УДК 004.09

СОЗДАНИЕ ПЛАГИНА В TEAMCITY ДЛЯ ИНТЕГРАЦИИ С ИНСТРУМЕНТАМИ АНАЛИЗА КОДА INTELLIJ IDEA

Кочетков Н.О. (федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»)
Научный руководитель – программист факультета ИТиП, Аксенов В.Е.
(федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»)

Аннотация. Данный проект нацелен на разработку плагина интегрируемого в CI/CD TeamCity с возможностью удобной и быстрой настройки для работы с анализом кода IntelliJ IDEA.

Введение. В настоящее время существует множество различных решений для статического анализа кода. Однако они все довольно разные. Одни базируются только на точечных инспекциях с единственным языком программирования, другие охватывают множество различных языков. Некоторые завязаны не на конкретный язык, а ищут какие-то определенные проблемы, например поиск уязвимостей.
Проблема большинства инструментов заключается в том, что у создателей различное, зачастую некомфортное для использования на практике, видение, как должен быть выстроен процесс настройки и сам UI для него. Поэтому было бы лучше придумать хорошую систему для любого пользователя, от обычного программиста, до системного инженера.
Цель работы заключается в написании плагина, позволяющего внедрить всю мощь инспекций IntelliJ IDEA, которые можно увидеть у себя в проекте при её использовании, в сервис непрерывной интеграции - TeamCity.

Основная часть. Прежде чем приходить к созданию собственной системы были изучены такие инструменты, как SonarQube, ESLint и т.д. По итогу изучения были выявлены их достоинства и недостатки при настройке и использовании пользователями.
В SonarQube было необходимо для использования поднимать их сервер, который ещё не давал никакого анализа, но уже требовал ресурсы и какие-то настройки. Для программиста будет очень долго производить все эти действия, только чтобы прогнать анализ. Помимо этого, необходимо было конфигурировать анализ через отдельные файлы с большим числом настроек, что не удобно. В интернете присутствует мало обучающих уроков от самих разработчиков, что повышает порог вхождения.
В случае с ESLint рассматриваются только JavaScript подобные языки. Для его работы необходим Node.js. ESLint хорошо решает свою задачу, однако является узкоспециализированным решением. Его также трудно назвать эталоном для конфигурирования.
В разработке плагина были учтены все позитивные и негативные моменты, в том числе описанные выше. Для удобного использования у вас появляется возможность добавить build step, который позволяет настроить дополнительный шаг при сборке вашей build конфигурации. Благодаря использованию IntelliJ IDEA исчезла проблема с индексацией и подтягиванием большинства зависимостей во время анализа на агентах, что необходимо было делать в других глобальных инструментах. Добавлена система выбора профайлов, которые позволяют определить инспекции, участвующие в анализе. Для того, чтобы можно было переиспользовать полученный результат анализа, была добавлена возможность выставлять Failure condition на основании статистики различных по критичности проблем после прогона или оформлять в виде тестов. В дальнейшем тесты можно будет назначать на людей, которые должны их решить, что не позволит получить собранный продукт плохо качества. Также поддержана возможность узнать между прогонами, что именно изменилось (добавилось и решилось), не все инструменты так умеют.

Выводы. Разработанный в данной работе плагин позволил использовать инспекции IntelliJ IDEA вне IDE и решил некоторые проблемы с запуском анализа.

Кочетков Н.О. (автор)	Подпись
Аксенов В.Е. (научный руководитель)	Подпись