Основы использования Git

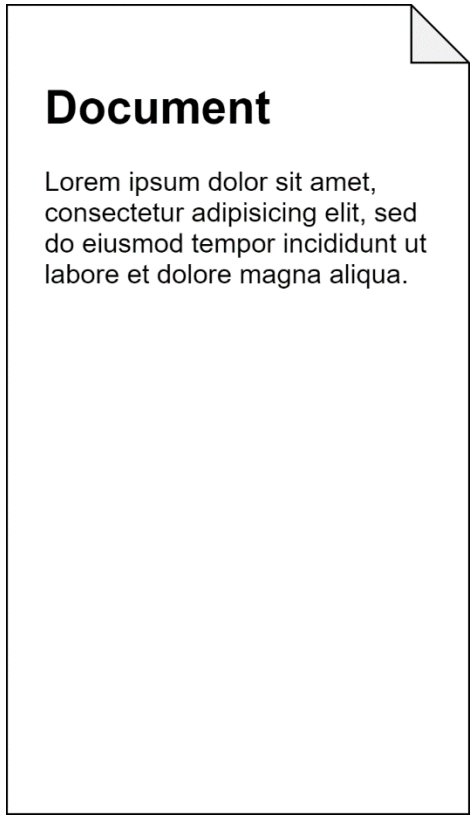
Git

Git — распределённая система управления версиями. Проект был создан Линусом Торвальдсом для управления разработкой ядра Linux. Первая версия выпущена 7 апреля 2005 года.

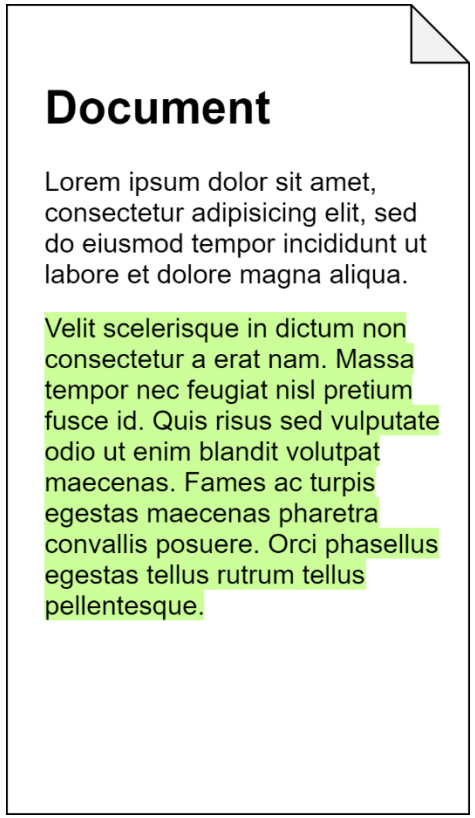
Система управления версиями (от англ. Version Control System, VCS или Revision Control System) — программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое.

Основы использования Git

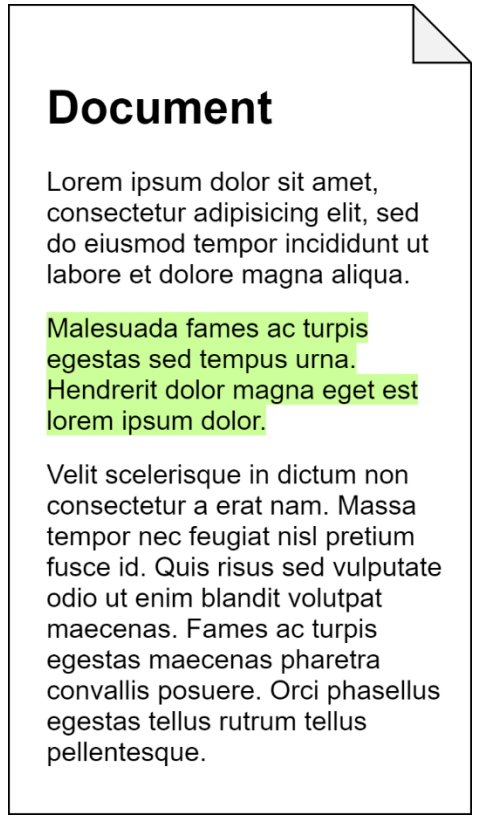
Версия 1



Версия 2



Версия 3



Основы использования Git

История систем контроля версий

SCCS (Source Code Control System) — первая система управления версиями, разработанная в Bell Labs в 1972 году

RCS (Revision Control System) — разработана в 1985 году.

CVS (Concurrent Versions System) — централизованная система управления версиями 1990г.

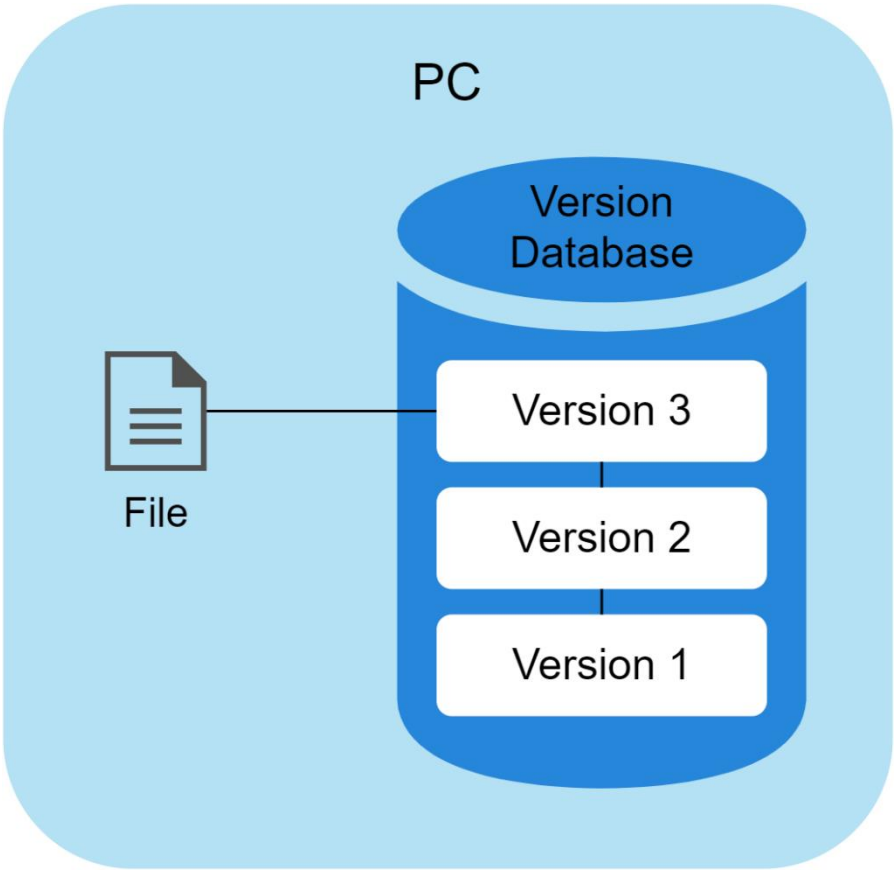
SVN (Subversion) — свободная централизованная система управления версиями, официально выпущенная в 2004 году компанией CollabNet.

Git – 2005 г.

Основы использования Git

Типы систем контроля версий

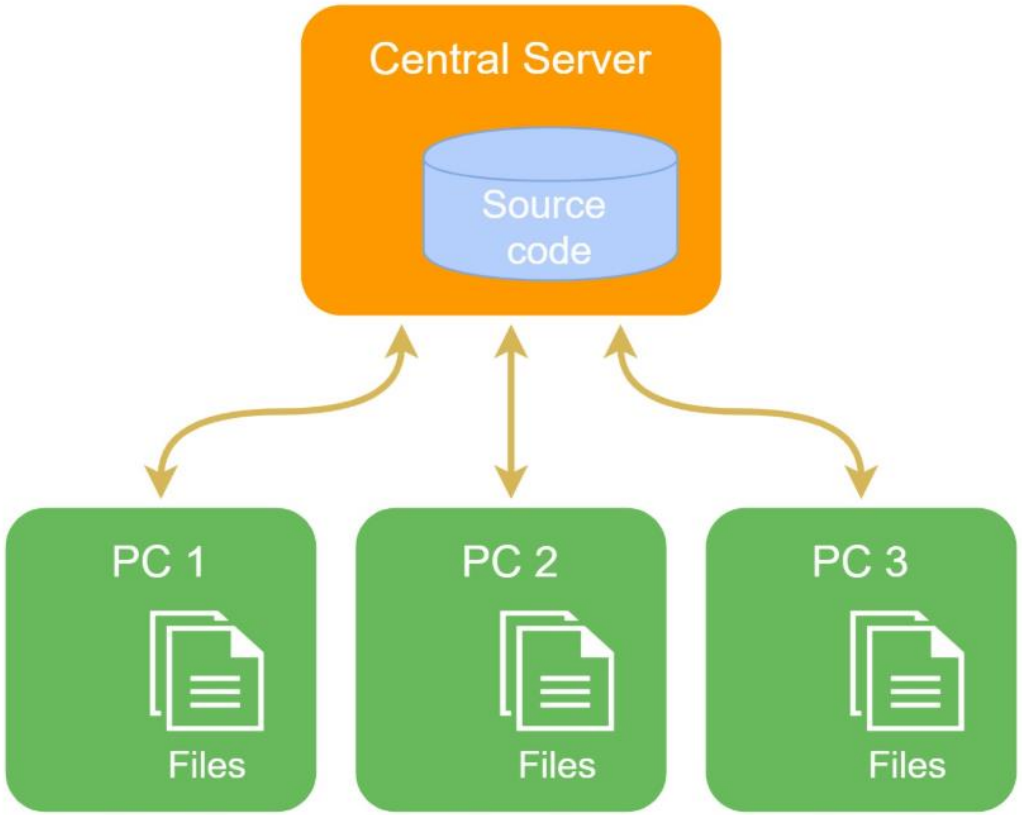
Локальная



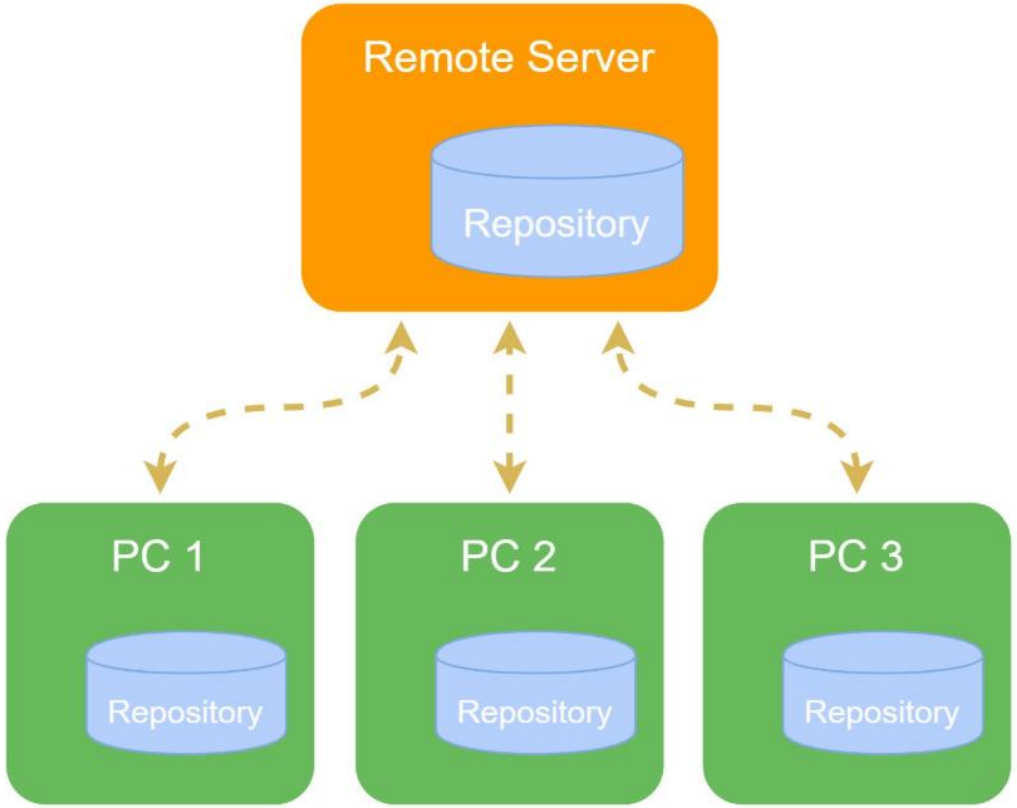
Основы использования Git

Типы систем контроля версий

Централизованная



Распределенная



Основы использования Git

Конфигурация

git config - утилита для вывода и изменения конфигурации для работы Git и внешнего вида.

Уровни конфигурации:

- Уровень системы: общие для всех пользователей --system
- Уровень пользователя: настройки конкретного пользователя --global.
- Уровень проекта: (.git/config) в репозитории проекта.

Настройки на каждом следующем уровне перекрывают настройки из предыдущих уровней. То есть значения в .git/config перекрывают конфигурацию уровня системы.

Основы использования Git

Основные команды

- **git init** – создает репозиторий
- **git add** – добавляет файлы в индекс для последующего коммита.
- **git commit** – используется для фиксации изменений в репозитории.
- **git diff** – отображение изменений.
- **git status** – вывод состояния файлов.
- **git log** – используется для вывода истории коммитов.
- **git help** – вывод справки, **git help <command>** - ввод доступных опций использования команды.

Основы использования Git

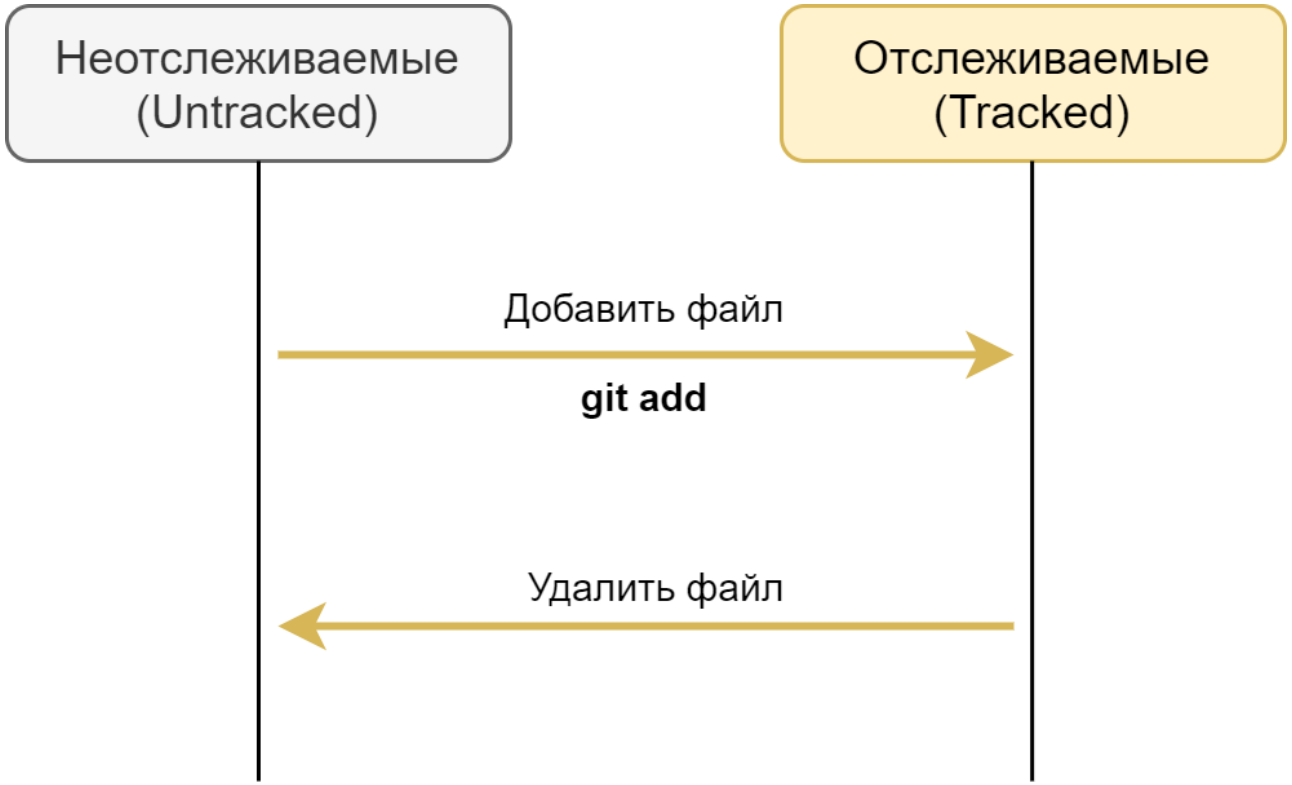
Коммит

```
commit 46b5b3a932ca403d7008617372e2a3cac2dd5eb0 (HEAD -> master)
Author: Anna Marhina <ann.marhina@gmail.com>
Date:   Thu Jul 23 19:26:10 2020 +0300

    Change index.html

diff --git a/index.html b/index.html
index 3c9ec5c..291e1ab 100644
--- a/index.html
+++ b/index.html
@@ -5,7 +5,7 @@
 </head>

 <body>
-    Hello
+    Hello Git
 </body>

 </html>
\ No newline at end of file
```

*результат выполнения команды git log -p

История коммитов

Состояния файлов



Основы использования Git

Состояния файлов

```
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        index.html

nothing added to commit but untracked files present (use "git add" to track)
```

Основы использования Git

Основы использования Git

git status

1. Все файлы в неизмененном состоянии

```
$ git status
On branch master
nothing to commit, working tree clean
```

2. Файл index.html в измененном состоянии

```
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
```

3. Файл index.html проиндексирован и готов к коммиту

```
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   index.html
```

.gitignore

.gitignore - файл, позволяющий указывать что в репозитории не следует отслеживать.

Что игнорировать:

- скомпилированный исходный код;
- пакеты и сжатые файлы;
- логи;
- базы данных;
- вспомогательные файлы систем разработки (Visual Studio).

Основы использования Git

.gitignore пример

```
# Исключить все файлы с расширением .a
*.a

# Но отслеживать файл lib.a, даже если он подпадает под исключение выше
!lib.a

# Исключить файл TODO в корневой директории, но не файл в subdir/TODO
/TODO

# Игнорировать все файлы в директории build/
build/

# Игнорировать файл doc/notes.txt, но не файл doc/server/arch.txt
doc/*.txt
```