

# **Composer 101**

**Dependency management in PHP & Laravel**

**You can download this presentation as PDF**

# What is Composer?

- Manages all PHP project dependencies
- Manages dependencies versions
- Handles PHP autoloading
- De-facto standard for installing PHP libraries



# Creating project

Optional package version

`--prefer-dist` would try to download and unzip files, without this flag composer would use `GIT`

```
composer create-project laravel/laravel=7.12.0 your-project-name --prefer-dist
```

Full package name consists of vendor & package name

Your project name

# Adding new dependency

Package name to install

Optionally, package version to install

```
composer require --dev barryvdh/laravel-ide-helper=2.7.0
```

--dev means add to development dependencies list

```
composer require mll-lab/laravel-graphql-playground=2.1.0
```

By default (without --dev) packages are added to default list

Some packages you need always (like `laravel/framework`), others you need for local development only

# Dependency version

Most commonly, you will see and use `^1.0` format. It means install the most recent version above or equal `1.0`, but before `2.0`

"Exactly 1.0.2": `1.0.2`

"Above or equal 1.0": `>=1.0`

"Above/equal 1.0, less than 2.0": `>=1.0 <2.0`

"Above/equal 1.0, but less than 1.1 or above/equal 1.2": `>=1.0 <1.1 || >=1.2`

"1.0.0 up to and including 2.1.0": `1.0.0 - 2.1.0`

"Anything equal/above 1.0.0, less than 1.1": `1.0.*`

"`^1.2.3`": `>=1.2 <2.0.0`

"`~1.2`": `>=1.2 <2.0.0`

"`~1.2.3`": `>=1.2.3 <1.3.0`

# Installing dependencies

If `composer.lock` does not exist, it will be created after running this command

```
composer install --prefer-dist  
composer install --no-dev
```

`--prefer-dist` to download and unzip files (not use GIT). It's faster, used on production servers mostly

`--no-dev` to not install development dependencies (installs everything by default)

# Updating dependencies

Update everything

Update only  
laravel/framework  
package

Update all packages  
from laravel vendor

```
composer update  
composer update laravel/framework  
composer update laravel/framework laravel/tinker  
composer update laravel/*
```

Update laravel/  
framework and  
laravel/tinker  
(you can list more  
packages)



# composer.json

Where dependencies are defined

- Specifies what libraries project needs
- Contains library version constraints
- Contains PHP and PHP extensions constraints

```
{
  "name": "vendor/project-name",
  "require": {
    "php": "^7.2.5",
    "fideloper/proxy": "^4.2",
    "fruitcake/laravel-cors": "^1.0",
    "guzzlehttp/guzzle": "^6.3",
    "laravel/framework": "^7.0",
    "laravel/tinker": "^2.0"
  },
  "require-dev": {
    "barryvdh/laravel-ide-helper": "2.7.0",
    "facade/ignition": "^2.0",
    "fzaninotto/faker": "^1.9.1",
    "mockery/mockery": "^1.3.1",
    "nunomaduro/collision": "^4.1",
    "phpunit/phpunit": "^8.5"
  }
}
```



# composer.lock

## What should be installed

- Locks libraries to specific versions
- Guarantees libraries will be installed in specified versions
- Automatically generated (do not edit)

# Global installation

Install package globally on the machine

```
composer global require laravel/installer
```

**macOS:** `$HOME/.composer/vendor/bin`

**Windows:** `%USERPROFILE%\AppData\Roaming\Composer\vendor\bin`

**GNU / Linux Distributions:** `$HOME/.config/composer/vendor/bin` or  
`$HOME/.composer/vendor/bin`

Add this to your PATH on your system to be able to use global composer tools, like Laravel Installer

Installing `laravel/installer` globally let's you create new Laravel projects like that

```
laravel new bloggl
```

# Scripts

- Run PHP code or shell script after or before certain actions
- Makes it easy to automate tasks

```
"scripts": {  
    "post-autoload-dump": [  
        "Illuminate\\Foundation\\ComposerScripts::postAutoloadDump",  
        "@php artisan package:discover --ansi"  
    ],  
    "post-root-package-install": [  
        "@php -r \"file_exists('.env') || copy('.env.example', '.env');\""br/>    ],  
    "post-create-project-cmd": [  
        "@php artisan key:generate --ansi"  
    ]  
}
```

# Autoloading

## And PHP namespaces

- Defines the namespace prefix for specific directory
- By default, Laravel makes every class in `app/` directory prefixed with `App` namespace

```
<?php
namespace App;

class Post extends Model {

}
```

```
"autoload": {
    "psr-4": {
        "App\\": "app/"
    },
    "classmap": [
        "database/seeds",
        "database/factories"
    ]
},
"autoload-dev": {
    "psr-4": {
        "Tests\\": "tests/"
    }
}
```