

Composer, autoloading PSR-4 i biblioteka phpdotenv

Czas: 2 × 45 min

praca indywidualna

Cel: Zarządzanie zależnościami projektu PHP Composerem: autoloading klas zgodny z PSR-4, instalacja biblioteki z Packagist i przeniesienie konfiguracji bazy do pliku `.env` poza repozytorium.

Po zajęciach potrafisz:

- tworzy plik `composer.json` poleceniem `composer init` i rozumie jego sekcje
- konfiguruje autoloading PSR-4 i zastępuje nim ręczne `require`
- instaluje bibliotekę, odczytuje ograniczenia wersji (`^5.7`) i rolę pliku `composer.lock`
- przechowuje dane dostępowe w `.env` (phpdotenv) z walidacją wymaganych zmiennych
- przygotowuje `.gitignore` tak, by `vendor/` i `.env` nie trafiły do repozytorium

Sprzęt i materiały

- komputer z Windows 11, XAMPP, Composer, Git, dostęp do internetu (Packagist)
- projekt z zadania 15 (lub folder `r2_16_start`) i plik `r2_16_sklep_oop.sql`

Oprogramowanie

- **Composer** — instalator Composer-Setup.exe; jako PHP wskaż `C:\xampp\php\php.exe`
getcomposer.org/download
- **Packagist** — strona biblioteki phpdotenv
packagist.org/packages/vlucas/phpdotenv
- **Git**
git-scm.com/downloads
- **XAMPP i Visual Studio Code** — VS Code:
<https://code.visualstudio.com/>
apachehttpd.apache.org/download.html

Przygotowanie przed zajęciami

- Sprawdź, czy na komputerze działa `composer --version`; jeśli nie, zainstaluj Composera (wymaga uprawnień — zgłoś nauczycielowi).

Bezpieczeństwo

- Plik `.env` zawiera dane dostępowe — nigdy nie wysyłaj go do repozytorium ani nie wklejaj do sprawozdania z prawdziwymi hasłami.
- Instaluj tylko biblioteki z wiarygodnych źródeł (Packagist, aktywne repozytorium, wiele pobrań); przed wdrożeniem uruchamiaj `composer audit`.

Przebieg zadania

1 composer init

Skopiuj projekt z zadania 15 do `htdocs\sklep_composer`. Polecenia wpisuj w terminalu VS Code w folderze projektu.

1. Uruchom `composer --version` i `php --version`; zanotuj wersje.
2. Uruchom `composer init`. Nazwa pakietu: `nazwisko/sklep-oop`, typ `project`, opis własny. Na pytania o interaktywne dodanie zależności odpowiedz `no`, na pytanie o mapowanie PSR-4 — `n` (dodasz je ręcznie). Obejrzyj powstały `composer.json`.

2 Autoload PSR-4

Zasada PSR-4: prefiks przestrzeni nazw odpowiada katalogowi, a reszta nazwy klasy — ścieżce pliku.

```
"autoload": {
    "psr-4": {
        "Sklep\\": "src/"
    }
}
```

1. Dopisz do `composer.json` sekcję z ramki i wykonaj `composer dump-autoload`. Obejrzyj folder `vendor/composer` — w którym pliku jest zapisane mapowanie `Sklep\\` → `src/`?
2. W `test.php` zastąp wszystkie `require_once` jednym `require __DIR__ . '/vendor/autoload.php'`; i uruchom testy — wynik ma być taki sam jak w zadaniu 15.
3. Dodaj nową klasę (np. `PocztaPolska`, jeśli jej nie masz) i sprawdź, że działa bez zmian w `test.php` poza testem. Kiedy po dodaniu klasy trzeba uruchomić `composer dump-autoload`, a kiedy nie?

3 Biblioteka phpdotenv i konfiguracja w .env

Zaimportuj `r2_16_sklep_oop.sql`.

```
# .env.example (wzór – trafia do repozytorium)
APP_ENV=dev
DB_HOST=localhost
DB_NAME=sklep_oop
DB_USER=root
DB_PASS=
```

1. Zainstaluj bibliotekę: `composer require vlucas/phpdotenv`. Zanotuj, jakie ograniczenie wersji dopisał Composer w `require` i jakie dodatkowe pakiety zainstalował (`composer show`, `composer show --tree`).
2. Utwórz `.env.example` (ramka) i jego kopię `.env`. Utwórz `config/bootstrap.php`, który łączy autoload, wczytuje `.env` (`Dotenv\Dotenv::createImmutable(...)->load()`), wymaga niepustych `DB_HOST`, `DB_NAME`, `DB_USER`, istnienia `DB_PASS` i wartości `APP_ENV` z listy `dev / prod` (`required()`, `notEmpty()`, `allowedValues()`), a na tej podstawie ustawia `display_errors`.
3. Napisz klasę `Sklep\Baza` z metodą statyczną `połącz(): PDO`, która tworzy jedno połączenie z danymi z `$_ENV`, oraz `Sklep\ProduktRepozytorium::wszystkie()` zwracającą tablicę obiektów `Produkt` z tabeli `produkty`.
4. Utwórz `public/index.php`: tabela produktów z bazy (netto, VAT, brutto) i przykładowe zamówienie z dwóch produktów. Otwórz `http://localhost/sklep_composer/public/`.
5. Sprawdź walidację: usuń z `.env` wiersz `DB_USER`, a potem zmień nazwę pliku `.env` — zanotuj komunikaty wyjątków. Przywróć plik.

4 Wersje, lock i repozytorium

1. Wyjaśnij w sprawozdaniu zapis `^5.7` (jakie wersje dopuszcza) i porównaj z `~5.7` oraz `5.7.*`. Odczytaj w `composer.lock` dokładną zainstalowaną wersję phpdotenv.
2. Utwórz `.gitignore` z wpisami `/vendor/` i `.env`. Wykonaj `git init`, `git add .`, `git status` — sprawdź, że na liście nie ma `vendor/` ani `.env`, ale są `composer.json`, `composer.lock` i `.env.example`. Zatwierdź zmiany (`git commit`).
3. Test odtworzenia: skopiuj repozytorium do nowego folderu poleceniem `git clone`, uruchom tam `composer install`, utwórz `.env` z `.env.example` i sprawdź działanie strony. Uruchom `composer audit` i `composer outdated`. Spakuj projekt bez `vendor/` jako `nazwisko_imie_zad16.zip`.

W sprawozdaniu umieść

- ☐ tabela wykonanych poleceń (krok.punkt) z poleceniami i fragmentami kodu
- ☐ plik `composer.json` i wynik `composer show --tree`
- ☐ komunikaty phpdotenv przy braku zmiennej i braku pliku `.env`
- ☐ zrzuty: wynik testów z autoloadem, strona `public/`, `git status` po dodaniu `.gitignore`
- ☐ archiwum `nazwisko_imie_zad16.zip` (bez `vendor/`) i odpowiedzi na pytania kontrolne

Pytania kontrolne

1. Czym różni się `composer install` od `composer update`? Które polecenie wykonasz po pobraniu projektu kolegi i dlaczego?
2. Dlaczego `composer.lock` dodajemy do repozytorium, a `vendor/` nie?
3. Jakie korzyści daje trzymanie konfiguracji w `.env` zamiast w kodzie PHP?
4. Co oznacza ograniczenie wersji `^5.7` i dlaczego jest bezpieczniejsze niż `*`?

Kryteria oceny

Kryterium	Punkty
<code>composer.json</code> z init, autoload PSR-4, testy działające przez <code>vendor/autoload.php</code>	5
Instalacja phpdotenv, <code>.env</code> / <code>.env.example</code> , bootstrap z walidacją zmiennych	4
Klasy <code>Baza</code> i <code>ProduktRepozytorium</code> , strona z produktami z bazy	4
Wersje i lock opisane poprawnie; <code>.gitignore</code> , commit, odtworzenie przez <code>composer install</code>	4
Zrzuty, archiwum, odpowiedzi na pytania	3
Razem	20

Sprawozdanie przygotuj na wzorze dołączonym do zadania (plik .docx o tym samym numerze).