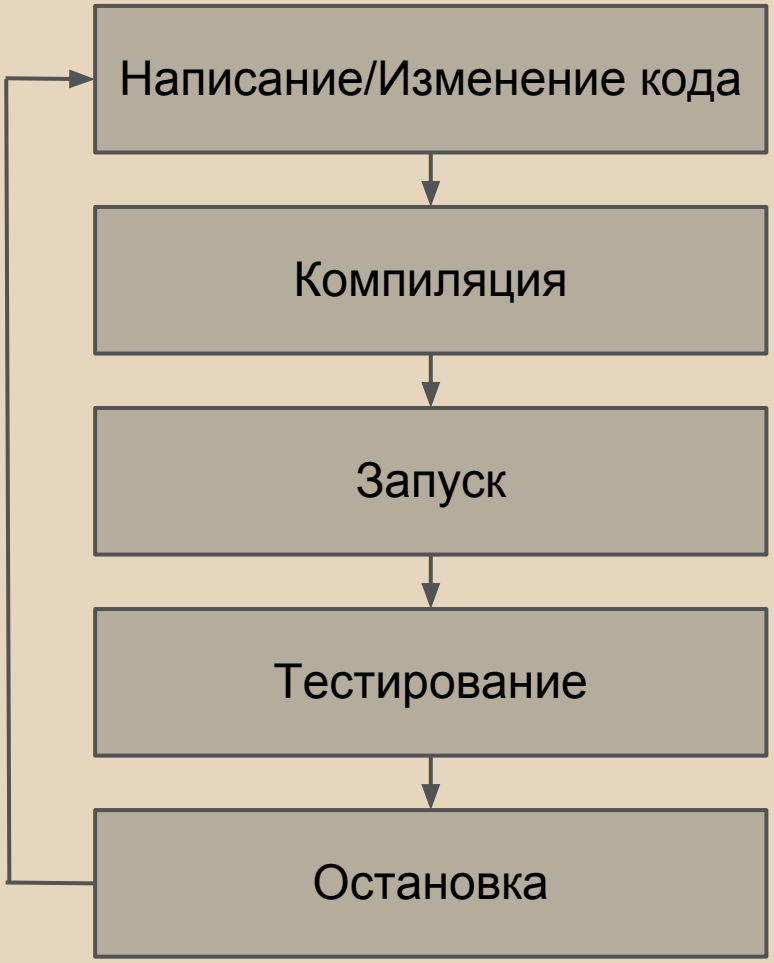
Замена классов, загруженных в JVM, на лету

студент: Лепенькин Я.А.
руководитель: Жданов Д.Ю.

СПбАУ НОЦ НТ

Цикл разработки ПО



Замена кода на лету

```
public int getInt(int x) {
    x += 4;
    x *= 12;
    return x;
}
```

```
public int getInt(int x) {
    return compute(x) + 5;
}
private int compute(int x){
    x += 4;
    x *= 12;
    return x;
}
```

Стандартные средства Java не позволяют производить такую замену.

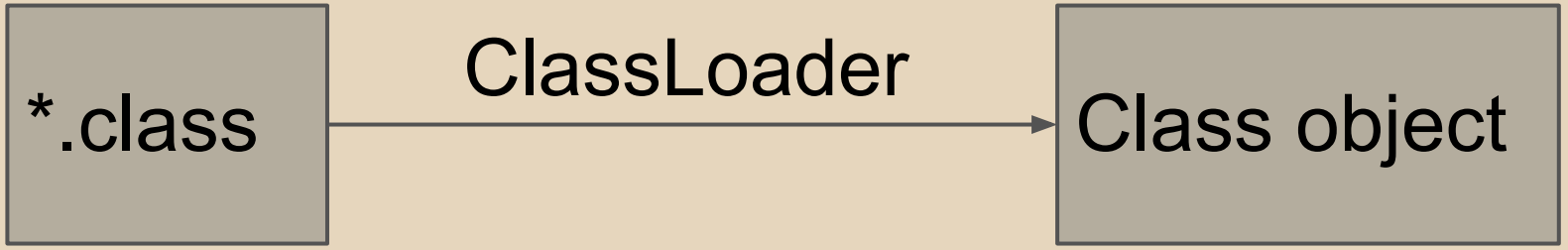
Цель

Предоставить пользователям виртуальной машины Java удобную возможность добавлять методы, поля загруженных классов.

Задачи

- изучение возможностей и ограничений HotSpot JVM по переопределению классов;
- анализ существующих решений;
- выбор оптимального подхода;
- реализация выбранного подхода в виде плагина для IntelliJ IDEA.

Загрузка классов

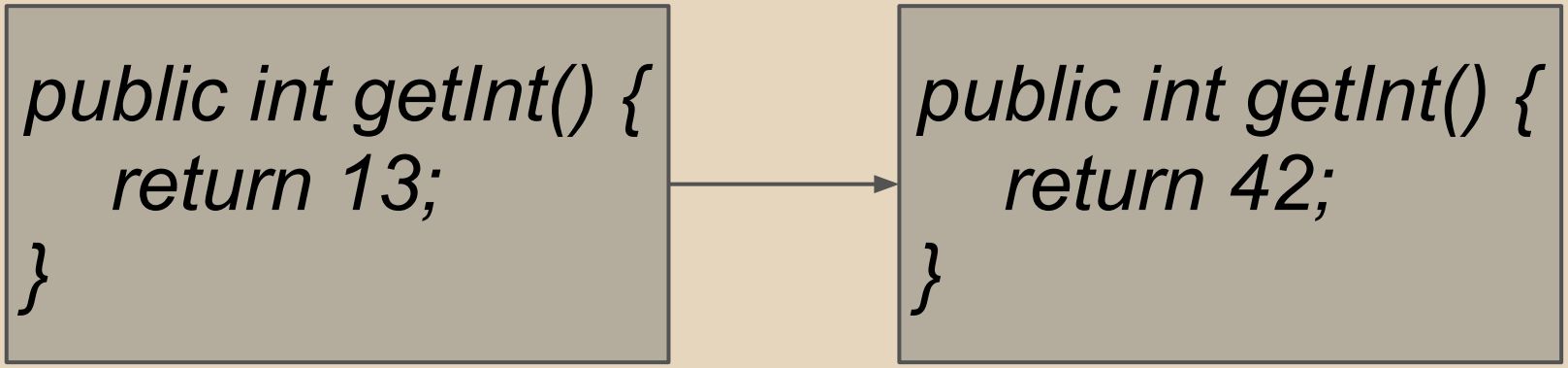


Не существует способа внутри приложения переопределить уже загруженный класс.

Переопределение классов

технология Java HotSwap

- появилась в Java 1.4
- использует Java Debug Interface
- поддерживается всеми современными IDE
- заменять можно лишь тела методов

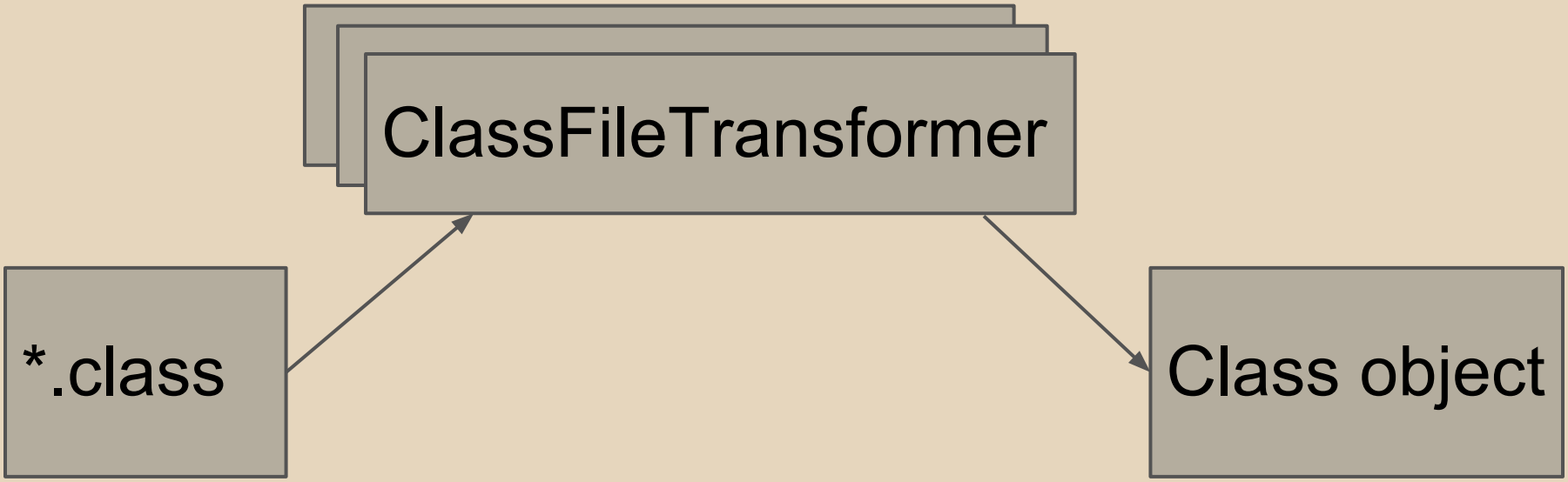


Возможные подходы

- модифицировать класс при загрузке таким образом, чтобы последующие изменения не затрагивали его структуру (JRebel, Javadaptor);
- использовать альтернативные виртуальные машины, поддерживающие полноценную замену классов (DCE VM, JVolve).

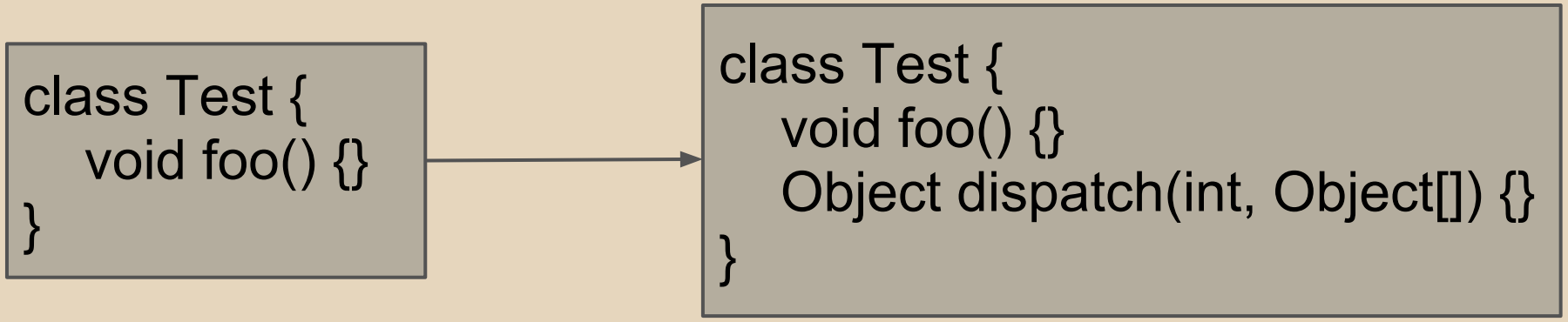
ClassFileTransformers

java.lang.instrument
Instrumentation.addClassFileTransformer(...)



Инструментирование байткода

При загрузке класса:



Инструментирование байткода

При добавлении метода:

```
class Test {
    void foo() {}
    Object getObj() {
        return new Object();
    }
}
```



```
class Test {
    void foo() {}
    Object dispatch(int id, Object[] as) {
        if (id == 1) {
            return new Object();
        }
    }
}
```

Дальнейшие вызовы getObj заменяются вызовом метода dispatch с соответствующими параметрами.

Dynamic Code Evolution VM

Поддерживает:

- добавление и удаление методов, полей;
- изменение иерархии наследования.

Модификация HotSpot JVM. Изменения не затрагивают подсистем выполнения.

Подход, основанный на инструментировании байткода

достоинства:

платформо-независимость

недостатки:

длительность первоначальной загрузки, появление дополнительных методов в стек-трейсе, использование Reflection API, необходимость дополнительных средств интеграции с отладчиком

Подход с использованием DCE VM

<u>достоинства:</u>

расширение стандартного Java HotSwap, не требует дополнительной интеграции со средствами разработки, доступный исходный код

<u>недостатки:</u>

платформо-зависимость

Плагин для IntelliJ IDEA

- установка Dynamic Code Evolution VM;
- автоматическое обновление DCEVM;
- изменение конфигураций запуска проектов пользователей.

Результаты

- изучен механизм Java HotSwap;
- изучены решения, использующие инструментирование байткода;
- собран JDK, DCE VM на linux/macos/windows;
- написан плагин для IntelliJ IDEA.

Спасибо за внимание!