**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?**
   1. Система управления базами данных
   2. Инструмент для управления зависимостями в PHP-проектах
   3. Шаблон для написания PHP-библиотек
   4. Редактор кода для PHP

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

1. **Что является основным файлом для конфигурации библиотеки в Composer?**
   1. composer.json
   2. config.php
   3. composer.lock
   4. dependencies.json

**Ответ:** a. composer.json

1. **Какую команду следует использовать для создания нового проекта с Composer?**
   1. composer init
   2. composer create
   3. composer new-project
   4. composer start

**Ответ:** a. composer init

1. **Как Composer управляет зависимостями для проекта?**
   1. Зависимости указываются вручную в файле composer.json, затем Composer автоматически загружает их с помощью команды composer install.
   2. Composer скачивает зависимости через систему пакетов.
   3. Composer использует файлы config.json для определения зависимостей.
   4. 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.

1. **Что делает команда composer install?**
   1. Загружает все зависимости, указанные в файле composer.json, и устанавливает их в проект.
   2. Устанавливает Composer в систему.
   3. Создает новый проект и устанавливает зависимости.
   4. Обновляет все установленные зависимости.

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

1. **Что такое composer.lock?**
   1. Файл, который содержит информацию о версиях всех установленных зависимостей.
   2. Файл, в котором Composer сохраняет настройки конфигурации проекта.
   3. Файл, в котором указаны доступные команды.
   4. Файл для хранения логов ошибок при установке зависимостей.

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

1. **Какую команду следует использовать для обновления зависимостей до последних версий в проекте?**
   1. composer upgrade
   2. composer update
   3. composer refresh
   4. composer change

**Ответ:** b. composer update

1. **Как указать зависимость для библиотеки в файле composer.json?**
   1. Добавить запись в раздел dependencies
   2. Добавить запись в раздел libs
   3. Указать в разделе require
   4. Указать в разделе packages

**Ответ:** c. Указать в разделе require

1. **Как Composer проверяет совместимость версий зависимостей?**
   1. Использует специальные версии и диапазоны версий, указанные в composer.json
   2. Все зависимости всегда устанавливаются в последней версии
   3. Совместимость проверяется только при первой установке
   4. Проверку совместимости проводит командная строка

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

1. **Как удалить зависимость из проекта с помощью Composer?**
   1. composer remove <package-name>
   2. composer uninstall <package-name>
   3. composer delete <package-name>
   4. composer unlink <package-name>

**Ответ:** a. composer remove <package-name>

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

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

1. **Каким образом можно подключить автозагрузку классов в проекте с использованием Composer?**
   1. Добавить секцию autoload в composer.json
   2. Установить специальный плагин для автозагрузки
   3. Использовать глобальный файл конфигурации
   4. Включить автозагрузку через настройки сервера

**Ответ:** a. Добавить секцию autoload в composer.json

1. **Что такое composer.json?**
   1. Это файл, который используется для настройки сервера веб-приложения.
   2. Это конфигурационный файл, в котором описываются зависимости и метаданные проекта.
   3. Это файл для установки PHP-расширений.
   4. Это лог-файл, в который Composer записывает установленные пакеты.

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

1. **Как Composer может работать с различными версиями PHP?**
   1. Используя файл composer.json для указания минимальной версии PHP
   2. Через команду composer switch-php-version
   3. Composer не поддерживает работу с несколькими версиями PHP
   4. Используя команду composer set-php-version

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

1. **Какая из следующих команд используется для создания библиотеки в Composer?**
   1. composer library-create
   2. composer init
   3. composer package
   4. composer start-library

**Ответ:** b. composer init

**16-25. Открытые вопросы**

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

**Открытый ответ:**

1. **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.

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

**Открытый ответ:**

1. **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"
* }
* ]

}

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

**Открытый ответ:**

1. **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.

1. **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:**

1. **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"
* }

}

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

**Открытый ответ:**

1. **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.

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

**Открытый ответ:**

1. **Настройка специфических версий PHP**:

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

}

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

**Открытый ответ:**

1. **Несовместимые зависимости**:

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

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

**Открытый ответ:**

1. **Обновление всех зависимостей**:

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

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

**Открытый ответ:**

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

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

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

**Открытый ответ:**

1. **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**

1. **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?**
   1. Это публичный репозиторий пакетов Composer
   2. Это локальный сервер для установки зависимостей в проекте
   3. Это инструмент для создания пакетов Composer
   4. Это система для версионирования зависимостей

**Ответ:** a. Это публичный репозиторий пакетов Composer

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

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

1. **Какие шаги нужно предпринять, чтобы ваш пакет стал доступен в репозитории Packagist?**
   1. Создать репозиторий на GitHub или другом хостинге, затем подключить его к Packagist
   2. Добавить ссылку на ваш пакет в composer.json и выполнить команду composer update
   3. Отправить архив с пакетом в Packagist
   4. Публиковать новый пакет с новой версией

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

1. **Какой тип репозитория обычно используется для размещения пакетов Composer?**
   1. Git репозиторий (например, GitHub)
   2. FTP-сервер
   3. Локальный сервер
   4. Платформа для управления проектами

**Ответ:** a. Git репозиторий (например, GitHub)

1. **Что необходимо указать в файле composer.json для правильной публикации пакета?**
   1. Только имя и описание пакета
   2. Название, версия и автозагрузку классов
   3. Лицензию и права доступа
   4. Все поля name, description, version, autoload, license

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

1. **Как обновить версию пакета в репозитории Packagist?**
   1. Выпустить новый тег в вашем репозитории Git и отправить изменения на Packagist
   2. Публиковать новый пакет с новой версией
   3. Изменить версию в файле composer.json и выполнить команду composer update
   4. Выполнить команду composer publish с новой версией

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

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

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

1. **Что происходит, если вы добавляете пакет в репозиторий Packagist без указания тегов Git?**
   1. Пакет не будет доступен для установки
   2. Packagist автоматически добавит тег для пакета
   3. Пакет будет установлен с последней доступной версией
   4. Пакет будет доступен только для разработки

**Ответ:** a. Пакет не будет доступен для установки

1. **Какие метаданные можно указать в файле composer.json при размещении пакета?**
   1. Только название и описание
   2. Название, описание, версия, авторы, лицензия, зависимости, автозагрузка
   3. Только описание и автозагрузка
   4. Название, версия и зависимости

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

1. **Какую команду нужно выполнить, чтобы проверить, правильно ли настроен пакет перед публикацией на Packagist?**
   1. composer validate
   2. composer check
   3. composer publish
   4. composer install

**Ответ:** a. composer validate

**Вопросы по командам Git для добавления пакетов и публикации на Packagist**

1. **Какая команда Git используется для создания нового репозитория на GitHub (или другом хостинге)?**
   1. git create
   2. git init
   3. git push origin
   4. git clone

**Ответ:** b. git init

1. **Какая команда Git используется для добавления всех изменений в текущем проекте перед коммитом?**
   1. git add .
   2. git commit -a
   3. git push
   4. git clone

**Ответ:** a. git add .

1. **Какой команды Git используется для отправки изменений на удалённый репозиторий?**
   1. git pull
   2. git push
   3. git commit
   4. git merge

**Ответ:** b. git push

1. **Как создать новый тег в Git для версии пакета?**
   1. git tag -a v1.0.0
   2. git commit -m "v1.0.0"
   3. git push v1.0.0
   4. git init v1.0.0

**Ответ:** a. git tag -a v1.0.0

1. **Как отправить новый тег на удалённый репозиторий после его создания?**
   1. git push origin v1.0.0
   2. git push --tags
   3. git commit --tags
   4. git merge origin v1.0.0

**Ответ:** a. git push origin v1.0.0

1. **Как проверить список всех тегов в репозитории?**
   1. git tags
   2. git show
   3. git tag
   4. git list-tags

**Ответ:** c. git tag

1. **Как удалить тег в Git?**
   1. git delete <tag>
   2. git tag -d <tag-name>
   3. git remove-tag <tag-name>
   4. git push --delete <tag-name>

**Ответ:** b. git tag -d <tag-name>

1. **Как синхронизировать локальные изменения с репозиторием, обновив его содержимое?**
   1. git fetch
   2. git update
   3. git sync
   4. git pull

**Ответ:** d. git pull

1. **Какая команда используется для создания новой ветки в Git?**
   1. git branch <branch-name>
   2. git create-branch <branch-name>
   3. git checkout -b <branch-name>
   4. git new-branch <branch-name>

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

1. **Как перейти на ветку в Git, в которой будет опубликован новый пакет?**
   1. git switch <branch-name>
   2. git checkout <branch-name>
   3. git pull <branch-name>
   4. git push <branch-name>

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

**Открытые вопросы**

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

**Открытый ответ:**

1. **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.

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

**Открытый ответ:**

1. **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).

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

**Открытый ответ:**

1. **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).

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

**Открытый ответ:**

1. **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.

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

**Открытый ответ:**

1. **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.

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

**Открытый ответ:**

1. **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).

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

**Открытый ответ:**

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

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

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

**Открытый ответ:**

1. **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.

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

**Открытый ответ:**

1. **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.

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

**Открытый ответ:**

1. **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.**