УНИВЕРСИТЕТ СИНЕРГИЯ

ПРИНЦИПЫ РАБОТЫ В СРЕДЕ ПРОГРАММИРОВАНИЯ PYCHARM

Занятие №5, МОДУЛЬ 1

Основы PyCharm, особенности данной среды разработки

Цель занятия: Изучение основ среды PyCharm.

Глоссарий

1. **Среда разработки –** система, в которой реализуется процесс создания и конфигурирования приложений, которые могут использоваться в различных областях.

2. **Приложение** - код, состоящий из одного или нескольких системных файлов, который вместе образуют единую систему.

1. Что такое **кодирование**?
2. Какие компоненты есть у ПК?
3. Что такое **ввод/вывод данных**?
4. Какими бывают программы?
5. Что такое **программа**?

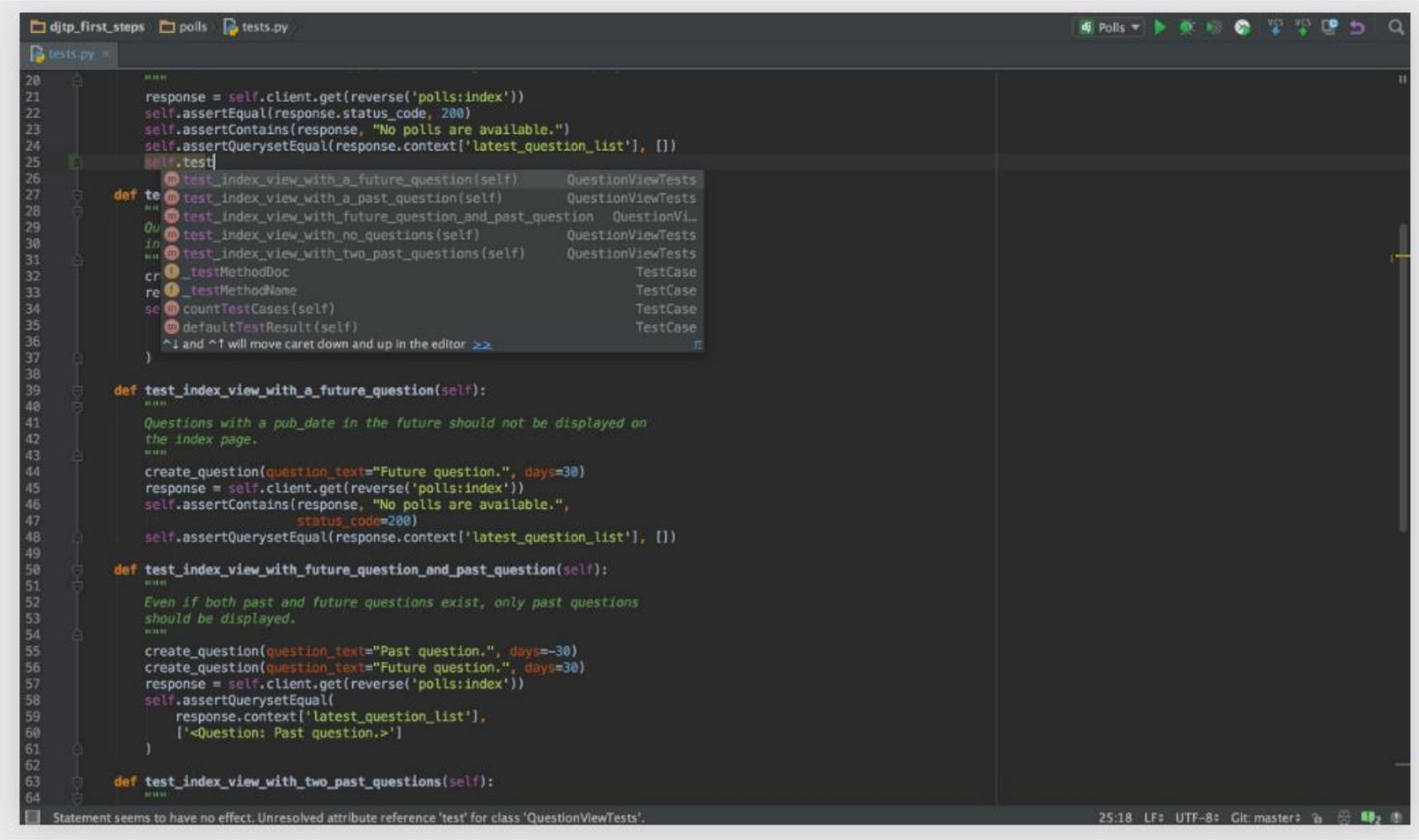
Среда PyCharm

Начать можно с того, что **PyCharm** обладает максимальной продуктивностью, а значит может позаботиться о рутинных задачах, чтобы архитектор-разработчик смог сосредоточиться на более важных вещах.

Работая в PyCharm, разработчики экономят время, так как для большинства задач не нужно отрывать руки от клавиатуры. В тоже время, PyCharm все знает о коде. Среда PyCharm имеет встроенный редактор, который может помочь пользователю в нахождение синтаксических, логических и прочих ошибок. Данная среда удобно подсвечивает ключевые слова, тела функций / циклов и прочее.

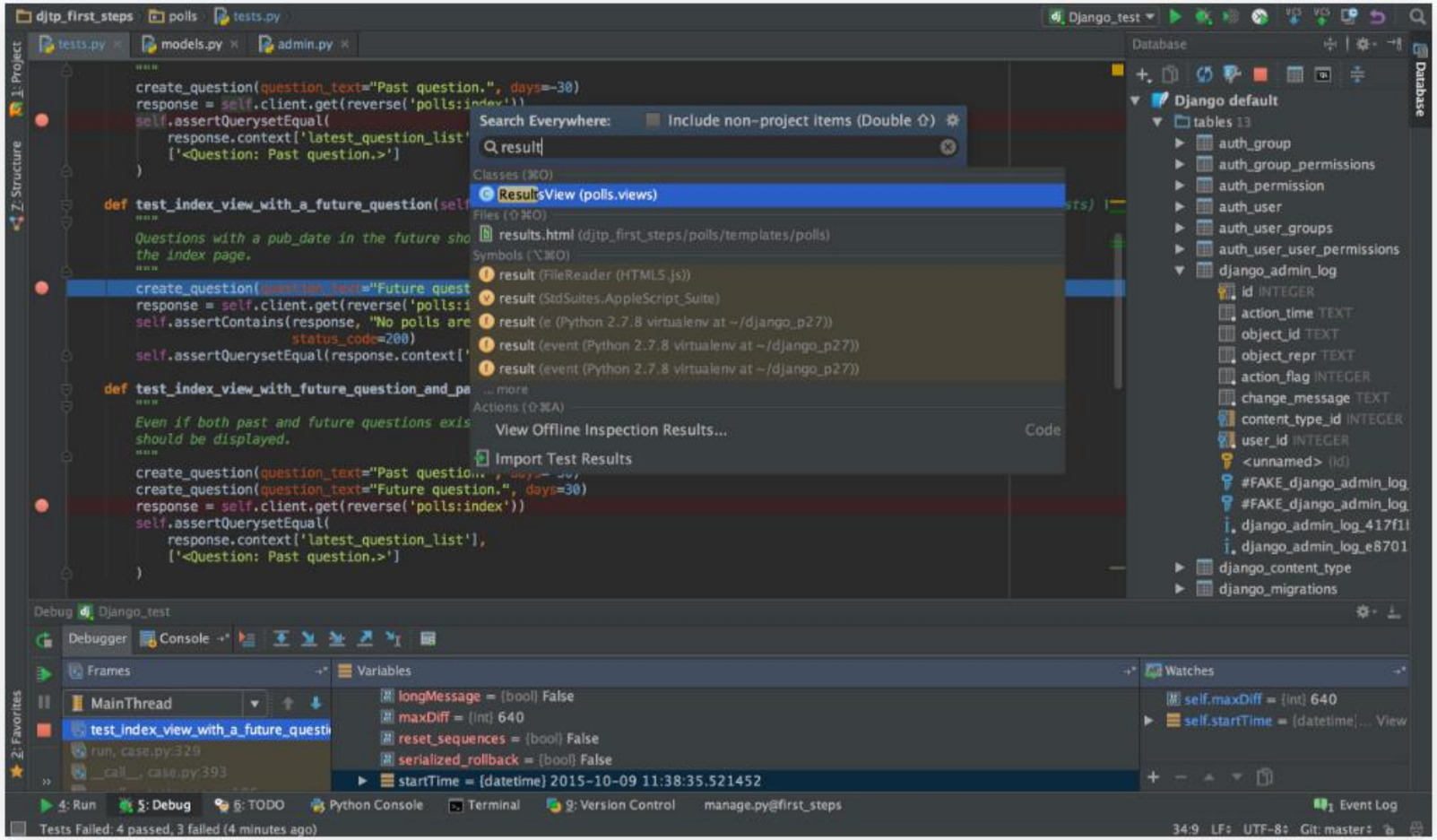
Умный редактор

Умный механизм анализа и редактирования кода обеспечивает точное автодополнение, поиск ошибок и быстрые исправления, удобную навигацию по коду и другие полезные функции:



Умный редактор

Все это дополняется повышением качества кода. PyCharm помогает писать красивый код, который легко поддерживать. IDE контролирует качество кода с помощью проверок соответствия требованиям PEP-8, умных рефакторингов и множества инспекций, а также оказывает помощь при тестировании:



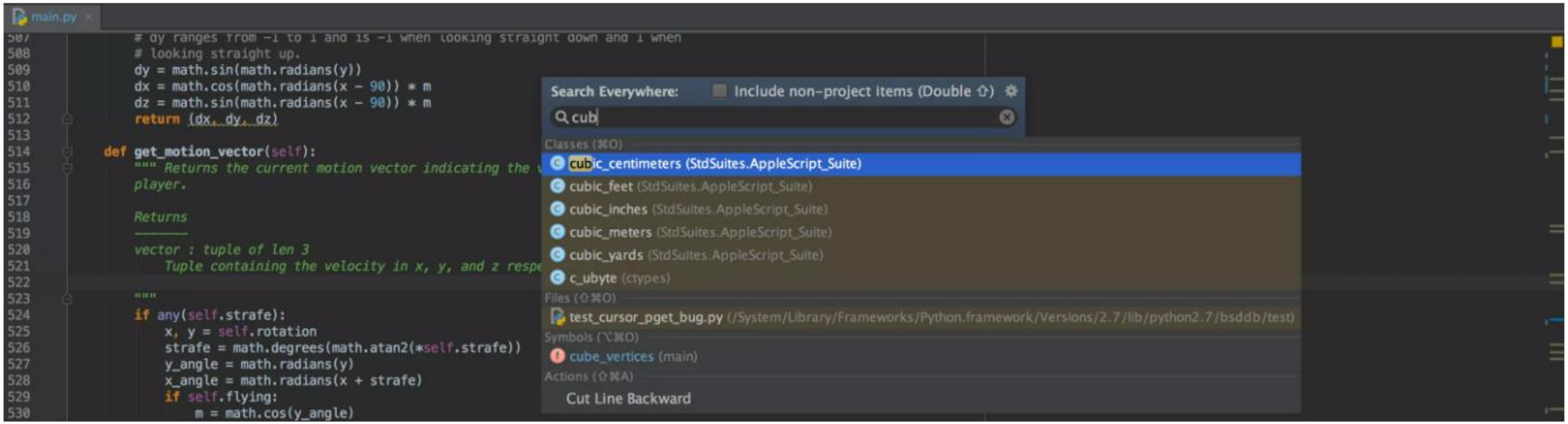
Умный редактор

С помощью умного детектора дубликатов PyCharm проверяет код на наличие дублирующихся фрагментов. IDE предложит список фрагментов, которые следует преобразовать, а рефакторинги помогут избавиться от повторяющегося кода.

Разработчик также может выбрать один из доступных стилей кода, разработанного для каждого из поддерживаемых языков. Помимо этого, в данной профессиональной среде разработки предусмотрены: сворачивание блоков кода, автоматическая расстановка скобок и кавычек, а также подсветка парных скобок и т.д.

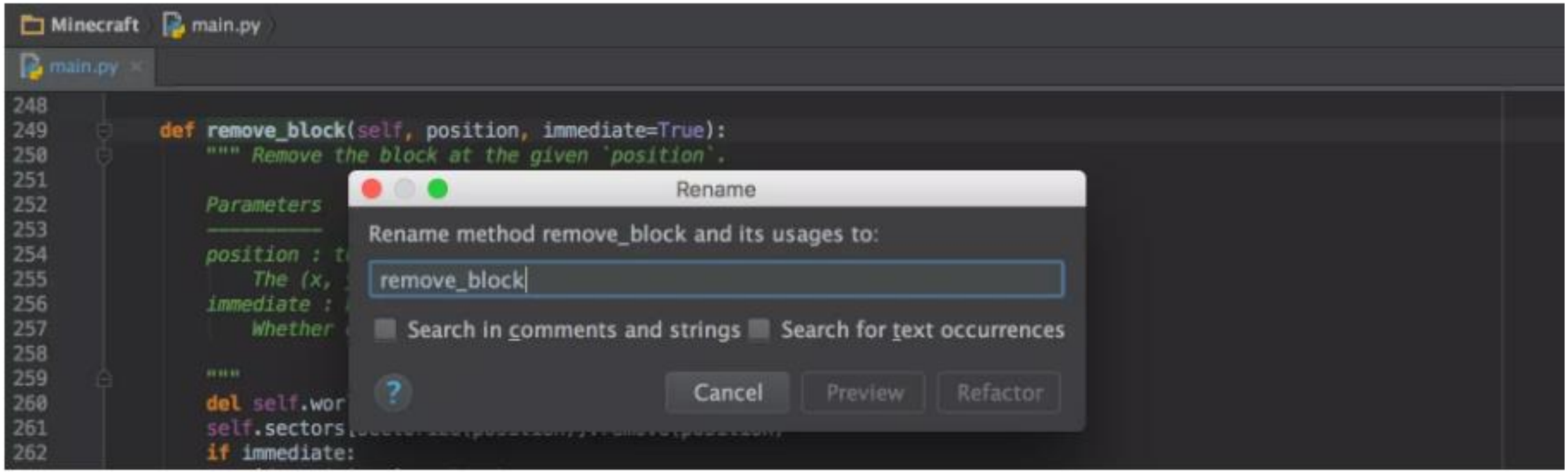
Удобная навигация

Функция *Search Everywhere* поможет архитекторам-разработчикам найти класс, файл, действие или элемент интерфейса IDE. Вызвать данную функцию можно двойным нажатием клавиши **Shift**, а затем вызвать поисковый запрос:



Быстрые и безопасные рефакторинги

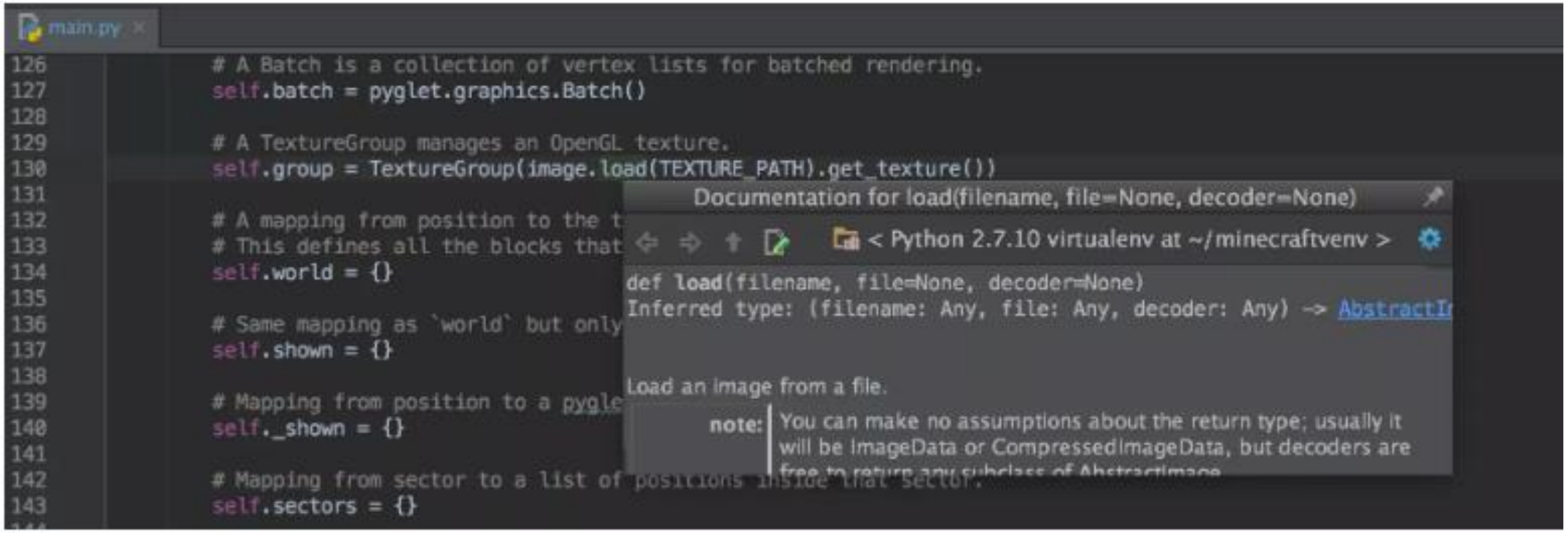
Рефакторинги **Rename** и **Move** применимы для файлов, функций, констант, классов, свойств, методов, параметров локальных и глобальных переменных:



Быстрые и безопасные рефакторинги

Для эффективной реорганизации кода доступны и другие рефакторинги: изменения сигнатуры, конвертация и пакет/модуль Python, создании функции верхнего уровня и т.д.

В разделе «**Документация**» разработчики могут просмотреть определения объектов и читать документацию, не покидая IDE:



Основные преимущества

Давайте еще раз выделим **основные преимущества** данной профессиональной среды:

- **Умный редактор**. В PyCharm легко редактировать код благодаря автодополнению и проверке кода, подсветке ошибок и быстрым исправлениям, а также автоматическому рефакторингу и удобной навигации.

- **Поддержка веб-фреймворков**.. PyCharm поддерживает современные фреймворки для веб-разработки: Django, Flask, Google App Engine, Pyramid и web2py.

Основные преимущества

- **Поддержка научных вычислений.** В PyCharm разработчики могут работать с Jupyter-ноутбуками, запускать команды в интерактивной консоли Python, подключать библиотеки Anaconda, а также работать с другими библиотеками для научных вычислений и анализа данных, включая Matplotlib и Numpy.

- **Разработка на разных языках.** Помимо Python, PyCharm поддерживает JavaScript, CoffeeScript, TypeScript, Cython, SQL, HTML/CSS, языки шаблонов, AngularJS, Node.js и другие технологии.

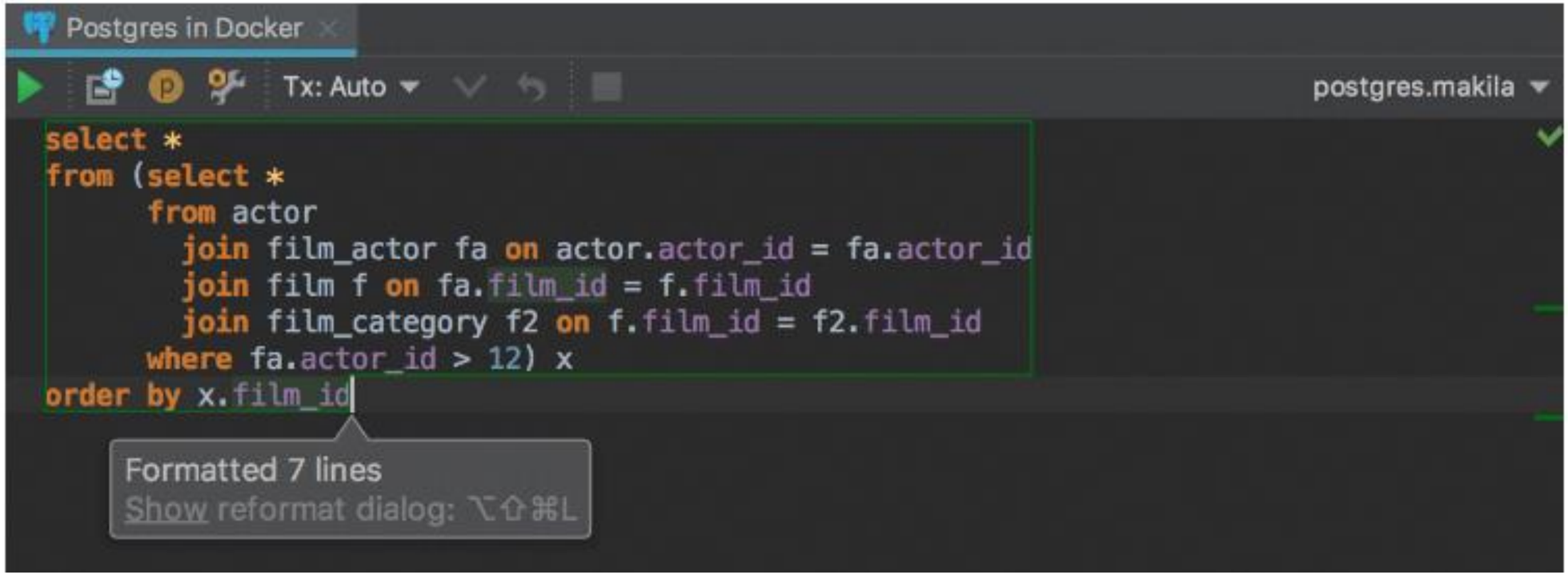
Основные преимущества

- **Возможности удаленной разработки.** В PyCharm разработчики могут запускать, отлаживать, тестировать и развертывать приложения на удаленных хостах или виртуальных машинах с помощью удаленных интерпретаторов, встроенного SSH-терминала и интеграции с Docker и Vagrant.

- **Встроенные инструменты для разработчиков.** PyCharm предоставляет встроенный отладчик и инструмент запуска тестов, профилировщик Python, встроенный терминал, инструмент для работы с базами данных и интеграцию с популярными системами контроля версий.

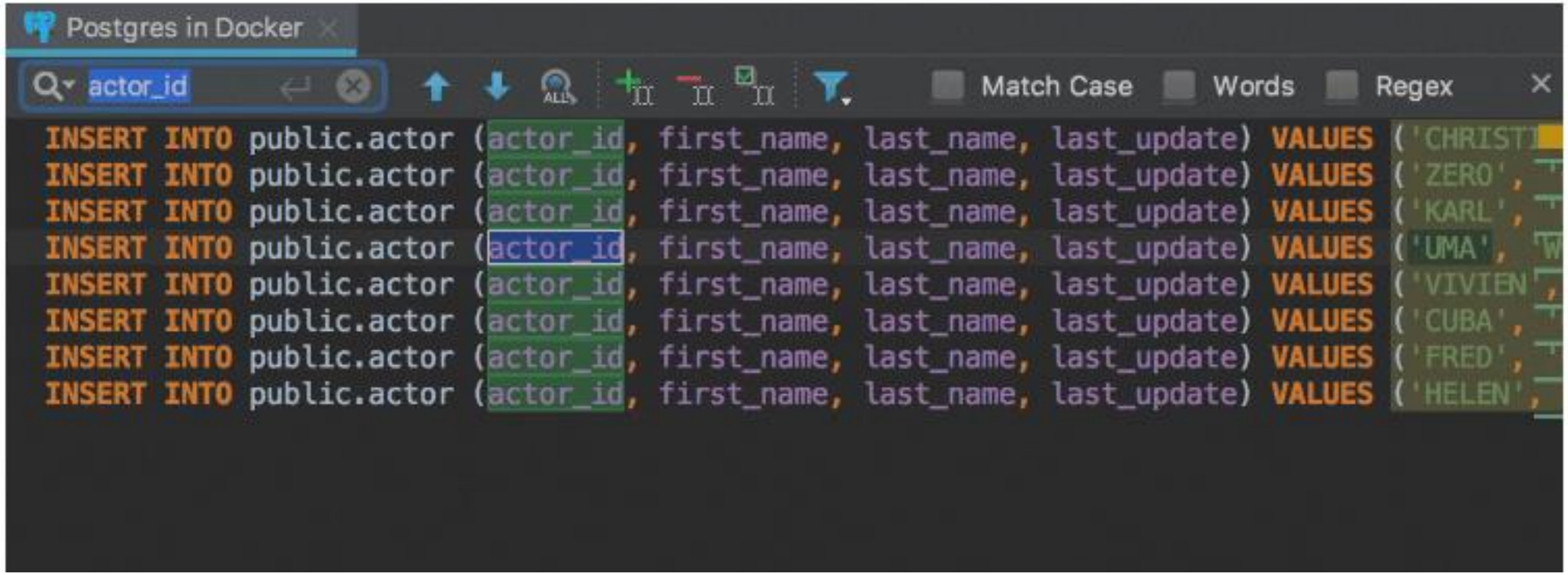
Форматирование кода. Мультикурсоры

DataGrip автоматически отформатирует *SQL*-код в соответствии с настройками стиля. Разработчикам не нужно тратить много времени на форматирование старого кода вручную - он может воспользоваться комбинацией клавиш **Ctrl + Alt + L**:



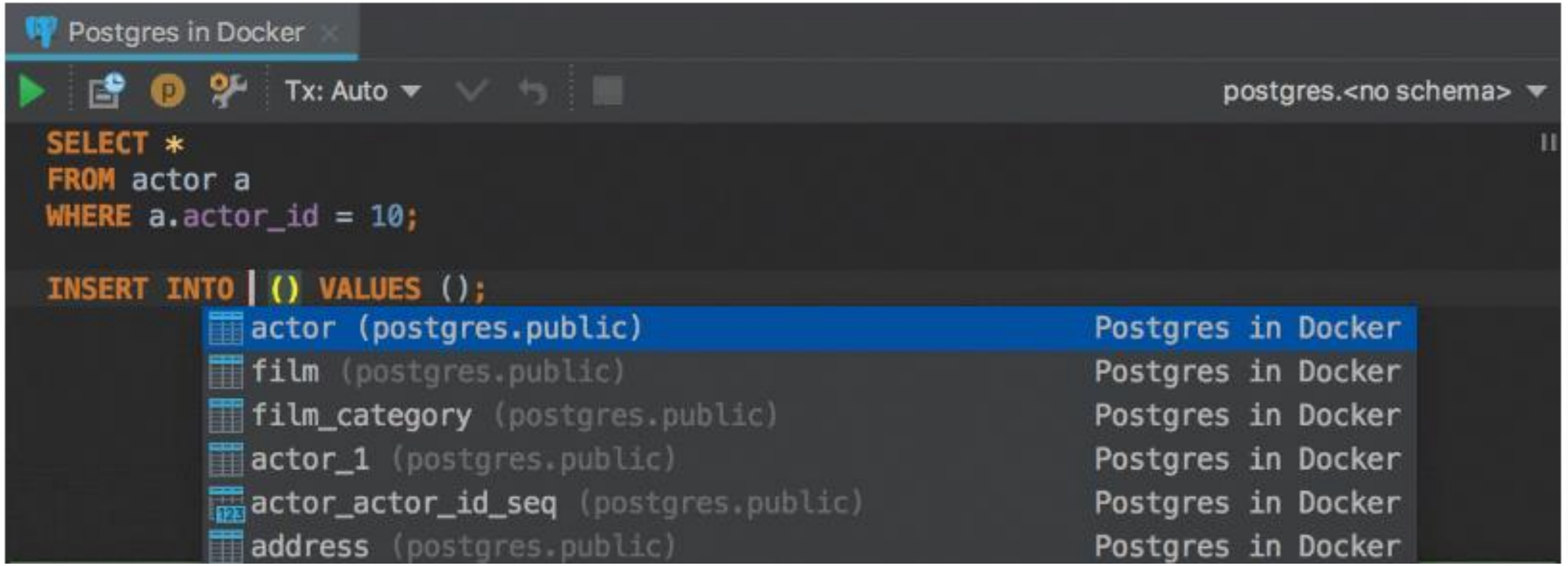
Форматирование кода.
Мультикурсоры

Мульти-курсоры - эффективный способ писать или менять запросы. Для того, чтобы расставить мульти-курсоры в *DataGrip*, нужно дважды нажать на клавишу **Ctrl** и, удерживая данную клавишу нажатой, использовать клавиши стрелок вверх/вниз. Также доступен текстовый поиск.
После нахождения нужного фрагмента текста, нужно нажать на иконку *Select All Occurrences*, которая находится справа от поля ввода на панели поиска:



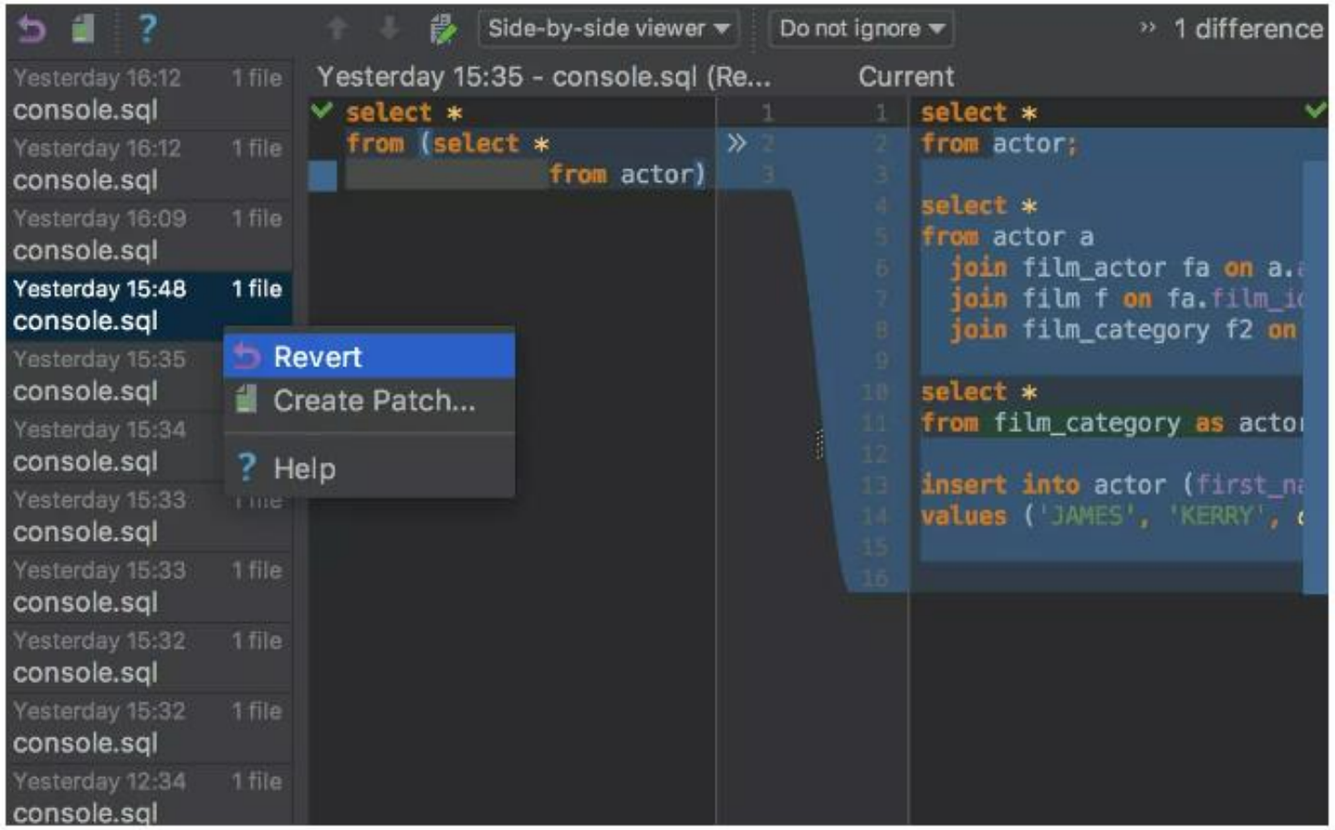
Шаблоны кода.
Локальная история

Архитекторы-разработчики могут использовать встроенные шаблоны кода, чтобы не писать все время одно и то же. Для отображения выпадающего списка шаблонов, нужно нажать комбинацию клавиш **Ctrl + J**:



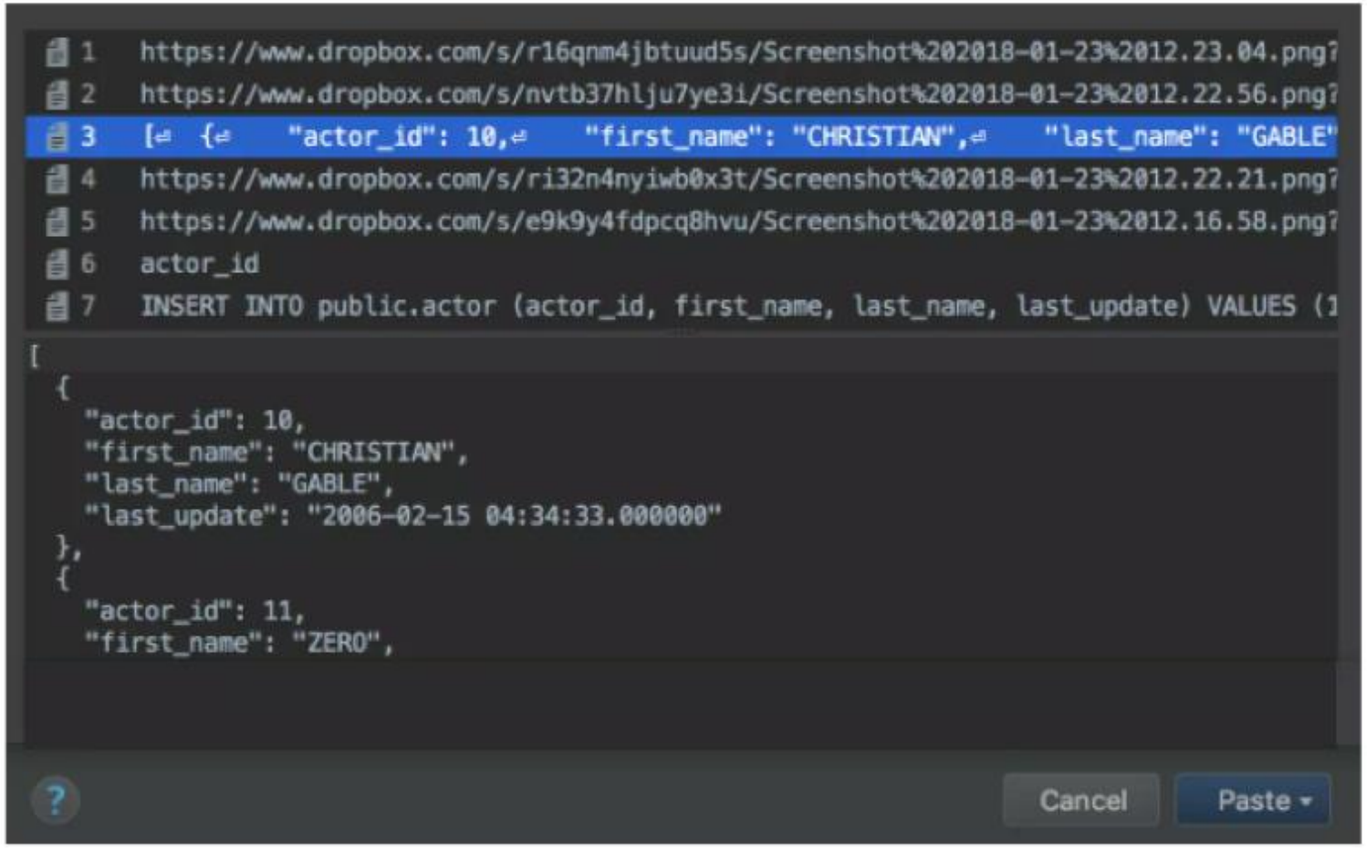
Шаблоны кода.
Локальная история

В той ситуации, если пользователю нужно сравнить версии,
то ему в этом может помочь интерфейс *Diff Viewer*:



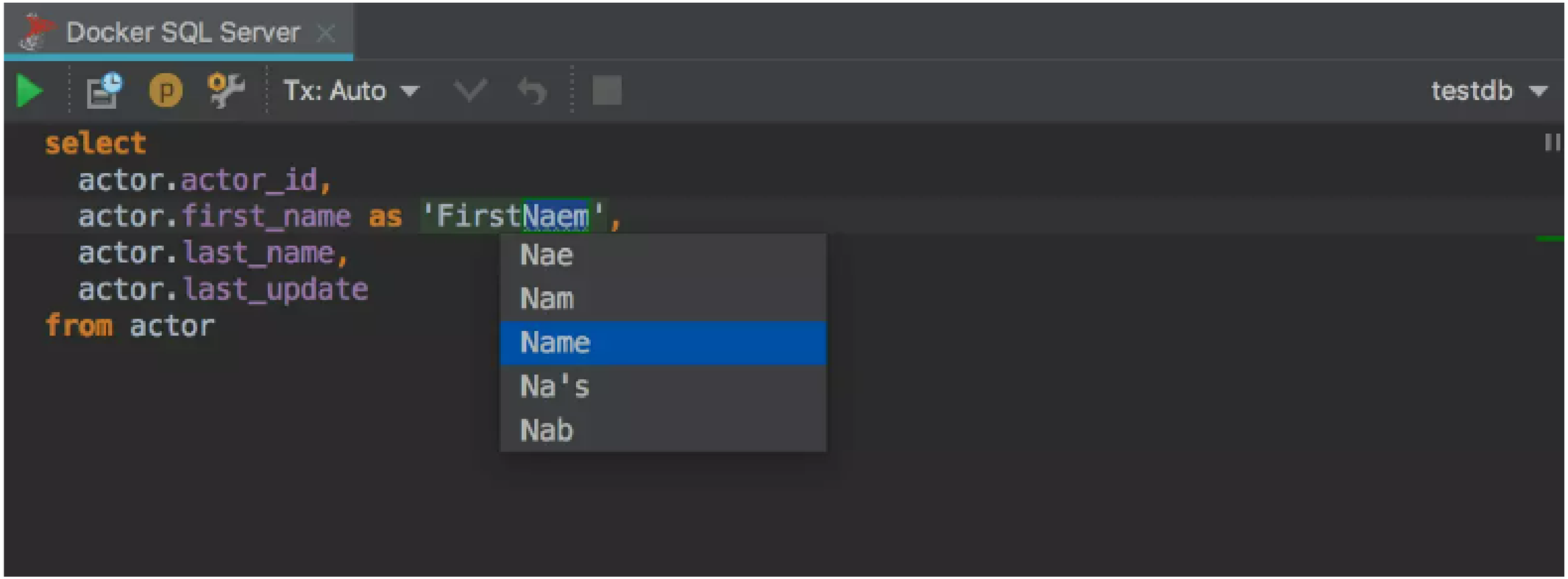
История буфера обмена. Проверка орфографии

В *DataGrip* хранится история буфера обмена. IDE записывает все, что пользователь копирует в буфер и хранит это, пока программа не будет закрыта окончательно:



История буфера обмена.
Проверка орфографии

DataGrip проверяет орфографию везде: в текстовых файлах и в данных таблиц.
Для того, чтобы исправить опечатку, пользователю нужно нажать **Alt + Enter**:



Языковые вставки.
Динамический SQL

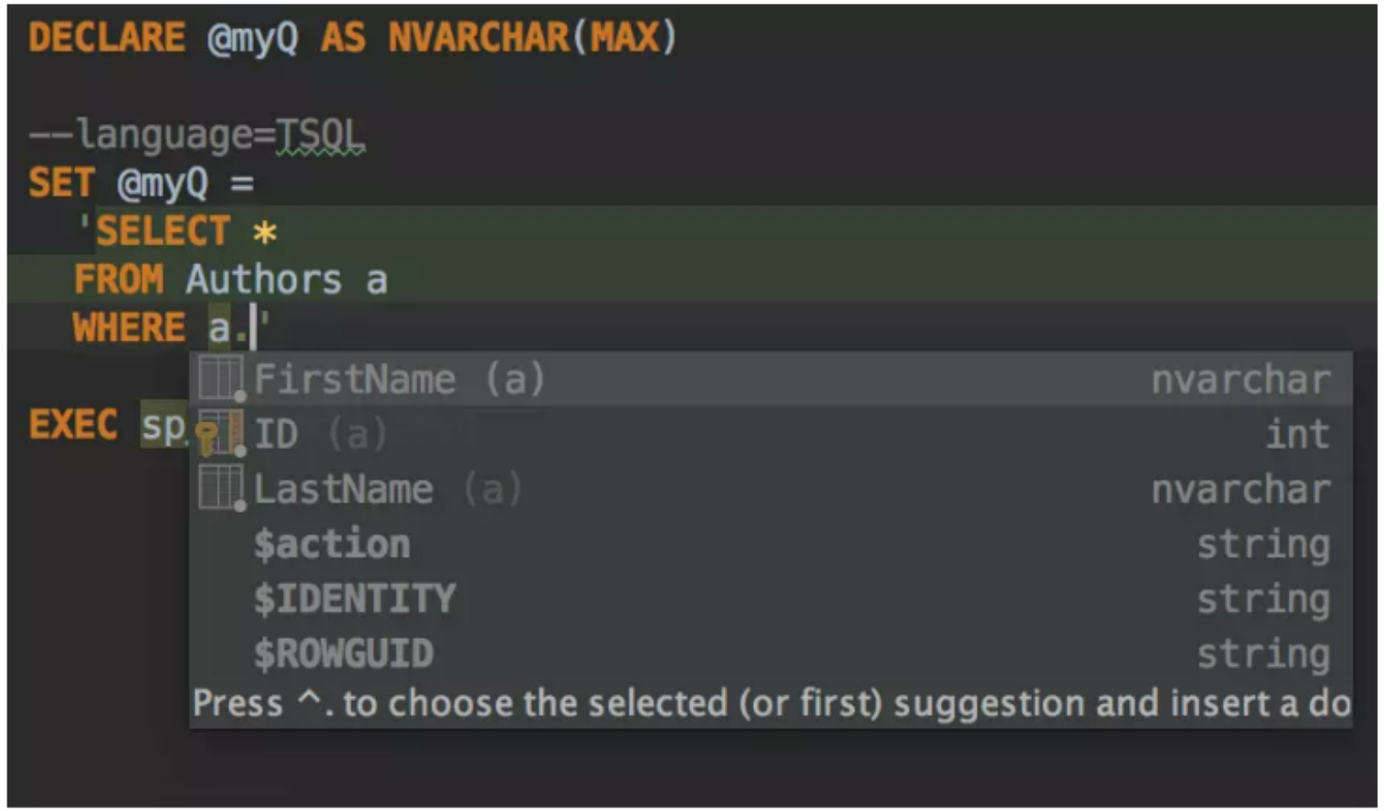
По умолчанию, *DataGrip* может определить только два формата литералов: *XML* и *JSON*.
Но можно и подсказать IDE, что внутри строки - код:

```
CREATE TABLE Simpsons
(
  json_data JSON
);

INSERT INTO Simpsons (json_data)
VALUES ('{
  "first_name": "Homer",
  "last_name": "Simpson",
  "age": "38",
  "address": {
    "street_address": "742 Evergreen Terrace",
    "city": "Springfield",
    "state": ""
  }
}');
```

Языковые вставки.
Динамический SQL

Для динамического SQL внутри строки, в это же время, заработают автодополнение, инспекции и другие привычные функции.
Посмотрим на реализацию динамического SQL в данной среде:



Языковые вставки.
Динамический SQL

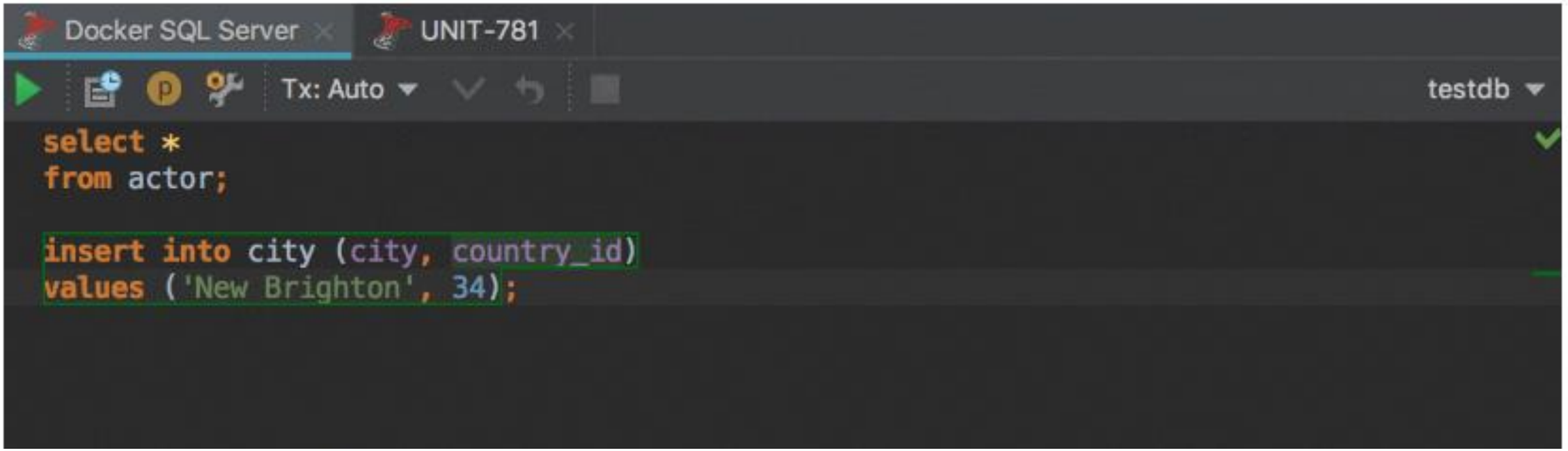
Как и в *Jupyter*, закомментировать код в *DataGrip* очень просто. Для этого нужно использовать клавиши для *построчного комментирования* или *блоками*, если это поддерживается SQL-диалектом:

```
SELECT *
FROM actor
  JOIN film_actor fa ON actor.actor_id = fa.actor_id
  JOIN actor a ON fa.actor_id = a.actor_id
--   JOIN actor a2 ON fa.actor_id = a2.actor_id
```

```
-- SELECT *
-- FROM actor
--   JOIN film_actor fa ON actor.actor_id = fa.actor_id
--   JOIN actor a ON fa.actor_id = a.actor_id
--   JOIN actor a2 ON fa.actor_id = a2.actor_id
```

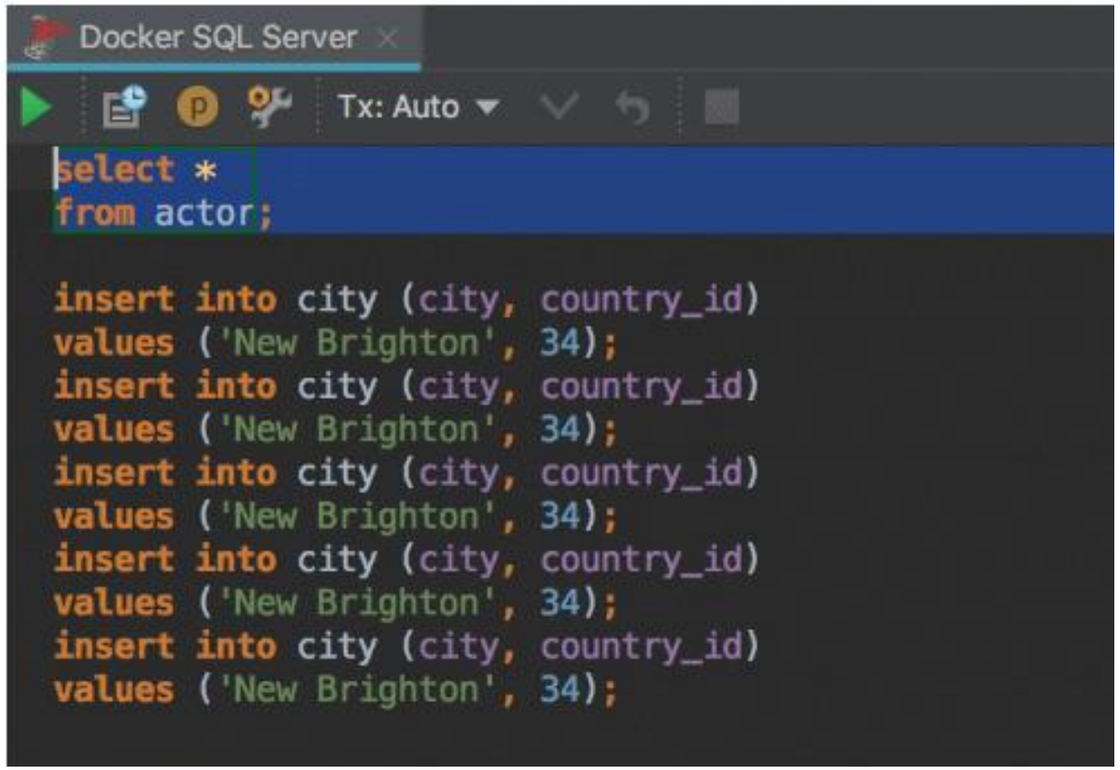
Языковые вставки.
Динамический SQL

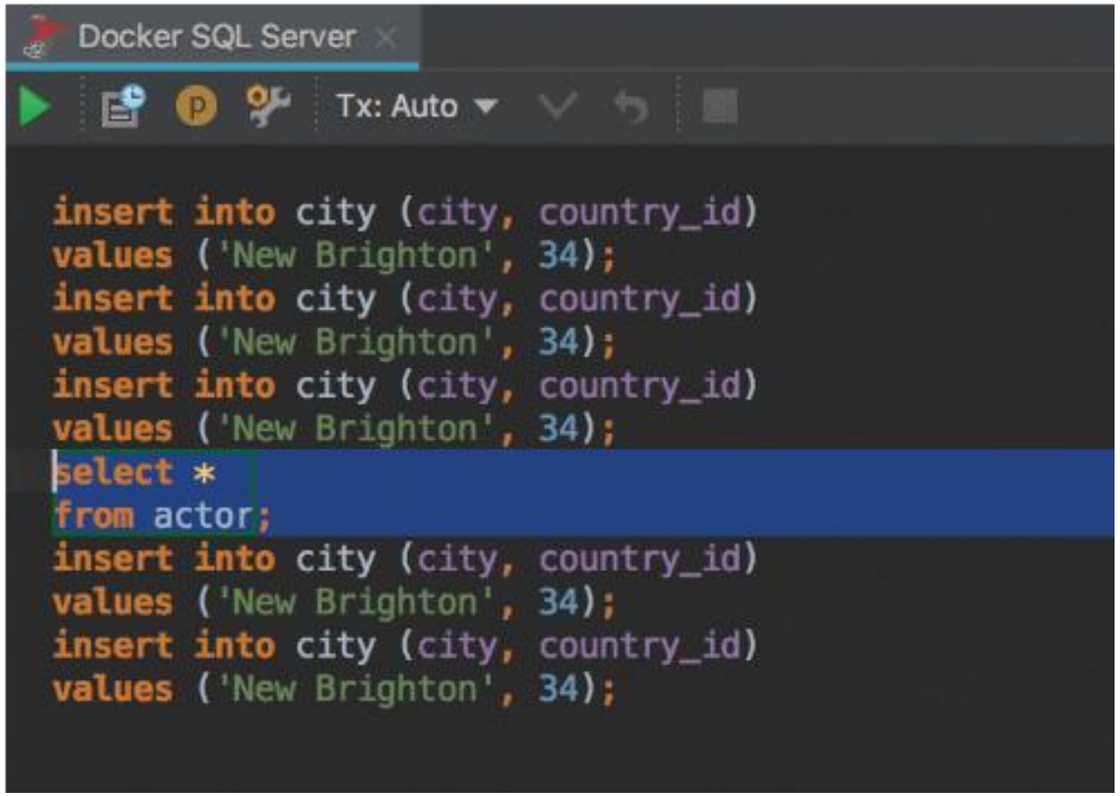
Кроме этого, в текстовом редакторе платформы **IntelliJ** встроена возможность навигации по коду.
Посмотрим на основные комбинации клавиш: «**Ctrl + Alt + <- / ->**» или «**Ctrl + Shift + Backspace**».
Посмотрим на пример работы с текстовым редактором платформы **IntelliJ**:



Языковые вставки.
Динамический SQL

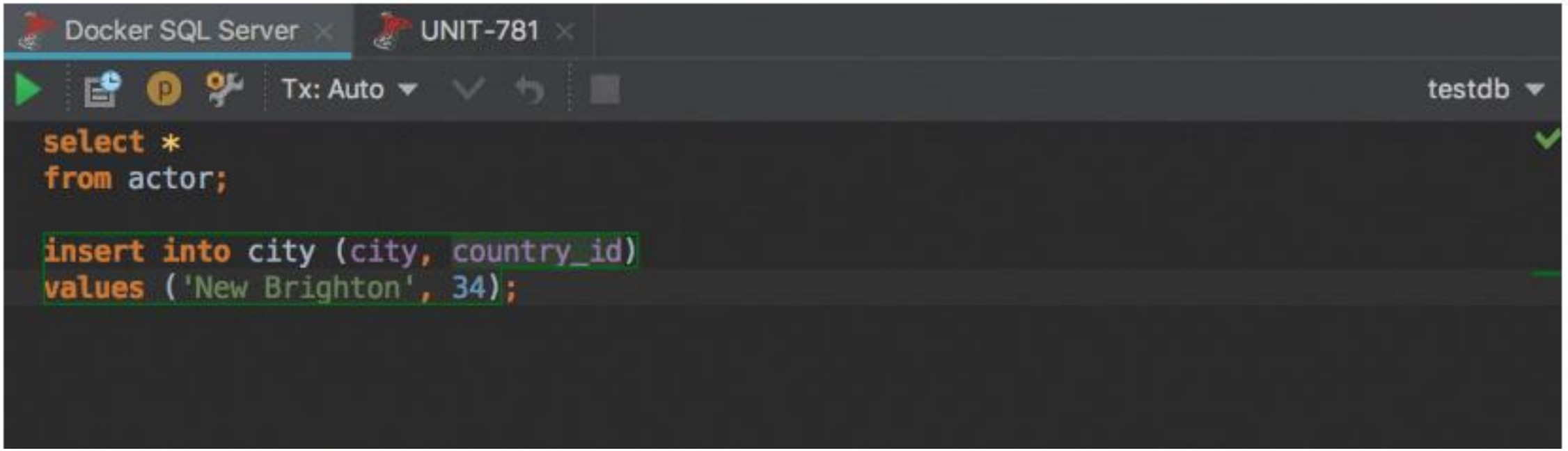
Данные операции работают и с частями кода:





Языковые вставки.
Динамический SQL

Кроме этого, в текстовом редакторе платформы **IntelliJ** встроена возможность навигации по коду.
Посмотрим на основные комбинации клавиш: «**Ctrl + Alt + <- / ->**» или «**Ctrl + Shift + Backspace**».
Посмотрим на пример работы с текстовым редактором платформы **IntelliJ**:



Основные отличия

Также рассмотрим **основные отличия** *Community* и *Pro* версий:

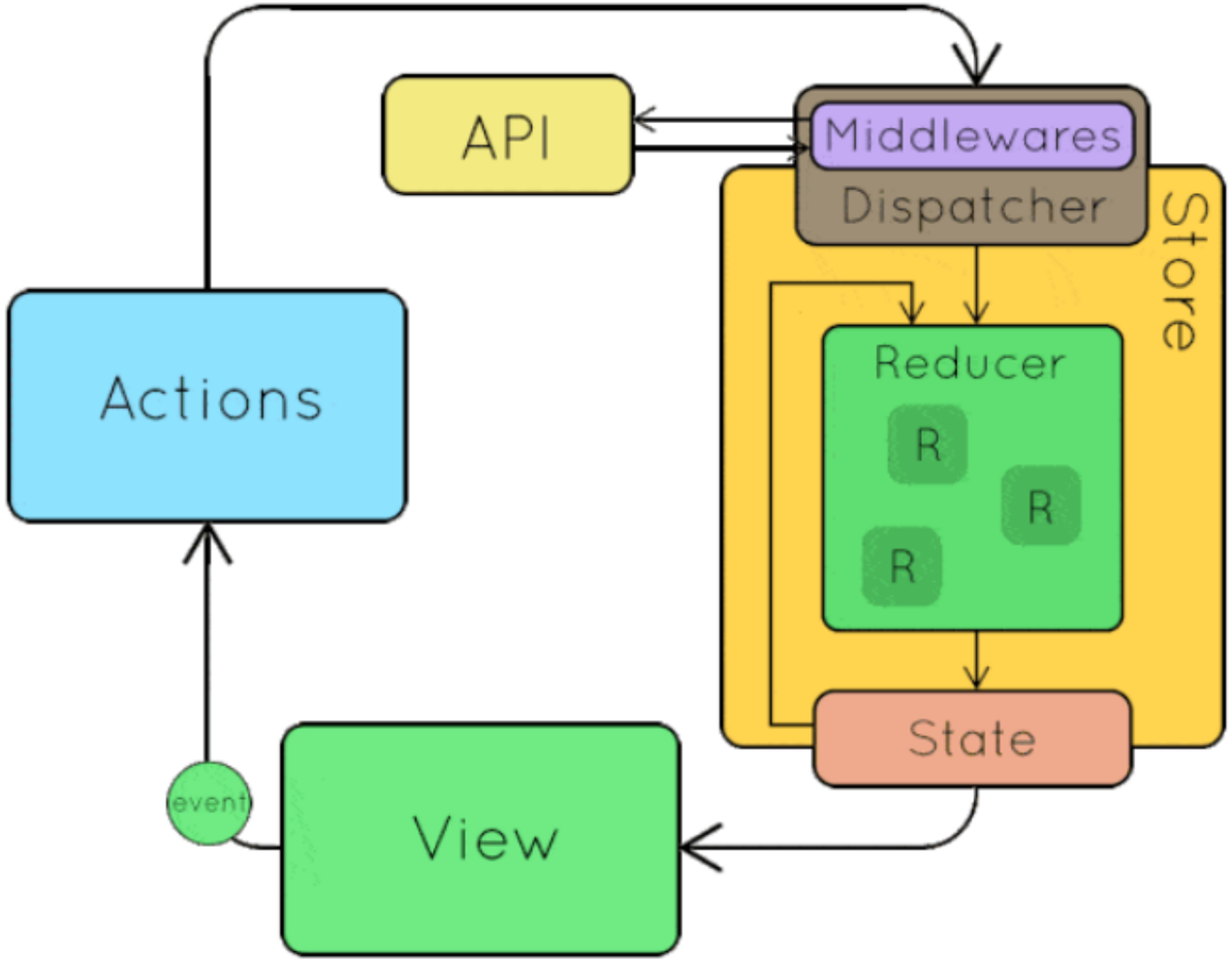
- *Community* не поддерживает *JavaScript*, *CSS* и другие веб-технологии. В данной версии можно работать только с языком *Python* и его вариациями, а также с *HTML*, *XML*, *JSON* и некоторыми другими форматами.

- То же самое можно сказать и о поддержке популярных фреймворков. **PyCharm** *Community Edition* поддерживает только фреймворки для *Python*, а *Pro*-версия - еще и *React*, *Angular* и другие инструменты веб-разработки, ориентированные на *JavaScript*.

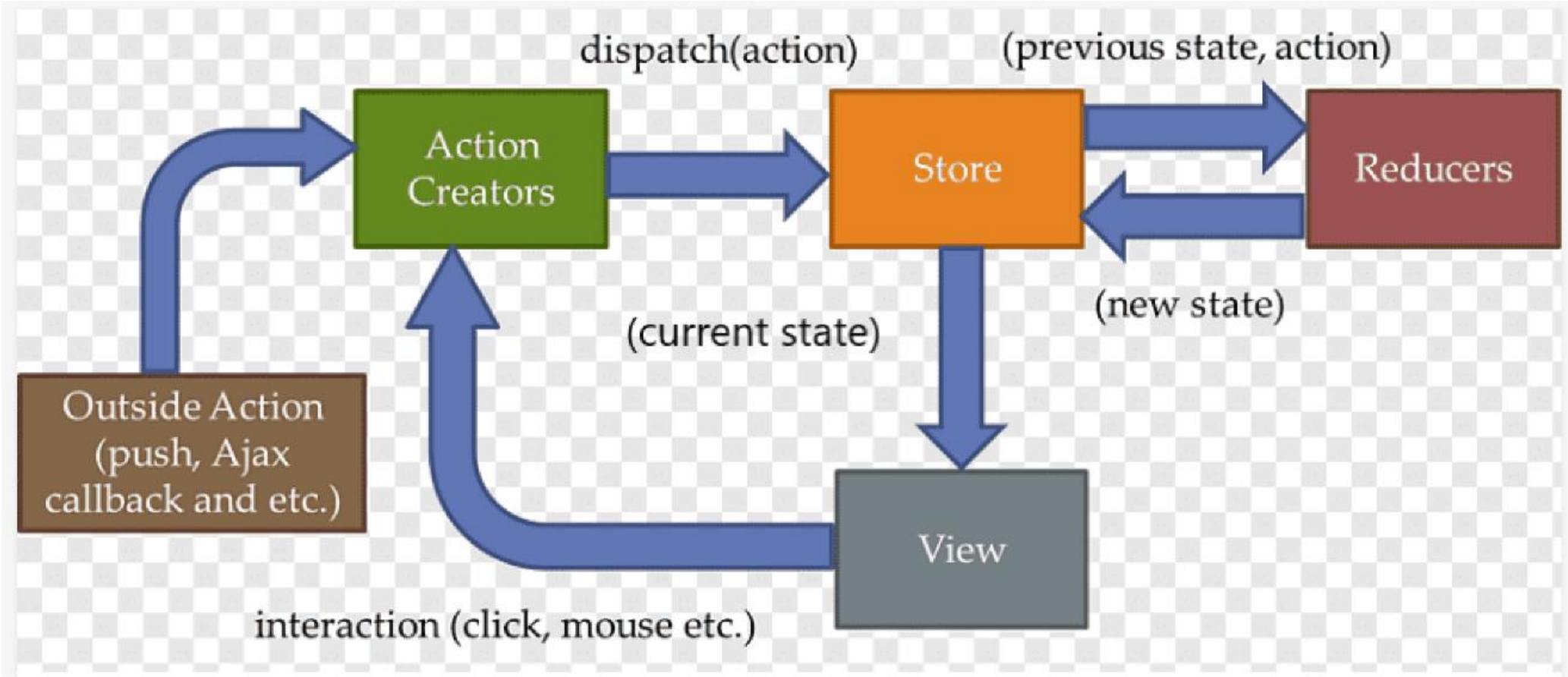
Основные отличия

- В *Community*-версии нет встроенных инструментов для работы с базами данных, в то время как в *Pro*-версии они есть.
- Инструменты для развертывания и контроля версий в *Community* устанавливаются отдельно как плагины, а в *Pro* они предустановлены.
- Возможности совместной разработки в бесплатной версии ограничены: не более трех человек и сессии не больше 30 минут.

Покажем как управлять состоянием
в *Angular* по мере росте простого
приложения для произвольной области:



Ниже приведена блок-схема данных **GitHub** на **React**

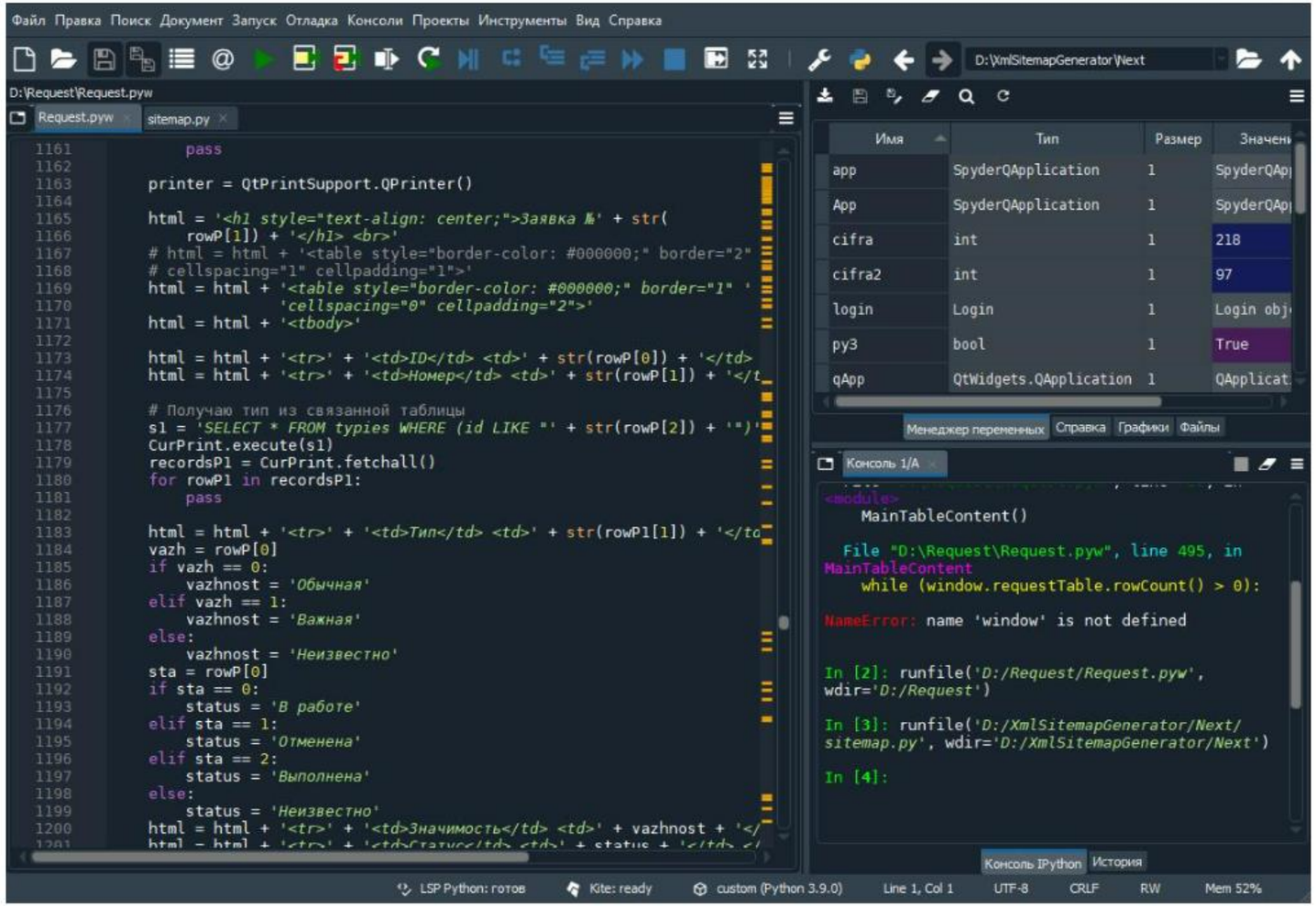


Full-stack разработка с *Python* - широкая область, где среда **PyCharm** может помочь:

- Многофункциональный *HTTP*-клиент для автоматизированного тестирования
- Поддержка фреймворка **web2py**: особый тип конфигурации *run/debug*, поддержка языка шаблонов, навигация между представлениями и шаблонами и шаблон проекта **web2py**
- Поддержка **Google App Engine** (**GAE**): настройка параметров GAE прямо в IDE, загрузка приложений из меню инструментов, просмотр файлов в логе, специальная консоль для запуска команд *appcfg.py* и шаблон проекта GAE
- Языковые вставки для встраивания поддержки SQL в строки на *Python* с автодополнением не только для команд SQL, но и для схемы заданного источника данных.

Введение в Spyder

Посмотрим на пример работы в *Spyder*:



УНИВЕРСИТЕТ СИНЕРГИЯ

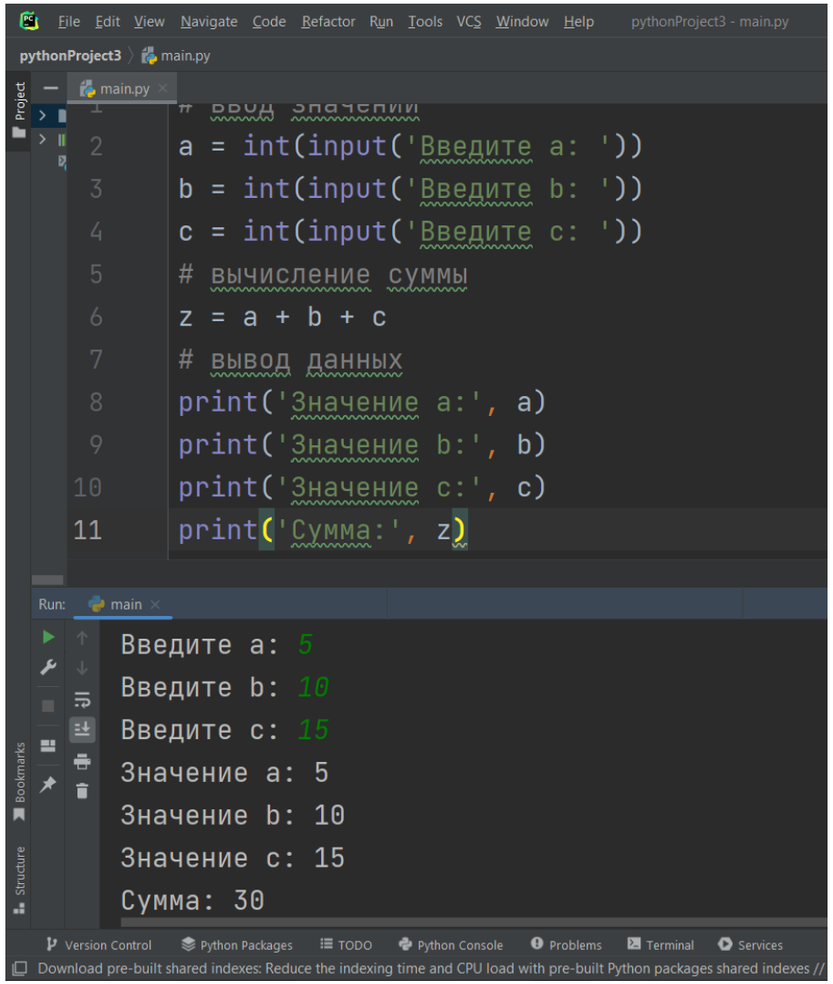
ПРАКТИЧЕСКИЕ ЗАДАЧИ

Задача 1

Создать программу в среде PyCharm, в которой вводятся с клавиатуры три переменные с типом данных **int**: **‘a’**, **‘b’** и **‘c’**. Затем в переменную **‘z’** записывается сумма значений данных переменных. Вывести на консоль значения введенных переменных и сумму.

Решение

Решение.
Напишем код для решения данной практической задачи и посмотрим на вывод:

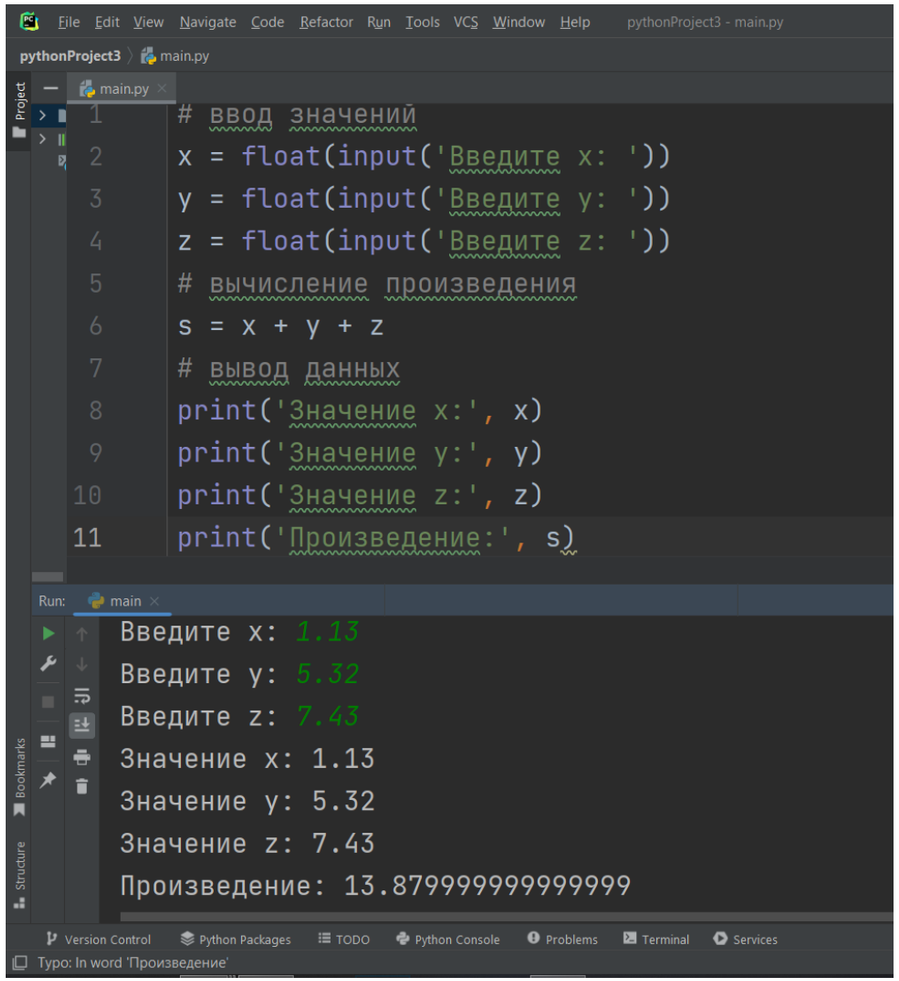


Задача 2

Создать программу в среде PyCharm, в которой вводятся с клавиатуры три переменные с типом данных **float**: **‘x’**, **‘y’** и **‘z’**. Затем в переменную **‘s’** записывается произведений значений данных переменных. Вывести на консоль значения введенных переменных и произведение.

Решение

Решение.
Для решения данной задачи понадобится следующий код:



1. Для чего используется *DataGrip*, при работе с приложениями?
2. Что такое **мульти-курсоры**?
3. Какую основную функцию выполняет операция *Select All Occurences*?
4. Что такое **редактор** и какими они бывают?
5. Какой этап разработки является самым посредственным в среде PyCharm?