Фатьянов М.А.

ИСР 1.1. Работа с репозиторием Git/GitHub:

базовые операции

1. Создание локального репозитория (git init)

Чтобы начать работу с системой контроля версий Git, нужно инициализировать репозиторий в папке проекта:

откройте терминал;

перейдите в директорию проекта;

выполните команду $git\ init$.

После этого в папке появится скрытая директория .git, где Git будет хранить:

историю изменений;

конфигурационные настройки репозитория.

2. Проверка состояния репозитория (git status)

Команда $git\ status$ показывает текущее состояние репозитория:

какие файлы изменены;

какие файлы отслеживаются;

есть ли незафиксированные изменения.

В новом репозитории команда выведет информацию о пустом состоянии проекта.

3. Добавление файлов в индекс (git add)

Команда $git\ add$ помещает файлы в «индекс» (область подготовки к коммиту):

для отдельного файла: $git\ add\ file.c$;

для группы файлов по маске: $git\ add\ "*.c"$;

для всех файлов в директории: $git\ add\ -A$.

Это первый шаг к фиксации изменений — без $git\ add$ коммит не увидит новые или изменённые файлы.

4. Фиксация изменений (git commit)

Коммит ($git\ commit$) — это «снимок» состояния репозитория на определённый момент. Чтобы создать коммит:

Добавьте файлы в индекс через $git\ add$;

Выполните $git\ commit\ -m\ "Описание изменений"$ (флаг -m задаёт комментарий к коммиту).

Каждый коммит имеет уникальный идентификатор и хранит:

список изменённых файлов;

метаданные (автор, дата, комментарий).

5. Отправка изменений на сервер (git push)

Чтобы загрузить локальные коммиты в удалённый репозиторий (например, на GitHub), используйте $git\ push$:

$git\ push\ <имя_репозитория>\ <ветка>$

Пример:

$git\ push\ origin\ master$

Где:

origin — стандартное имя удалённого репозитория;

master — ветка по умолчанию (в новых репозиториях может называться main).

6. Клонирование репозитория (git clone)

Чтобы получить копию удалённого репозитория на свой компьютер, выполните:

$git\ clone\ <ссылка_на_репозиторий>$

Эта команда:

создаёт локальную копию проекта;

автоматически настраивает связь с удалённым репозиторием (обычно как origin).

7. Получение обновлений с сервера (git pull)

Если в удалённом репозитории появились новые изменения, синхронизируйте их с локальной версией:

$git\ pull\ <имя_репозитория>\ <ветка>$

Пример:

$git\ pull\ origin\ master$

Команда $git\ pull$ сочетает два действия:

$git\ fetch$ (загрузка изменений с сервера);

$git\ merge$ (слияние изменений с локальной веткой).

8. Работа с ветками (git branch, git checkout)

Создание ветки:

$git\ branch\ <имя_ветки>$

Например:

$git\ branch\ amazing_new_feature$

Переключение на ветку:

$git\ checkout\ <имя_ветки>$

Пример:

$git\ checkout\ amazing_new_feature$

По умолчанию существует ветка master (или main). Ветки позволяют:

новые функции изолированно;

избегать конфликтов с основной версией кода.

9. Слияние веток (git merge)

Чтобы объединить изменения из одной ветки в другую (например, из amazing_new_feature в master):

Переключитесь на целевую ветку:

$git\ checkout\ master$

Выполните слияние:

$git\ merge\ amazing_new_feature$

Если конфликтов нет, Git автоматически создаст коммит слияния.

Отправьте изменения на сервер:

$git\ push\ origin\ master$

Удаление ненужной ветки:

$git\ branch\ -d\ amazing_new_feature$