Паттерн Singleton

Есилевич Александр

13 декабря 2011 г.

Назначение

Гарантирует, что у класс есть только один экземпляр, и предоставляет к нему глобальную точку доступа

Примеры

- файловая система
- журнал приложения
- оконный менеджер (графическая система)

Реализация с помощью глобальной переменной

- Даёт глобальный доступ к объекту
- Не запрещает создавать другие экземпляры класса

Паттерн Singleton

```
class Singleton {
    public static instance() {
        if(inst == null) {
            // create instance
        }

        return inst;
    }

    // constructor is not public
    protected Singleton() {}

    // public singleton operations

    private static Singleton inst;
}
```

Пример использования: Abstract Factory

```
class ComponentFactory {
    public static instance() {
        if(inst == null) {
            String style = readStyleFromConfig();
            if(style == "style1")
                inst = new Style1Factory();
            else
                inst = new Style2Factory();
        }

        return inst;
    }

    protected ComponentFactory() {}

    public abstract SystemButton createButton();
    public abstract SystemEditBox createEditBox();

    private static ComponentFactory inst;
}
```

Пример использования: Abstract Factory

```
SystemButton button1 =
    ComponentFactory.instance().createButton();
button1.setListener(...);

SystemButton button2 =
    ComponentFactory.instance().createButton();
button2.setListener(...);
```

Проблемы реализации

Как сделать так, чтобы наследников `ComponentFactory` мог создавать только сам `ComponentFactory`?

- В C++ – сделать конструкторы наследников закрытыми и объявить `ComponentFactory` другом;
- В Java – использовать пакеты и использовать package-private модификатор доступа
- Сделать конструкторы наследников закрытыми, определить метод `instance` в каждом из них, и вызывать эти методы в `ComponentFactory.instance`.

Результаты

- контролируемый доступ к единственному экземпляру
- не засоряет пространство имён глобальными переменными
- допускает уточнение операций с помощью подклассов
- допускает переменное число экземпляров