

Пиротехническое шоу

Задание: Написать код который будет моделировать шоу с фейерверками, используя все возможные комбинации форм, цветов и действий. Каждый фейерверк представляет собой уникальную комбинацию этих характеристик. Ваша задача - написать асинхронный код, который имитирует запуск каждого фейерверка и затем выводит сообщение о его завершении.

Вам понадобится три списка, список формы, цвета и действия.

```
shapes = ["circle", "star", "square", "diamond", "heart"]
colors = ["red", "blue", "green", "yellow", "purple"]
actions = ["change_color", "explode", "disappear"]
```

Каждый фейерверк должен сгенерировать два сообщения:

1. Первое сообщение выводится при "запуске" фейерверка. Оно должно содержать информацию о цвете, форме и действии фейерверка в формате: `f"Запущен {color} {shape} салют, в форме {action}!!!"`, где {color}, {shape} и {action} - это одно из значений свойств фейерверка.
2. Второе сообщение выводится по истечении времени полета фейерверка и сообщает о его "завершении". Оно должно быть в формате: `f"Салют {color} {shape} завершил выступление {action}"`, где {color}, {shape} и {action} также берутся из списка свойств фейерверка.

```
# Запуск салюта с комбинированными значениями
Запущен yellow star салют, в форме change_color!!!

# Отчёт о успешном запуске
Салют yellow star завершил выступление change_color
```

Подсказка: Используйте функцию `product()` из модуля `itertools` для создания декартова произведения всех возможных комбинаций форм, цветов и действий всего должно получиться 75 уникальных комбинаций.

```
combinations = list(itertools.product(shapes, colors, actions))
```

Определите асинхронную функцию `launch_firework()`, которая принимает данные о комбинации значений салюта, эта функция должна:

- Выводить сообщение о запуске фейерверка с указанием его цвета, формы и действия.

```
f"Запущен {color} {shape} салют, в форме {action}!!!"
```

- "Замораживать" выполнение функции `launch_firework()` на время полета фейерверка, используя `asyncio.sleep()`, на случайное значение, например от 1 до 5 сек.

```
await asyncio.sleep(random.randint(1, 5))
```

- Выводить сообщение о завершении выступления фейерверка.

```
f"Салют {color} {shape} завершил выступление {action}"
```

Определите асинхронную функцию (точку входа) `main()`, которая создает список задач, используя корутину `launch_firework()` для каждого фейерверка, и ожидает завершения всех задач, используя `asyncio.gather()`.

Ваша программа должна выводить следующие сообщения:

```
...
...
...
Запущен red heart салют, в форме explode!!!
Запущен red circle салют, в форме disappear!!!
Салют purple heart завершил выступление change_color
Запущен purple circle салют, в форме explode!!!
Запущен yellow square салют, в форме disappear!!!
Салют yellow diamond завершил выступление change_color
Запущен red square салют, в форме explode!!!
Салют green diamond завершил выступление disappear
Запущен purple diamond салют, в форме disappear!!!
Салют green diamond завершил выступление change_color
Запущен yellow circle салют, в форме disappear!!!
Запущен blue diamond салют, в форме disappear!!!
...
...
...
```

Помните, что ключевая идея асинхронного программирования - возможность выполнять несколько задач "одновременно" без блокирования основного потока. Так, вашей программе нет смысла ожидать полного цикла запуска каждого салюта по отдельности. Вместо этого, с использованием `asyncio.gather()`, код должен запустить все салюты одновременно, и, в то время как они "выступают/летят/взрываются", программа переключается между ними, не ожидая окончания действия каждого конкретного салюта. Это позволяет значительно оптимизировать время выполнения программы, делая более эффективной и быстрой работу нашего фаершоу.