Feature toggle

Станислав Цыганов
alloy.stas@gmail.com
@DevAlloy

RAMBLER&Co

Задачи

- Настройки пользователя
- A/B тестирование
- Удаленное управление фичами
- Категоризация пользователей

О чем поговорим

- Что такое feature toggle
- Их категории
- Возможные подходы к реализации

Feature toggle

- Наличие состояния приложения, исходя из которого те или иные фичи считаются включёнными или выключёнными
- Состоянием может быть: конфигурационный файл с сервера, состояние сети, версия iOS
- Ветвление логики происходит в toggle point

Концепция

- Наличие базового кода

- Наличие переключаемого/выключаемого кода

Категории

Основные метрики:

- Продолжительности жизни
- Степень гибкости

Экспериментальные

- Очень динамичные (могут меняться по несколько раз во время одного запуска)

- Сюда же можно отнести фичи, которыми может управлять пользователь

- Фичи для A/B тестирования

Системные

- Фичи, которые могут сильно влиять на производительность системы
- Потенциально опасные фичи, которые может понадобиться удаленно отключить

Разрешительные

- Выделение премиальных пользователей
- Внутреннее тестирование фичи

Возможные подходы

Зависят от:

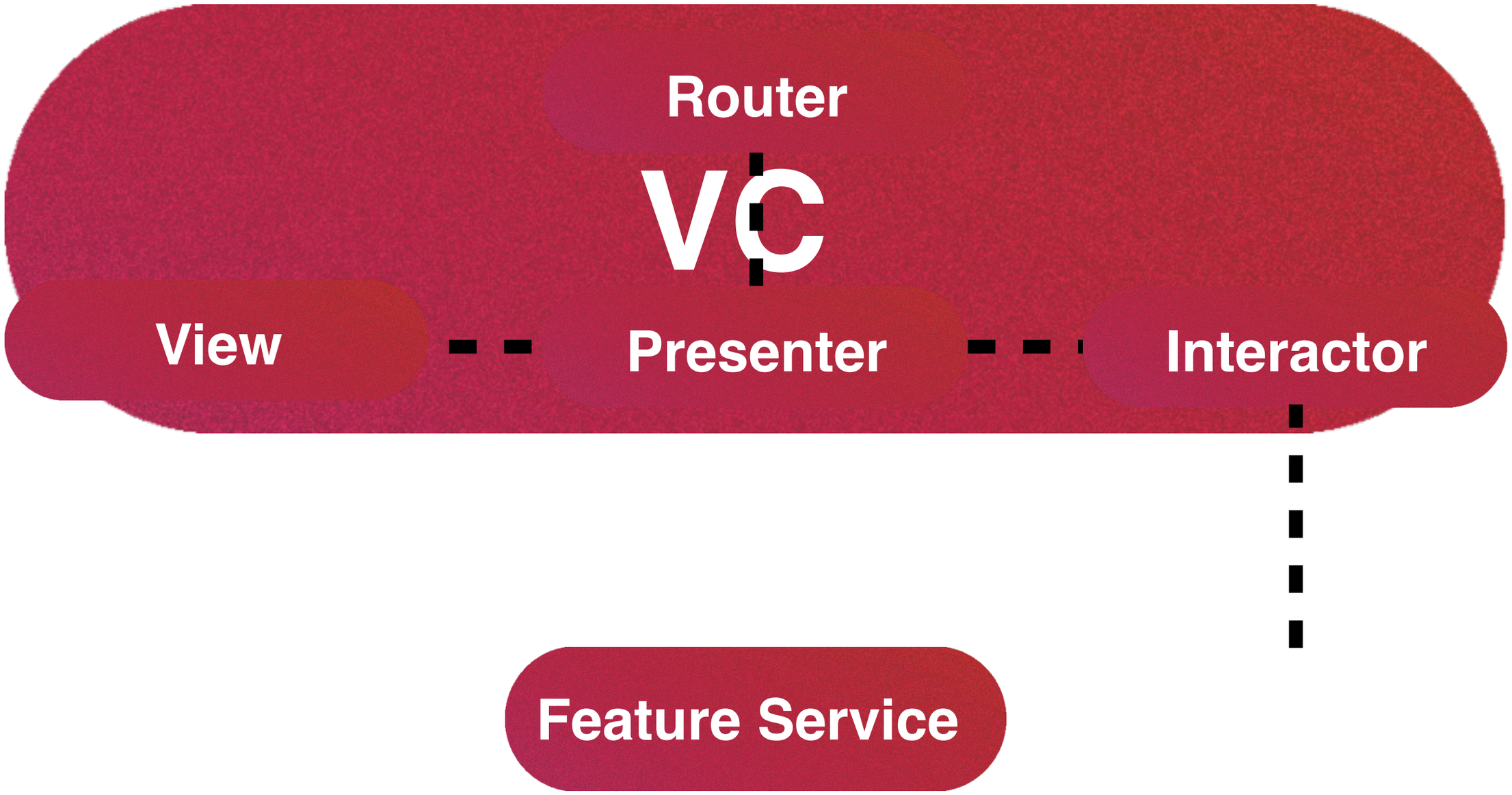
- Динамичности фичи
- Архитектуры

Динамические фичи

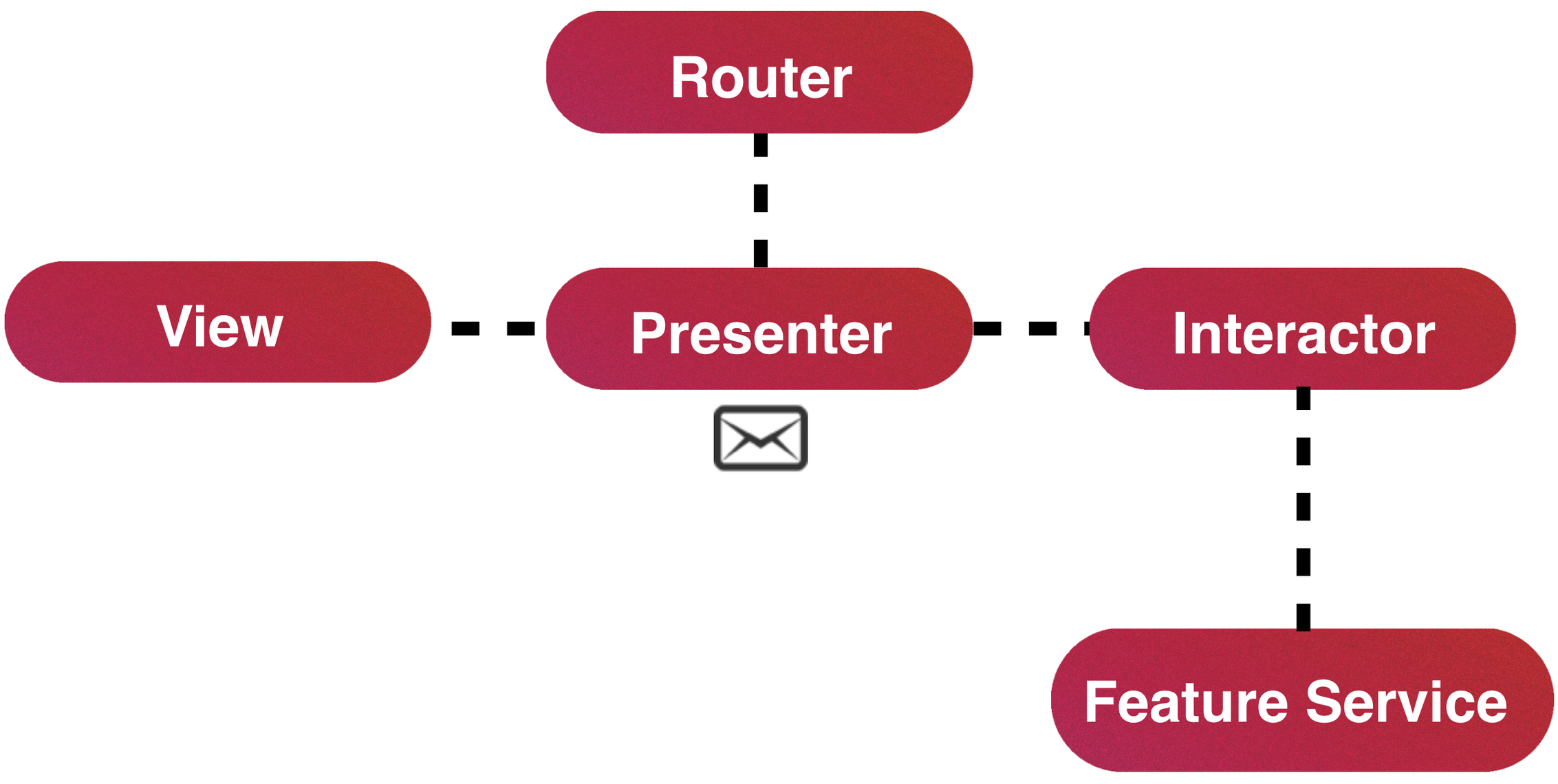
Динамические фичи

- Сервис настроек
- Реализация интерактора

Сервис настроек



Сервис настроек



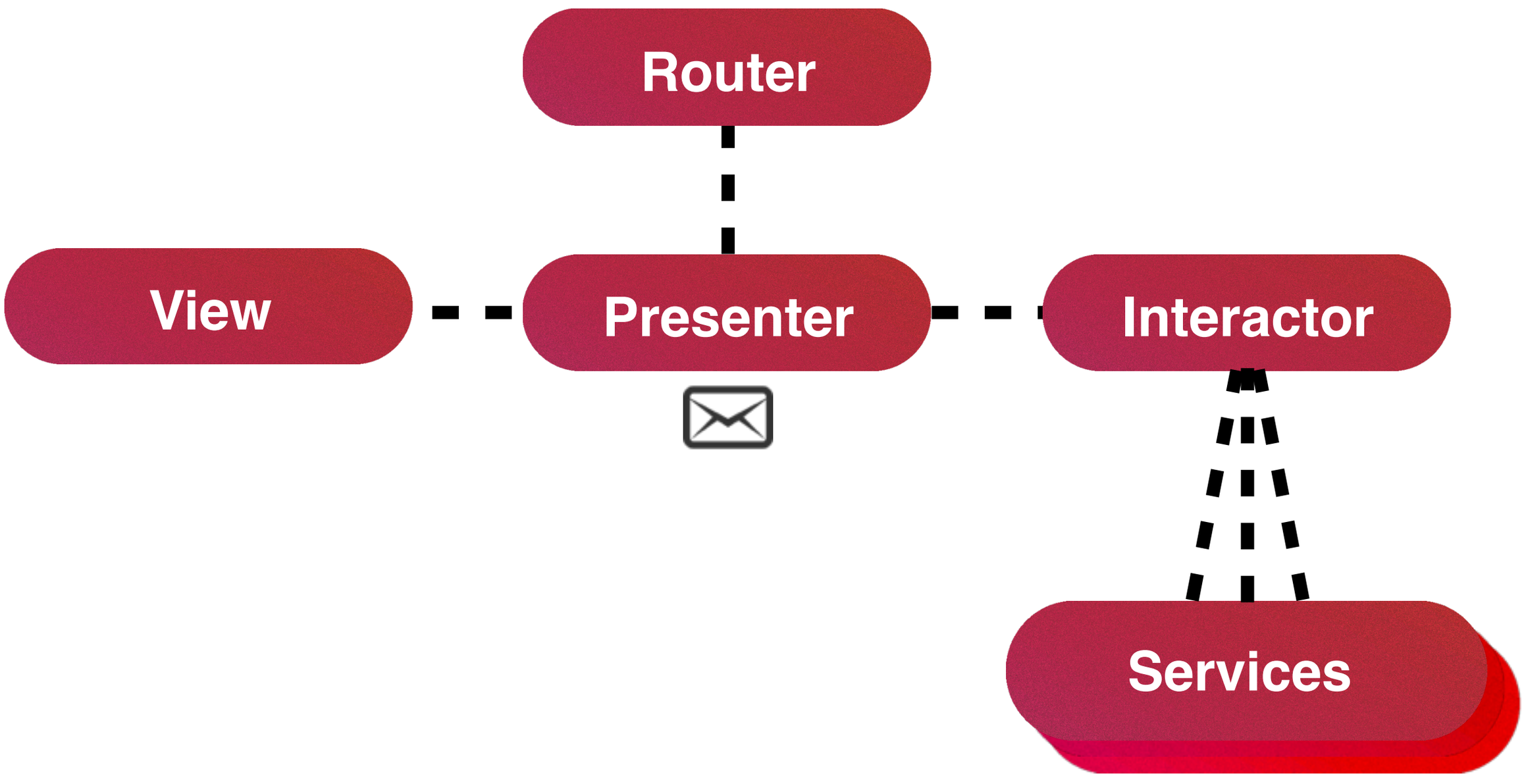
Сервис настроек

```
- (BOOL)featureEnabled {
    BOOL reachable = [self.reachability ...];
    BOOL highestVersion = [self.versioningClient ...];
    return reachable && highestVersion;
}
```

Сервис настроек

- Отвечает за включенность фич
- Интерфейс может слишком сильно разрастаться

Реализация интерактора



Реализация интерактора

```
- (BOOL)featureEnabled {
    BOOL userAuthorized = [self.authService ...];
    BOOL autoPlayEnabled = [self.settingsService ...];
    return userAuthorized && autoPlayEnabled;
}
```

Различия

- Позволяет избежать большого сервиса
- Менее переиспользуем

Presenter/VC

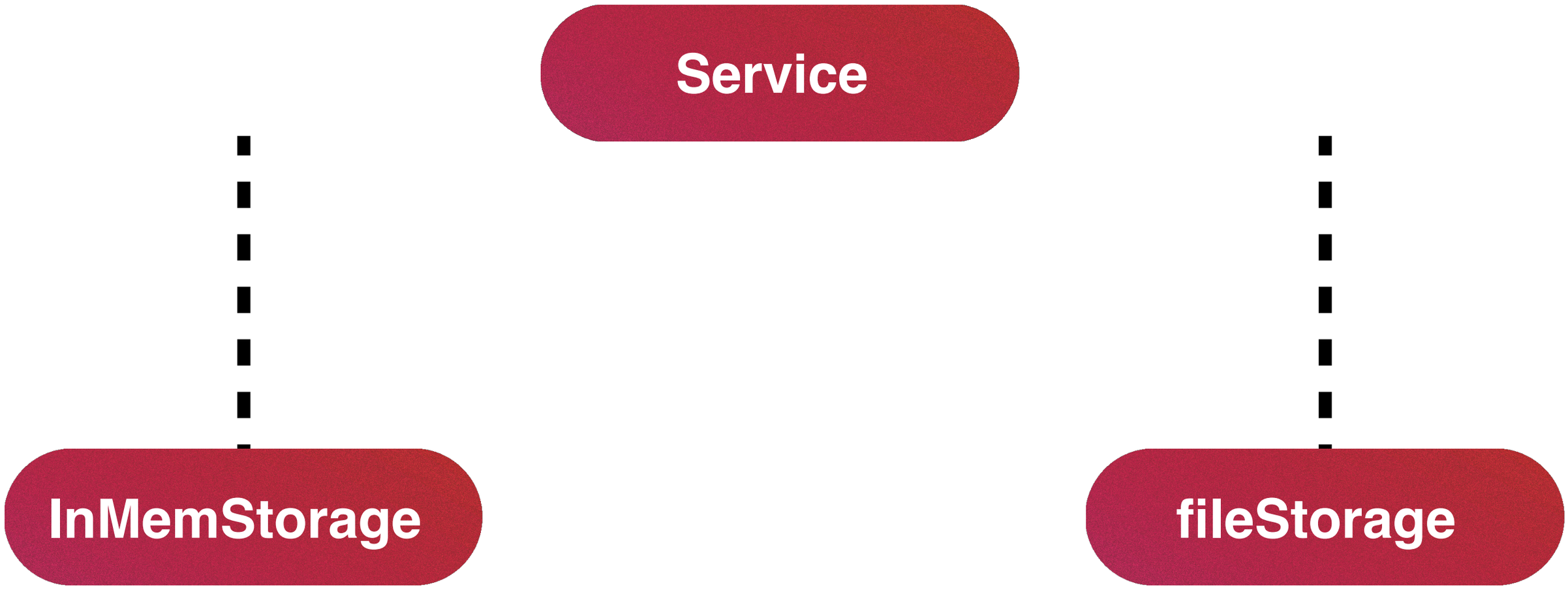
```
- (void)foo {
    BOOL featureEnabled = [self.interactor featureEnabled];
    if (featureEnabled) {
        // do feature work
    }
}
```

Статические фичи

- Инъекция зависимостей
- Передача конфигурации

Инъекция зависимостей

- Правильные зависимости могут быть выбраны при инициализации



Инъекция зависимостей

- Может быть реализована через инъекцию фабрики

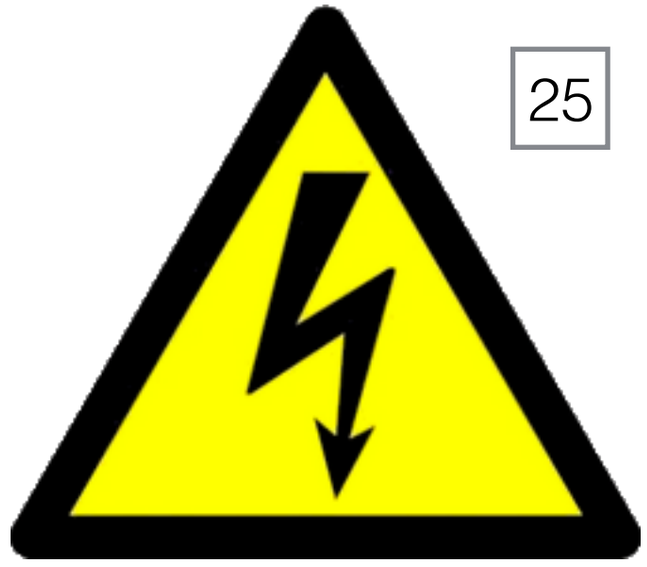
```
- (void)foo {
    if (featureEnabled) {
        self.storage = [self.storageFabric memStorage];
    } else {
        self.storage = [self.storageFabric fileStorage];
    }
}
```

Передача конфигурации

- Конфигурация устанавливается извне

```
- (void)foo {
    if (self.configuration.featureEnabled) {
        // ...
    } else {
        // ...
    }
}
```

Внимание!



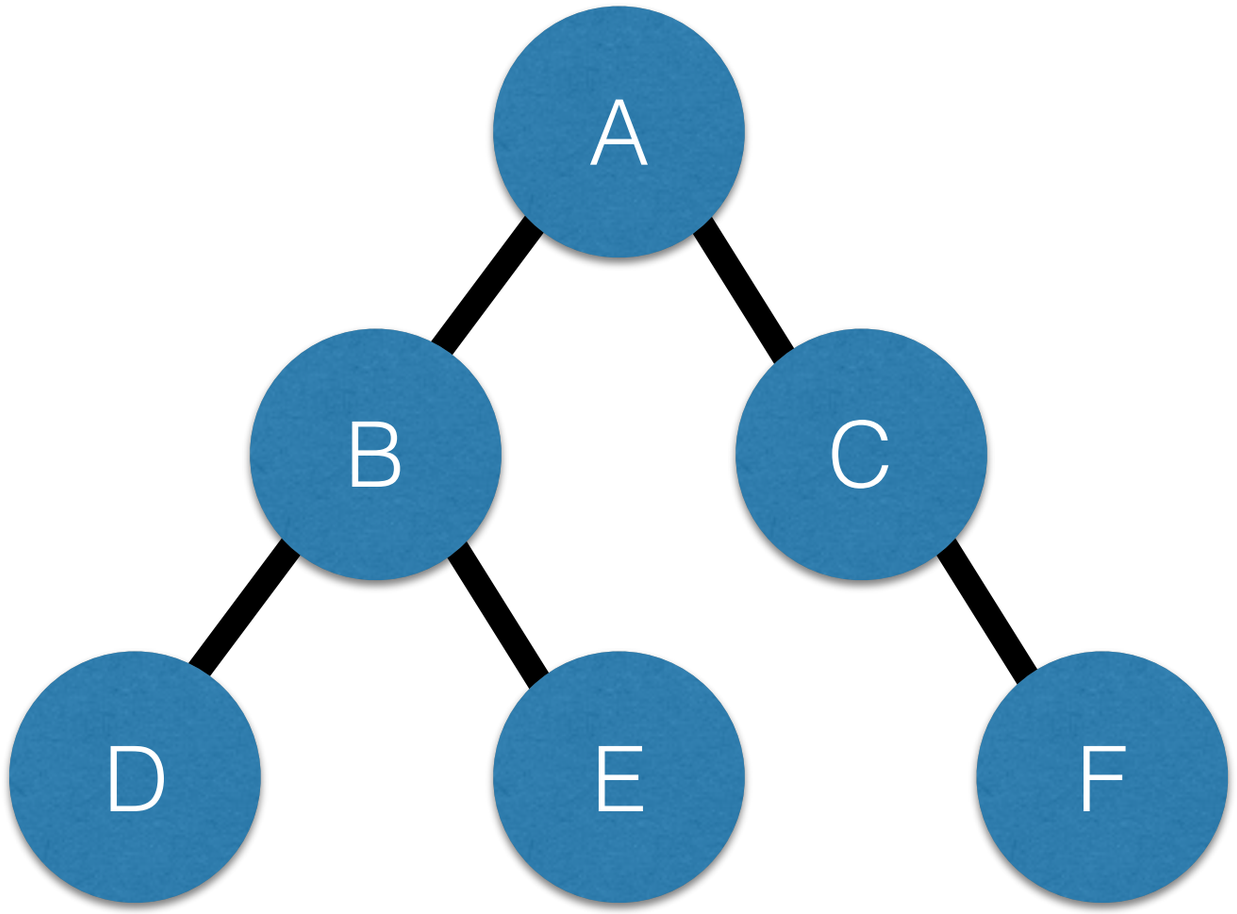
- Не стоит инкапсулировать абсолютно все фичи
- Не стоит делать зависимые друг от друга фичи
- Разделение места ветвления и принятия решения

Избыточная инкапсуляция

- Дополнительные усилия

- Невозможность/высокая сложность надстройки над фичей

Зависимые фичи



- Сложность тестирования
- Сложность расширения и рефакторинга

Toggle point

- Выделение объекта для принятия решения
- Получения конфига извне
- Правильная конфигурация модуля

Что получаем?

- Инкапсуляцию фич
- Относительно простой переход между категориями feature toggle
- Спокойствие за "опасные" фичи

Итоги

- Feature toggle - подход, где поведения компонентов приложения может меняться в зависимости от состояния приложения

- Выделяются основные категории фич:
  Экспериментальные,
  Системные и Разрешительные

- Подходы к реализации статических и динамических фич