­­МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Вятский государственный университет»**

**(ФГБОУ ВО «ВятГУ»)**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Отчет по лабораторной работе №5 дисциплины

«Разработка программных систем»

Выполнил студент группы ИВТб-31\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Монахов А.М./

Проверил преподаватель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Чистяков Г.А. /

Киров 2022

# Постановка задачи

Необходимо сконфигурировать рабочее окружение с системой контроля версий Git, а также провести ряд преобразований над проектом, полученным ранее. Для выполнения лабораторной работы необходимо решить следующие задачи.

* Установить расширение Git для Intell IJidea.
* Зарегистрировать проект, выполненный в ходе предыдущей лабораторной работы, на ресурсе, представляющем пространство для проектов Subversion.
* Добавить ссылку на созданный репозиторий в Intell IJidea.
* Синхронизировать репозиторий с проектом.
* Изменить функционал проекта.
* Обновить файлы в репозитории.
* Окатить сделанные изменения с помощью информации из репозитория.
* Создать побочную ветвь проекта и произвести конфликтующие изменения в стволовой и побочной ветвях.
* Разрешить возникший конфликт посредством встроенных инструментов Git.

2. Детализированное описание хода выполнения лабораторной работы

В Intell IJidea уже реализована интеграция с Git, поэтому загружаем проект из предыдущей лабораторной работы. Выбираем вкладку VCS/Enable Version Control integration, где в открытом окне выбираем Git. Таким способом мы подключаем Git к нашему проекту.

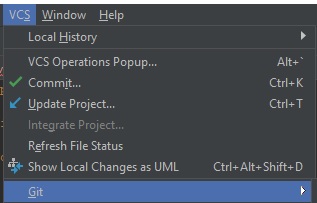


Рис 1. – Подключенный Git.

Далее был подключен репозиторий на GitHub и добавлен в наш проект.

Связываем и загружаем наш проект на GitHub:

Производим фиксирование наших изменений VCS/Commit, в открывшемся окне обязательно пишем комментарий и выбираем файлы, которые нужно зафиксировать. После чего жмем кнопку Commit.

Изображение выглядит как текст

Автоматически созданное описание

Далее нужно загрузить зафиксированные файлы в репозиторий: Git\Push, нажимаем Push.

Изображение выглядит как текст, снимок экрана, монитор, экран

Автоматически созданное описание

Рис 4. Выбор репозитория загрузки.

Нажимаем кнопку Push и после чего наш проект успешно загружается.

Производим изменения проекта, например, изменение текста у кнопок на главной форме и обновляем файлы внутри репозитория. Git/Commit устанавливаем комментарий изменения и нажимаем кнопку Commit, после Git/Push. Либо нажимаем по кнопке Commit and Push - загружаем изменения в ветку и репозиторий на GitHub.



Рис 5. – Фиксирование изменений проекта.

Откат сделанных изменений производится через вкладку Git, где выбираем вкладку Show Git Log, после чего выбираем нужную нам версию проекты, в нашем случае предыдущую, нажимаем правой кнопкой и выбираем пункт Revert Commit. После чего происходит откат к выбранной версии.



Рис 6. – Откат изменений проекта.

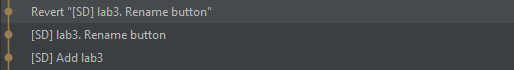


Рис 7. – Фиксирование отката.

Создание побочной ветки производится так: выбираем VCS/Git/Branches в появившемся окне выбираем New Branch, вводим название ветки и жмем OK, после чего у нас создается ветка с выбранным именем

Изображение выглядит как текст

Автоматически созданное описание

Рис 8. – Создание новой ветки.

Изменяем содержимое побочной ветки, например, имя класса GUI было переименовано в MyGUI, и загрузили изменения в ветку.

Далее выбираем основную ветку и изменяем имя класса GUI на другое GUIMain, чтобы возникла конфликтующая ситуация, так же загружаем изменения в ветку.

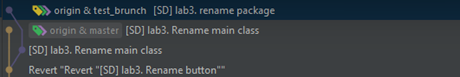


Рис 10. – Фиксирование изменений в главной ветке и в побочной.

Далее переходим в местер ветку и сливем все из тестовой ветки в мастер ветку.

Изображение выглядит как текст, снимок экрана, монитор, экран

Автоматически созданное описание

После чего у нас появится окно конфликтов, где мы можем принять изменения либо главной ветки, либо побочной, выбираем все файлы и принимаем изменения главной ветки. После чего все изменения неконфликтующие войдут в главную ветку, а конфликтующие изменения останутся от главной ветки

Изображение выглядит как текст

Автоматически созданное описание

Рис 11. – Конфликтные файлы.

Изображение выглядит как текст, снимок экрана, монитор, экран

Автоматически созданное описание

Рис 12. – окно разрешения конфликтив.

Изображение выглядит как текст

Автоматически созданное описание

Рис 12. – Объединение веток.

# Вывод

В результате лабораторной работы были основные навыки работы с Git: подключение Git к проекту, фиксация изменений проекта и их загрузка в текущую ветку; откат к выбранной версии проекта; создание новых побочных ветвей; фиксация изменений как в главной ветке, так и побочных; слияние главной ветки и побочной; решение возникших конфликтных ситуаций при слиянии встроенными средствами Git.