Тестовый контроль « Основы создания библиотек Composer»

Данный блок вопросов делится на две части: часть касательно базовых возможностей composer и часть по разработке пакетов для их коллективного использования

Часть 1. Основы работы с composer

1-15. Вопросы с выбором

1. **Что такое Composer?**
    a. Система управления базами данных
    b. Инструмент для управления зависимостями в PHP-проектах
    c. Шаблон для написания PHP-библиотек
    d. Редактор кода для PHP

Ответ: b. Инструмент для управления зависимостями в PHP-проектах

2. **Что является основным файлом для конфигурации библиотеки в Composer?**
    a. composer.json
    b. config.php
    c. composer.lock
    d. dependencies.json

Ответ: a. composer.json

3. **Какую команду следует использовать для создания нового проекта с Composer?**

    a. composer init

    b. composer create

    c. composer new-project

    d. composer start

Ответ: a. composer init

4. **Как Composer управляет зависимостями для проекта?**

    a. Зависимости указываются вручную в файле composer.json, затем Composer автоматически загружает их с помощью команды composer install.

    b. Composer скачивает зависимости через систему пакетов.

    c. Composer использует файлы config.json для определения зависимостей.

    d. Зависимости добавляются с помощью команды composer dependencies.

Ответ: a. Зависимости указываются вручную в файле composer.json, затем Composer автоматически загружает их с помощью команды composer install.

5. **Что делает команда composer install?**

    a. Загружает все зависимости, указанные в файле composer.json, и устанавливает их в проект.

    b. Устанавливает Composer в систему.

    c. Создает новый проект и устанавливает зависимости.

    d. Обновляет все установленные зависимости.

Ответ: a. Загружает все зависимости, указанные в файле composer.json, и устанавливает их в проект.

6. **Что такое composer.lock?**

a. Файл, который содержит информацию о версиях всех установленных зависимостей.

b. Файл, в котором Composer сохраняет настройки конфигурации проекта.

c. Файл, в котором указаны доступные команды.

d. Файл для хранения логов ошибок при установке зависимостей.

Ответ: a. Файл, который содержит информацию о версиях всех установленных зависимостей.

7. **Какую команду следует использовать для обновления зависимостей до последних версий в проекте?**

a. composer upgrade

b. composer update

c. composer refresh

d. composer change

Ответ: b. composer update

8. **Как указать зависимость для библиотеки в файле composer.json?**

a. Добавить запись в раздел dependencies

b. Добавить запись в раздел libs

c. Указать в разделе require

d. Указать в разделе packages

Ответ: c. Указать в разделе require

9. **Как Composer проверяет совместимость версий зависимостей?**

a. Использует специальные версии и диапазоны версий, указанные в composer.json

b. Все зависимости всегда устанавливаются в последней версии

c. Совместимость проверяется только при первой установке

d. Проверку совместимости проводит командная строка

Ответ: a. Использует специальные версии и диапазоны версий, указанные в composer.json

10. **Как удалить зависимость из проекта с помощью Composer?**
    a. composer remove <package-name>
    b. composer uninstall <package-name>
    c. composer delete <package-name>
    d. composer unlink <package-name>

Ответ: a. composer remove <package-name>

11. **Что такое автозагрузка в Composer?**
    a. Метод, который автоматически загружает файлы классов PHP по мере их необходимости.
    b. Утилита, которая автоматически обновляет зависимости при их установке.
    c. Система, которая подключает базу данных в проект.
    d. Механизм, который автоматически генерирует новые версии для библиотек.

Ответ: a. Метод, который автоматически загружает файлы классов PHP по мере их необходимости.

12. **Каким образом можно подключить автозагрузку классов в проекте с использованием Composer?**
    a. Добавить секцию autoload в composer.json
    b. Установить специальный плагин для автозагрузки
    c. Использовать глобальный файл конфигурации
    d. Включить автозагрузку через настройки сервера

Ответ: a. Добавить секцию autoload в composer.json

13. **Что такое composer.json?**

a. Это файл, который используется для настройки сервера веб-приложения.

b. Это конфигурационный файл, в котором описываются зависимости и метаданные проекта.

c. Это файл для установки PHP-расширений.

d. Это лог-файл, в который Composer записывает установленные пакеты.

Ответ: b. Это конфигурационный файл, в котором описываются зависимости и метаданные проекта.

14. **Как Composer может работать с различными версиями PHP?**

a. Используя файл composer.json для указания минимальной версии PHP

b. Через команду composer switch-php-version

c. Composer не поддерживает работу с несколькими версиями PHP

d. Используя команду composer set-php-version

Ответ: a. Используя файл composer.json для указания минимальной версии PHP

15. **Какая из следующих команд используется для создания библиотеки в Composer?**

a. composer library-create

b. composer init

c. composer package

d. composer start-library

Ответ: b. composer init

16-25. Открытые вопросы

16. **Объясните, что такое файл composer.json и какие ключевые разделы он должен содержать для библиотеки.**

Открытый ответ:

16. **Файл** composer.json:

- Это основной файл конфигурации для проекта, использующего Composer. Он содержит информацию о проекте и его зависимостях.
- Ключевые разделы для библиотеки:
  - name: Имя библиотеки в формате vendor/package.
  - description: Краткое описание библиотеки.
  - require: Список зависимостей, необходимых для работы библиотеки.
  - autoload: Настройки автозагрузки классов.
  - version: Версия библиотеки.
  - authors: Информация об авторах библиотеки.
  - license: Лицензия библиотеки.

17. **Как вы можете добавить дополнительный репозиторий для загрузки библиотек в вашем проекте с использованием Composer?**

Открытый ответ:

17. **Добавление дополнительного репозитория**:

- В файле composer.json можно добавить раздел repositories, чтобы указать дополнительные репозитории для загрузки библиотек.
- Пример:
- {
- "repositories": [
- {
- "type": "vcs",
- "url": "https://github.com/vendor/package"
- }
-]

}

18. **Опишите процесс создания и настройки автозагрузки классов с использованием Composer.**

Открытый ответ:

18. **Создание и настройка автозагрузки классов**:

- В файле composer.json добавляется раздел autoload.
- Пример:
- {
- "autoload": {
- "psr-4": {
- "Vendor\\Namespace\\": "src/"
- }
- }

}

- После этого выполняется команда composer dump-autoload, чтобы сгенерировать файл автозагрузки.

19. **Как можно ограничить версии зависимостей для библиотеки в файле composer.json? Приведите пример записи для зависимости с версией выше 1.2.3 и ниже 2.0.0.**

Открытый ответ:

19. **Ограничение версий зависимостей**:

- В разделе require можно указать ограничения на версии зависимостей.
- Пример для версии выше 1.2.3 и ниже 2.0.0:
- {
- "require": {
- "vendor/package": ">1.2.3 <2.0.0"
- }

}

20. Что такое командная строка Composer и как она используется для управления зависимостями в проекте?

Открытый ответ:

20. **Командная строка Composer**:

- Composer предоставляет командную строку для управления зависимостями.
- Основные команды:
    - composer install: Устанавливает зависимости, указанные в composer.lock.
    - composer update: Обновляет зависимости до последних версий, соответствующих указанным ограничениям.
    - composer require: Добавляет новую зависимость.

21. Как в Composer можно настроить использование специфических версий PHP для различных проектов?

Открытый ответ:

21. **Настройка специфических версий PHP**:

- В файле composer.json можно указать требуемую версию PHP в разделе require.
- Пример:
- {
- "require": {
- "php": "^7.4"
- }

}

22. Что происходит, если зависимости не совместимы между собой при попытке установить их с помощью Composer?

Открытый ответ:

22. **Несовместимые зависимости**:

- Если зависимости не совместимы между собой, Composer выдаст ошибку и не установит их.
- Необходимо разрешить конфликт вручную, изменив версии зависимостей.

23. **Как обновить все зависимости проекта, которые были указаны в composer.json, до последних доступных версий?**

Открытый ответ:

23. **Обновление всех зависимостей**:

- Команда composer update обновляет все зависимости до последних доступных версий, соответствующих указанным ограничениям.

24. **Каким образом Composer решает проблему с конфликтующими зависимостями, когда разные библиотеки требуют разные версии одной и той же зависимости?**

Открытый ответ:

24. **Решение проблемы с конфликтующими зависимостями**:

- Composer использует алгоритм разрешения зависимостей, чтобы найти совместимые версии.
- Если конфликт не может быть разрешен, Composer выдаст ошибку.

25. **Как бы вы создали библиотеку с использованием Composer и подключили её к вашему проекту? Опишите пошагово процесс.**

Открытый ответ:

25. **Создание и подключение библиотеки**:

- **Шаг 1**: Создайте структуру проекта библиотеки.
- **Шаг 2**: Создайте файл composer.json с необходимыми разделами (name, description, require, autoload и т.д.).
- **Шаг 3**: Выполните команду composer install для установки зависимостей.
- **Шаг 4**: В основном проекте добавьте библиотеку в раздел require файла composer.json.

- **Шаг 5**: Выполните команду composer update для установки библиотеки в основной проект.

Вопросы на сопоставление

26.**Сопоставьте команды Composer с их действиями.**

Команды:

1. composer init
2. composer install
3. composer update
4. composer remove <package-name>
5. composer dump-autoload

Действия:

A) Устанавливает все зависимости, указанные в composer.json, и генерирует файл composer.lock.
B) Удаляет указанную зависимость из проекта.
C) Инициализирует новый проект и создает файл composer.json.
D) Обновляет все зависимости проекта до последних доступных версий.

E) Перегенерирует файлы автозагрузки.

Ответ:

Сопоставим команды Composer с их действиями:

1. **composer init** - C) Инициализирует новый проект и создает файл composer.json.
2. **composer install** - A) Устанавливает все зависимости, указанные в composer.json, и генерирует файл composer.lock.
3. **composer update** - D) Обновляет все зависимости проекта до последних доступных версий.
4. **composer remove <package-name>** - B) Удаляет указанную зависимость из проекта.
5. **composer dump-autoload** - E) Перегенерирует файлы автозагрузки.

Таким образом, правильные сопоставления:

- **1-C**

- **2-A**
- **3-D**
- **4-B**
- **5-E**

Часть 2. Размещение пакета в репозитории пакетов Composer и Команды Git для добавления пакетов

Вопросы с выбором

1. **Что такое Packagist?**
    a. Это публичный репозиторий пакетов Composer
    b. Это локальный сервер для установки зависимостей в проекте
    c. Это инструмент для создания пакетов Composer
    d. Это система для версионирования зависимостей

Ответ: a. Это публичный репозиторий пакетов Composer

2. **Как добавить свой пакет в репозиторий Packagist?**
    a. Зарегистрироваться на сайте Packagist и добавить URL своего репозитория
    b. Использовать команду composer publish
    c. Отправить пакет через GitHub
    d. Добавить пакет вручную в файл composer.json

Ответ: a. Зарегистрироваться на сайте Packagist и добавить URL своего репозитория

3. **Какие шаги нужно предпринять, чтобы ваш пакет стал доступен в репозитории Packagist?**
    a. Создать репозиторий на GitHub или другом хостинге, затем подключить его к Packagist
    b. Добавить ссылку на ваш пакет в composer.json и выполнить команду composer update

c. Отправить архив с пакетом в Packagist

d. Публиковать новый пакет с новой версией

Ответ: a. Создать репозиторий на GitHub или другом хостинге, затем подключить его к Packagist

4. **Какой тип репозитория обычно используется для размещения пакетов Composer?**

a. Git репозиторий (например, GitHub)

b. FTP-сервер

c. Локальный сервер

d. Платформа для управления проектами

Ответ: a. Git репозиторий (например, GitHub)

5. **Что необходимо указать в файле composer.json для правильной публикации пакета?**

a. Только имя и описание пакета

b. Название, версия и автозагрузку классов

c. Лицензию и права доступа

d. Все поля name, description, version, autoload, license

Ответ: d. Все поля name, description, version, autoload, license

6. **Как обновить версию пакета в репозитории Packagist?**

a. Выпустить новый тег в вашем репозитории Git и отправить изменения на Packagist

b. Публиковать новый пакет с новой версией

c. Изменить версию в файле composer.json и выполнить команду composer update

d. Выполнить команду composer publish с новой версией

Ответ: a. Выпустить новый тег в вашем репозитории Git и отправить изменения на Packagist

7. **Как добавить частный репозиторий в Composer для размещения пакетов, не публикуемых в Packagist?**
    a. Использовать параметр repositories в файле composer.json
    b. Добавить ссылку на репозиторий в настройки сервера
    c. Указать URL в командной строке при установке зависимостей
    d. Создать новую ветку в репозитории Packagist

Ответ: a. Использовать параметр repositories в файле composer.json

8. **Что происходит, если вы добавляете пакет в репозиторий Packagist без указания тегов Git?**
    a. Пакет не будет доступен для установки
    b. Packagist автоматически добавит тег для пакета
    c. Пакет будет установлен с последней доступной версией
    d. Пакет будет доступен только для разработки

Ответ: a. Пакет не будет доступен для установки

9. **Какие метаданные можно указать в файле composer.json при размещении пакета?**
    a. Только название и описание
    b. Название, описание, версия, авторы, лицензия, зависимости, автозагрузка
    c. Только описание и автозагрузка
    d. Название, версия и зависимости

Ответ: b. Название, описание, версия, авторы, лицензия, зависимости, автозагрузка

10. **Какую команду нужно выполнить, чтобы проверить, правильно ли настроен пакет перед публикацией на Packagist?**
    a. composer validate

b. composer check

c. composer publish

d. composer install

Ответ: a. composer validate

Вопросы по командам Git для добавления пакетов и публикации на Packagist

12. **Какая команда Git используется для создания нового репозитория на GitHub (или другом хостинге)?**

a. git create

b. git init

c. git push origin

d. git clone

Ответ: b. git init

13. **Какая команда Git используется для добавления всех изменений в текущем проекте перед коммитом?**

a. git add .

b. git commit -a

c. git push

d. git clone

Ответ: a. git add .

13. **Какой команды Git используется для отправки изменений на удалённый репозиторий?**

a. git pull

b. git push

c. git commit

d. git merge

Ответ: b. git push

14. **Как создать новый тег в Git для версии пакета?**

    a. git tag -a v1.0.0
    b. git commit -m "v1.0.0"
    c. git push v1.0.0
    d. git init v1.0.0

Ответ: a. git tag -a v1.0.0

15. **Как отправить новый тег на удалённый репозиторий после его создания?**

    a. git push origin v1.0.0
    b. git push --tags
    c. git commit --tags
    d. git merge origin v1.0.0

Ответ: a. git push origin v1.0.0

16. **Как проверить список всех тегов в репозитории?**

    a. git tags
    b. git show
    c. git tag
    d. git list-tags

Ответ: c. git tag

17. **Как удалить тег в Git?**

    a. git delete <tag>
    b. git tag -d <tag-name>
    c. git remove-tag <tag-name>
    d. git push --delete <tag-name>

Ответ: b. git tag -d <tag-name>

18. **Как синхронизировать локальные изменения с репозиторием, обновив его содержимое?**

    a. git fetch

    b. git update

    c. git sync

    d. git pull

Ответ: d. git pull

19. **Какая команда используется для создания новой ветки в Git?**

    a. git branch <branch-name>

    b. git create-branch <branch-name>

    c. git checkout -b <branch-name>

    d. git new-branch <branch-name>

Ответ: c. git checkout -b <branch-name>

20. **Как перейти на ветку в Git, в которой будет опубликован новый пакет?**

    a. git switch <branch-name>

    b. git checkout <branch-name>

    c. git pull <branch-name>

    d. git push <branch-name>

Ответ: b. git checkout <branch-name>

Открытые вопросы

21. **Как зарегистрировать свой пакет на Packagist и какие шаги нужно выполнить для его размещения?**

Открытый ответ:

21. **Как зарегистрировать свой пакет на Packagist и какие шаги нужно выполнить для его размещения?**

- Чтобы зарегистрировать пакет на Packagist, необходимо выполнить следующие шаги:
    1. Создать учётную запись на Packagist.
    2. Создать репозиторий на GitHub (или другом сервисе) и добавить в него файл composer.json.
    3. Убедиться, что в репозитории есть теги версий (например, v1.0.0).
    4. Перейти на Packagist, нажать "Submit" и ввести URL вашего репозитория.
    5. Packagist автоматически проверит репозиторий и добавит его в свой список пакетов.

22. **Как настроить репозиторий на GitHub (или другом сервисе) для корректной работы с Packagist?**

Открытый ответ:

22. **Как настроить репозиторий на GitHub (или другом сервисе) для корректной работы с Packagist?**

- Для корректной работы с Packagist необходимо:
    1. Убедиться, что репозиторий публичный (или настроить доступ для Packagist, если он частный).
    2. Включить в репозиторий файл composer.json с корректными данными.
    3. Создать теги версий для релизов (например, v1.0.0).

23. **Какие требования предъявляются к версии пакета, чтобы он корректно отображался в репозитории Packagist?**

Открытый ответ:

23. **Какие требования предъявляются к версии пакета, чтобы он корректно отображался в репозитории Packagist?**

- Версия пакета должна соответствовать формату семантического версионирования (например, 1.0.0).
- Версия должна быть указана в теге Git (например, v1.0.0).

24. **Как обновить пакет на Packagist после того, как новая версия была выпущена в вашем репозитории? Опишите весь процесс.**

Открытый ответ:

24. **Как обновить пакет на Packagist после того, как новая версия была выпущена в вашем репозитории? Опишите весь процесс.**

- Для обновления пакета на Packagist:
    1. Создайте новый тег версии в репозитории (например, v1.1.0).
    2. Packagist автоматически обнаружит новый тег и обновит информацию о пакете.

25. **Как добавить свой собственный частный репозиторий пакетов в проект с помощью Composer? Какие параметры для этого необходимо указать в composer.json?**

Открытый ответ:

25. **Как добавить свой собственный частный репозиторий пакетов в проект с помощью Composer? Какие параметры для этого необходимо указать в composer.json?**

- Чтобы добавить частный репозиторий, в composer.json нужно указать:
- {
- "repositories": [
- {
- "type": "vcs",
- "url": "https://github.com/username/repository"
- }
-]

}

- Если репозиторий частный, можно добавить параметры аутентификации в auth.json.

26. **Какие проблемы могут возникнуть, если в репозитории пакета отсутствуют теги Git, и как это можно исправить?**

Открытый ответ:

26. **Какие проблемы могут возникнуть, если в репозитории пакета отсутствуют теги Git, и как это можно исправить?**

- Без тегов Git Packagist не сможет определить версии пакета.
- Для исправления нужно создать теги версий (например, git tag v1.0.0 и git push origin v1.0.0).

27. **Какие поля должны быть обязательно указаны в файле composer.json, чтобы пакет корректно отображался на Packagist?**

Открытый ответ:

27. **Какие поля должны быть обязательно указаны в файле composer.json, чтобы пакет корректно отображался на Packagist?**

- Обязательные поля: name, description, version, type, license, authors.

28. **Что нужно учитывать при управлении зависимостями для пакетов, размещённых на частных репозиториях в Composer?**

Открытый ответ:

28. **Что нужно учитывать при управлении зависимостями для пакетов, размещённых на частных репозиториях в Composer?**

- Убедитесь, что у вас есть доступ к частному репозиторию.
- Настройте аутентификацию в auth.json для доступа к частным репозиториям.

29. **Опишите процесс публикации нового пакета в репозитории Composer, начиная с его создания до публикации на Packagist.**

Открытый ответ:

29. **Опишите процесс публикации нового пакета в репозитории Composer, начиная с его создания до публикации на Packagist.**

- Создайте репозиторий на GitHub и добавьте composer.json.

- Создайте теги версий.
- Зарегистрируйтесь на Packagist и добавьте URL вашего репозитория.
- Packagist автоматически добавит ваш пакет в свой список.

30. **Как Composer взаимодействует с другими репозиториями, если пакет не был опубликован на Packagist? Как указать дополнительные репозитории для установки зависимостей?**

Открытый ответ:

30. **Как Composer взаимодействует с другими репозиториями, если пакет не был опубликован на Packagist? Как указать дополнительные репозитории для установки зависимостей?**

- Composer может взаимодействовать с другими репозиториями, указав их в composer.json:
- {
- "repositories": [
- {
- "type": "vcs",
- "url": "https://github.com/username/repository"
- }
-]

}

- Это позволяет Composer находить и устанавливать пакеты из указанных репозиториев.

Вопросы на сопоставление

Сопоставьте шаги с действиями при публикации пакета в Packagist.

Шаги:

1. Создание репозитория на GitHub
2. Создание файла composer.json
3. Отправка изменений в Packagist

4. Добавление тега Git
5. Проверка валидности пакета

Действия:

A) Нужно запустить команду composer validate для проверки правильности структуры.
B) Указываются метаданные пакета, включая название, описание, версия и зависимости.
C) После создания репозитория необходимо опубликовать его на Packagist.
D) С помощью тега Git Composer может определить версию пакета.

E) Пакет должен быть размещён в публичном репозитории, таком как GitHub.

Ответ:

Для публикации пакета в Packagist, шаги и действия сопоставляются следующим образом:

1. **Создание репозитория на GitHub** - E) Пакет должен быть размещён в публичном репозитории, таком как GitHub.
2. **Создание файла composer.json** - B) Указываются метаданные пакета, включая название, описание, версия и зависимости.
3. **Отправка изменений в Packagist** - C) После создания репозитория необходимо опубликовать его на Packagist.
4. **Добавление тега Git** - D) С помощью тега Git Composer может определить версию пакета.
5. **Проверка валидности пакета** - A) Нужно запустить команду composer validate для проверки правильности структуры.

Таким образом, правильные сопоставления: 1-E, 2-B, 3-C, 4-D, 5-A.