

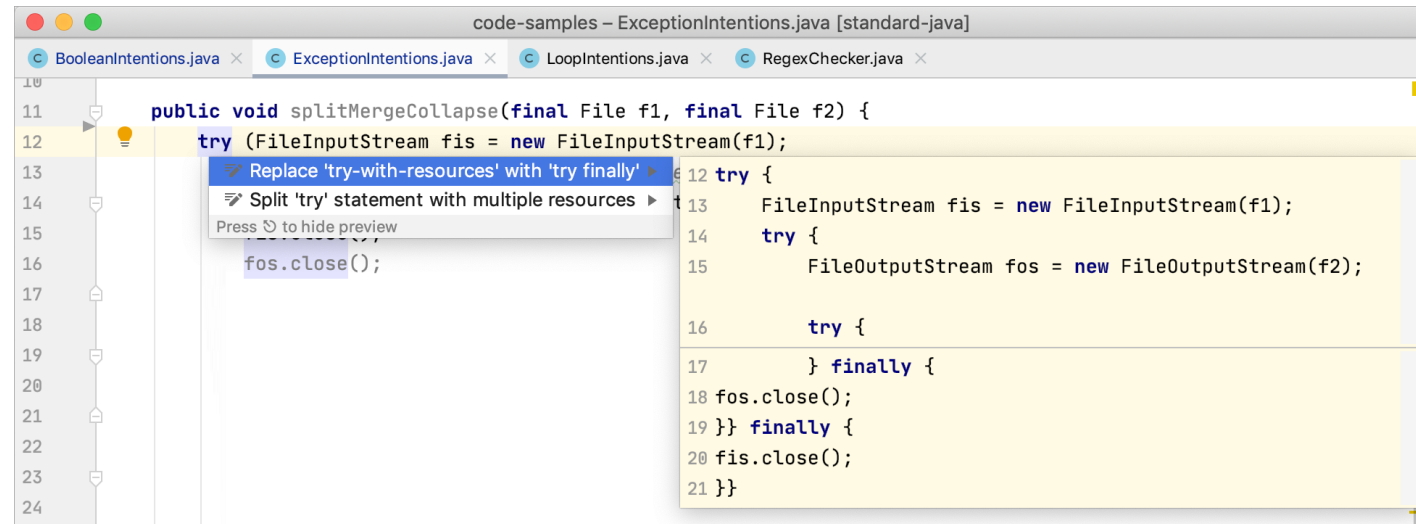
Рекомендация улучшений кода на Java в IntelliJ IDEA

Кутленков Дмитрий, 17.Б11-мм

Научный руководитель – к.т.н. Брыксин Т. А.

Рецензент – программист ООО “ИнтеллиДжей Лабс” Красильщиков Д.В.

Суть работы



The screenshot shows the IntelliJ IDEA IDE with a Java file named 'ExceptionIntentions.java'. The code is as follows:

```
11 public void splitMergeCollapse(final File f1, final File f2) {
12     try (FileInputStream fis = new FileInputStream(f1);
13          FileOutputStream fos = new FileOutputStream(f2);
14          try {
15              fos.close();
16          } finally {
17              fis.close();
18          })
19     }
20 }
21 }
```

An intention menu is open over the try statement, showing the following options:

- Replace 'try-with-resources' with 'try finally'
- Split 'try' statement with multiple resources

The 'Split 'try' statement with multiple resources' option is selected, and a preview of the resulting code is shown below the menu:

```
12 try {
13     FileInputStream fis = new FileInputStream(f1);
14     try {
15         FileOutputStream fos = new FileOutputStream(f2);
16
17         try {
18             fos.close();
19         } finally {
20             fis.close();
21         }
22     }
23 }
```

- В среде разработки **IntelliJ IDEA** существует механизм “Intention actions”, позволяющий программисту выполнять автоматические преобразования кода (рефакторинг)
- Зачастую полезность применения намерения субъективна или зависит от контекста
- Иногда эти действия применяются несколько раз подряд (образуют цепочки)

Пример



Which one is better?

First:

```
1     }
2
3     private void deleteStagingFilesAsync()
4     {
5         List<File> files = listFiles(baseStagingDir, null);
6         if (!files.isEmpty()) {
7             new Thread(() -> {
8                 for (File file : files) {
9                     try {
10                        Files.deleteIfExists(file.toPath());
```

Second:

```
1     }
2
3     private void deleteStagingFilesAsync()
4     {
5         List<File> files = listFiles(baseStagingDir,
6             null);
7         if (!files.isEmpty()) {
8             new Thread(() -> {
9                 for (File file : files) {
10                    try {
```

Пример



Which one is better?

First:

```
1         if (prestoTableName.equalsIgnoreCase(pinotTableName)) {
2             return pinotTableName;
3         }
4     }
5     throw new PinotException(PINOT_UNCLASSIFIED_ERROR, Optional.empty(), "Unable to find the presto table " + prestoTableName + " in " + allTables);
6 }
7
8 @Override
9 public PinotTableHandle getTableHandle(ConnectorSession session, SchemaTableName tableName)
10 {
```

Second:

```
1     }
2 }
3     throw new PinotException(PINOT_UNCLASSIFIED_ERROR,
4         Optional.empty(),
5         java.lang.String.format("Unable to find the presto table %s in %s",
6             prestoTableName,
7             allTables));
8 }
9
10 @Override
```

Цели и задачи

Цель: разработка плагина для **IntelliJ IDEA**, способного классифицировать намерения и цепочки намерений, предлагая пользователю те из них, чье применение улучшает код.

Задачи:

- Разработать инструмент для сбора данных (возможных преобразований кода), применить его к некоторой кодовой базе и привести данные к удобному для разметки виду
- На размеченных данных обучить модель для классификации вариантов преобразований
- Создать плагин, использующий полученную модель

Существующие решения

- *Haas et al.*¹ – функция оценки для рефакторинга «Извлечение метода»
 - *Yue et al.*² – машинное обучение для поиска мест, к которым стоит применить рефакторинг
 - *Fontana et al.*³ – те же методы для распознавания запахов кода
-
- Существующие решения не учитывают цепочки преобразований и не предоставляют возможности мгновенного применения рефакторинга

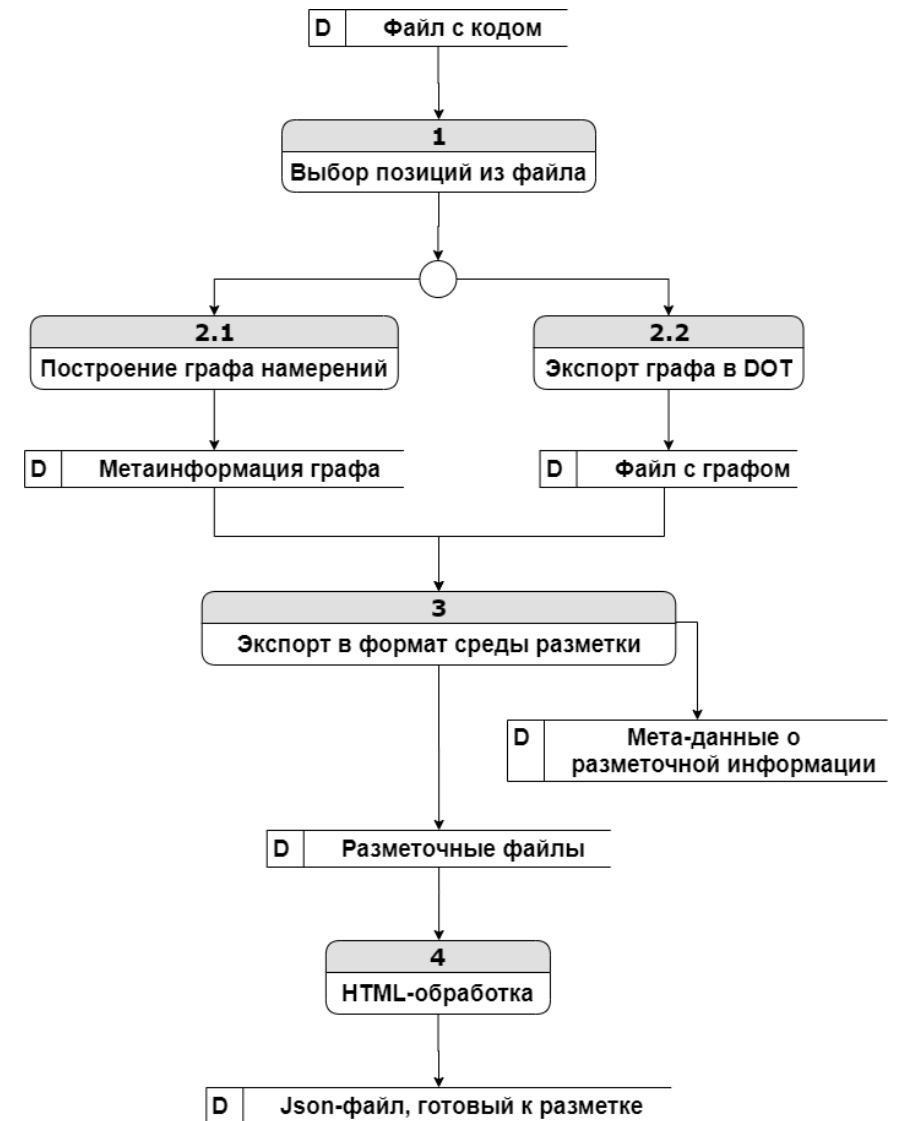
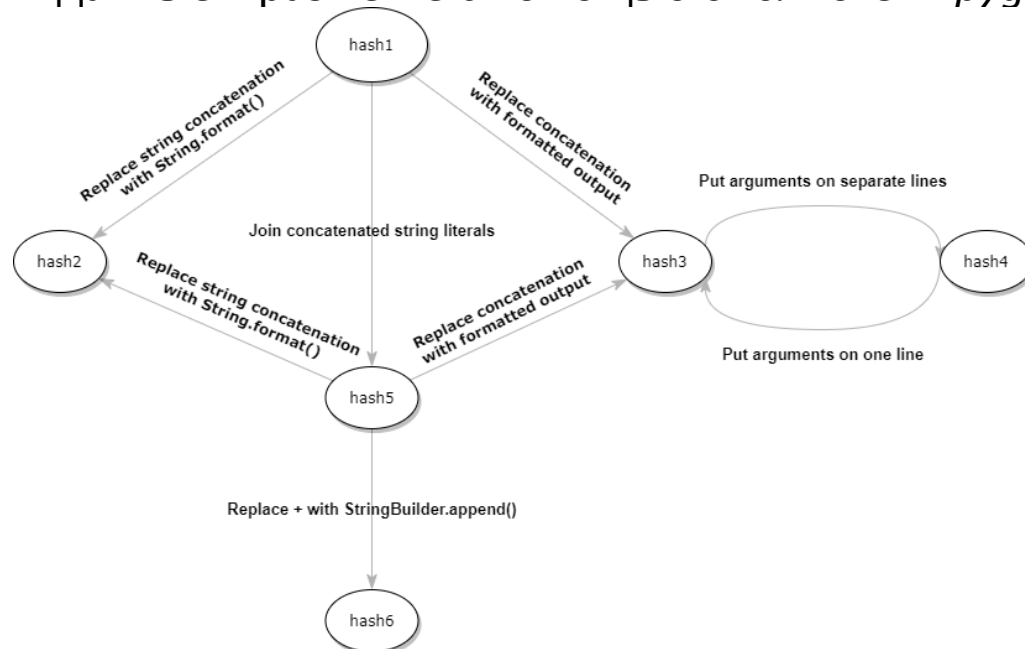
1 - Haas, Roman & Hummel, Benjamin. (2016). Deriving Extract Method Refactoring Suggestions for Long Methods. 144-155. 10.1007/978-3-319-27033-3_10.

2 - R. Yue, Z. Gao, N. Meng, Y. Xiong, X. Wang and J. D. Morgenthaler. (2018). "Automatic Clone Recommendation for Refactoring Based on the Present and the Past," 2018 IEEE ICSME. 115-126. 10.1109/ICSME.2018.00021.

3 - F. A. Fontana, M. Zanoni, A. Marino and M. V. Mäntylä. (2013). "Code Smell Detection: Towards a Machine Learning-Based Approach," 2013 IEEE ICSM. 396-399. 10.1109/ICSM.2013.56.

Инструмент для извлечения вариантов кода

- К каждой позиции кода пытаемся применить все возможные намерения
- Повторяем с полученными результатами
- Получаем **граф намерений**
- образуем пары из изначального варианта и всех остальных
- Готовим данные к разметке с помощью библиотеки *rugments*



Векторизация и метрики

- Векторизация кода производится с помощью **метрик**
- Было реализовано 9 метрик, которые измеряют **влияние на восприятие кода**
- Помимо абсолютных значений, производится подсчет **разностных метрик** между двумя участками кода

Количественные метрики	Статистические метрики
Число строк, число переносов строки внутри выражения, число параметров, число отступов и др.	Средняя длина имени параметра, максимальная длина строки внутри выражения

Датасет и метрики моделей

- 2100 объектов полученных из проекта с открытым кодом guava
- Для тестирования cross-project были взяты еще два проекта – dbeaver и presto – с 2037 и 500 примеров соответственно
- Отсутствующие значения заменяются на -1
- **Precision** (точность) = $\frac{TP}{TP+FP}$, **Recall** (полнота) = $\frac{TP}{TP+FN}$
- **F-мера** = $2 * \frac{precision * recall}{precision + recall}$, то есть их среднее гармоническое

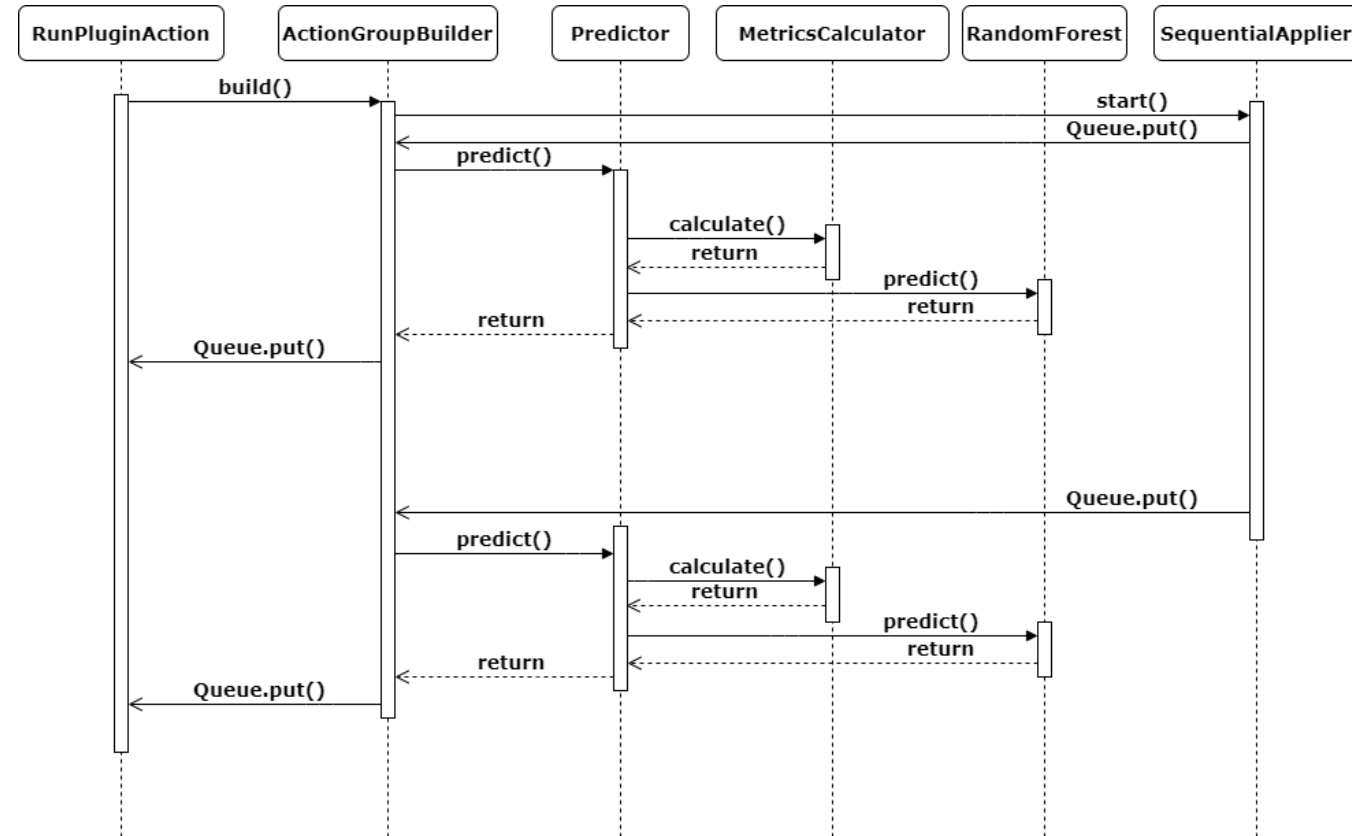
Эксперименты с моделями

- Реализации моделей были взяты из библиотеки **Weka**
- В результате была выбрана модель **Random Forest**

	Внутри проекта		Cross-project dbeaver		Cross-project presto	
Модель	Precision	F-мера	Precision	F-мера	Precision	F-мера
Random Forest	0.87	0.86	0.86	0.80	0.67	0.69
J48	0.87	0.87	0.80	0.79	0.70	0.70
AdaBoostM1	0.80	0.81	0.74	0.74	0.64	0.66
Naive Bayes	0.74	0.75	0.78	0.78	0.70	0.71

Прототип плагина

- Пользователь запускает плагин комбинацией клавиш
- Пользователю показывается динамически обновляемый список применимых цепочек
- Динамическая загрузка достигается за счет запуска отдельного потока для построения графа намерений и предсказания результатов



Апробация

- 5 опытных программистов (2-4 года работы) использовали плагин в течение недели
- 80% положительно оценили качество советов
- 60% хотели бы использовать плагин в дальнейшем
- Была отмечена хорошая работа плагина на строках и удобство механизма предпросмотра
- Среди недостатков было отмечено малое количество ситуаций, в которых плагин предлагал рефакторинг

Демонстрация прототипа

- Работа плагина

```
if (connectedNum > 0) {  
    message.insert( offset: 0, str: "Connections reopened: " + connectedNum + " (of " + totalNum + ")");  
} else if (message.length() == 0) {  
    message.insert( offset: 0, str: "All connections (" + totalNum + ") are alive!");  
}  
if (error != null) {  
    UIUtils.showErrorDialog(  
        shell,  
        "Invalidate data source [" + context.getDataSource() + "]: " +  
        "Error while connecting to the datasource", // +  
        error);  
}
```

Choose Intention

- Replace string concatenation with String.format()
- Replace string concatenation with String.format(), Put arguments on separate lines
- Replace concatenation with formatted output
- Replace concatenation with formatted output, Put arguments on separate lines
- Put arguments on separate lines
- Put arguments on separate lines, Replace concatenation with formatted output
- Put arguments on separate lines, Replace concatenation with formatted output, Put arguments on separate lines

Press Ctrl+Shift+I to open preview



```
if (connectedNum > 0) {  
    message.insert( offset: 0, String.format("Connections reopened: %d (of %d)",  
        connectedNum,  
        totalNum));  
} else if (message.length() == 0) {  
    message.insert( offset: 0, str: "All connections (" + totalNum + ") are alive!");  
}
```

Демонстрация прототипа

- Интеграция с механизмом предпросмотра намерений

Choose Intention

- Replace string concatenation with `String.format()`
- Replace string concatenation with `String.format()`, Put arguments on separate lines**
- Replace string concatenation with `String.format()`, Put arguments on separate lines, Put arguments on one line
- Replace concatenation with formatted output
- Replace concatenation with formatted output, Put arguments on separate lines
- Put arguments on separate lines
- Put arguments on separate lines, Replace concatenation with formatted output
- Put arguments on separate lines, Replace concatenation with formatted output, Put arguments on separate lines
- Put arguments on separate lines, Put arguments on one line
- Put arguments on separate lines, Put arguments on one line, Replace concatenation with formatted output
- Press Escape to hide preview

```
104 message.insert(0, String.format("Connections reopened: %d (of %d)",
105     connectedNum,
106     totalNum));
```

Результаты

Результаты:

- Создан инструмент для извлечения вариантов кода.
- Создан набор метрик кода, на основании которых модель будет принимать решение.
- Проведена серия экспериментов с моделями и выбрана модель на основе деревьев решений.
- Создан прототип плагина, использующий полученную модель.
- Проведена апробация плагина на пользователях. Получены в целом положительные результаты.

Репозитории:

- <https://github.com/SacredArrow/idea-intentions-plugin>
- <https://github.com/SacredArrow/usableIntentionsPlugin>