Макрорасширения для среды разработки на языке Scala

Студент: Осипов Станислав

Руководитель: Подхалюзин Александр Викторович

Академический Университет 2014

Особенности Scala

- Альтернатива Java для JVM
- Сочетание функциональной парадигмы с объектно-ориентированным подходом
- Строгая статическая унифицированная система типов
- Обратная совместимость с Java
- Активно развивается

Цели и задачи

Цель:

Обеспечить в среде разработки возможность предоставлять пользователю информацию о методах, сгенерированных во время компиляции

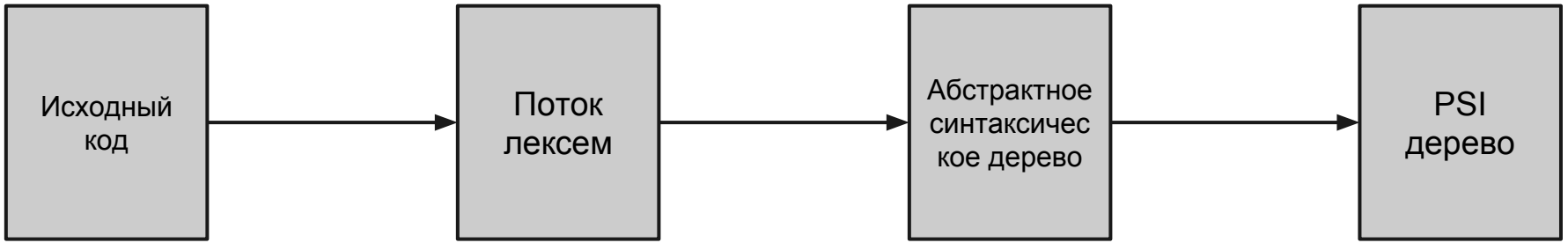
Задачи:

- Анализ возможных решений
- Разработка предметно-ориентированного языка для создания расширений
- Реализация возможности динамически добавлять расширения
- Тестирование

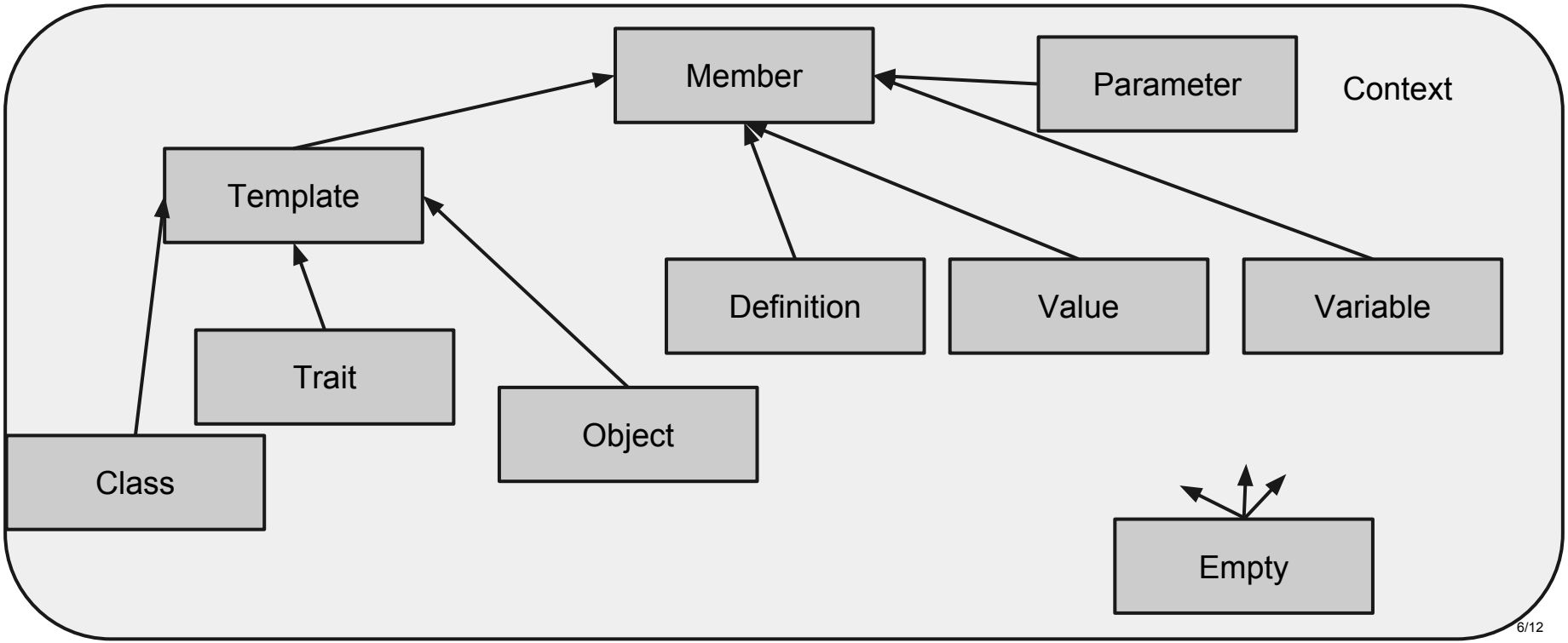
Возможные подходы

- Обработка семантики конструкций языка
  - +: Подходит для стандартных случаев
  - -: Требует значительных усилий для поддержки произвольных расширений
- Анализ class-файлов
  - +: Не требует написания дополнительного кода от разработчика
  - -: Устаревшая информация, проблемы с производительностью
- Динамические расширения
  - +: Поддержка произвольных макросов и расширений компилятора
  - -: Нужно писать дополнительный код

Анализ кода в среде разработки

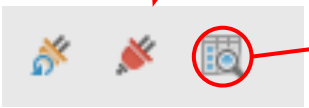


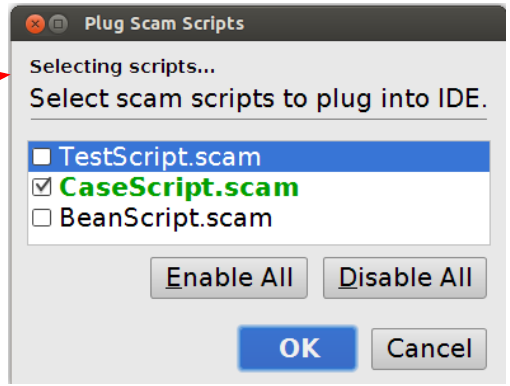
Структура предметно-ориентированного языка



Динамическая загрузка: Scam скрипты

```
import ...

for(c <- Template.fromContext) {
  if(c.hasAnnotation(Annotation("SynCase"))) {
    c.companion += Method(_ => "apply", Seq(String_), String_)
  }
}
```

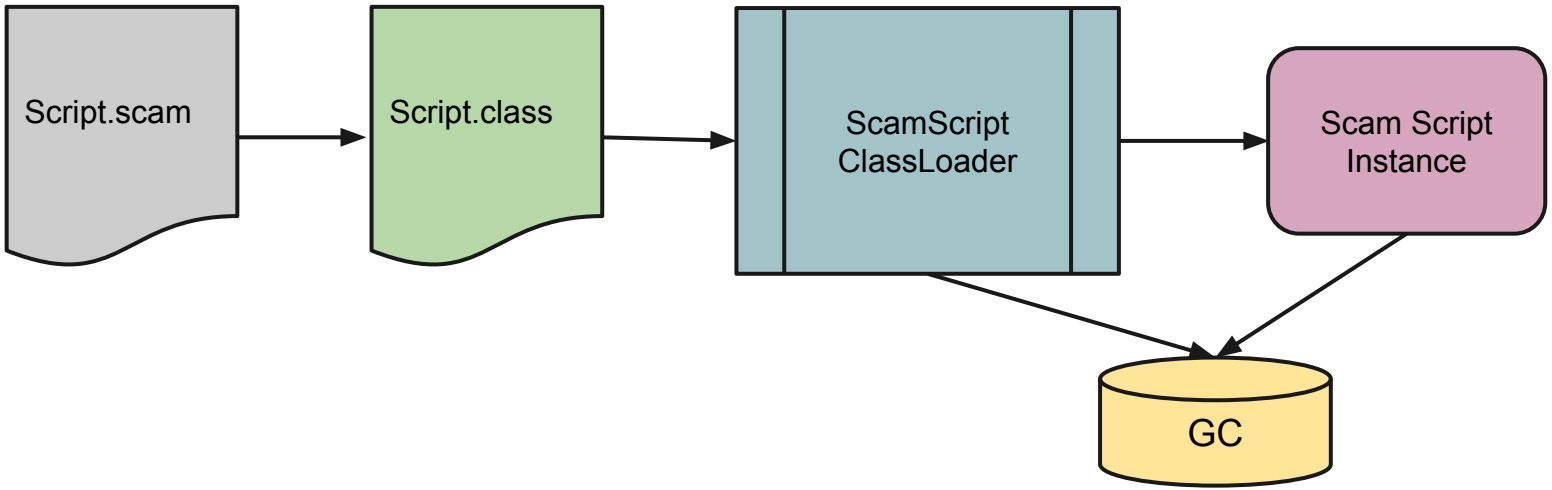




Специальный тип Scala файла с дополнительным интерфейсом и расширенной областью видимости

Менеджер позволяет находить скрипты во всем проекте и динамически их загружать

Динамическая загрузка: компиляция



Тестирование

Обработка семантики BeanProperties и Case Classes в среде разработки заменена на соответствующие скрипты. Протестирована корректная работа рефакторингов и других сервисов

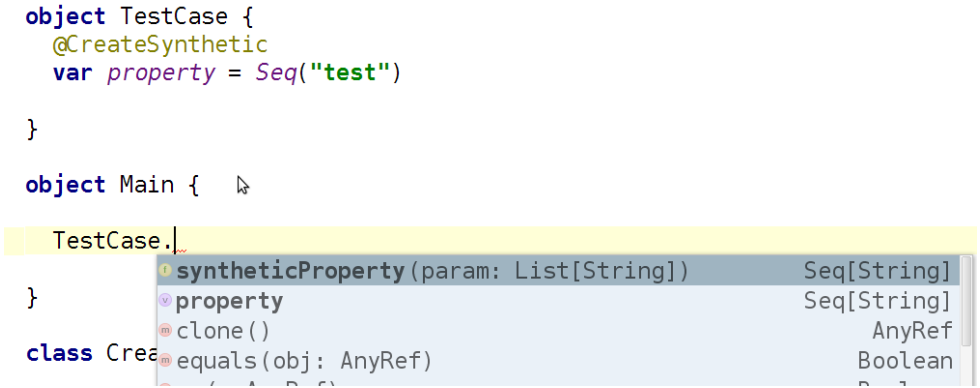
```
val getter= Method("get" + _.capitalize, Seq(), _.getScalaType)
val is = Method("is" + _.capitalize, Seq(), _.getScalaType)
val setter = Method("set" + _.capitalize, Seq(_.getScalaType), Unit_)

for (h <- Member.fromContext) {
  if (h.hasAnnotation(Annotation("BeanProperty"))) {
    h.asVariable.containingClass.add(Seq(getter, setter))
    h.asValue.containingClass.add(getter)
  }
  if (h.hasAnnotation(Annotation("BooleanBeanProperty"))) {
    h.asVariable.containingClass.add(Seq(is, setter))
    h.asValue.containingClass.add(is)
  }
}
```

Технологии

- Scala
- Java
- Intellij Platform
  - PSI Framework
  - Index Framework
  - Persistent API
  - Test Framework
  - ...

Результат

- Разработан инструмент, позволяющий среде разработки анализировать код с учетом методов, сгенерированных во время компиляции.
- Реализована динамическая загрузка расширений написанных на DSL
- Инструмент интегрирован в Scala Plugin for Intellij IDEA



Исходный код :
https://github.com/OsipovStas/intellij-scala

Спасибо за внимание