Добрый день, можно ли поподробнее рассказать про наследование с повторяющимся именем

# Define a new FlatPageAdmin class FlatPageAdmin(FlatPageAdmin):

fieldsets = ( (None, {'fields': ('url', 'title', 'content', 'sites')}), (\_('Advanced options'), { 'classes': ('collapse',), 'fields': ( 'enable\_comments', 'registration\_required', 'template\_name', ), }), )

3 комментария

7 апреля 2023 г.

Анатолий Семенчук

14:03

Добрый день.

В данном примере используется наследование с повторяющимся именем, которое означает, что вы создаете новый класс FlatPageAdmin, который наследует свойства и методы из родительского класса FlatPageAdmin. Здесь родительский FlatPageAdmin является стандартным классом Django, который предоставляет административный интерфейс для модели FlatPage

При наследовании с повторяющимся именем, вы можете изменять или расширять функциональность родительского класса. В данном случае, вы изменяете свойство fieldsets, которое определяет группы полей, отображаемых на странице административного интерфейса для модели FlatPage

9 апреля 2023 г.

**P**

Peter Kirbasov

18:55

а можно вместо этого просто переприсвоить атрибут класса ?

# Define a new FlatPageAdmin FlatPageAdmin.fieldsets = ( (None, {'fields': ('url', 'title', 'content', 'sites')}), (\_('Advanced options'), { 'classes': ('collapse',), 'fields': ( 'enable\_comments', 'registration\_required', 'template\_name', ), }), )

Тимур Суворов

23:08

Добрый вечер! Извиняюсь за задержку с ответом, не сработало оповещение о новом сообщении. Да, это будет работать, но это неправильно, так как вы меняете атрибут у родительского класса. Это ведет к нарушению принципа Open-Closed Principle (OCP) из SOLID, который утверждает, что классы должны быть открыты для расширения, но закрыты для изменения. Если атрибут родительского класса изменяется напрямую, то это может привести к нежелательным побочным эффектам в других частях программы, которые используют этот класс. Поэтому мы сначала наследуемся, а потом в наследованном классе изменяем атрибут. Тем самым не затрагивая основной класс.