**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное автономное образовательное учереждение высшего образования

Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Мегафакультет трансляционных информационных технологий

Факультет информационных технологий и программирования

Лабораторная работа № 1

По дисциплине «Инструментальные средства разработки ПО»

Разработка плагина для IDE

Выполнил студент группы №M32081

*Баязитова Диана Маратовна*

Проверил

*Приискалов Роман Андреевич*

*САНКТ-ПЕТЕРБУРГ*

*2021*

**IDE**

Интегрированная среда разработки, ИСP (англ. Integrated development environment — IDE), также единая среда разработки, ЕСР — комплекс программных средств, используемый программистами для разработки программного обеспечения (ПО).

Среда разработки включает в себя:

* текстовый редактор,
* транслятор (компилятор и/или интерпретатор),
* средства автоматизации сборки,
* отладчик.

Иногда содержит также средства для интеграции с системами управления версиями и разнообразные инструменты для упрощения конструирования графического интерфейса пользователя. Многие современные среды разработки также включают браузер классов, инспектор объектов и диаграмму иерархии классов — для использования при объектно-ориентированной разработке ПО.

**Intellij IDEA**

IntelliJ IDEA — интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python, разработанная компанией JetBrains.

Первая версия появилась в январе 2001 года и быстро приобрела популярность как первая среда для Java с широким набором интегрированных инструментов для рефакторинга, которые позволяли программистам быстро реорганизовывать исходные тексты программ. Дизайн среды ориентирован на продуктивность работы программистов, позволяя сконцентрироваться на функциональных задачах, в то время как IntelliJ IDEA берёт на себя выполнение рутинных операций.

Начиная с шестой версии продукта IntelliJ IDEA предоставляет интегрированный инструментарий для разработки графического пользовательского интерфейса. Среди прочих возможностей, среда хорошо совместима со многими популярными свободными инструментами разработчиков, такими как CVS, Subversion, Apache Ant, Maven и JUnit. В феврале 2007 года разработчики IntelliJ анонсировали раннюю версию плагина для поддержки программирования на языке Ruby.

Начиная с версии 9.0, среда доступна в двух редакциях: Community Edition и Ultimate Edition. Community Edition является полностью свободной версией, доступной под лицензией Apache 2.0, в ней реализована полная поддержка Java SE, Kotlin, Groovy, Scala, а также интеграция с наиболее популярными системами управления версиями. В редакции Ultimate Edition, доступной под коммерческой лицензией, реализована поддержка Java EE, UML-диаграмм, подсчёт покрытия кода, а также поддержка других систем управления версиями, языков и фреймворков.

Поддерживаемые языки:

* Java
* JavaScript
* CoffeeScript
* HTML/XHTML/HAML
* CSS/SASS/LESS
* XML/XSL/XPath
* YAML
* ActionScript/MXML
* Python
* Ruby
* Haxe
* Groovy
* Scala
* SQL
* PHP
* Kotlin
* Clojure
* Си
* C++
* Go
* Rust
* Dart

Каждый компонент IntelliJ IDEA создан для того, чтобы максимально повысить продуктивность разработки. Умный редактор кода в сочетании с эргономичным дизайном делают разработку не только эффективной, но и приятной.

После индексирования исходного кода IntelliJ IDEA предоставляет массу возможностей для быстрой и эффективной разработки: умное автодополнение, анализ кода в реальном времени и надежные рефакторинги.

Text

Description automatically generated

Все необходимые инструменты уже под рукой, а значит не нужно искать и устанавливать плагины для интеграции с системами контроля версий и поддержки популярных языков и фреймворков.

Graphical user interface, text

Description automatically generated

В то время как базовое автодополнение предлагает имена классов, методов, полей и ключевых слов в области видимости, умное автодополнение предлагает только те элементы кода, которые актуальны в текущем контексте.

Graphical user interface, text

Description automatically generated

Несмотря на то, что IntelliJ IDEA — в первую очередь IDE для Java, она понимает и предоставляет интеллектуальную помощь при написании кода на SQL, JPQL, HTML, JavaScript и многих других языках и позволяет редактировать код, написанный не на Java, внутри строковых литералов Java-кода.

Graphical user interface, text, application, chat or text message

Description automatically generated

IntelliJ IDEA анализирует однообразные задачи в процессе разработки и автоматизирует их, чтобы вы могли сосредоточиться на общей картине.

Работая над функциональностью и дизайном IDE, JetBrains позаботились о том, чтобы разработчика ничто не отвлекало, и сделали все возможное, чтобы устранить или минимизировать риск потери контекста.

IntelliJ IDEA понимает, что вы хотите сделать, и автоматически вызывает соответствующие инструменты.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

IntelliJ IDEA не только помогает писать код в редакторе, но и упрощает работу в целом. Вы сможете легко и быстро заполнить поле, найти элемент в списке, открыть нужное окно, поменять настройки и т.д.

**Разработка плагина для Intellij IDEA**

Для создания плагинов доступны три поддерживаемых рабочих процесса. Рекомендуемый рабочий процесс для новых проектов - использовать шаблон GitHub или использовать Gradle для создания всего с нуля. Старый рабочий процесс Plugin DevKit по-прежнему поддерживает существующие проекты.

IntelliJ IDEA поддерживает использование Gradle для автоматического создания новых проектов плагинов и автоматического выполнения всех необходимых настроек в build.gradle. Это также может быть использовано для преобразования существующего плагина в Gradle.

Плагин вводит следующие задачи

|  |  |
| --- | --- |
| **Task** | **Описание** |
| buildPlugin | Собирает плагин и подготавливает ZIP-архив к развертыванию. |
| patchPluginXml | Собирает все файлы plugin.xml в источниках и заполняет атрибуты с / до сборки и версии. |
| downloadRobotServerPlugin | Скачивает плагин робот-сервер, необходимый для запуска тестов пользовательского интерфейса. |
| prepareSandbox | Создает правильную структуру плагина, копирует исправленные XML-файлы плагина и заполняет каталог песочницы всем этим. |
| prepareTestingSandbox | Подготавливает песочницу, которая будет использоваться при запуске тестов. |
| prepareUiTestingSandbox | Подготавливает песочницу, которая будет использоваться при запуске тестов пользовательского интерфейса. |
| buildSearchableOptions | Создает индекс компонентов пользовательского интерфейса (также известных как параметры с возможностью поиска) для плагина путем запуска автономного экземпляра IDE.  Обратите внимание, что это задача runIde с предопределенными аргументами, и все свойства задачи runIde также применяются к задачам buildSearchableOptions. |
| jarSearchableOptions | Создает файл jar с возможностью поиска для распространения с плагином. |
| runIde | Выполняет экземпляр IntelliJ IDEA с плагином, который вы разрабатываете. |
| runIdeForUiTests | Выполняет экземпляр IntelliJ IDEA, готовый к запуску тестов пользовательского интерфейса с плагином, который вы разрабатываете. |
| publishPlugin | Загружает архив распространения плагина на https://plugins.jetbrains.com. |
| runPluginVerifier | Запускает инструмент проверки подключаемого модуля IntelliJ для проверки двоичной совместимости с указанными сборками IntelliJ IDE. |
| verifyPlugin | Проверяет полноту и содержимое дескрипторов plugin.xml, а также структуру архива плагина. |
| signPlugin | Подписывает ZIP-архив предоставленным ключом с помощью библиотеки marketplace-zip-signer. |
| listProductsReleases | Перечисляет доступные двоичные выпуски IDE, которые можно использовать с проверкой подключаемых модулей. |

Мой плагин

Требования: виджет в статус бар, который отображает время, проведенное за конкретным проектом.

Три режима работы трекера:

1. Работает
2. Пауза
3. Остановлен

В зависимости от режима меняется цвет трекера:

1. Зеленый (работает)
2. Желтый (пауза)
3. Красный (остановлен)

Трекер входит в режим паузы при бездействии в течение 5 секунд (время задано произвольно, можно изменить)

Стек технологий:

1. Java 8
2. Gradle
3. IntelliJ IDEA Plugin API

Пример работы:

1. Работа виджета остановлена пользователем



1. Виджет работает



1. Пользователь бездействует в проекте, виджет приостановил свою работу



Ссылка на гитхаб: https://github.com/DianaBayazitova/TimeTracker