**Отчет по Git**

Во время разработки требуется хранить как минимум 2 версии исходного кода программы: версию находящийся в активной разработке и последнюю гарантировано рабочую версию.

Копия программы нужна чтобы **продемонстрировать** заказчику или чтобы вернуться к ней, если с текущей версией случится что-нибудь **плохое**.

Решить эту проблемы, автоматизировать рутинные операции призвана **система управления версиями**.

**Система управления версиями** (Version Control System, VCS) – программное обеспечение для облегчения работы с изменяющейся информацией.

Git — один из видов **VCS**.

В целом VCS можно разделить таким образом:

* **Локальные** — все файлы хранятся только **в вашей ОС**, например, разложены по папкам с версиями.
* **Централизованные** — проект хранится **на** **сервере**, а ваша рабочая версия включает только текущий набор файлов.
* **Распределенные** — копии проекта (и вся информация о версиях) располагаются **не только на сервере**, **но и на** нескольких **клиентских машинах**, чтобы обеспечить устойчивость к отказу сервера.

**Репозиторием** называют хранилище вашего кода и историю его изменений. Git работает **локально** и все ваши репозитории хранятся в определенных папках на жестком диске.

**Commit/Коммит** — это способ **сохранения изменений в коде**. Каждый commit содержит информацию о том, **что было изменено** в коде **и кем** были внесены эти изменения. Они позволяют разработчикам отслеживать изменения в своем (или чужом) коде и возвращаться к предыдущим версиям, если это необходимо.

**Ветка** — это **набор** commit, которые идут **друг за другом**. У ветки **есть название**, основную ветку чаще всего называют master. Если говорить простыми словами, то ветка master — это наш проект.

1. **Навигация через Git Bash**

* clear — Очистить консоль
* pwd — Показать текущий каталог
* ls - Показать файлы в данной папке, включая и скрытые
* ls -f — Показать файлы в данной папке, кроме скрытых
* cd c:/ — Перейти в конкретный каталог
* cd - — Вернуться назад
* cd .. — Выйти на 1 уровень вверх
* cd ../.. — Выйти на 2 уровня вверх
* mkdir — Создать папку
* cd !$ — Перейти в только что созданную папку
* mkdir -p {app1,app2} — Создать сразу несколько папок
* mkdir -p app/{css,js} — Создать сразу несколько вложенных папок
* touch index.html — Создать файл index.html
* touch app/{css/main.css,js/main.js,index.html} — Создать сразу несколько файлов, никаких лишних пробелов быть не должно
* touch — позволяет создавать файлы
* rm test — Удалить пустую папку test
* rm -r test — Удалить папку test с файлами внутри неё
* mv app1/\*.\* app2 — Переместить все файлы из папки app1 в папку app2

1. **Работа с репозиторием**

* git init – создание репозитория в текущей директории
* git status – проверка статуса директории (если репозитория нет, то будет выведено сообщение)
* git add <название\_файла> – добавление файла в отслеживаемые (Для того, чтобы добавить все файлы сразу то в место файла нужно поставить **точку** или --all. Просмотреть можно через git status)
* git clone <cсылка> <новое\_название> - клонирование существующего репозитория
* get fetch <названия репозитория> - копирование изменений удаленного репозитория
* git commit -m “описание” – создание коммита
* git ls-files – просмотр отслеживаемых файлов
* git log – просмотр всех коммитов
* git diff – показывает разницу между текущим состоянием файлов и предыдущим коммитом
* git diff <исходная\_ветка> <целевая\_ветка> - показывает разницу между двух веток
* git branch <название\_ветки> – создание новой ветки (если не указывать название ветки, то будет отображен список всех веток)
* git branch -d <название\_ветки> - удаление ветки
* git checkout <название\_ветки> – переключение между ветками
* git pull origin master – обновление локальной ветки master
* git merge <название\_ветки> - слияние ветки
* git push origin master – отправка изменений ветки master на удаленный репозиторий (нужно обязательно находиться на этой ветке)
* git clone <адрес\_репозитория> - клонирование существующего репозитория через ссылку

Источники:

* <https://github.com/VetrovSV/Programming/blob/master/git_lec.pdf>
* <https://habr.com/ru/articles/541258/>
* <https://selectel.ru/blog/tutorials/git-setup-and-common-commands/>
* <https://git-scm.com/book/ru/v2/%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D1%8B-Git-%D0%A1%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-Git-%D1%80%D0%B5%D0%BF%D0%BE%D0%B7%D0%B8%D1%82%D0%BE%D1%80%D0%B8%D1%8F>
* <https://gist.github.com/sasstr/03d819bd98272ae261dd29ef731e9a6f>
* <https://www.learnqa.ru/tpost/z1k65h4um1-chto-takoe-commit-v-git>
* <https://habr.com/ru/articles/542616/>
* <https://docs.github.com/ru/get-started/using-git/getting-changes-from-a-remote-repository>

Подготовил **Занин Вячеслав ВМК-21**