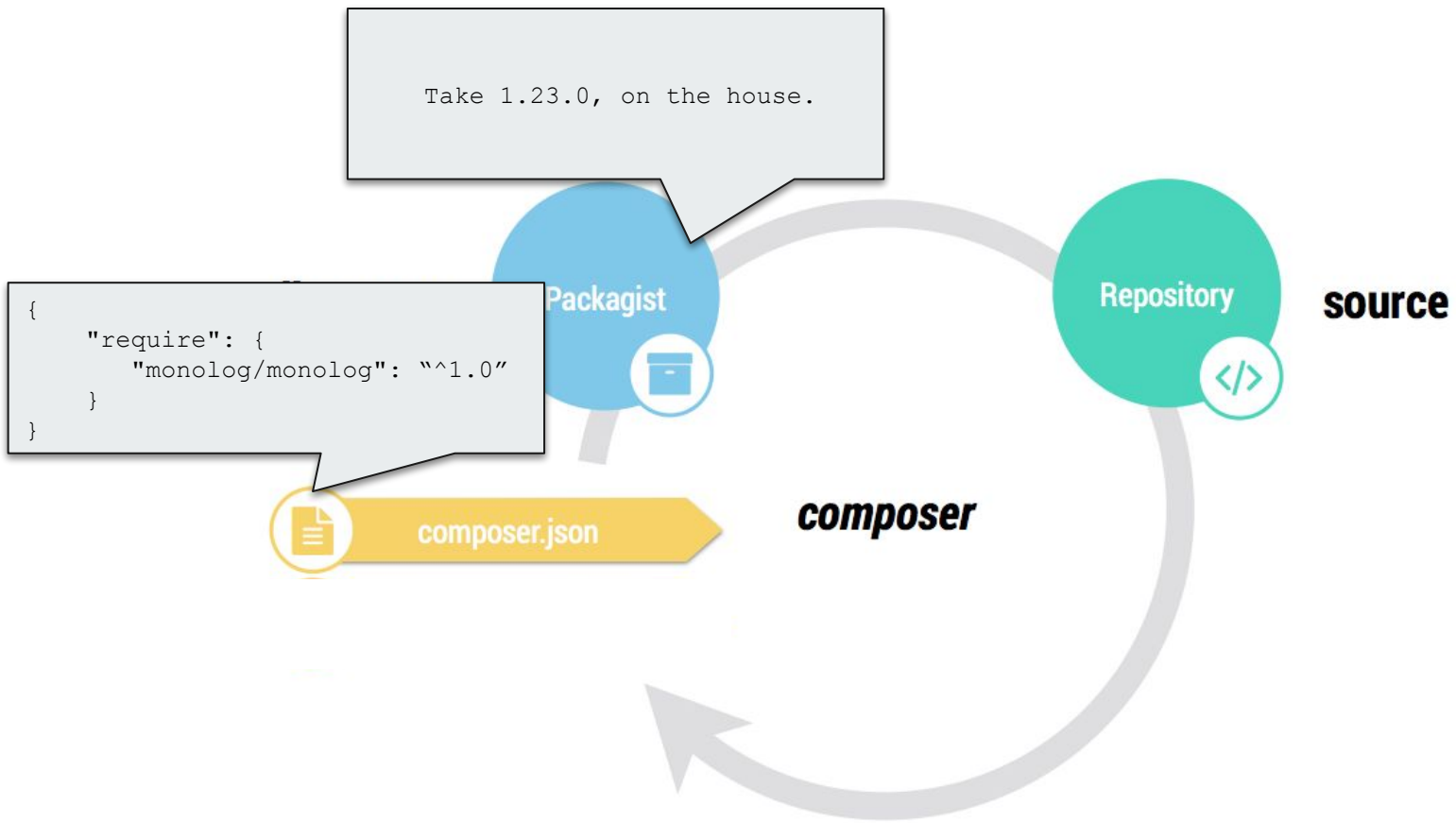
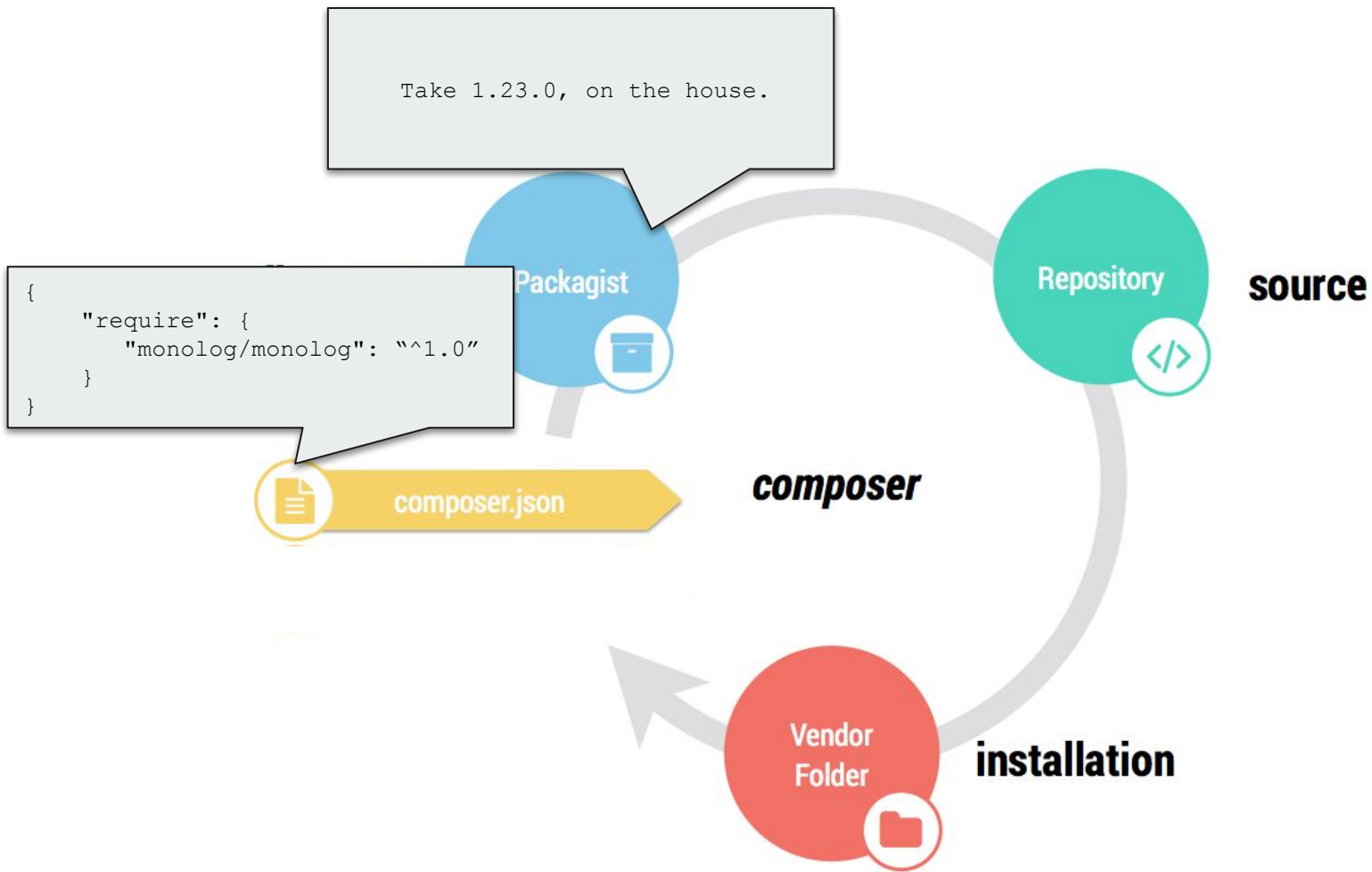
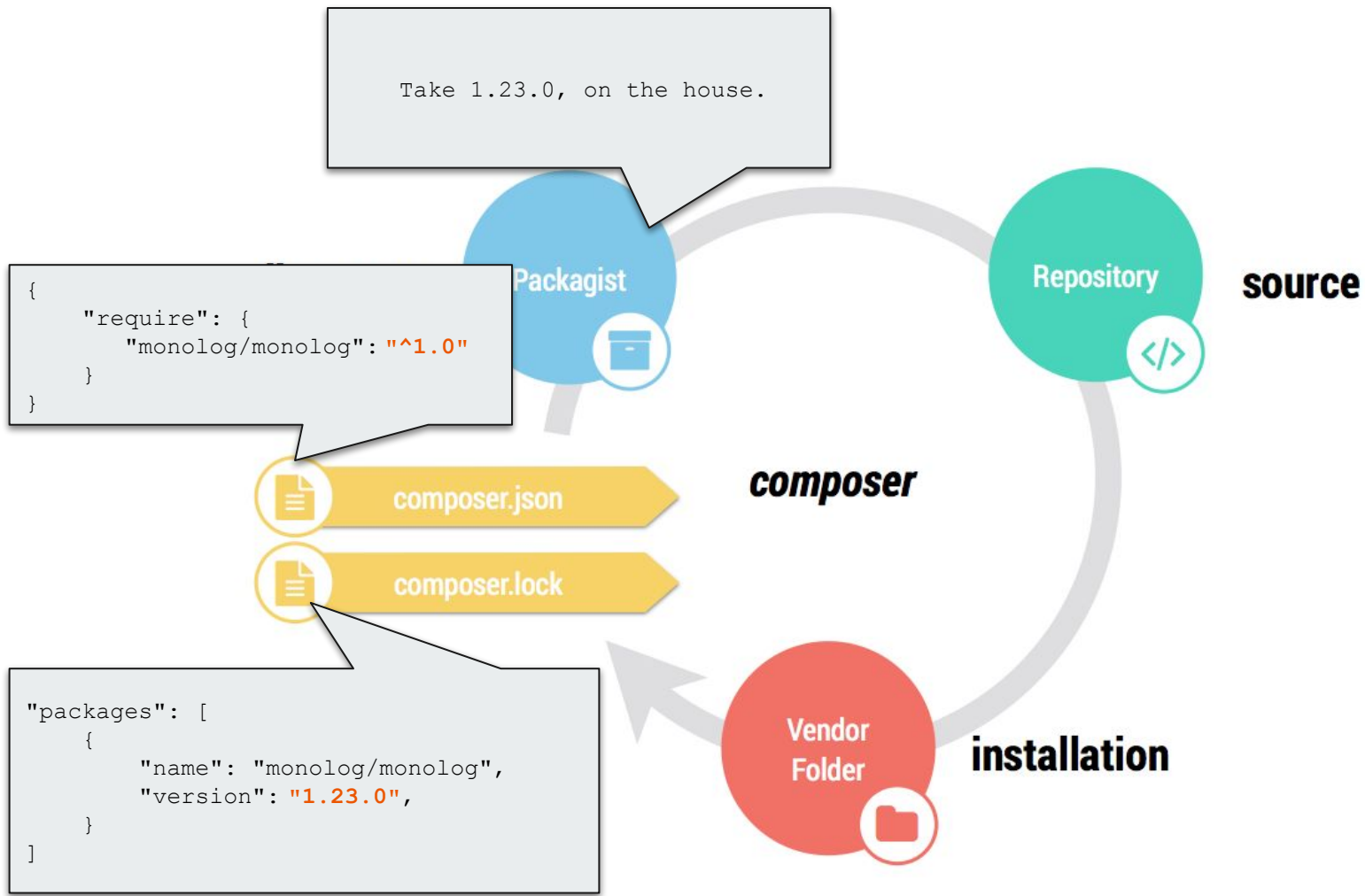


Image from "Composer the right way," Rafael Dohms









The existence of composer.lock

Fundamentally changes the behavior of `composer install`.

Grok this.





If composer.lock DOES NOT exist

`composer install` will:

- Discover all available dependency versions
- Determine which versions of packages should be installed.
- Create `composer.lock`
- Install the exact dependencies defined in `composer.lock`.



If composer.lock DOES exist

`composer install` will:

- ~~Discover all available dependency versions~~
- ~~Determine which versions of packages should be installed.~~
- ~~Create composer.lock~~
- Install the exact dependencies defined in composer.lock.

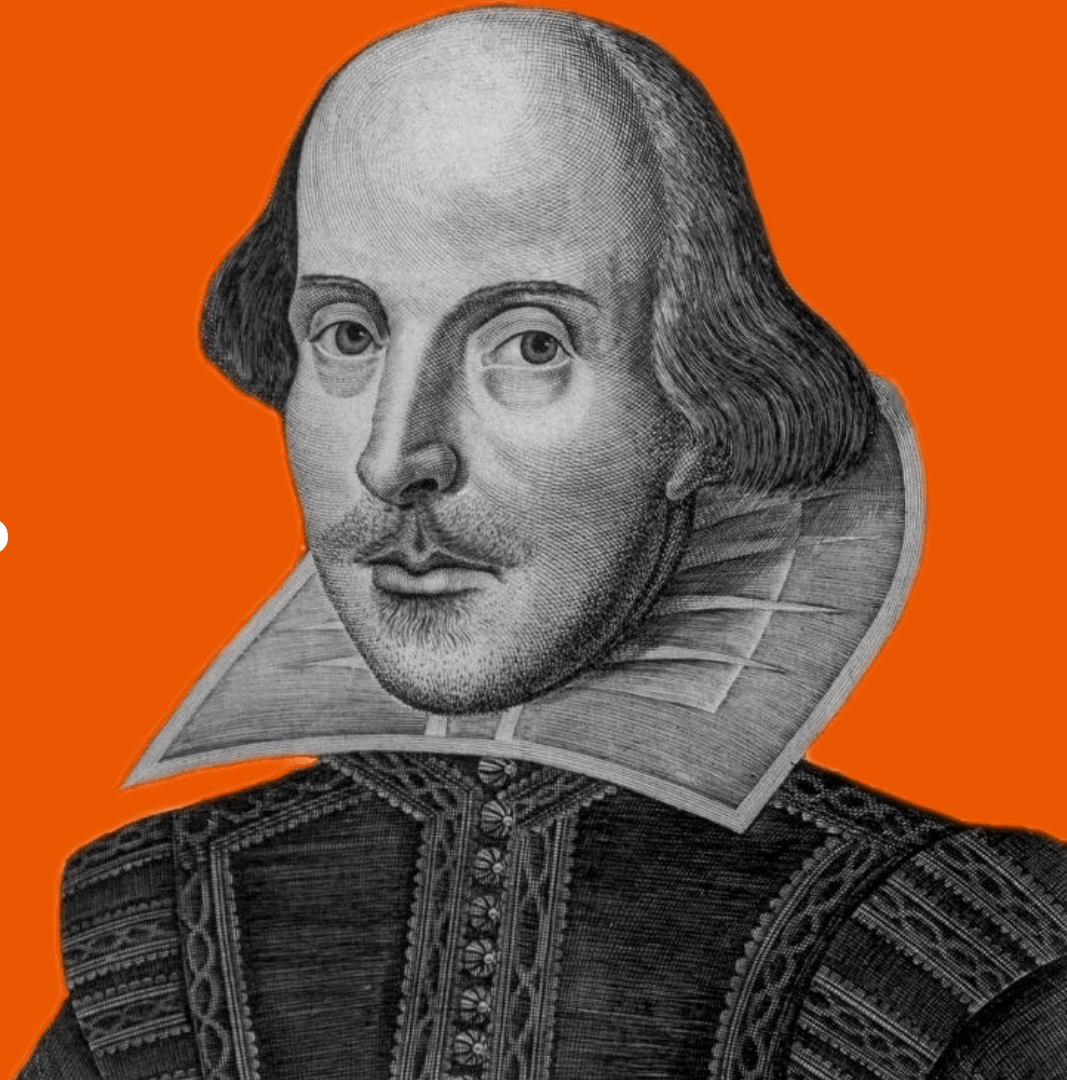


Once you're “locked in”

`composer update` will:

- Discover all available dependency versions
- Determine which versions of packages should be installed.
- ~~Create~~ **update** `composer.lock`
- Install the exact dependencies defined in `composer.lock`.

**To commit or
not to commit?**



Commit composer.lock.
But not vendor.




Commit composer.lock, not vendor.

Composer has indicated that this is the **best practice** because it **avoids the following problems**:

- Large VCS repository size and diffs when you update code.
- Duplication of the history of all your dependencies in your own VCS.
- Adding dependencies installed via git to a git repo will show them as submodules.

<https://getcomposer.org/doc/faqs/should-i-commit-the-dependencies-in-my-vendor-directory.md>



If you commit composer.lock and not vendor

Everyone gets the exact same
dependencies.

But your repo is much lighter, and
you avoid git submodule hell.

```
$ git clone [something] my-new-project
$ cd my-new-project

# Populate vendor.
$ composer install
```

How do I get vendor on prod?

—



This assumes you have scripted deployments

Like, a build server or CI server that can:

- Run `composer install` for you.
- Commit and push vendor to prod.

We're going to get into this later in [advanced topics](#).

WAT.



**Don't worry,
you can still commit vendor.**

Easy, documented alternative.

<https://getcomposer.org/doc/faqs/should-i-commit-the-dependencies-in-my-vendor-directory.md>

Recap.





What did we learn?

Composer concepts:

- dependency management
- composer files/dirs
 - composer.json
 - composer.lock
 - vendor
 - bin
- packagist
- versions & constraints
- what to commit

Composer commands:

- init
- require
- install
- update

That was vanilla Composer.



Drupal complicates things.



Drupal considerations

- Drupal modules aren't on Packagist
- Drupal doesn't use Semantic Versioning. E.g., 8.x-2.0.
- Drupal doesn't install modules, themes, etc in vendor.


Drupal core doesn't fully support Composer out-of-the-box, hence the [Composer Initiative](#).

But [there are tools that address these issues](#).



Composer template for Drupal projects

<https://github.com/drupal-composer/drupal-project>



Use it as a template for a new project.

Inherit default composer.json,
preconfigured.

```
$ composer create-project  
drupal-composer/drupal-project:8.x-dev  
some-dir  
--stability dev  
--no-interaction
```

Preconfigured with what?

—



Drupal.org as a package source

```
"repositories": [  
  {  
    "type": "composer",  
    "url": "https://packages.drupal.org/8"  
  }  
],
```




Packagist

Sinop
75
Atrable
300

\$2.50

3



Packagist

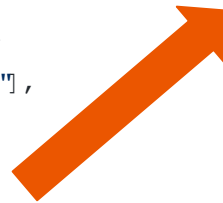
Drupal.org



Correct project install locations

```
"require": {
    "composer/installers": "^1.2"
},
"extra": {
    "installer-paths": {
        "web/core": ["type:drupal-core"],
        "web/libraries/{$name}": ["type:drupal-library"],
        "web/modules/contrib/{$name}": ["type:drupal-module"],
        "web/profiles/contrib/{$name}": ["type:drupal-profile"],
        "web/themes/contrib/{$name}": ["type:drupal-theme"],
        "drush/contrib/{$name}": ["type:drupal-drush"]
    }
}
```

- composer.json
- composer.lock
- vendor
- web
 - core
 - libraries
 - modules/contrib
 - profiles/contrib
 - themes/contrib
 - drush/contrib

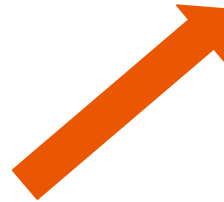




Required dependencies, like Drupal

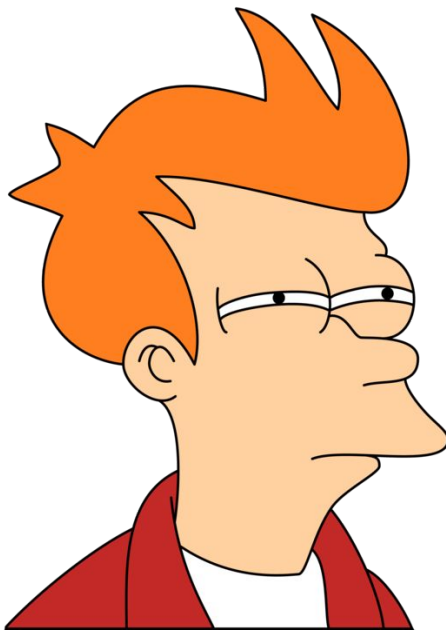
```
"require": {  
    "drupal/core": "~8.5.1"  
},
```

- composer.json
- composer.lock
- vendor
- web
 - core
 - libraries
 - modules/contrib
 - profiles/contrib
 - themes/contrib
 - drush/contrib



Required dependencies, like Drupal

```
"require": {  
    "drupal/core": "~8.5.1"  
},
```



- composer.json
- composer.lock
- vendor
- web
 - core
 - libraries
 - modules/contrib
 - profiles/contrib
 - themes/contrib
 - drush/contrib

Drupal scaffold

```
"require": {  
    "drupal-composer/drupal-scaffold": "^2.2"  
},
```

- composer.json
- composer.lock
- vendor
- web



- .htaccess
- core
- index.php
- libraries
- modules/contrib
- profiles/contrib
- robots.txt
- themes/contrib
- drush/contrib



A prepopulated .gitignore

```
web/core  
web/libraries  
web/modules/contrib  
web/profiles/contrib  
web/themes/contrib  
web/drush/contrib  
vendor
```

Patches



```
"require": {
    "cweagans/composer-patches": "^1.6"
},
"extra": {
    "patches": {
        "drupal/core": {
            "Clear Twig caches on deploys":
                "https://www.drupal.org/files/issues/2752961-130.patch"
        }
    }
}
```




Drupal community tools

```
"require": {  
    "drupal/console": "^1.0.2",  
    "drush/drush": "^9.0.0"  
},
```

Lots more.



Common Drupal tasks with Composer





Installing a module

Will be added to
[web/modules/contrib](#) thanks to
composer/installers config.

```
$ composer require drupal/token
```

downloaded
drupal/token 8.1.1.

**But Drupal doesn't use
semver?**

8.X-1.1

—

~~8.X-1.1~~

—

~~8.x~~-1.1.0

—

8.X-1.1 => 1.1.0

—

Drupal project versions



Drupal.org format	Translated semver format
{core.x}-{major}.{minor}-{stability}	{major}.{minor}.0-{stability}
8.x-3.4	3.4.0
8.x-2.10-rc2	2.10.0-rc2
8.x-1.0-alpha5	1.0.0-alpha5
7.x-1.x-dev	1.x-dev



Updating core

Updates drupal core only.

```
$ composer update drupal/core
```



Updating core

Updates **drupal core** and
all the packages that it requires.


```
$ composer update drupal/core  
--with-all-dependencies
```



Updating modules

Updates the **token module** and
all the packages that it requires.

```
$ composer update drupal/token  
--with-all-dependencies
```



Updating all the things

Updates **everything** (all modules, themes, core, and the packages they depend on) **within bounds of version constraints**.

```
$ composer update
```

Now you try!





Hands on tasks

- Install Composer
- Try Composer without Drupal
 - Create project
 - Require dependency
 - Implement
 - Test
- Try with drupal-project
 - Create project
 - require module
 - Require something from packagist



System Requirements

PHP 5.3.2+

For today's workshop, please use an OSX or Linux OS.

Windows users, please consider using Drupal VM.

bit.ly / 2qcyQ7w

grasmash.github.io/drupal-composer-training

For hands-on guide.

We will be here to answer questions and assist.


30 minutes.



Advanced topics.



Deploying to hosting.



Let's generate a build artifact.

A what?



Things you need for
development
process



Things you need for
production
website

Development



Artifact



Development (SASS)

```
#navbar {  
  width: 80%;  
  border: solid red 1px;  
  float: left;  
  padding: 5px;  
  
  ul {  
    list-style-type: none;  
  }  
  li {  
    float: right;  
    a {  
      font-weight: bold;  
      color: red;  
    }  
  }  
}
```

Artifact (CSS)

```
#navbar {  
  width: 80%;  
  border: solid red 1px;  
  float: left;  
  padding: 5px;  
}  
  
#navbar ul {  
  list-style-type: none;  
}  
  
#navbar ul li {  
  float: right;  
}  
  
#navbar ul li a {  
  font-weight: bold;  
  color: red;  
}
```

Development

```
.git/  
.gitignore  
.idea/  
.travis.yml  
.vagrant/  
README.md  
Vagrantfile  
composer.json  
composer.lock  
config/  
drush/custom  
patches/  
scripts/  
tests/  
web/modules/custom  
web/themes/custom
```

Artifact

Development

```
.git/  
.gitignore  
.idea/  
.travis.yml  
.vagrant/  
README.md  
Vagrantfile  
composer.json  
composer.lock  
config/  
drush/custom  
patches/  
scripts/  
tests/  
web/modules/custom  
web/themes/custom
```

Artifact

```
.git/  
.gitignore  
composer.json  
composer.lock  
config/  
drush/  
scripts/  
web/modules/custom  
web/themes/custom
```

Development

```
.git/  
.gitignore  
.idea/  
.travis.yml  
.vagrant/  
README.md  
Vagrantfile  
composer.json  
composer.lock  
config/  
drush/custom  
patches/  
scripts/  
tests/  
web/modules/custom  
web/themes/custom
```

Artifact

```
.git/  
.gitignore  
composer.json  
composer.lock  
config/  
drush/custom  
drush/contrib  
scripts/  
web/core  
web/libraries  
web/modules/contrib  
web/modules/custom  
web/modules/contrib  
web/themes/custom  
vendor/
```

How do we make an artifact?

—

With a script

That turns the source code in your git repo's branch into the artifact.

This is typically run by a build server or a CI server. But you can run it locally too.



Simple example



Create a build branch

```
$ git checkout -b master-build
```




create a dedicated branch
for build artifacts



Install production dependencies

```
$ composer install --no-dev --optimize-autoloader --prefer-dist
```



exclude require-dev
dependencies



Install production dependencies

```
$ composer install --no-dev --optimize-autoloader --prefer-dist
```



Improve autoloading performance. Up
to 37% PHP performance boost!



Install production dependencies

```
$ composer install --no-dev --optimize-autoloader --prefer-dist
```



avoid cloning .git repos to vendor



Remove any git submodules

```
$ find 'vendor' -type d | grep '\.git' | xargs rm -rf
```

```
$ find 'web' -type d | grep '\.git' | xargs rm -rf
```



Force add .gitignored dirs with vendor files

```
$ git add -f 'web/core'
```

```
$ git add -f 'web/modules/contrib'
```

```
$ git add -f 'web/themes/contrib'
```

```
$ git add -f 'web/profiles/contrib'
```

```
$ git add -f 'web/libraries'
```

```
$ git add -f 'vendor'
```



Commit and push upstream

```
$ git commit -m "Prepping for 1.0.0 release."
```

```
$ git push origin master-build
```

Troubleshooting Composer



**Dependency resolution is
complicated**



Common issues

- Your platform (PHP version) is incompatible with your requirements
- Your application requires two packages that are incompatible with each other
- Your application requires a non-existent/invalid version of a package
- You execute a command that has too narrow a scope



Common fixes

- Change your system's version of PHP, or set a minimum compatibility level
- Change the version constraint for one or more of your requirements
- Change the command that you're running so that more packages can be updated at once



Common issues

- Your platform (PHP version) is incompatible with your requirements
- Your application requires two packages that are incompatible with each other
- Your application requires a non-existent/invalid version of a package
- You execute a command that has too narrow a scope



Set PHP version

```
"config": {  
  "platform": {  
    "php": "5.6"  
  }  
},
```



Incompatible package versions

Your composer.json requires two packages that are incompatible with each other.

```
"require": {  
    "phpunit/phpunit": "^7.0.1",  
    "phpunit/php-timer": "~1.0"  
}
```

Your requirements could not be resolved to an installable set of packages.

Problem 1

- phpunit/phpunit 7.1.1 requires phpunit/php-timer ^2.0 -> satisfiable by phpunit/php-timer[2.0.0, 2.0.x-dev] but these conflict with your requirements or minimum-stability.
- phpunit/phpunit 7.1.0 requires phpunit/php-timer ^2.0 -> satisfiable by phpunit/php-timer[2.0.0, 2.0.x-dev] but these conflict with your requirements or minimum-stability.
- phpunit/phpunit 7.0.3 requires phpunit/php-timer ^2.0 -> satisfiable by phpunit/php-timer[2.0.0, 2.0.x-dev] but these conflict with your requirements or minimum-stability.
- phpunit/phpunit 7.0.2 requires phpunit/php-timer ^2.0 -> satisfiable by phpunit/php-timer[2.0.0, 2.0.x-dev] but these conflict with your requirements or minimum-stability.
- phpunit/phpunit 7.0.1 requires phpunit/php-timer ^2.0 -> satisfiable by phpunit/php-timer[2.0.0, 2.0.x-dev] but these conflict with your requirements or minimum-stability.
- Installation request for phpunit/phpunit ^7.0.1 -> satisfiable by phpunit/phpunit[7.0.1, 7.0.2, 7.0.3, 7.1.0, 7.1.1].

Installation failed, reverting ./composer.json to its original content.

**The error output is confusing.
Even for us.**



Common fixes

- Change your system's version of PHP
- **Change the version constraint for one or more of your requirements**
- Change the command that you're running so that more packages can be updated at once



Package versions

```
"require": {  
    "phpunit/phpunit": "^7.0.1",  
    "phpunit/php-timer": " ^2.0"  
}
```



Common fixes

- Change your system's version of PHP
- Change the version constraint for one or more of your requirements
- **Change the command that you're running so that more packages can be updated at once**



Update granularity

- `composer update [packages]`
- `composer update [packages] --with-all-dependencies`
- `composer update`

Thank you!



Sprints are on Friday!

Rate us.

We want feedback, and we want to be invited back.

