

18.02.2022

# Антипаттерны

Четвериков Александр

Java Lab, Andersen

[www.andersenlab.com](http://www.andersenlab.com)

[link to telegram](#)



# Паттерн

Паттерн — это повторяемая архитектурная конструкция для решения часто встречающихся проблем или ситуаций, возникающих при проектировании приложения.



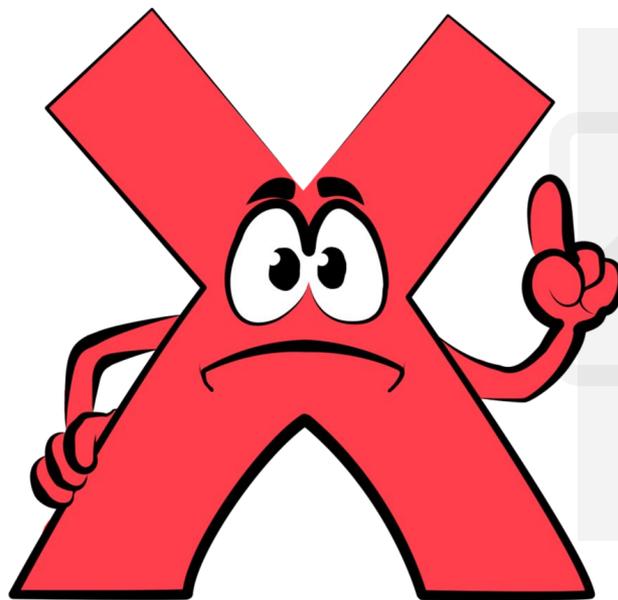
# Антипаттерн

Антипаттерн — это распространенный подход к решению класса часто встречающихся проблем, который является неэффективным, рискованным или непродуктивным.



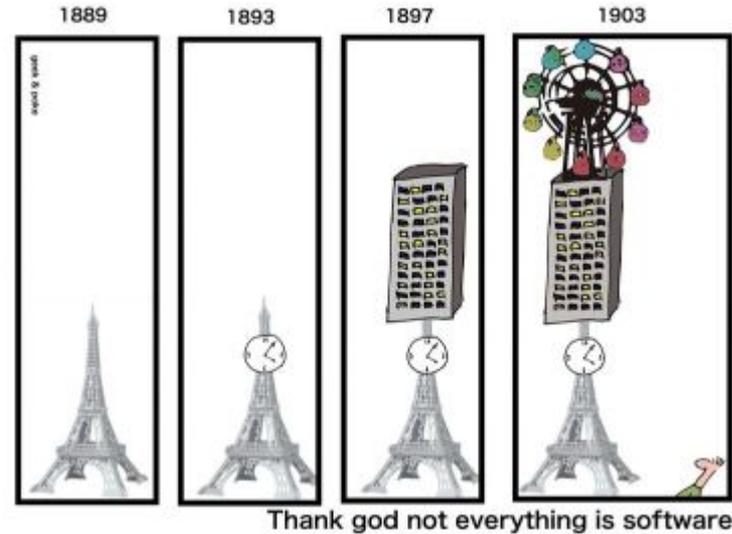
# Типы антипаттернов

1. **Development Anti Pattern** — антипаттерны архитектуры.
2. **Management Anti Pattern** — антипаттерны в области управления.
3. **Architectural antipatterns** — антипаттерны проблемы разработки, возникающие при написании программы.



# God object

Божественный объект — антипаттерн, который описывает излишнюю концентрацию слишком большого количества разношерстных функций, хранения большого количества разнообразных данных (объект, вокруг которого вращается приложение).



# Singleton

Одиночка — это самый простой паттерн, гарантирующий, что в однопоточном приложении будет единственный экземпляр некоторого класса, и предоставляющий глобальную точку доступа к этому объекту.

Но паттерн ли это или антипаттерн?



# Минусы синглтона

1. Глобальное состояние.
2. Синглтон нарушает один из принципов SOLID – Single Responsibility Principle.
3. Зависимость обычного класса от синглтона не видна в интерфейсе класса.
4. Наличие синглтона снижает тестируемость.



# Poltergeist

Бесполезные (полтергейстные) классы – это классы без зависимостей, используются для вызова методов другого класса или просто добавляют ненужный слой абстракции.

```
public class UserManager {  
    private UserService service;  
    public UserManager(UserService userService) {  
        service = userService;  
    }  
    User createUser(User user) {  
        return service.create(user);  
    }  
    String findEmailById(Long id) {  
        return service.findById(id).getEmail();  
    }  
    User findUserByEmail(String email) {  
        return service.findByEmail(email);  
    }  
}
```



# Hard code

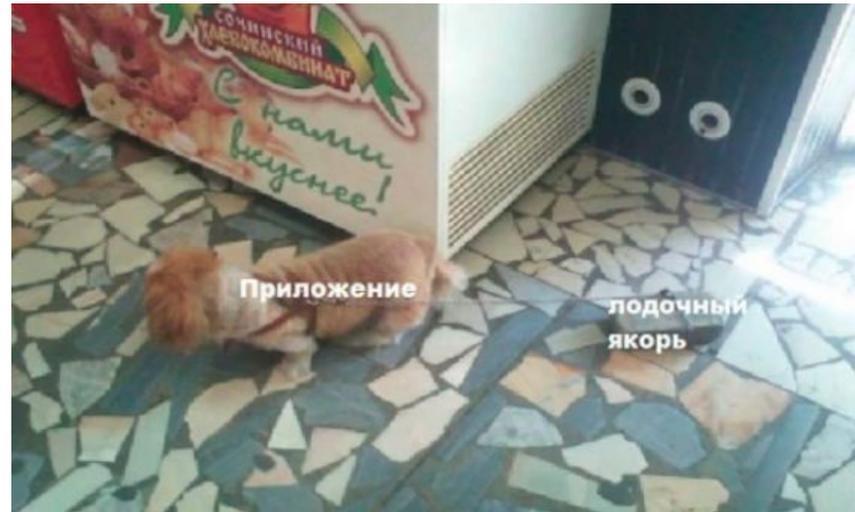
Суть данного антипаттерна в том, что код сильно привязан к конкретной аппаратной конфигурации и/или системному окружению, что сильно усложняет перенос его на другие конфигурации. Данный антипаттерн тесно связан с магическими числами (они часто переплетаются).

```
public Connection buildConnection() throws Exception {  
    Class.forName("com.mysql.cj.jdbc.Driver");  
    connection = DriverManager.getConnection(  
        "jdbc:mysql://localhost:3306/someDb?character  
        Encoding=UTF-8&characterSetResults=UTF-  
8&serverTimezone=UTC", "user01", "12345qwert");  
    return connection;  
}
```



# Boat anchor

**Лодочный якорь** в контексте антипаттернов означает хранение неиспользуемых частей системы, которые остались после какой-то оптимизации или рефакторинга. Также некоторые части кода могли быть оставлены «на будущее», вдруг придётся ещё их использовать. По сути, это делает из кода мусорное ведро.



# Analytical paralysis

Аналитический паралич — суть паттерна заключается в чрезмерном анализировании ситуации при планировании, так что решение или действие не предпринимаются, по сути парализуя разработку.



# Public Morozov

Шуточный антипаттерн публик Морозов - это класс, который открывает доступ ко всем полям и методам класса-предка, не зависимо от их модификаторов видимости.



# Используемая литература

<http://www.habr.com/>

<http://www.javenuе.info/>

<https://javarush.ru/>

<https://yandex.ru/images/>



THANK YOU



**ANDERSEN**

[www.andersenlab.com](http://www.andersenlab.com)

link to telegram