



Возможности среды разработки для анализа, модификации и ассистирования при написании кода на Python

Отчёт по учебной практике в форме «Производственное задание»

Кузнецов Илья Александрович
Группа 24.M71-ММ

Научный руководитель:
профессор кафедры Системного программирования, д.т.н. Д. В. Кознов
Консультант:
ведущий инженер ключевых проектов ООО «Техкомпания Хуавэй» Н. В. Тропин

Санкт-Петербургский государственный университет
Программная инженерия

- Saint-Petersburg Research Center (SRC)
- SRC IDE – мультязыковая среда разработки, основной компонент семейства продуктов SRC¹
- SRC IDE поддерживала Java
- Для Python модель кода была в разработке и уже позволяла приступить к реализации основных возможностей языкового сервера и клиента, однако требовала доработки и стабилизации базовых интерфейсов

¹ [Повторно используемая инфраструктура мультязыковой среды разработки для языка Python // Кузнецов И. А., 2023 // СПбГУ](#)

Цель и задачи

Цель: реализация с использованием модели кода основных возможностей SRC IDE для Python: анализ кода, действий для исправления проблем и реструктуризации кода, а также ассистентов для написания кода

Задачи выпускной квалификационной работы:

- провести обзор моделей кода и возможностей в инструментах разработки для языка Python
- реализовать основные возможности SRC IDE для языка Python с использованием модели кода
- провести многостороннее тестирование разработанного решения, создав соответствующую инфраструктуру
- настроить многосторонний мониторинг разработанного решения, создав соответствующую инфраструктуру
- выполнить апробацию разработанного решения с помощью мониторинга и пользователей SRC IDE для Python

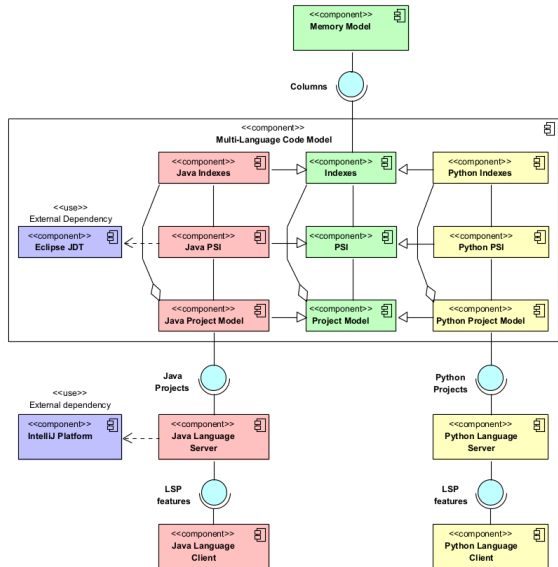
- Статические анализаторы – предлагают долгий, но наиболее точный анализ кода на Python
 - MyPy, PyRight
- Среды разработки – предлагают быстрый и точный анализ кода, исправления обнаруженных проблем и действия по реструктуризации кода, а также ассистируют при написании кода на Python
 - SRC IDE, PyCharm, VSCode (Pylance), Cursor

- Анализы (инспекции, диагностики) – обнаруживают и интерпретируют проблемы в коде, определяя исправления
- Исправления (фиксы) – это действия по модификации кода для устранения проблем, найденных инспекциями
- Реструктуризации (рефакторинги) – это действия по модификации кода для изменения его структуры, абстракции или упрощения
- Ассистенты (помощники) – предоставляют детальную информацию о коде и позволяют взаимодействовать с LLM для объяснения кода и применения рекомендованных модификаций

Обзор предметной области

Архитектура мультязыковой SRC IDE¹:

- Memory Model
 - Indexes
- Python Code Model
 - AST, PSI
 - Resolve, Type Inference, IR
 - Project Model
- Python Language Server
 - Features
- Python Language Client
 - Features
 - UI



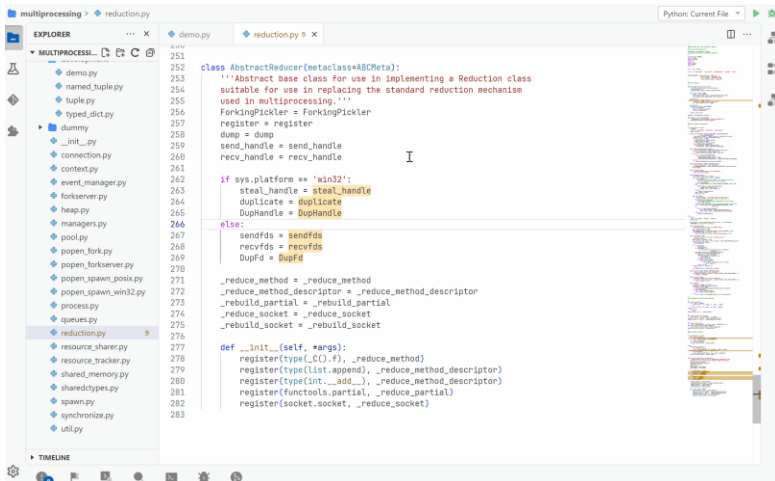
Обзор предметной области

- Производительность и точность возможностей инструментов разработки напрямую зависят от модели кода
- Качество и полнота возможностей инструментов разработки определяется соответствием спецификациям Python и библиотек, покрытием различных сценариев как в правильном, так и в неправильном коде, а также дизайном UX и UI

	Pylance (PyRight)	PyCharm	SRC IDE
Flask (5072 QR)	32.96 %	29.20 %	39.33 %
Jedi (10884 QR)	32.77 %	45.61 %	60.63 %
PyTorch (599914 QR)		54.29 %	61.36 %
Tensorflow (491977 QR)		65.96 %	84.09 %
Django (202578 QR)	28.75 %	59.16 %	76.08 %
Numpy (85004 QR)		61.51 %	87.64 %
ToolBench (2913 QR)	57.56 %	62.92 %	79.71 %
SciPy (123591 QR)		32.37 %	85.63 %
Scikit-learn (90175 QR)		33.78 %	69.52 %

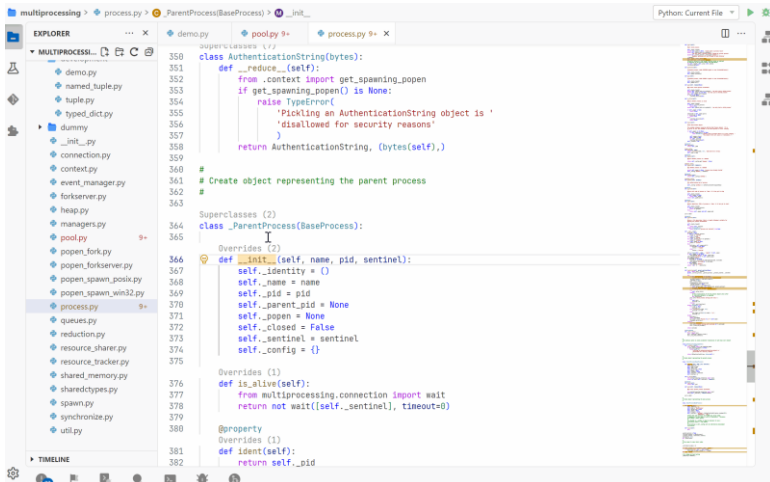
Ход работы: инспекции

- Реализовано более 60-ти различных инспекций
- Некоторые из наиболее важных: Syntax, Resolve (Import, Simple, Qualified), Expected Types, Unused, Redeclaration, Shadowing, Version Compatibility, ...



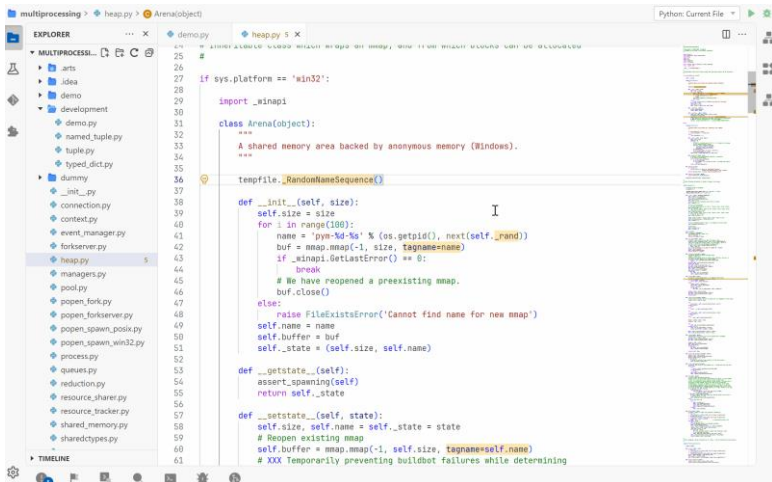
Ход работы: исправления

- Реализовано более 40-ка различных исправлений
- Некоторые из наиболее важных: Install, Import, Create, Remove Class, Function, Field, Variable, ...



Ход работы: реструктуризации

- Реализовано более 10-ти различных реструктуризаций
- Некоторые из наиболее важных: Rename, Rearrange, Inline, Extract Variable, Method, ...

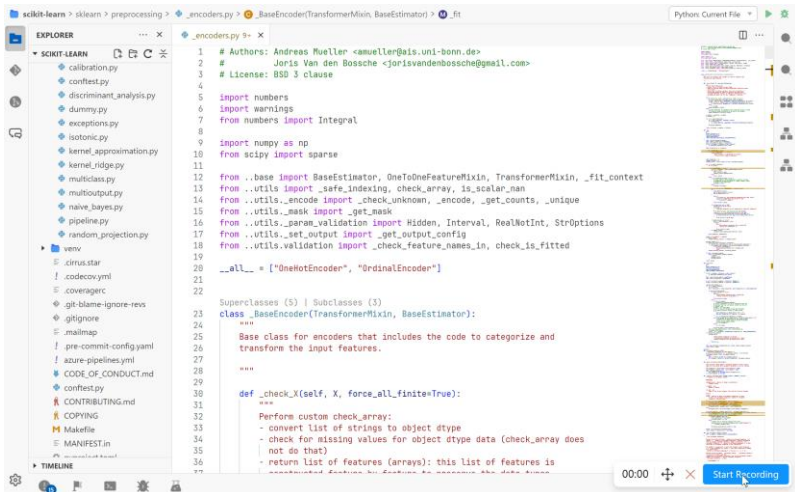


```
40 multiprocessing > heap.py > Arena(object) Python: Current File
EXPLORER
MULTIPROCESSING...
arts
.idea
demo
development
demo.py
named_tuple.py
tuple.py
typed_dict.py
dummy
_init_.py
connection.py
context.py
event_manager.py
forkserver.py
heap.py
managers.py
pool.py
popen_fork.py
popen_forkserver.py
popen_spawn_posix.py
popen_spawn_win32.py
process.py
queues.py
reduction.py
resource_sharer.py
resource_tracker.py
shared_memory.py
sharedtypes.py
TIMELINE

demo.py
25
26
27 if sys.platform == 'win32':
28
29 import _winapi
30
31 class Arena(object):
32 """
33 A shared memory area backed by anonymous memory (Windows).
34 """
35
36 tempfile._RandomNameSequence()
37
38 def __init__(self, size):
39     self.size = size
40     for i in range(100):
41         name = 'pym-%d-%s' % (os.getpid(), next(self._rand))
42         buf = mmap.mmap(-1, size, tagname=name)
43         if _winapi.GetLastError() == 0:
44             break
45         # We have reopened a preexisting mmap.
46         buf.close()
47     else:
48         raise FileExistsError('Cannot find name for new mmap')
49     self.name = name
50     self.buffer = buf
51     self._state = (self.size, self.name)
52
53 def __getstate__(self):
54     assert _spawning(self)
55     return self._state
56
57 def __setstate__(self, state):
58     self.size, self.name = self._state = state
59     # Reopen existing mmap
60     self.buffer = mmap.mmap(-1, self.size, tagname=self.name)
61     # XXX Temporarily preventing buildbot failures while determining
```

Ход работы: ассистенты

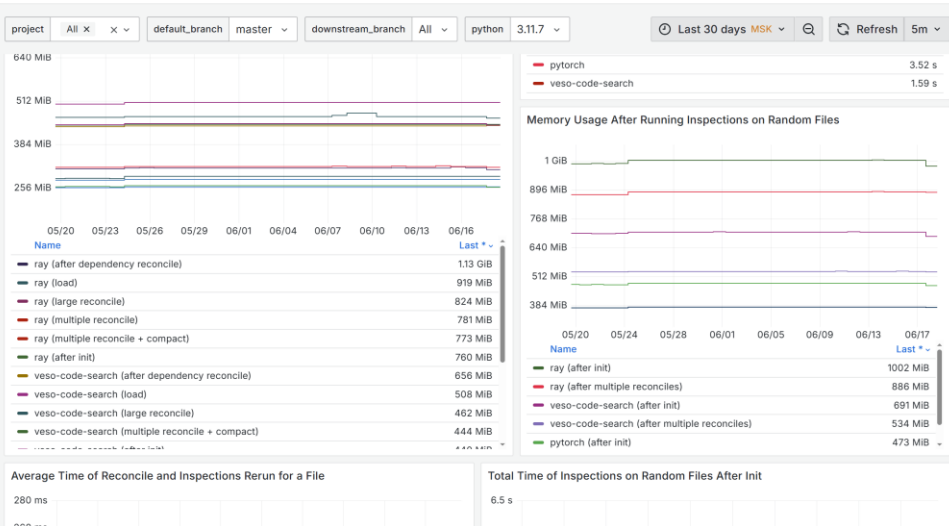
- Реализовано 3 ассистента: Hover, Signature Help, Inlay Hints
- AI Chat планируется интегрировать с другими возможностями среды разработки



- Модульное
- Интеграционное
- Регрессионное
- Комплексное

Ход работы: мониторинг

- Ведётся мониторинг производительности, улучшений и регрессий в состоянии разработанного решения на основе VictoriaMetrics и Grafana



Заключение

Для достижения **цели выпускной квалификационной работы** были получены следующие **результаты**:

- рассмотрены модели кода и возможности в статических анализаторах – MyPy, PyRight, и средах разработки – SRC IDE, PyCharm, VSCode (Pylance), Cursor
- на основе модели кода были реализованы основные возможности SRC IDE для Python, а именно: анализы кода, действия для исправления проблем и реструктуризации кода, а также ассистенты для написания кода
- проведено модульное, интеграционное, регрессионное и комплексное тестирование разработанного решения, создана соответствующая инфраструктура
- настроен мониторинг производительности и точности разработанного решения, создана инфраструктура на основе базы данных VictoriaMetrics и системы визуализации Grafana
- выполнена апробация разработанного решения на основе мониторинга, а также отзывов, сценариев и обнаруженных проблем у пользователей SRC IDE для Python