

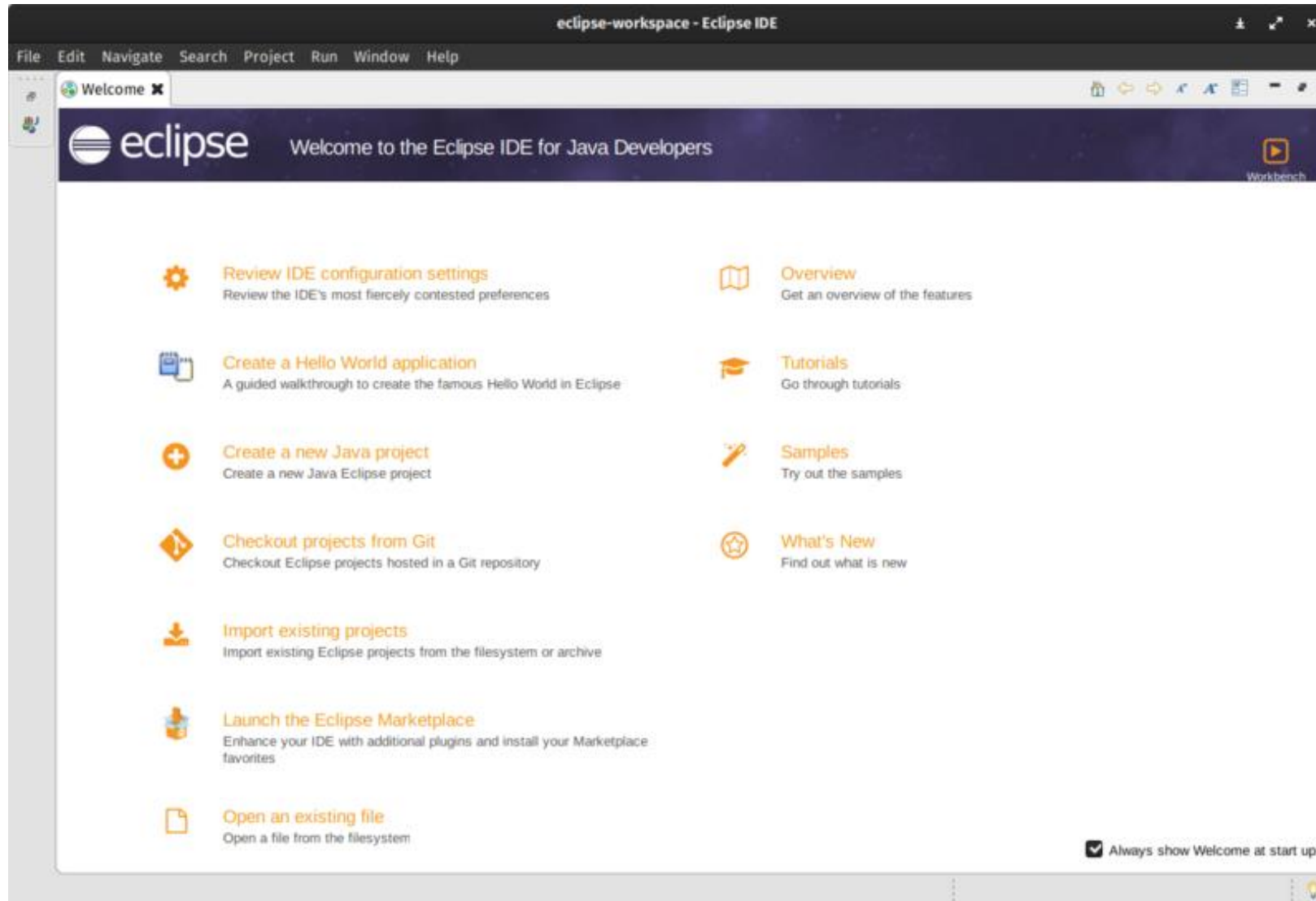
Тема 2. Интегрированная среда разработки ПО

1. Основные компоненты среды разработки
2. Приёмы работы
3. Поддержка совместной работы над проектом

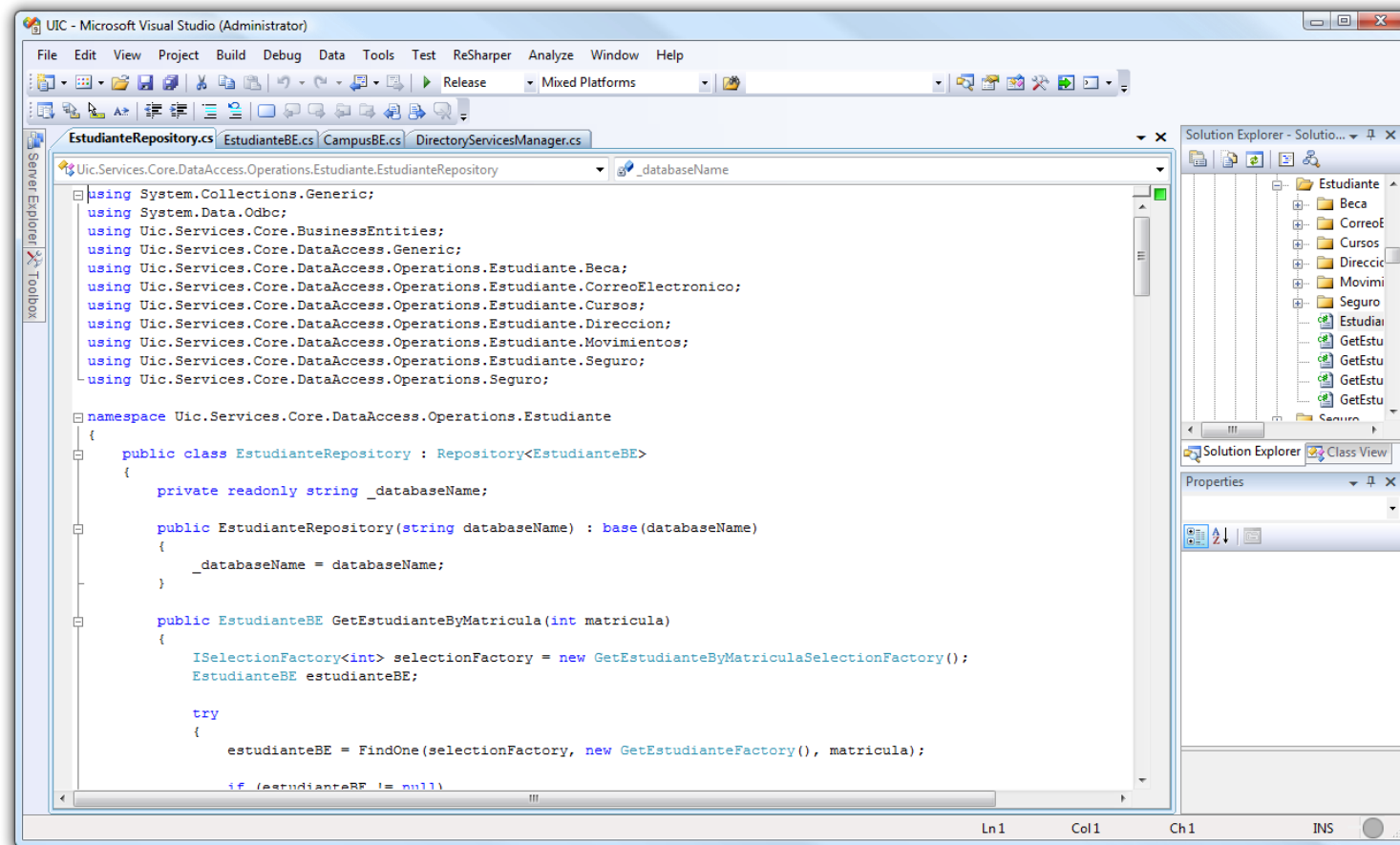
Основные компоненты IDE

- редактор кода
- средства сборки проекта
- средства запуска проекта
- средства отладки
- средства тестирования
- поддержка совместной работы
- расширения

Примеры сред. Eclipse



Примеры сред. MS Visual Studio



Live Share в Visual Studio 2019

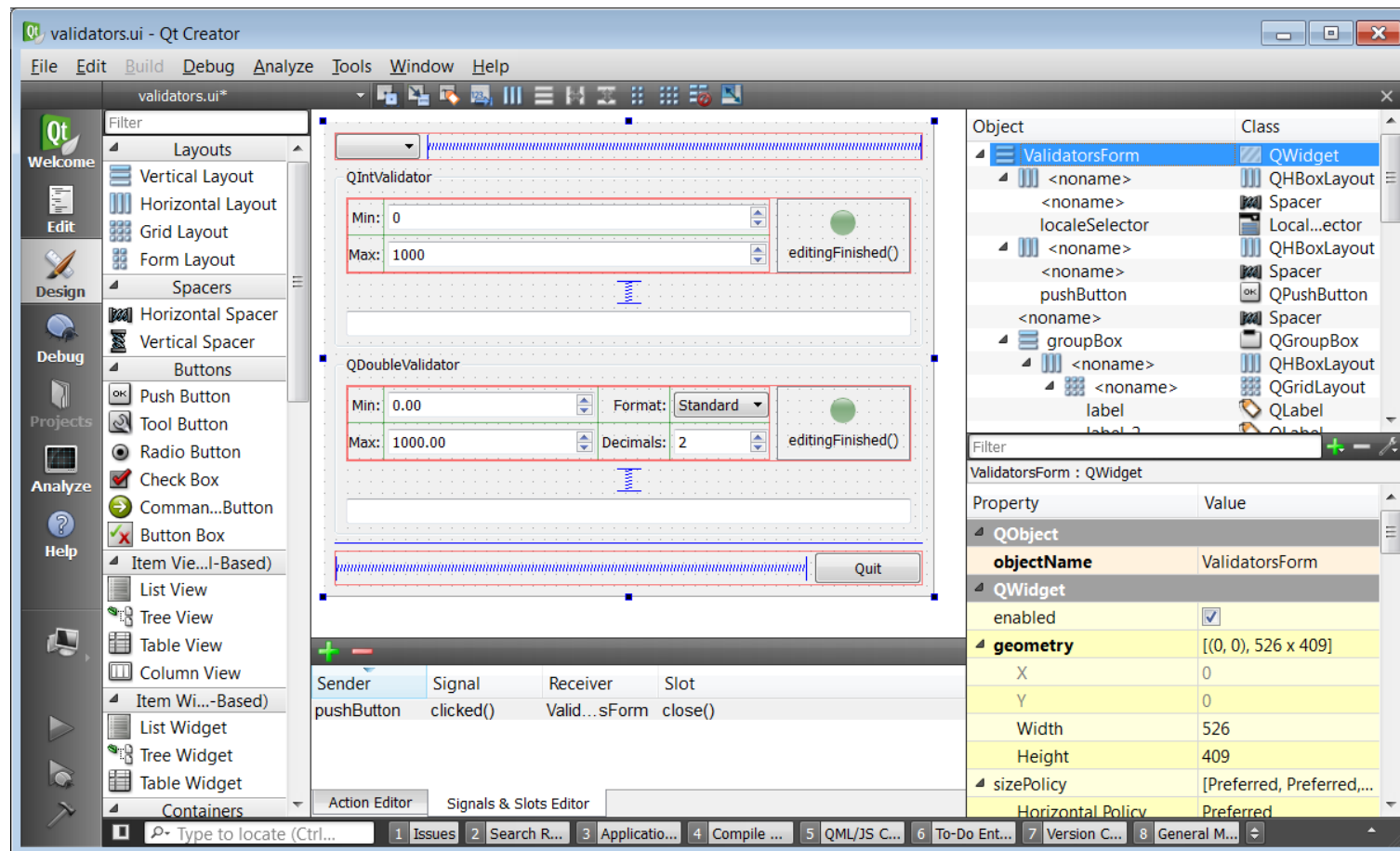
The screenshot displays the Visual Studio 2019 interface during a Live Share session. On the left, the 'SESSION DETAILS' sidebar is expanded, showing a list of participants and shared resources. The participants list includes Jon W Chu (Header.js:12), Amanda Silver (GuestbookGrid.js:13), and PJ Meyer (GuestbookGrid.js:9). Below this, shared servers (localhost:3000, REST API) and terminals (bash) are listed. The audio participants section shows Jon W Chu with a muted icon, and Amanda Silver and PJ Meyer with active audio icons.

The main editor area on the right shows a JavaScript file with the following code:

```
1 import GridArrow from "../GridArrow";
2 import GridLegend from "../GridLegend";
3 import GuestbookGridCell from "../GuestbookGridCell";
4
5 export default class GuestbookGrid extends Component {
6   constructor(props) {
7     super(props)
8     this.state = PJ Meyer
9     | signatures: signatures
10   }
11 }
12
13 Amanda Silver
14 render() {
15   const cells = this.state.signatures.map((signature, index) => (
16     <GuestbookGridCell key={index} {...signature} />
17   ));
18 }
```

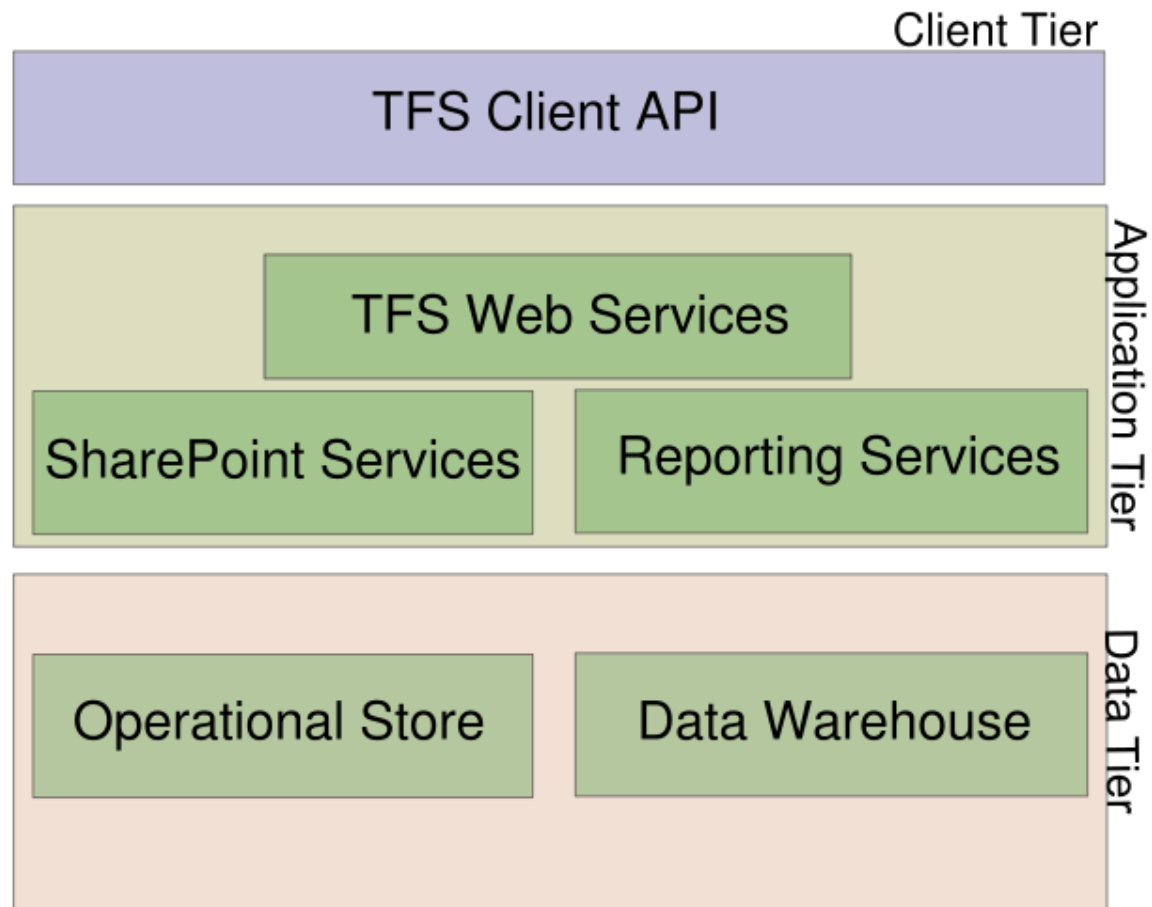
Annotations in the code include a purple box around 'PJ Meyer' on line 8 and a yellow box around 'Amanda Silver' on line 13. A green highlight covers the map function and its return statement on lines 15-16.

Примеры сред. Qt Creator

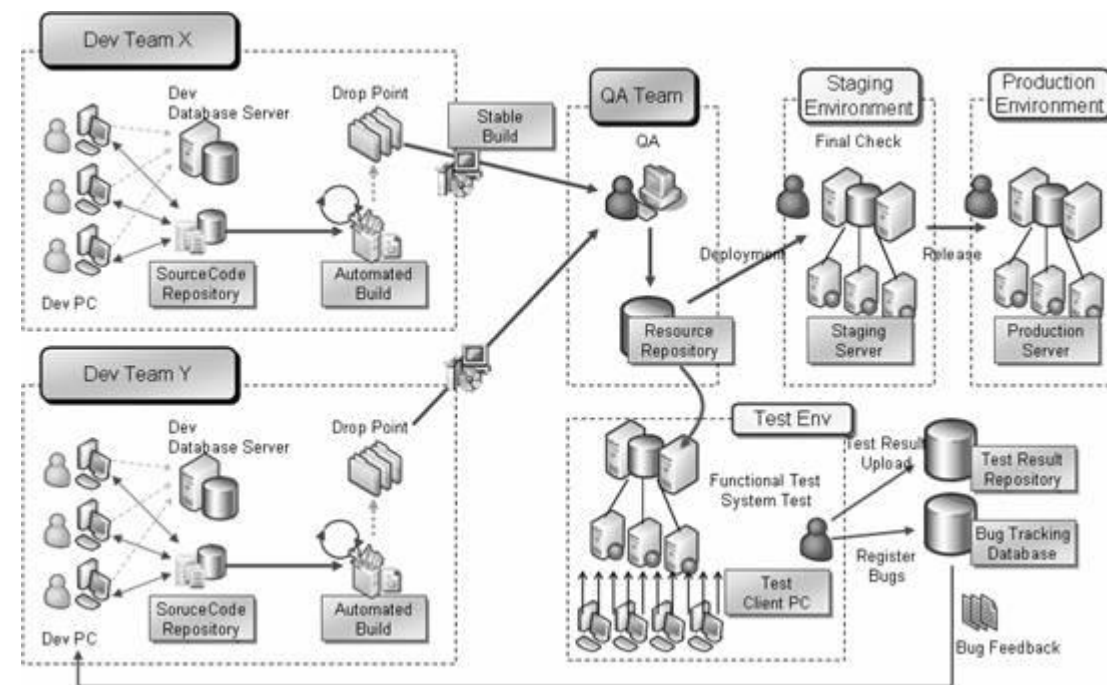
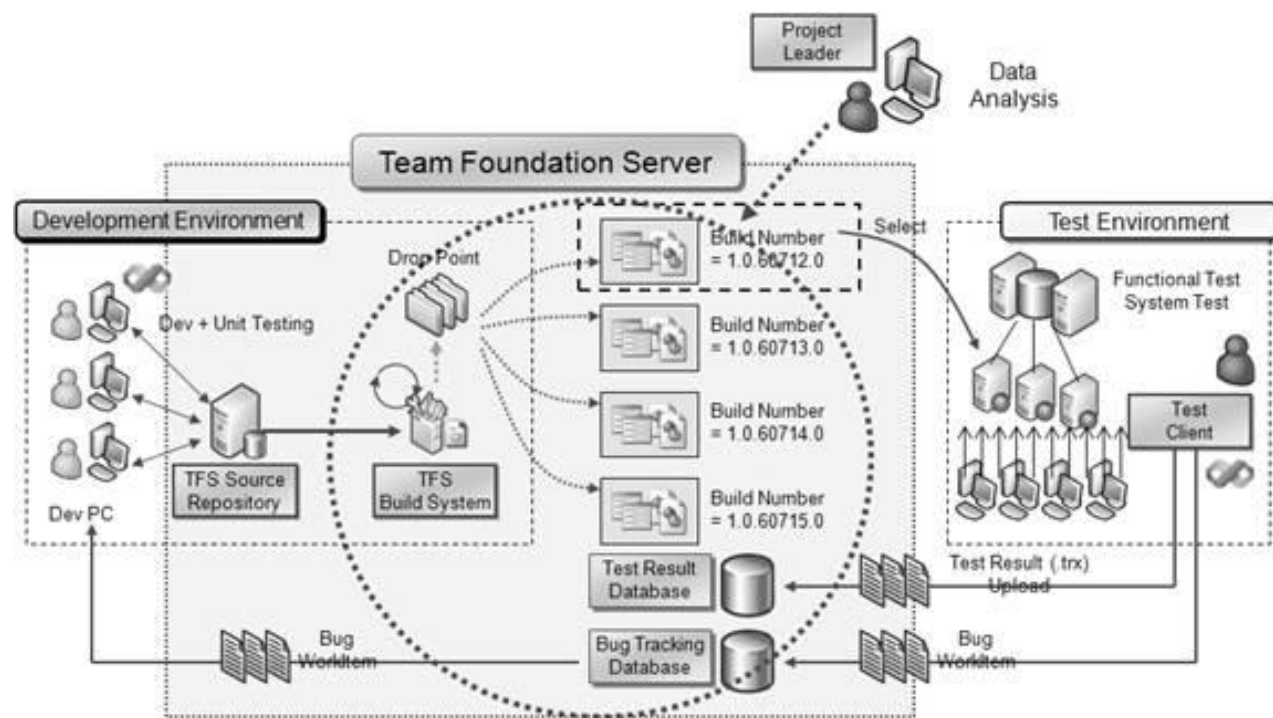


Примеры сред. Azure DevOps Server

Архитектура TFS



Логическая структура



Тема 3. Системы управления версиями

Назначение систем управления версиями

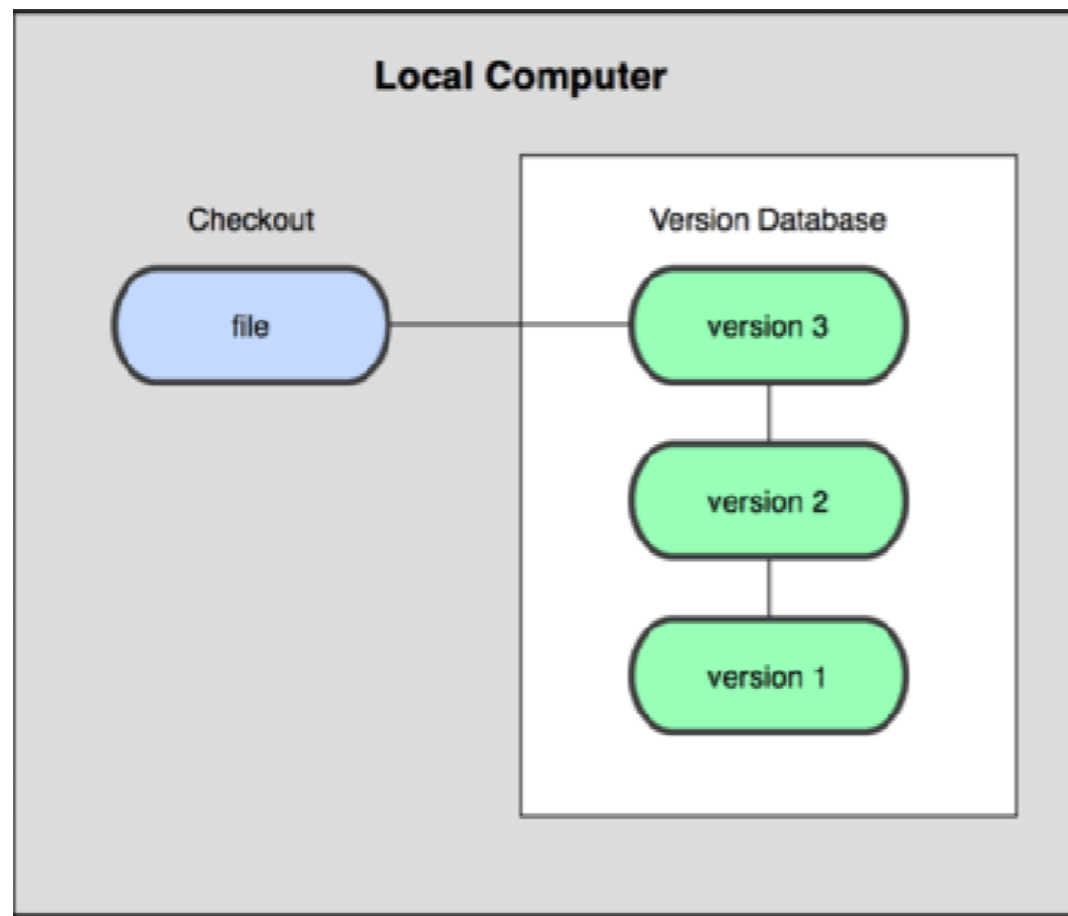
Виды систем управления версиями

Git: основные команды и порядок работы

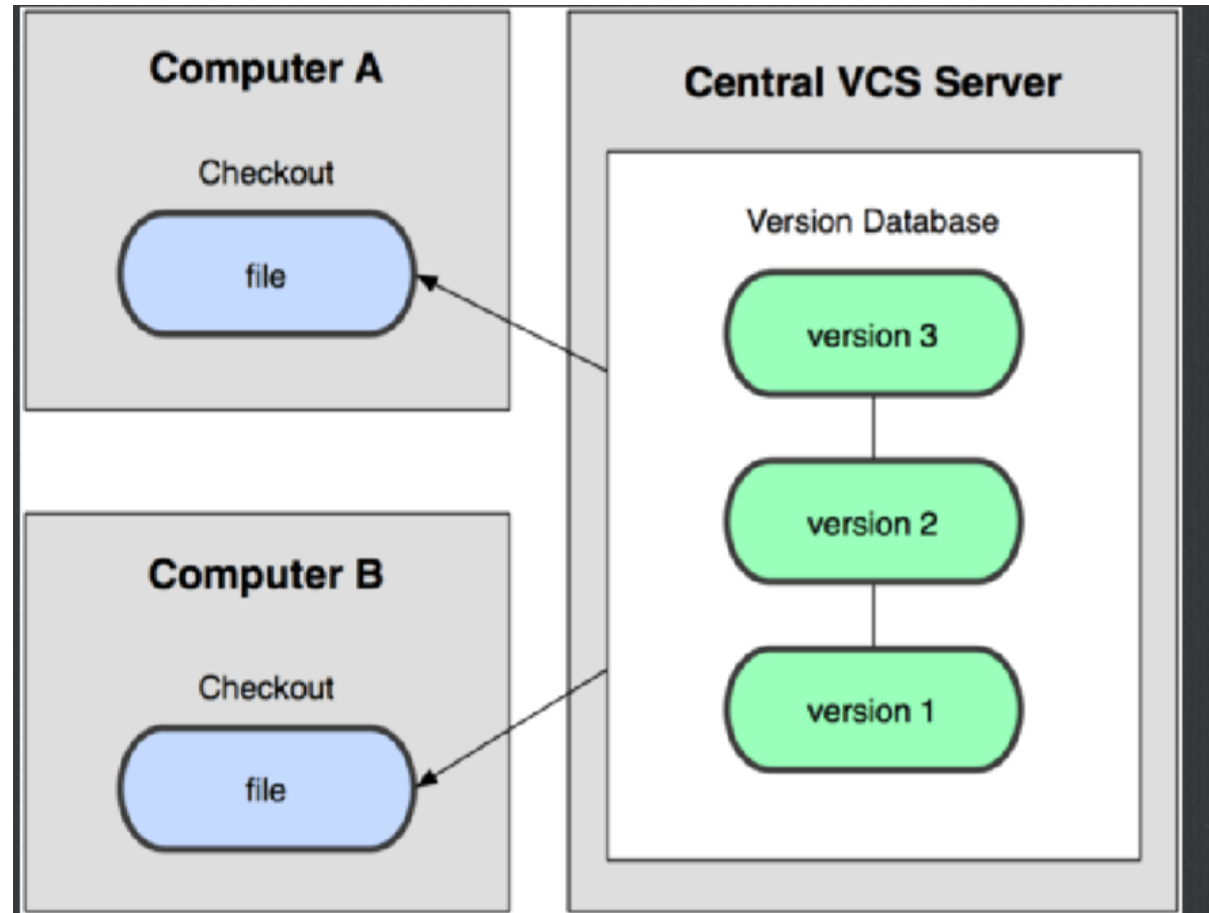
Зачем управлять версиями?

- дисциплина
- совместная работа
- архив всех версий (история)
- восстановление
- экономия дискового пространства

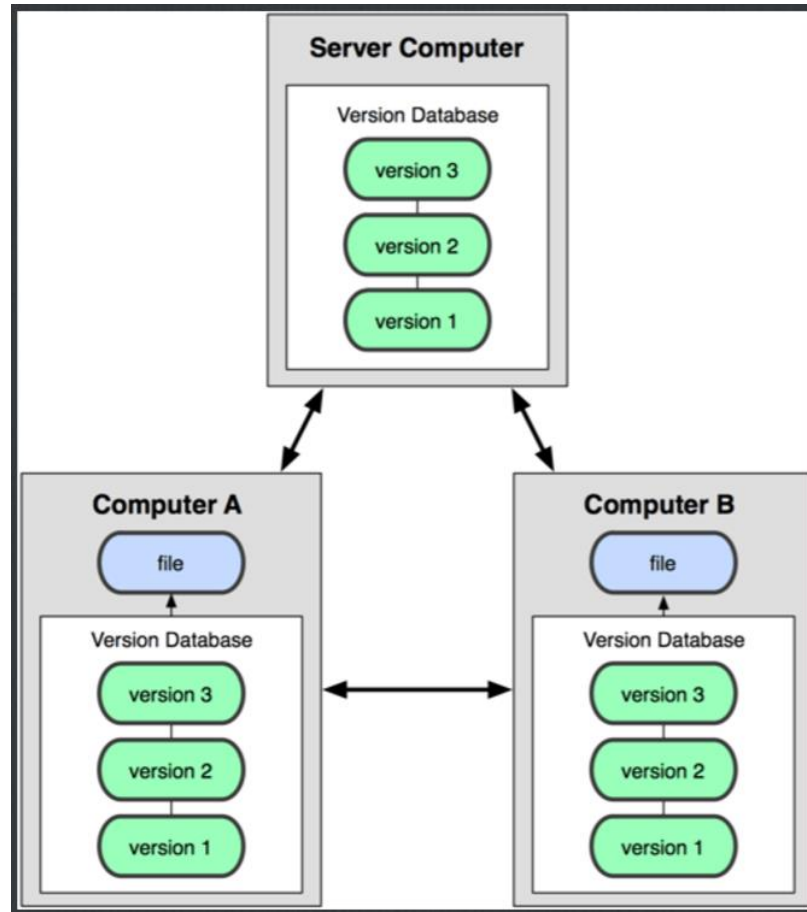
Локальное управление версиями



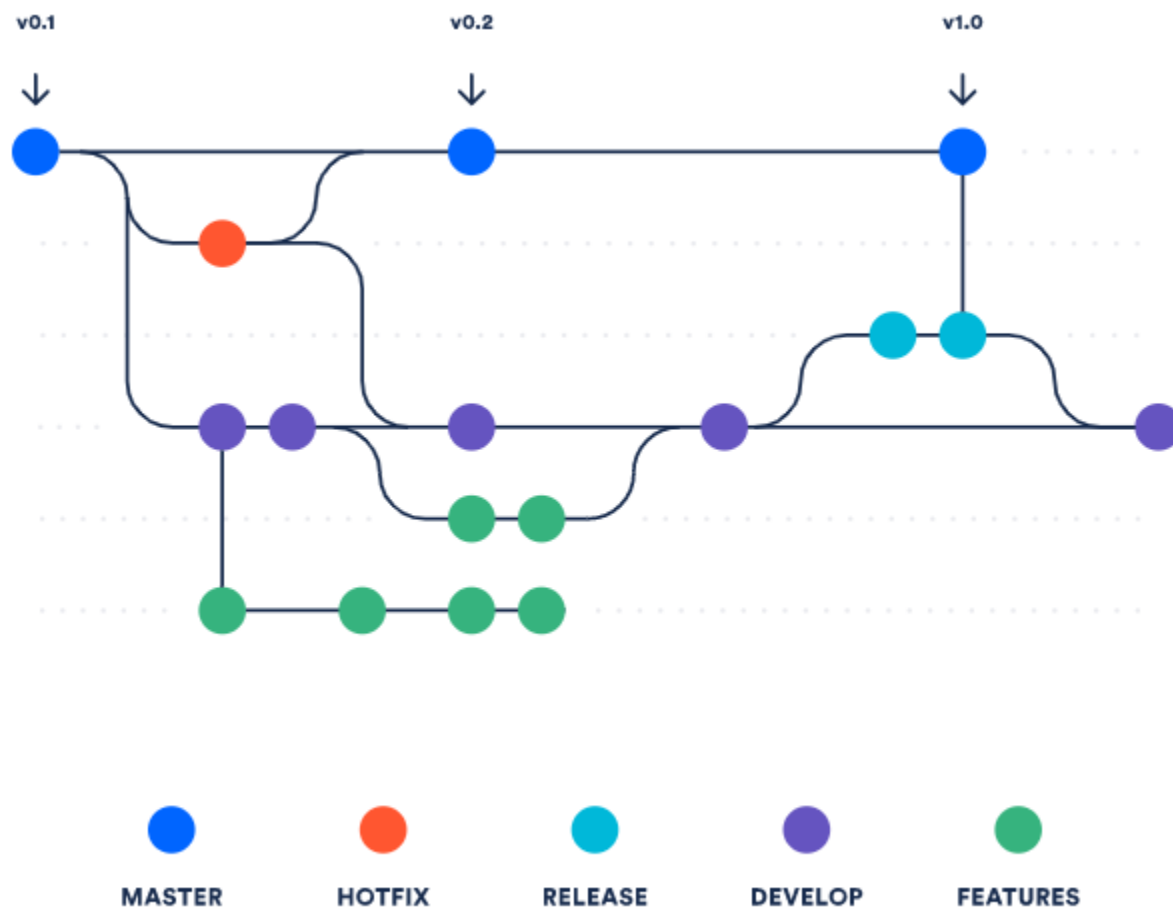
Централизованное управление версиями



Распределённое управление версиями



Как это выглядит



- Система одновременных версий (CVS)
- Apache Subversion (SVN)
- Git
- Mercurial
- Preforce

Сравнение версий

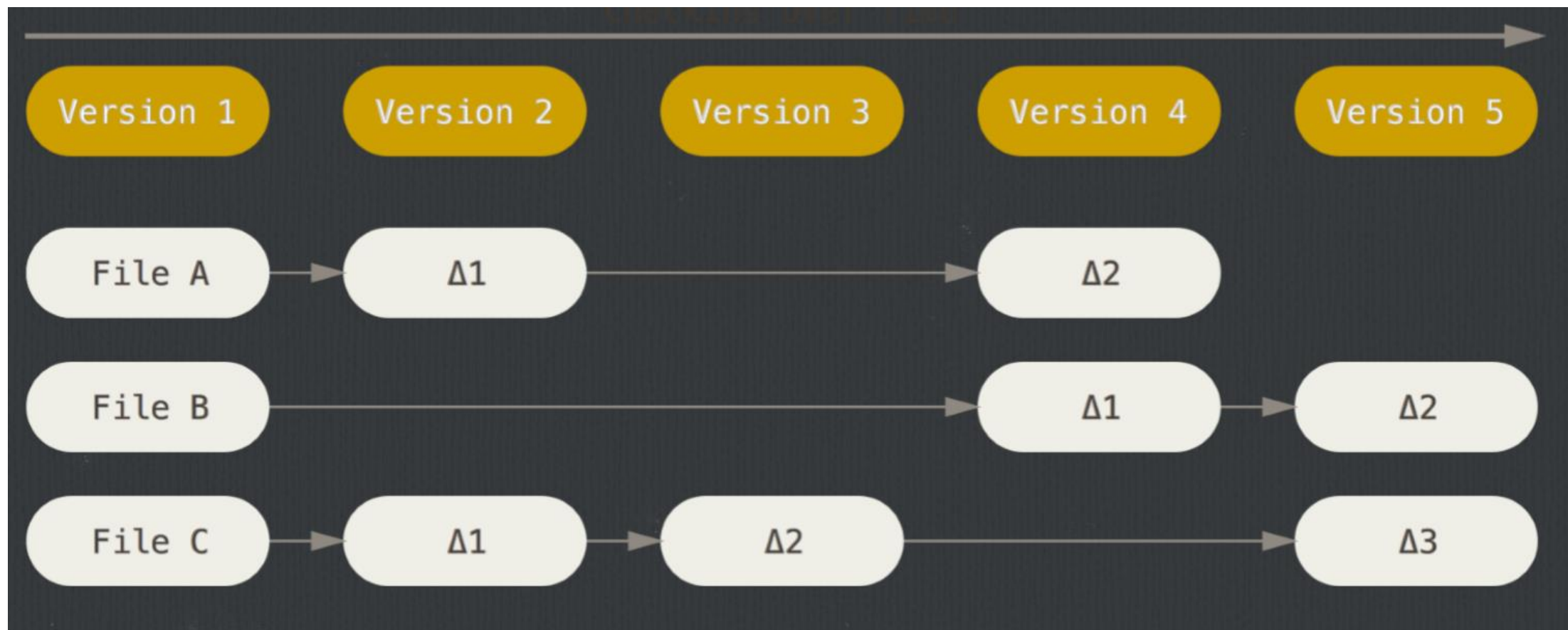
Программное обеспечение Архитектура сети Разрешение конфликтов Статус разработки

Git	Распределенная	Слияние	Активная
Mercurial	Распределенная	Слияние	Активная
SVN	Клиент-сервер	Слияние или блокировка	Активная
CVS	Клиент-сервер	Слияние	Только обслуживание

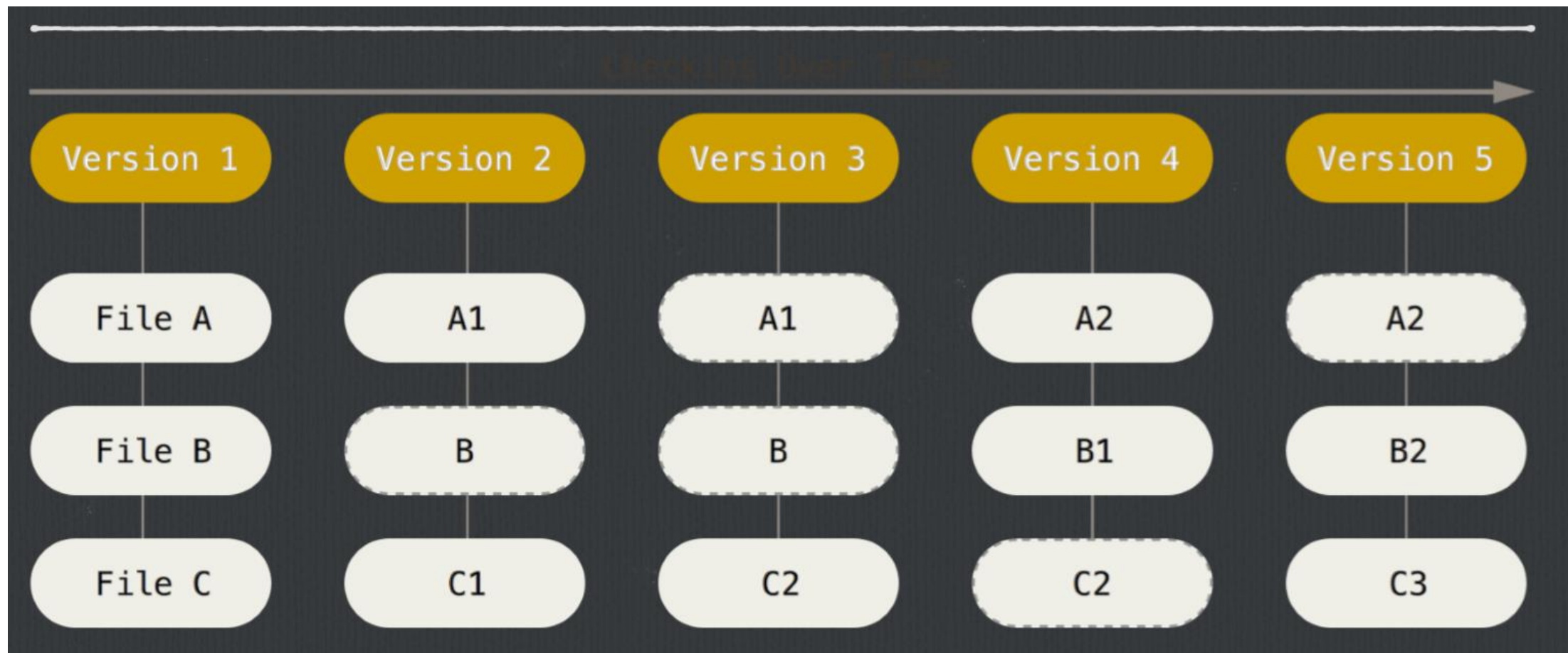
Git

- Скорость
- простой дизайн
- поддержка нелинейной разработки
- Распределённость
- поддержка больших проектов

Хранение изменений



Хранение состояний



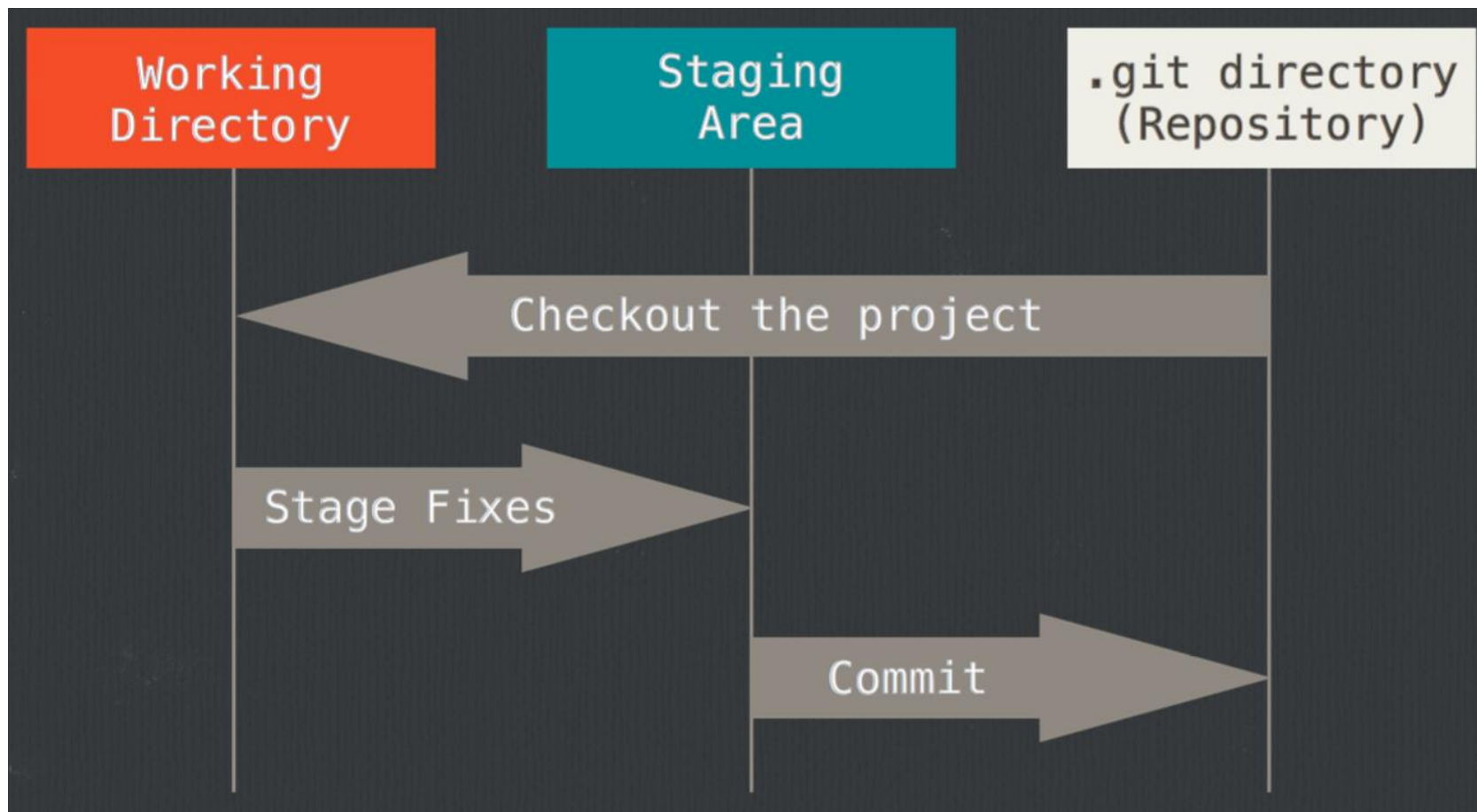
Особенности Git

- локальность изменений
- целостность данных
- добавление данных

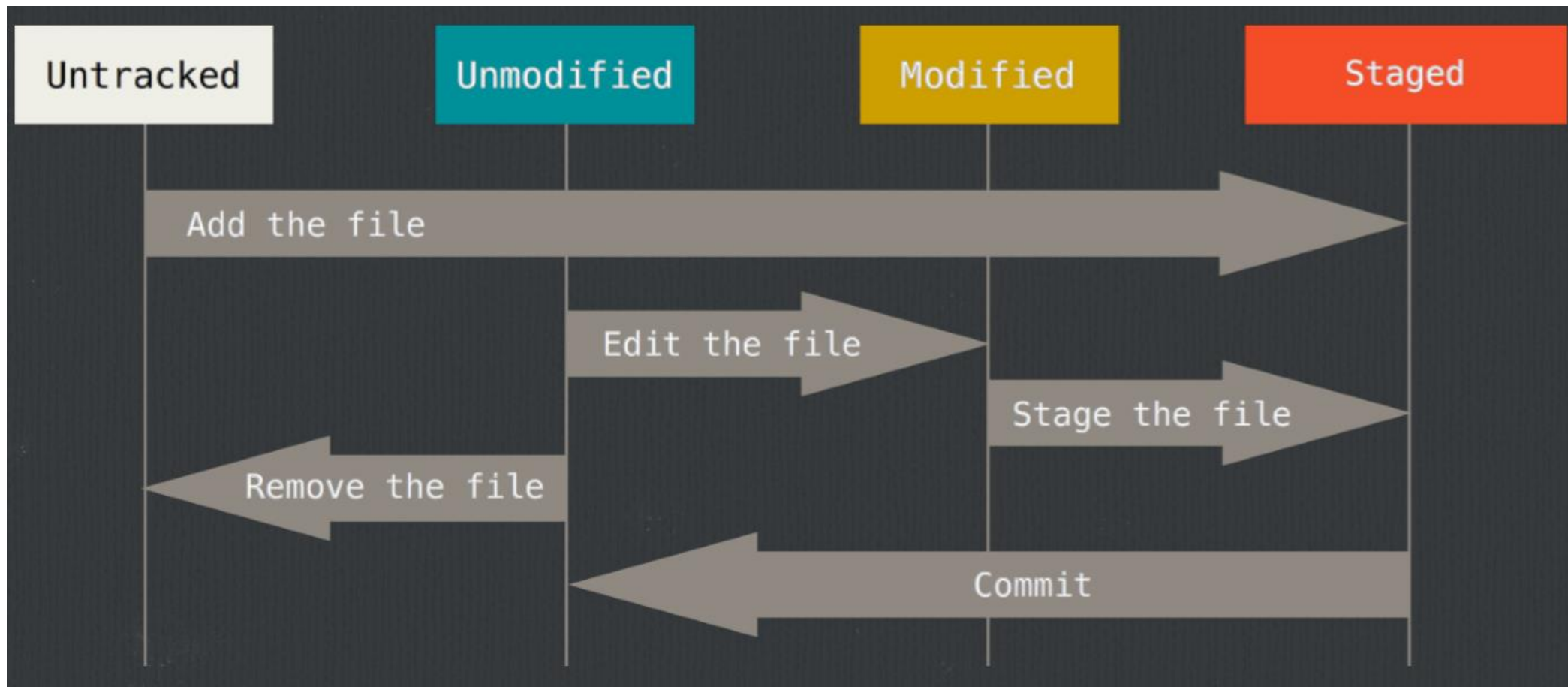
Три состояния файлов проекта

- закреплённое (committed)
- изменённое (modified)
- подготовленное (staged)

Три секции Git-проекта



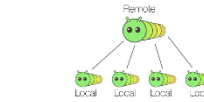
Жизненный цикл состояний файлов проекта



Настройка и работа с Git

```
$ git config --global user.name "John Doe"
$ git config --global user.email johndoe@example.com
```

Git Cheat Sheet



Git is a **distributed** version control system
Everyone that has a local copy of the repository at all times. Cannot be partially checked out, it's all or nothing.

Git \neq GitHub

Git is **not** GitHub
Git is a repository, where files are stored and managed. GitHub is a hosting service that keeps your Git repo in the cloud.

```
> git
```

Git CLI is **very powerful**.

OK, that's not a "fact", but once you get used to it, you'll be able to use the commands anywhere. You can also use CLI and GUIs interchangeably on the same Git repository.



There are **many ways** to get there
People use Git in many different ways, and projects may follow different Git flows. Don't be shy, ask around.

Terminology

repository - where files are stored, can be remote or local

remote repository - repository in hosting server, also referred to as origin

local repository - repository in local development machine

branch - a stream of work where commits are kept, can be remote or local

remote branch - branch with published commits (commits that have been pushed)

local branch - branch with unpublished commits, only developer can see, not shared

commit - a change unit, it is a scope of changes that are kept in sequential order

master branch - main stream of work, must always be stable

working branch - developer's stream of work, sandbox

staging area/index - keeps files to include in next commit

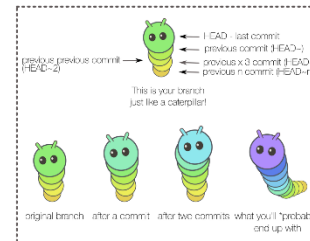
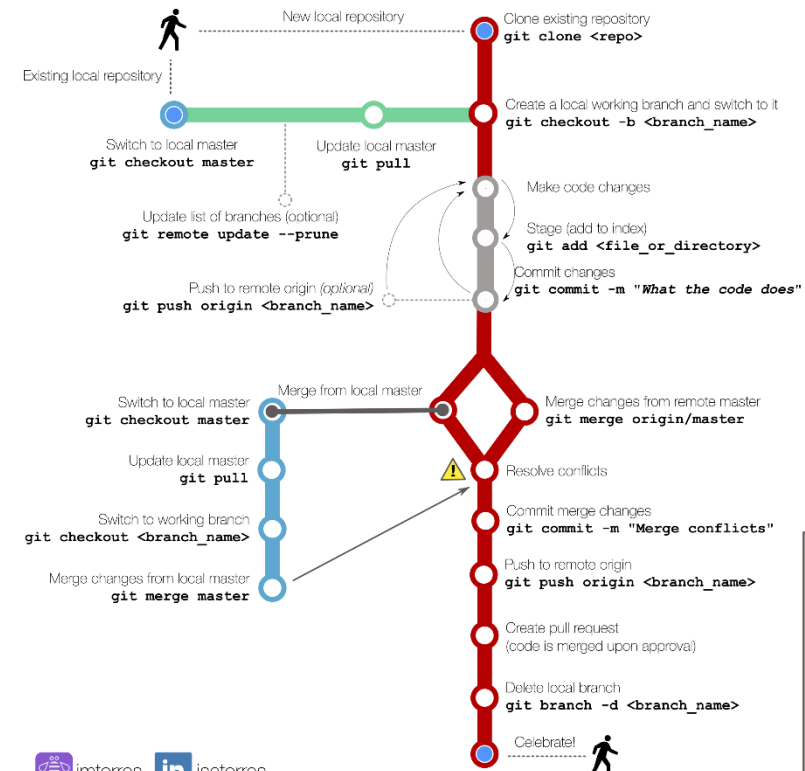
push - publish changes to remote

pull - merge remote changes into local changes

pull request - code review process to allow a change be integrated to the master branch

Simple flow of getting things done with Git

Scenario: developer working in individual branch, team code (stable) lives in the master branch



FREE!

Commands that you can use anytime

- See the status of the working branch
git status
- See the commit history
git log
- See all branches (remote - since last remote update)
git branch -a

Resolve conflicts

No need to panic! Use a merge tool, or...

```
(remove) <<<<<< HEAD
(merge)   your local code
(remove)  =====
(merge)   code in master
(remove)  >>>>>> master
```

Mark the conflict resolved
git add <file>

Want to start over? Roll back all the merged changes
git reset --hard HEAD



Git Cheat Sheet



Create a Repository

From scratch -- Create a new local repository

```
$ git init [project name]
```

Download from an existing repository

```
$ git clone my_url
```

Observe your Repository

List new or modified files not yet committed

```
$ git status
```

Show the changes to files not yet staged

```
$ git diff
```

Show the changes to staged files

```
$ git diff --cached
```

Show all staged and unstaged file changes

```
$ git diff HEAD
```

Show the changes between two commit ids

```
$ git diff commit1 commit2
```

List the change dates and authors for a file

```
$ git blame [file]
```

Show the file changes for a commit id and/or file

```
$ git show [commit]:[file]
```

Show full change history

```
$ git log
```

Show change history for file/directory including diffs

```
$ git log -p [file/directory]
```

Working with Branches

List all local branches

```
$ git branch
```

List all branches, local and remote

```
$ git branch -av
```

Switch to a branch, my_branch, and update working directory

```
$ git checkout my_branch
```

Create a new branch called new_branch

```
$ git branch new_branch
```

Delete the branch called my_branch

```
$ git branch -d my_branch
```

Merge branch_a into branch_b

```
$ git checkout branch_b
```

```
$ git merge branch_a
```

Tag the current commit

```
$ git tag my_tag
```

Make a change

Stages the file, ready for commit

```
$ git add [file]
```

Stage all changed files, ready for commit

```
$ git add .
```

Commit all staged files to versioned history

```
$ git commit -m "commit message"
```

Commit all your tracked files to versioned history

```
$ git commit -am "commit message"
```

Unstages file, keeping the file changes

```
$ git reset [file]
```

Revert everything to the last commit

```
$ git reset --hard
```

Synchronize

Get the latest changes from origin (no merge)

```
$ git fetch
```

Fetch the latest changes from origin and merge

```
$ git pull
```

Fetch the latest changes from origin and rebase

```
$ git pull --rebase
```

Push local changes to the origin

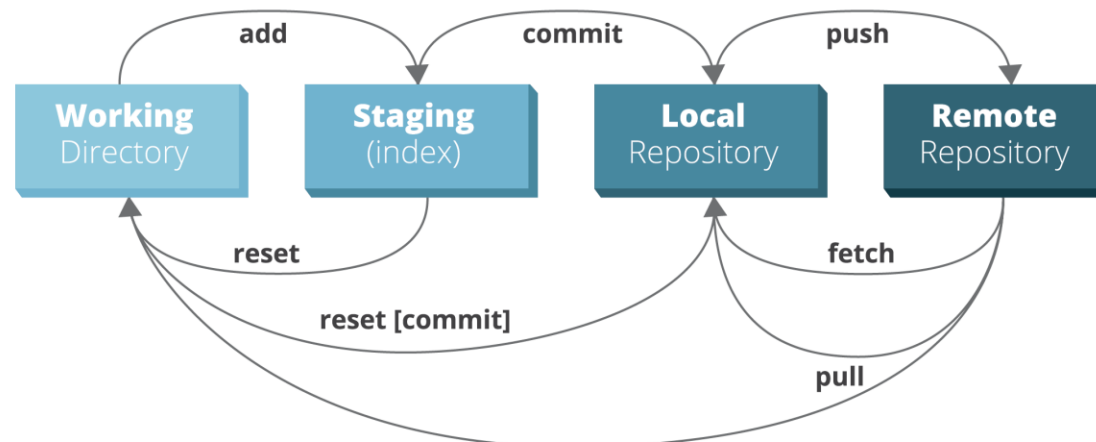
```
$ git push
```

Finally!

When in doubt, use git help

```
$ git command --help
```

Or visit <https://training.github.com/> for official GitHub training.



Git Extensions

The screenshot displays the Git Extensions application window, titled "gitextensions (master) - Git Extensions [DEBUG]". The interface includes a menu bar (Start, Repository, Navigate, View, Commands, GitHub, Plugins, Tools, Help) and a toolbar with icons for file operations and repository management. The left sidebar shows a tree view of the repository structure, including branches (72) and tags (170). The main area displays a commit history table with columns for commit message, author, date, and hash. The selected commit is highlighted in blue.

Commit Message	Author	Date	Hash
5834-skip-grid-refresh-after-navigational-script	RussKie	14 hours ago	29157f3c
origin/master Merge pull request #6261 from pmiossec/fix_unittest	RussKie	14 hours ago	ec51e441
pmiossec/fix_unittest (Try to) Fix unit test that fails only on CI server	Philippe Miossec	21 hours ago	1ca4fbc3
pmiossec/better_download_links README: Improve download links	Philippe Miossec	1 day ago	8e6ea009
DmitryZhelнин/fix-6221-dashboard-drag-and-drop Add ability to drag and drop a f...	Dmitry Zhelnin	1 day ago	93df11ac
Merge pull request #6259 from DmitryZhelнин/update-contributors-txt	RussKie	1 day ago	3d8a57c1
DmitryZhelнин/update-contributors-txt Signed off the project's Contributors Certific...	Dmitry Zhelnin	1 day ago	e18c8fd8
Sign developer certiciate of origin (#6258)	Donatas	1 day ago	4ef634e8
Merge pull request #6199 from pmiossec/show-remote-branch_updated	RussKie	2 days ago	5cd7fd89
Update changelog	RussKie	2 days ago	a338f607

The bottom panel shows the details of the selected commit (8e6ea009):

- Author: Philippe Miossec <pmiossec@gmail.com>
- Author date: 1 day ago (17/2/19 12:24:07 am)
- Commit date: 23 hours ago (17/2/19 11:28:22 pm)
- Commit hash: 8e6ea009bf99033c5e035d9384c0bdc9ce8fbef3
- Parent: 3d8a57c155

The commit message is "README: Improve download links". The commit includes a README file with the following content:

```
by:
* pointing towards the last release
* going directly to the bottom of the page (a little hacky :)
* adding a link to install with Chocolatey
```

Partly fix #6252

Related links: [View on GitHub](#), [Issue 6252](#)

Contained in no branch

Contained in no tag

Derives from tag: [v3.00.00-rc2](#) + 186 commits

Git Extension

