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

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.
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.
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.
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
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

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

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>

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.

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

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.

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

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

16-25. Открытые вопросы

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

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

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

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.**

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

21. **Как в Composer можно настроить использование специфических версий PHP для различных проектов?**
22. **Что происходит, если зависимости не совместимы между собой при попытке установить их с помощью Composer?**
23. **Как обновить все зависимости проекта, которые были указаны в composer.json, до последних доступных версий?**
24. **Каким образом Composer решает проблему с конфликтующими зависимостями, когда разные библиотеки требуют разные версии одной и той же зависимости?**
25. **Как бы вы создали библиотеку с использованием Composer и подключили её к вашему проекту? Опишите пошагово процесс.**

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.

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

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

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

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

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

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

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

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

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

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

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
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
13. **Какой команды Git используется для отправки изменений на удалённый репозиторий?**
    a. git pull
    b. git push
    c. git commit
    d. git merge
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
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
16. **Как проверить список всех тегов в репозитории?**

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

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>
18. **Как синхронизировать локальные изменения с репозиторием, обновив его содержимое?**
    a. git fetch
    b. git update
    c. git sync
    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>
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>

Открытые вопросы

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

22. **Как настроить репозиторий на GitHub (или другом сервисе) для корректной работы с Packagist?**
23. **Какие требования предъявляются к версии пакета, чтобы он корректно отображался в репозитории Packagist?**
24. **Как обновить пакет на Packagist после того, как новая версия была выпущена в вашем репозитории? Опишите весь процесс.**
25. **Как добавить свой собственный частный репозиторий пакетов в проект с помощью Composer? Какие параметры для этого необходимо указать в composer.json?**
26. **Какие проблемы могут возникнуть, если в репозитории пакета отсутствуют теги Git, и как это можно исправить?**
27. **Какие поля должны быть обязательно указаны в файле composer.json, чтобы пакет корректно отображался на Packagist?**
28. **Что нужно учитывать при управлении зависимостями для пакетов, размещённых на частных репозиториях в Composer?**
29. **Опишите процесс публикации нового пакета в репозитории Composer, начиная с его создания до публикации на Packagist.**
30. **Как Composer взаимодействует с другими репозиториями, если пакет не был опубликован на Packagist? Как указать дополнительные репозитории для установки зависимостей?**

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.