
DrupalEurope
Darmstadt, Germany
Sep 10 - 14, 2018
www.drupaleurope.org

Lesser known perks of using composer

Drupal Europe 2018

Drupal + Technology

TRACK SUPPORTED BY

platform.sh

Special Thanks to..

Mohit Aghera

Drupal developer @Axelerant

Drupal.org: mohit_aghera

Twitter: @mohit_rocks

Quick Introduction

- Why do we need it?
- How do we use it?

Drupal Composer initiative

Follow updates here: https://www.drupal.org/project/ideas/issues/2958021

Composer scripts

- Define script in composer.json and run
- Helps to automate tasks
- Supports various events.

Composer Scripts

```
"scripts": {
    "pre-autoload-dump": "Drupal\\Core\\Composer\\Composer::preAutoloadDump",
    "post-autoload-dump": "Drupal\\Core\\Composer\\Composer::ensureHtaccess",
    "post-package-install": "Drupal\\Core\\Composer\\Composer::vendorTestCodeCleanup",
    "post-package-update": "Drupal\\Core\\Composer\\Composer::vendorTestCodeCleanup",
    "drupal-phpunit-upgrade-check": "Drupal\\Core\\Composer\\Composer::upgradePHPUnit",
    "drupal-phpunit-upgrade": "@composer update phpunit/phpunit --with-dependencies --no-progress",
```

Composer Plugins

- Helps to overcome limitation of composer scripts
- Run independently - irrespective of root package
- Helps in creating distributable package

Composer Plugins

- “type” : “composer-plugin”
- “extra” : Contains class name
- Add special package called “composer-plugin-api”

```
{
    "name": "my/plugin-package",
    "type": "composer-plugin",
    "require": {
        "composer-plugin-api": "^1.1"
    },
    "extra": {
        "class": "My\\Plugin"
    }
}
```

Composer Plugins

drupal-composer/drupal-scaffold

```php
class Plugin implements PluginInterface, EventSubscriberInterface, Capable {

  /**
   * @var \DrupalComposer\DrupalScaffold\Handler
   */
  protected $handler;

  /**
   * {@inheritdoc}
   */
  public function activate(Composer $composer, IOInterface $io) {...}

  /**
   * {@inheritdoc}
   */
  public static function getSubscribedEvents() {
    return array(
      PackageEvents::POST_PACKAGE_INSTALL => 'postPackage',
      PackageEvents::POST_PACKAGE_UPDATE => 'postPackage',
      PackageEvents::POST_PACKAGE_UNINSTALL => 'postPackage',
      ScriptEvents::POST_INSTALL_CMD => 'postCmd',
      ScriptEvents::POST_UPDATE_CMD => 'postCmd',
    );
  }

  /**
   * Post package event behaviour.
   *
   * @param \Composer\Installer\PackageEvent $event
   */
  public function postPackage(PackageEvent $event) {
    $this->handler->onPostPackageEvent($event);
  }
```

Composer Plugins

```
class Plugin implements PluginInterface, Capable {
    public function activate(Composer $composer, IOInterface $io) {}

    public function getCapabilities() {
        return array(
            'Composer\Plugin\Capability\CommandProvider' =>
    'My\Composer\CommandProvider',
        );
    }
}
```

Patching

- Requirement to update core of system
- Bugs that are not merged

Patching

```
    "extra": {
        "enable-patching": true,
        "installer-types": [
            "bower-asset",
            "npm-asset"
        ],
        "drupal-scaffold": {
            "excludes": [
                ".htaccess"
            ]
        },
        "patches": {
            "drupal/search_api_spellcheck": {
                "Warning when no spelling suggestions":
"patches/search_api_spellcheck--no-suggestions-warning.patch"
            },
            "drupal/config_ignore": {
                "Apply config ignore to drush exports": "https://www.drupal
.org/files/issues/2018-04-19/support-for-export-2857247-18.patch"
            }
        },
```

Keeping data up to date

- Pull and process remote data automatically
- Implement Composer Plugins

Keeping data up to date

Benefits:

- No need to have separate sync code
- Helps to solve problems in continuous integration as data will already be there

Limitations:

- It is meant for retrieving and storing data - no live sync.

Commit Hooks

```
commit f7a4c88a0b4b5cc24446aba576dad1d6f12a5087 (HEAD -> master)
Author: [redacted]
Date:   [redacted]

    Adding captainhook json configuration file.

commit e6d632edeb9ae091fe8439e0c19110e144c46bf9
Author: [redacted]
Date:   [redacted]

    Fixing bugs
```

Commit Hooks

- GIT hooks helps to enforce code quality
- Ensure commit messages are proper
- Coding standards validation
- Sharing GIT hooks with composer has more advantages

Commit Hooks

There are several libraries that supports this type of implementations:

- https://github.com/sebastianfeldmann/captainhook
- https://github.com/bruli/php-git-hooks
- https://github.com/php-composter/php-composter
- https://github.com/BrainMaestro/composer-git-hooks

```
{
  "commit-msg": {
    "enabled": true,
    "actions": [
      {
        "action": "\\SebastianFeldmann\\CaptainHook\\Hook\\Message\\Action\\Regex",
        "options": {
          "regex": "#.*#"
        }
      }
    ]
  },
  "pre-commit": {
    "enabled": true,
    "actions": [
      {
        "action": "./vendor/bin/phpcs --standard=psr2 src tests"
      }
    ]
  },
  "pre-push": {
    "enabled": true,
    "actions": [
      {
        "action": "composer test"
      }
    ]
  }
}
```

Automatting project structure

- Composer helps to generate essential files and other code
- Use case : drupal-composer/drupal-project

```
"scripts": {
    "drupal-scaffold": [
        "DrupalComposer\\DrupalScaffold\\Plugin::scaffold"
    ],
    "pre-install-cmd": [
        "DrupalProject\\composer\\ScriptHandler::checkComposerVersion"
    ],
    "pre-update-cmd": [
        "DrupalProject\\composer\\ScriptHandler::checkComposerVersion"
    ],
    "post-install-cmd": [
        "DrupalProject\\composer\\ScriptHandler::createRequiredFiles"
    ],
    "post-update-cmd": [
        "DrupalProject\\composer\\ScriptHandler::createRequiredFiles"
    ]
},
```

Example from “ScriptHandler.php”

```php
public static function createRequiredFiles(Event $event) {
  $fs = new Filesystem();
  $drupalFinder = new DrupalFinder();
  $drupalFinder->locateRoot(getcwd());
  $drupalRoot = $drupalFinder->getDrupalRoot();

  $dirs = [
    'modules',
    'profiles',
    'themes',
  ];

  // Required for unit testing
  foreach ($dirs as $dir) {
    if (!$fs->exists( files: $drupalRoot . '/'. $dir)) {
      $fs->mkdir( dirs: $drupalRoot . '/'. $dir);
      $fs->touch( files: $drupalRoot . '/'. $dir . '/.gitkeep');
    }
  }

  // Prepare the settings file for installation
  if (!$fs->exists( files: $drupalRoot . '/sites/default/settings.php') and
$fs->exists( files: $drupalRoot . '/sites/default/default.settings.php')) {
    $fs->copy( originFile: $drupalRoot . '/sites/default/default.settings.php',
 targetFile: $drupalRoot . '/sites/default/settings.php');
    require_once $drupalRoot . '/core/includes/bootstrap.inc';
    require_once $drupalRoot . '/core/includes/install.inc';
    $settings['config_directories'] = [
      CONFIG_SYNC_DIRECTORY => (object) [
        'value' => Path::makeRelative( path: $drupalFinder->getComposerRoot()
. '/config/sync', $drupalRoot),
        'required' => TRUE,
      ],
```

Thank you..

Become a Drupal contributor, Friday from 9 AM

- First timers workshop
- Mentored contribution
- General contribution