۱ .دستور `composer init`چه کاری انجام می‌دهد؟

جواب: این دستور برای ساخت فایل `composer.json`استفاده می‌شود. این فایل مثل نقشه پروژه است که اطلاعاتی درباره پکیج‌ها، نام پروژه، autoloadو ... ذخیره می‌کند.

---

۲ .فایل `composer.json`چه کاربردی دارد؟

جواب: در این فایل مشخص می‌شود پروژه چه پکیج‌هایی نیاز دارد و Composer چطور باید autoloadرا انجام دهد. مثل یک لیست خرید برای dependency هاست.

---

۳ .چرا نام پکیج در `composer.json`باید با `^[a-z0-9-]+$` regex هماهنگ باشد؟

جواب: چون Composer از یک استاندارد جهانی برای نام‌گذاری پکیج‌ها استفاده می‌کند (حروف کوچک، عدد و خط تیره) تا تداخل ایجاد نشود.

---

۴ .پوشه `vendor`چیست؟

جواب: محلی است که Composer تمام پکیج‌های نصب‌شده و فایل autoload را در آن قرار می‌دهد. این پوشه نباید دستی تغییر داده شود.

---

۵ .آیا پوشه `vendor`بلافاصله بعد از `composer init`ساخته می‌شود؟

جواب: خیر، فقط وقتی ساخته می‌شود که یا autoload تعریف شود یا دستورات `composer install`یا `composer dump-autoload`اجرا شوند.

---

۶ .فایل `.gitignore`که Composer می‌سازد چه نقشی دارد؟

جواب: باعث می‌شود پوشه vendor و فایل‌های بی‌استفاده وارد مخزن گیت نشوند چون قابل بازتولید هستند.

---

۷. دستور composer install چه کاری انجام می‌دهد؟

جواب: پکیج‌هایی که در composer.json تعریف شده‌اند را نصب می‌کند و نتیجه را در فایل composer.lock ذخیره می‌کند.

---

۸. فرق composer.json و composer.lock چیست؟

جواب composer.json: لیست کلی پکیج‌ها را نگه می‌دارد، ولی composer.lock نسخه دقیق هر پکیج نصب‌شده را ذخیره می‌کند تا در سیستم‌های دیگر دقیقاً همان نسخه‌ها نصب شوند.

---

۹. اگر composer.json تغییر کند چه باید کرد؟

جواب: باید دستور composer install یا composer update را زد تا تغییرات اعمال و lock آپدیت شود.

---

۱۰. دستور composer dump-autoload چه کاری می‌کند؟

جواب: فایل autoload را دوباره تولید می‌کند. مخصوصاً وقتی namespace یا مسیر کلاس‌ها تغییر کرده باشد.

---

۱۱. فایل autoload_psr4.php کجا قرار دارد؟

جواب: در مسیر vendor/composer/ ساخته می‌شود و در آن mapping بین namespaceو مسیر پوشه‌ها نگه‌داری می‌شود.

---

۱۲ .اگر namespace در کلاس با مسیر پوشه اشتباه باشد چه می‌شود؟

جواب Composer :نمی‌تواند کلاس را پیدا کند و خطای Class not foundمی‌دهد.

---

۱۳ .چرا باید require vendor/autoload.phpدر پروژه اضافه شود؟

جواب: چون این فایل مسئول بارگذاری همه کلاس‌ها و پکیج‌هاست. بدون آن Composer کاری انجام نمی‌دهد.

---

۱۴ .autoload در Composer چه مزیتی نسبت به requireدستی دارد؟

جواب: باعث می‌شود نیاز نباشد مسیر تک‌تک فایل‌ها را بنویسیم. فقط یکبار autoload را صدا می‌زنیم و بقیه خودکار بارگذاری می‌شود.

---

۱۵ .PSR-4 چیست؟

جواب: یک استاندارد برای mapping بین namespace و مسیر پوشه‌ها است که توسط Composerبرای autoload استفاده می‌شود.

---

۱۶ .چرا استفاده از namespace در پروژه‌های بزرگ ضروری است؟

جواب: چون از تداخل نام کلاس‌ها جلوگیری می‌کند و ساختار پروژه را منظم‌تر می‌کند.

---

۱۷. اگر پوشه vendor پاک شود چه باید کرد؟

جواب: کافی است دوباره `composer install` را اجرا کنیم تا بر اساس `composer.lock` همان پکیج‌ها دوباره نصب شوند.

---

۱۸. چرا vendor نباید داخل گیت آپلود شود؟

جواب: چون حجم زیادی دارد و قابل بازتولید است. تنها `composer.json` و `composer.lock` باید در مخزن باشند.

---

۱۹. اگر نسخه PHP با پکیج‌ها سازگار نباشد چه اتفاقی می‌افتد؟

جواب Composer: خطای dependency می‌دهد و اجازه نصب نمی‌دهد تا نسخه سازگار انتخاب شود.

---

۲۰. فرق `composer update` و `composer install` چیست؟

جواب:

- `install`: بر اساس نسخه‌های موجود در `composer.lock` نصب می‌کند.
- `update`: جدیدترین نسخه‌های سازگار با `composer.json` را نصب و lock را بازنویسی می‌کند.

---

۲۱. فایل `composer.phar` چیست؟

جواب: نسخه قابل حمل Composer است. می‌توان آن را در هر مسیر کپی و اجرا کرد.

---

۲۲ .چرا composer.phar معمولاً در `/usr/local/bin` قرار می‌گیرد؟

جواب: چون آن مسیر داخل PATH است و می‌توان از هرجای ترمینال دستور `composer` را اجرا کرد.

---

۲۳ .autoload فقط مخصوص کلاس‌هاست؟

جواب: خیر، می‌تواند برای فایل‌های ساده PHP هم استفاده شود (با بخش `files`در composer.json).

---

۲۴ .چرا وقتی `composer install`می‌زنیم پیام "Nothing to install" می‌آید؟

جواب: چون در `composer.json`هیچ پکیجی تعریف نشده است.

---

۲۵ .وقتی پکیج جدیدی نصب کنیم چه اتفاقی می‌افتد؟

جواب Composer :آن پکیج را داخل `vendor/`می‌ریزد، در `composer.json`اضافه می‌کند و در `composer.lock`نسخه دقیقش را ثبت می‌کند.

---

۲۶ .آیا می‌توانیم autoload را بدون Composer انجام دهیم؟

جواب: بله، با `spl_autoload_register`می‌شود، ولی Composer کار را راحت‌تر و استاندارد می‌کند.

---

۲۷ .چرا Composer محبوب‌ترین dependency manager در PHP است؟

جواب: چون هم dependency ها را مدیریت می‌کند و هم autoloading را پشتیبانی می‌کند.

---

۲۸ .اگر autoload درست کار نکند چه بررسی‌هایی باید کرد؟

جواب: چک کنیم namespace درست باشد، پوشه درست در composer.json ثبت شده باشد و `composer dump-autoload`اجرا شده باشد.

---

۲۹ .آیا فایل `composer.lock`باید داخل گیت قرار بگیرد؟

جواب: بله، چون تضمین می‌کند همه توسعه‌دهندگان یک نسخه دقیق از پکیج‌ها را داشته باشند.

---

۳۰ .چطور می‌توانیم بررسی کنیم autoload درست کار می‌کند؟

جواب: کافی است یک کلاس با namespace درست بسازیم، فقط یکبار `require` `vendor/autoload.php`را اضافه کنیم و ببینیم بدون `require`دستی کار می‌کند.

📌۵ تمرین عملی با جواب تشریحی

تمرین ۱

🔹یک پروژه جدید بساز، دستور `composer init`بزن و نام پکیج را `myapp/demo` قرار بده.

🔹چه فایل‌هایی ساخته می‌شود؟

✅جواب:

- composer.jsonساخته می‌شود.
- اگر git فعال باشد .gitignore ساخته می‌شود.
- پوشه vendor ساخته نمی‌شود تا وقتی install یا dump-autoload زده شود.

تمرین ۲

🔻در composer.jsonبخش autoload را اینطور اضافه کن:

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

🔻پوشه src/بساز و یک فایل Test.phpبا کلاس App\Testایجاد کن.

🔻با composer dump-autoloadتست کن.

✅جواب:

وقتی در فایل اصلی بنویسیم:

```
require 'vendor/autoload.php';
use App\Test;
$test = new Test();
```

کلاس بدون require دستی لود می‌شود.

تمرین ۳

🔻پوشه vendor را پاک کن و دوباره composer installبزن.

🔻چه چیزی اتفاق می‌افتد؟

✅جواب:
پوشه vendor دوباره ساخته می‌شود و تمام dependency ها براساس `composer.lock` نصب می‌شوند.

تمرین ۴

🔷یک پکیج مثل `nesbot/carbon`را نصب کن:

```
composer require nesbot/carbon
```

🔷چه فایل‌هایی تغییر می‌کنند؟

✅جواب:

- پکیج در vendor نصب می‌شود.
- در composer.json به بخش require اضافه می‌شود.
- composer.lockبروزرسانی می‌شود.

تمرین ۵

🔷فرض کن namespace اشتباه تعریف کردی:

```
"autoload": {
  "psr-4": {
    "Core\\": "cores/"
  }
}
```

)درحالی‌که پوشه واقعی `core/`است(

🔷چه اتفاقی می‌افتد؟

✅**جواب:**

Composer**کلاس‌ها را پیدا نمی‌کند و خطای** `Class 'Core\Something' not found` **می‌دهد.**

باید namespace **و مسیر را هماهنگ کنیم و دوباره** `composer dump-autoload` **بزنیم.**