Метаклассы

Задание 1. Метакласс для ограничения наследования

Создайте метакласс FinalMeta, который запретит наследование от классов, использующих этот метакласс. Это может использоваться, когда у вас есть класс, который должен оставаться финальным и не наследоваться далее.

Условия:

- Поднятие исключения при попытке создания класса-наследника.
- Поддержка пользовательских сообщений для исключения, которые объясняют причину запрета.
- Возможность указать список исключений, от которых можно наследоваться.

Задание 2. Автоматическое создание сериализуемых классов

Напишите метакласс SerializableMeta, который будет добавлять к классу методы сериализации и десериализации (to_dict и from_dict). Метакласс должен автоматически определять все атрибуты класса и включать их в результат работы метода to_dict. Это полезно для работы с API и базами данных.

Условия:

- Атрибуты должны сериализоваться рекурсивно, если они содержат экземпляры других классов.
- Метод from_dict должен восстанавливать экземпляр класса из словаря.
- Исключение для private-атрибутов (начинающихся с _).

Задание 3. Создание иерархии проверок и ограничений с метаклассами

Разработайте метакласс ValidatedMeta, который проверяет наличие обязательных методов и атрибутов в классе, наследующем от него, и поднимает исключение, если чего-то не хватает. Такой метакласс может использоваться для создания иерархии интерфейсов с обязательными методами, необходимыми для правильной работы класса.

Условия:

- Проверка на наличие атрибутов, заданных в параметре required_attributes метакласса.
- Если у класса отсутствует метод, указанный в параметре required_methods, поднимается исключение.
- Использовать кастомные исключения, описывающие недостающие атрибуты или методы.

Singleton

Задание 1. Настройки конфигурации приложения

- Разработайте класс Config, который хранит глобальные настройки приложения. Он должен быть Singleton.
- Пусть Config поддерживает обновление настроек на основе JSON файла и загрузку по запросу.
- Добавьте возможность хранить различные профили конфигураций, например, для разработки и продакшена.
- Реализуйте методы для безопасного обновления значений конфигурации, а также блокировки изменений в режиме "только для чтения».

Задание 2. Система управления доступом с единой точкой входа

- Создайте класс AccessControl, который будет хранить права доступа пользователей на уровне приложения.

- Используйте Singleton для хранения всех данных о правах доступа, чтобы избежать дублирования прав для пользователей и ролей.
- Реализуйте возможность изменения прав доступа и уведомление об изменениях для активных сессий.
- Настройте проверку прав пользователя при выполнении действий, таких как доступ к данным, внесение изменений и пр.

Задание 3. Кэширование данных в распределенной системе

- Разработайте Singleton класс Cache, который управляет кэшем данных.
- Реализуйте кэширование данных с функциями get, set, и delete, а также управление временем жизни кэша.
- Добавьте возможность записи данных в локальную память.
- Настройте автоматическую синхронизацию кэша между несколькими экземплярами приложения, чтобы данные оставались актуальными.