



PHP Composer

Eko Kurniawan Khannedy

Eko Kurniawan Khannedy

- Technical architect at one of the biggest ecommerce company in Indonesia
- 10+ years experiences
- www.programmerzamannow.com
- youtube.com/c/ProgrammerZamanNow





Eko Kurniawan Khannedy

- Telegram : [@khannedy](https://t.me/khannedy)
- Facebook : fb.com/ProgrammerZamanNow
- Instagram : instagram.com/programmerzamannow
- Youtube : youtube.com/c/ProgrammerZamanNow
- Telegram Channel : t.me/ProgrammerZamanNow
- Email : echo.khannedy@gmail.com



Sebelum Belajar

- PHP Dasar
- PHP Object Oriented Programming
- PHP 8
- PHP Database
- PHP Web



Agenda

- Pengenalan Dependency Management
- Pengenalan Composer
- Membuat Project
- Autoload
- Repository
- Upload ke Repository
- Download dari Repository
- Dan lain-lain

Pengenalan Composer



Sebelum Dependency Management

- Saat kita membuat aplikasi, biasanya kita sering sekali membutuhkan library atau framework
- Sebelum menggunakan dependency management, jika kita membutuhkan library atau framework, maka kita perlu download library atau framework tersebut secara manual. Setelah itu kita masukkan ke dalam kode program kita
- Jika library nya sederhana, mungkin masih mudah untuk mengelolanya, tapi bagaimana jika ternyata library tersebut membutuhkan library lain, dan tidak hanya satu, tapi ada banyak?
- Alhasil kita harus download semua library yang dibutuhkan oleh library yang kita gunakan
- Belum lagi, kita harus tahu versi berapa library lain yang digunakan oleh library yang kita gunakan
- Dan kadang project kita pun menjadi bengkak karena ukuran library yang terlalu banyak



Dependency Management

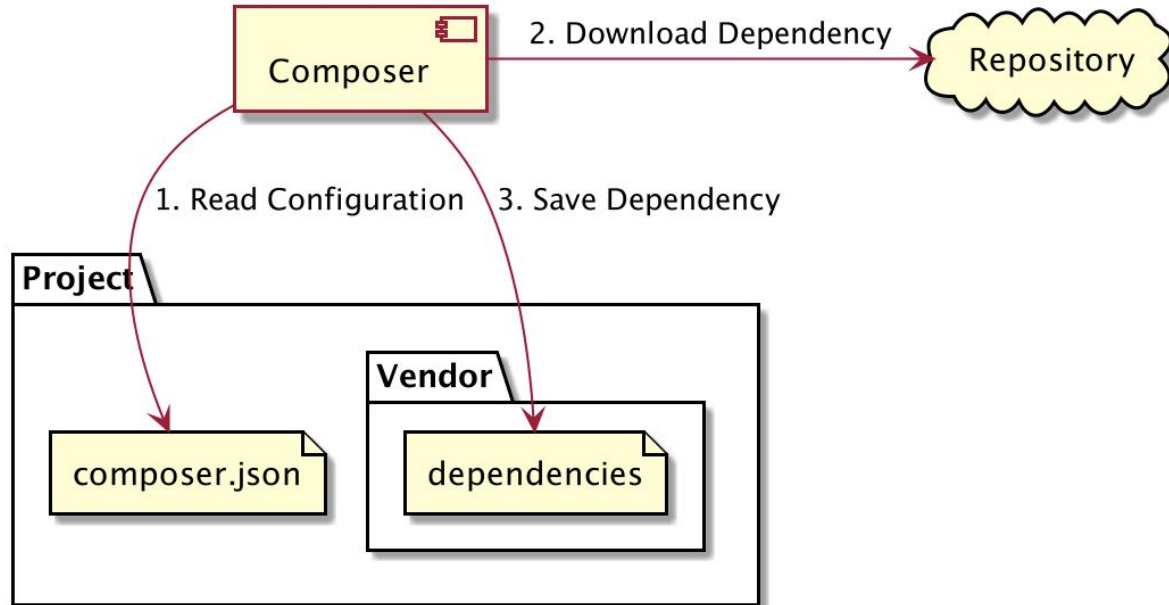
- Masalah yang sudah kita bahas sebelumnya, bisa kita hindari jika kita menggunakan Dependency Management Tool
- Dependency Management Tool bertugas untuk mendownload semua library yang kita butuhkan beserta library yang dibutuhkan oleh library yang kita gunakan
- Semua dilakukan secara otomatis
- Dan juga Dependency Management Tool bisa download library sesuai dengan versi yang dibutuhkan secara otomatis



Pengenalan Composer

- Composer adalah salah satu Dependency Management Tool yang populer untuk PHP
- Composer terinspirasi dari dependency management NPM (nodejs) dan Bundler (ruby)
- Dengan Composer, kita cukup membuat konfigurasi file yang berisi dependency yang kita butuhkan
- Composer akan secara otomatis download semua library dan dependency nya yang dibutuhkan sesuai dengan versi yang kita gunakan
- Kita juga bisa update versi library yang kita gunakan dengan mudah hanya dengan mengubah versi di file konfigurasi yang terdapat di project kita
- <https://getcomposer.org/>

Cara Kerja Composer



Menginstall Composer



Menginstall Composer

- <https://getcomposer.org/download/>
- Untuk menginstall Composer sangatlah sederhana, kita cukup download binary file composer, lalu masukkan ke dalam path directory bin PHP, atau bisa gunakan directory terpisah, namun kita perlu menambahkan ke PATH lagi jika menggunakan directory terpisah
- Setelah menginstall Composer di PATH, kita bisa gunakan perintah berikut di terminal / command line untuk mengecek versi composer yang sudah terinstall :
composer --version

Membuat Project Composer



Membuat Project Composer

- Untuk membuat project composer sangat sederhana, kita cukup membuat file composer.json pada project PHP kita
- Oleh karena itu, kita bisa menambahkan di project baru ataupun di project lama
- Namun jika ingin secara otomatis, kita bisa menggunakan perintah :
composer init



Composer Init

```
→ belajar-php-composer composer init
```

```
Welcome to the Composer config generator
```

This command will guide you through creating your composer.json config.

Package name (<vendor>/<name>) [khannedy/belajar-php-composer]: programmerzamannow/belajar-php-composer

Description []: Belajar PHP Composer

Author [Eko Kurniawan Khannedy <echo.khannedy@gmail.com>, n to skip]:

Minimum Stability []:

Package Type (e.g. library, project, metapackage, composer-plugin) []: project

License []:



composer.json

```
1 {
2     "name": "programmerzamannow/belajar-php-composer",
3     "description": "Belajar PHP Composer",
4     "type": "project",
5     "authors": [
6         {
7             "name": "Eko Kurniawan Khannedy",
8             "email": "echo.khannedy@gmail.com"
9         }
10    ],
11    "require": {}
12 }
```




Menginstall Dependency

- Setelah kita membuat project Composer, selanjutnya kita perlu menginstall dependency
- Walaupun sampai sekarang kita belum menambah dependency apapun, tapi itu tidak masalah
- Untuk menginstall dependency, kita bisa menggunakan perintah :
composer update
- Perintah composer update, akan menginstall semua dependency yang terdapat di file **composer.json**, lalu semua dependency akan di update di file **composer.lock**
- Hasil semua instalasi dependency, akan disimpan di folder vendor



composer update

```
→ belajar-php-composer composer update
Loading composer repositories with package information
Updating dependencies
Nothing to modify in lock file
Writing lock file
Installing dependencies from lock file (including require-dev)
Nothing to install, update or remove
Generating autoload files
```

Hello World



Hello World

- Composer secara otomatis akan menyimpan semua library yang kita gunakan di directory vendor
- Lantas bagaimana cara include file library nya?
- Kita tidak perlu melakukan satu per satu, kita cukup menggunakan file `/vendor/autoload.php` yang sudah secara otomatis di generate oleh composer
- File `autoload.php` tersebut bisa secara otomatis meng-include class-class yang kita butuhkan di library
- Jadi di file PHP kita, kita cukup hanya menambahkan `autoload.php` saja



Kode : Hello World

```
PHP HelloWorld.php x
1  <?php
2
3  require_once __DIR__ . '/vendor/autoload.php';
4
5  echo "Hello World" . PHP_EOL;
6
```



Autoload



Autoload

- Composer menggunakan fitur PHP Class Autoload untuk load class yang terdapat di directory vendor
- <https://www.php.net/manual/en/language.oop5.autoload.php>
- Selain itu, kita juga bisa memasukkan source code di project kita ke autoload agar kita tidak perlu melakukan include satu per satu file PHP class nya
- Namun agar bisa autoload, ada standar yang perlu kita ikuti



Kode : Autoload

```
] ,  
"autoload": {  
    "psr-4": {  
        "ProgrammerZamanNow\\": "src/"  
    }  
},  
"require": {}  
}
```




composer dump-autoload

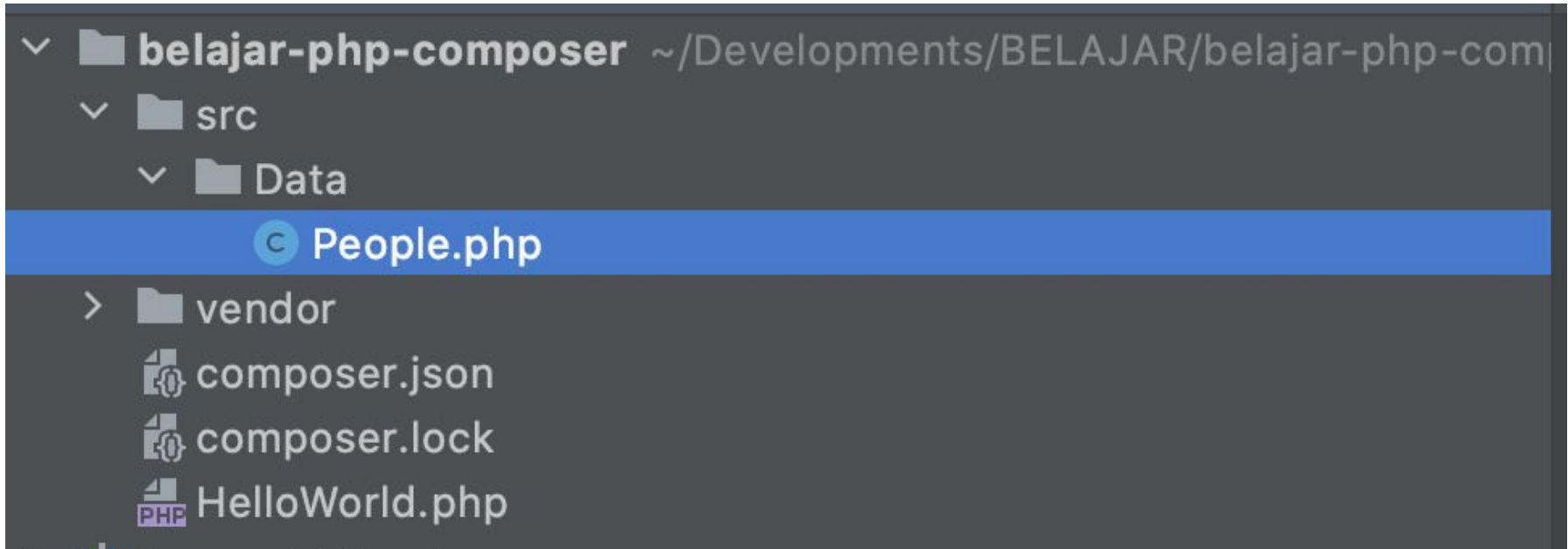
- Setelah kita menambah autoload, kita perlu melakukan generate ulang file autoload.php
- Untuk melakukan itu, kita bisa menggunakan perintah :
composer dump-autoload



Aturan Pembuatan Source Code

- Pada composer.json sebelumnya, kita sudah menggunakan namespace ProgrammerZamanNow di directory src
- Artinya jika kita import class di namespace ProgrammerZamanNow, nama dia akan include file di directory src
- Jika namespace nya ProgrammerZamanNow, maka foldernya adalah src, jika namespace nya ProgrammerZamanNow\Data, maka foldernya adalah src/Data
- Sedangkan untuk nama file, harus sama dengan nama class, jika nama class nya People, maka nama file harus People.php

Struktur Directory





Kode : Class People

```
namespace ProgrammerZamanNow\Data;

class People
{
    public function __construct(private string $name)
    {
    }

    public function sayHello(string $name): string
    {
        return "Hello $this->name";
    }
}
```



Kode : Menggunakan Autoload

```
require_once __DIR__ . '/vendor/autoload.php';  
  
use ProgrammerZamanNow\Data\People;  
  
$people = new People("Eko");  
  
echo $people->sayHello("Budi") . PHP_EOL;
```

Repository



Repository

- Repository merupakan tempat semua dependency di simpan
- Secara default, composer menggunakan repository Packagist
- <https://packagist.org/>
- Kita juga bisa menambahkan repository selain packagist jika mau :
<https://getcomposer.org/doc/05-repositories.md>

Menambah Dependency



Menambah Dependency

- Salah satu keuntungan menggunakan Composer adalah, kita bisa dengan mudah menambahkan dependency library yang kita butuhkan
- Terdapat dua jenis library di composer, library yang digunakan ketika development, dan library yang digunakan ketika aplikasi berjalan
- Contoh library yang digunakan ketika development adalah unit test misalnya
- Untuk menambah library, kita bisa tambahkan di composer dengan attribute require (untuk library aplikasi), dan require-dev (untuk library development)



Composer Dependency

```
{  
  "require": {  
    "library1" : "version",  
    "library2" : "version"  
  },  
  "require-dev": {  
    "librarydev1" : "version",  
    "librarydev2" : "version"  
  }  
}
```



Versi Library

- Saat menambahkan library, kita perlu menentukan versi berapa yang akan kita gunakan
- Ada beberapa cara menggunakan versi library di composer, semuanya tertulis di halaman resmi composer : <https://getcomposer.org/doc/articles/versions.md>
- Version di Composer mengikuti Semantic Versioning <https://semver.org/>



Kode : Menambah Library Monolog

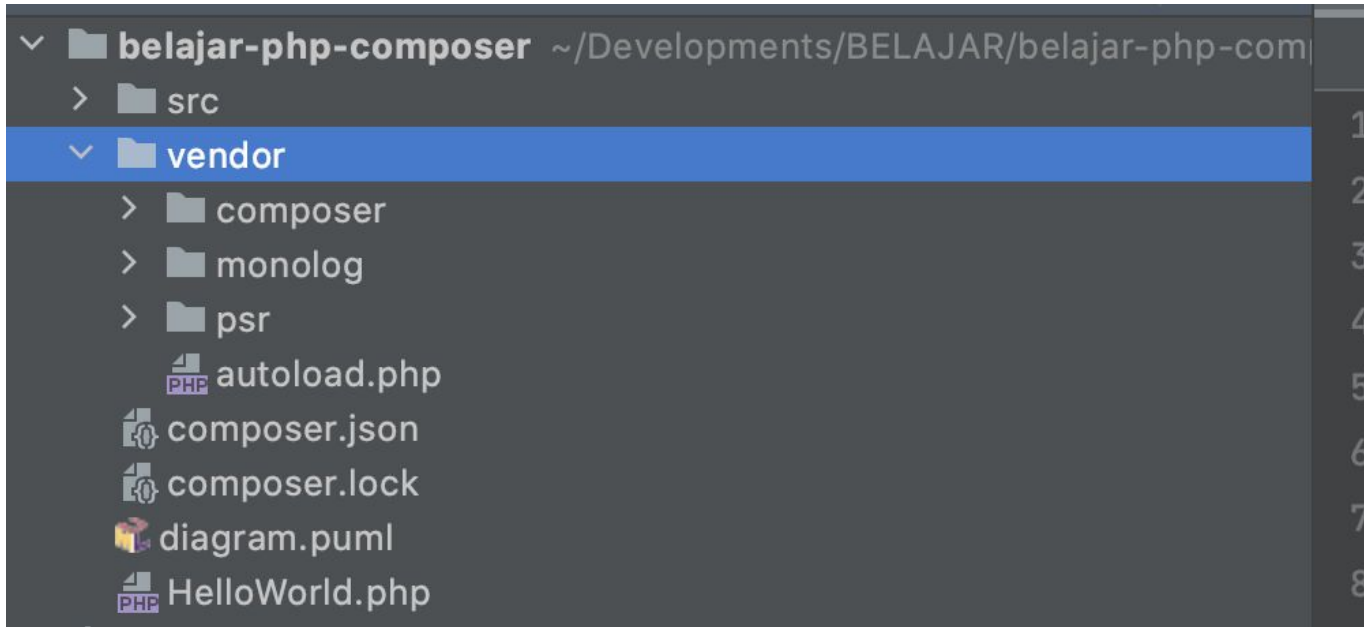
```
},  
"require": {  
  "monolog/monolog": "2.2.0"  
}  
}
```



Jangan Lupa

- Setelah menambah library, biasakan melakukan update : composer update
- Dan melakukan generate autoload : composer dump-autoload

Hasil Composer Update





Kode : Mencoba Monolog

```
require_once __DIR__ . '/vendor/autoload.php';

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

$log = new Logger('ProgrammerZamanNow');
$log->pushHandler(new StreamHandler('application.log', Logger::INFO));

$log->info('Hello World');
$log->info('Selamat Belajar Composer');

```

Membuat Library



Library

- Sekarang kita sudah tahu bagaimana menggunakan Composer di project aplikasi kita
- Sekarang pertanyaannya, bagaimana jika kita ingin membuat library?
- Library yang bisa digunakan di project kita atau oleh project orang lain?
- Composer tidak hanya mendukung pembuatan project, namun juga library



Membuat Library

```
→ belajar-php-composer-library composer init
```

```
Welcome to the Composer config generator
```

This command will guide you through creating your composer.json config.

Package name (<vendor>/<name>) [khannedy/belajar-php-composer-library]: programmerzamannow/belajar-php-composer-library

Description ☐

Author [Eko Kurniawan Khannedy <echo.khannedy@gmail.com>, n to skip]:

Minimum Stability ☐

Package Type (e.g. library, project, metapackage, composer-plugin) ☐

License ☐

Define your dependencies.

Would you like to define your dependencies (require) interactively [yes]? no

Would you like to define your dev dependencies (require-dev) interactively [yes]? no



Kode : composer.json

```
],  
  "require": {  
    "php": ">=8.0"  
  },  
  "autoload": {  
    "psr-4": {  
      "ProgrammerZamanNow\\Belajar\\": "src/"  
    }  
  }  
}
```



Kode : Class Customer

```
namespace ProgrammerZamanNow\Belajar;  
  
class Customer  
{  
    public function __construct(private string $name)  
    {  
    }  
  
    public function sayHello(string $name): string  
    {  
        return "Hello $name, My Name is $this->name";  
    }  
}
```

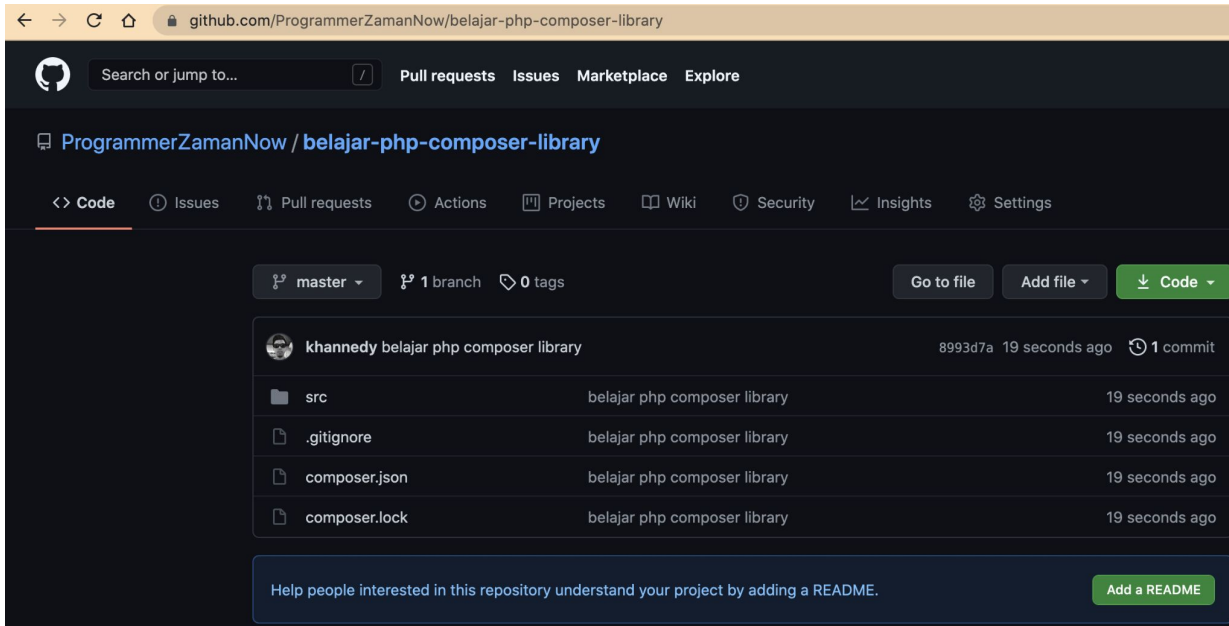
Upload ke Repository



Upload ke Repository

- Setelah selesai membuat library, kita bisa upload library yang sudah kita buat ke Git Repository
- Ada banyak Git Repository yang gratis, contohnya adalah Github
- Pada course ini saya tidak akan membahas tentang Git, karena Git dibuat dalam course terpisah

Contoh Git Repository



The screenshot shows a GitHub repository page for 'ProgrammerZamanNow/belajar-php-composer-library'. The page is dark-themed. At the top, there's a navigation bar with the GitHub logo, a search bar, and links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below this, the repository name is displayed. A secondary navigation bar includes links for 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. The 'Code' link is underlined. Below the navigation, there's a section for the repository's state: 'master' branch, '1 branch', and '0 tags'. To the right of this are buttons for 'Go to file', 'Add file', and 'Code'. The main content area shows a list of files and folders: 'src', '.gitignore', 'composer.json', and 'composer.lock'. Each item has a file icon, the name, the repository name, and the time since the last commit. At the bottom, there's a prompt to 'Add a README'.

github.com/ProgrammerZamanNow/belajar-php-composer-library

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

ProgrammerZamanNow / belajar-php-composer-library

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

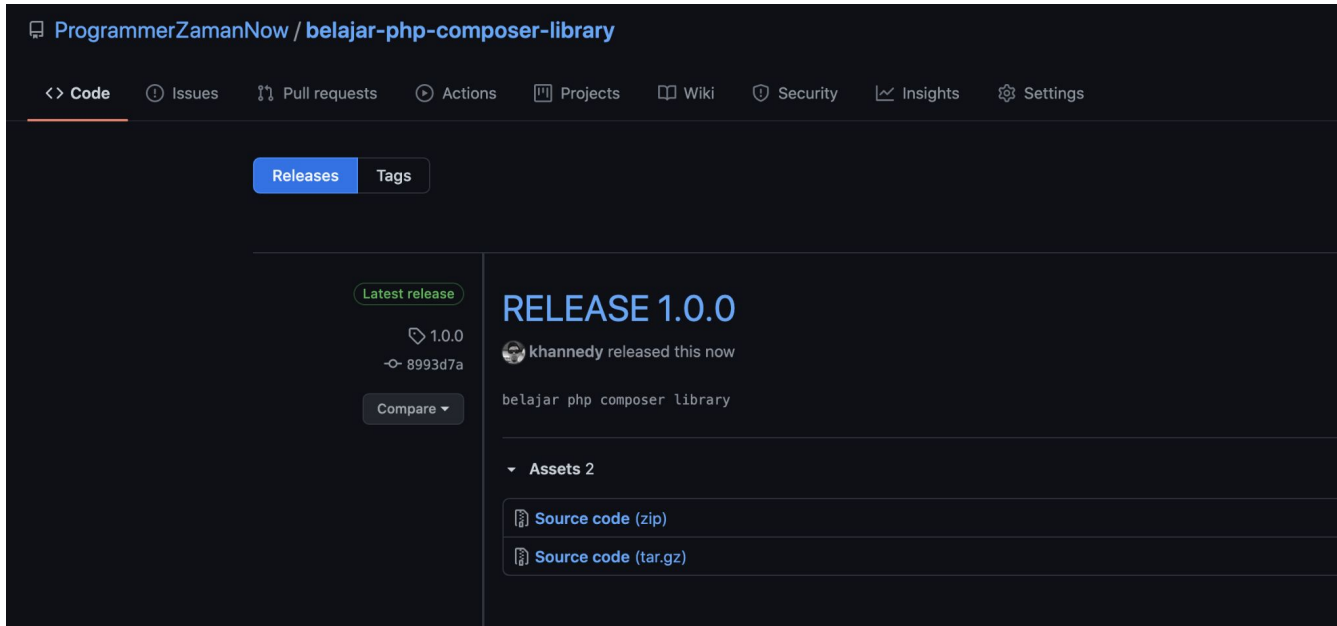
master 1 branch 0 tags Go to file Add file Code

khannedy belajar php composer library 8993d7a 19 seconds ago 1 commit

src	belajar php composer library	19 seconds ago
.gitignore	belajar php composer library	19 seconds ago
composer.json	belajar php composer library	19 seconds ago
composer.lock	belajar php composer library	19 seconds ago

Help people interested in this repository understand your project by adding a README. Add a README

Membuat Release atau Tag



The screenshot displays the GitHub interface for the repository 'ProgrammerZamanNow / belajar-php-composer-library'. The top navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The 'Releases' tab is selected, showing a list of releases. The latest release, 'RELEASE 1.0.0', is highlighted, indicating it was released 'this now' by user 'khannedy'. The release details show the source code in both zip and tar.gz formats.

ProgrammerZamanNow / [belajar-php-composer-library](#)

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Releases Tags

Latest release

1.0.0
8993d7a

Compare

RELEASE 1.0.0

khannedy released this now

belajar php composer library

Assets 2

- Source code (zip)
- Source code (tar.gz)

Download dari Repository



Download dari Repository

- Setelah library kita di upload ke repository, kita bisa menggunakan library tersebut di project kita
- Untungnya composer juga terintegrasi dengan Git repository, jadi kita bisa menambahkan git repository sebagai composer repository



Kode : Menambah Dependency

```
"repositories": [  
  {  
    "type": "vcs",  
    "url": "https://github.com/ProgrammerZamanNow/belajar-php-composer-library"  
  }  
],  
"require": {  
  "php": ">=8.0",  
  "programmerzamannow/belajar-php-composer-library": "1.0.0",  
  "monolog/monolog": "2.2.0"  
}
```



Kode : Hello Library

```
<?php

require_once __DIR__ . '/vendor/autoload.php';

$customer = new \ProgrammerZamanNow\Belajar\Customer("Eko");
echo $customer->sayHello("Budi");
```

Upgrade Versi Library



Upgrade Versi Library

- Saat membuat library, sudah pasti kita akan melakukan proses update dan upgrade
- Untuk melakukan upgrade library, caranya cukup mudah, kita hanya tinggal membuat update kode library, lalu membuat release atau tag baru



Kode : Update Kode Library

```
public function sayHello(string $name = "Guest"): string
{
    return "Hello $name, My Name is $this->name";
}
```

Membuat Release atau Tag Baru

ProgrammerZamanNow / belajar-php-composer-library

Watch 0

☆

<> Code

🕒 Issues

🔗 Pull requests

🔄 Actions

📁 Projects

📖 Wiki

🛡 Security

📊 Insights

⚙ Settings

Releases

Tags

Edit release

Delete


Latest release

2.0.0

914ac45

Compare

RELEASE 2.0.0

 khannedy released this now

Update Name

Assets 2

Source code (zip)

Source code (tar.gz)



Kode : Update Versi Library

```
"repositories": [  
  {  
    "type": "vcs",  
    "url": "https://github.com/ProgrammerZamanNow/belajar-php-composer-library"  
  },  
],  
"require": {  
  "php": ">=8.0",  
  "programmerzamannow/belajar-php-composer-library": "2.0.0",  
  "monolog/monolog": "2.2.0"  
}
```


Submit ke Packagist



Packagist

- Khusus OpenSource library, kita juga bisa submit ke Packagist
- Caranya pun sangat mudah, kita cukup registrasi, lalu submit repository kita di Github
- Packagist bisa secara otomatis mendeteksi versi library kita sesuai dengan Tag atau Release di Git

Submit ke Packagist

 *The PHP Package Repository*

Composer 2 is available!

Browse

Search packages...

Submit package

Repository URL (Git/Svn/Hg)

Check

Trying to share private code?

Use **Private Packagist** to share code through Composer without publishing it for everyone on Packagist.org.

Please make sure you have read the **package naming** rules for your package. The authoritative name of your package is the name in your `composer.json` file inside the main branch of your repository. Do not change it after that.

Do not submit forks of existing packages. If you forked a package to patch it, use **VCS Repositories** instead. If you intend on maintaining the fork free of charge, you can request to become a **maintainer** of the original package.

If you need help or if you have any questions please ask in the **community**.

Library di Packagist

★ programmerzamannow/belajar-php-composer-library

composer require programmerzamannow/belajar-php-composer-library

Warning: This package is not installable via Composer 1.x, please make sure you [upgrade](#) to Composer 2+.
[Read more about our Composer 1.x deprecation policy.](#)

There is no license information available for the latest version (2.0.0) of this package.

Belajar PHP Composer Library

[Abandon](#) [Delete](#) [Update](#) [Edit](#)

2.0.0

2021-05-18 09:01 UTC

requires

• php: >=8.0

requires (dev)

None

suggests

None

Maintainers



Details

[github.com/ProgrammerZamanNow/...](#)

[Source](#)

[Issues](#)

Installs: 0

Dependents: 0

Suggesters: 0

Security: 0

Stars: 0

Watchers: 0

Forks: 0

Open Issues: 0

dev-master



2.0.0



1.0.0





Keuntungan Menggunakan Packagist

- Kita tidak perlu menambahkan repository git di composer.json satu per satu
- Bisa sync secara otomatis ketika ada release versi baru



Kode : Update composer.json

```
"autoload": {  
    "psr-4": {  
        "ProgrammerZamanNow\\": "src/"  
    }  
},  
"require": {  
    "php": ">=8.0",  
    "programmerzamannow/belajar-php-composer-library": "2.0.0",  
    "monolog/monolog": "2.2.0"  
}
```

Fitur Lainnya



Fitur Lainnya

- Sebenarnya, sampai disini teman-teman sudah bisa mulai menggunakan composer
- Karena tujuan composer memang untuk dependency management
- Namun selain dependency management, masih banyak lagi fitur-fitur tambahan yang terdapat di composer
- Kita akan bahas sekilas saja, dan teman-teman bisa explore lebih dalam jika tertarik dengan fitur-fitur lainnya diluar dependency management



Script

- Fitur ini bisa digunakan untuk membuat custom script
- Dengan custom script ini, kita bisa menggunakan composer untuk membuat perintah misal :
composer script-saya



Kode : Contoh Script

```
    "scripts": {  
      "jalankan-server": "php -S localhost:8080",  
      "hello": "echo 'Hello World'"  
    }  
  }
```



Private Repository

- Saat kita membuat aplikasi di perusahaan, sudah pasti project dan library nya tidak akan public, alias private
- Composer juga mendukung private repository dengan menambahkan authentication
- Tiap jenis repository berbeda-beda cara authentication nya
- Kita bisa lihat di dokumentasinya
<https://getcomposer.org/doc/articles/authentication-for-private-packages.md>



Plugin

- Kita juga bisa mengubah atau meningkatkan fungsionalitas composer itu sendiri
- Untuk hal ini, di composer terdapat fitur plugin
- <https://getcomposer.org/doc/articles/plugins.md>



Vendor Binary

- Kadang library tidak hanya berisi kode PHP saja
- Kadang library juga berisi binary file untuk dieksekusi, contohnya jika teman-teman menggunakan library PHPUnit
- Fitur ini di composer bernama Vendor Binary
- <https://getcomposer.org/doc/articles/vendor-binaries.md>



Dan lain-lain

- <https://getcomposer.org/doc/>

Materi Selanjutnya



Materi Selanjutnya

- PHP Unit Test
- Perbanyak Studi Kasus
- Belajar Framework (Laravel atau Codeigniter)