ParMooN (Parallel Mathematical Object Oriented Numerics)

Sashikumaar Ganesan

jointly with ...

Sangeeta Yadav,

Computational Mathematics Group
Department of Computational and Data Sciences
Indian Institute of Science, Bangalore, India

Version 1.0 (2019)

Version Control System

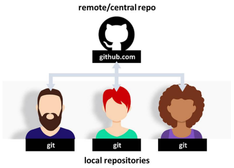
It is a software that helps software developers to work together and maintain a complete history of their work.The functions of a VCS are as follows:

1. Allows developers to work simultaneously.
2. Does not allow overwriting each others changes.
3. Maintains a history of every version.

Types of VCS :

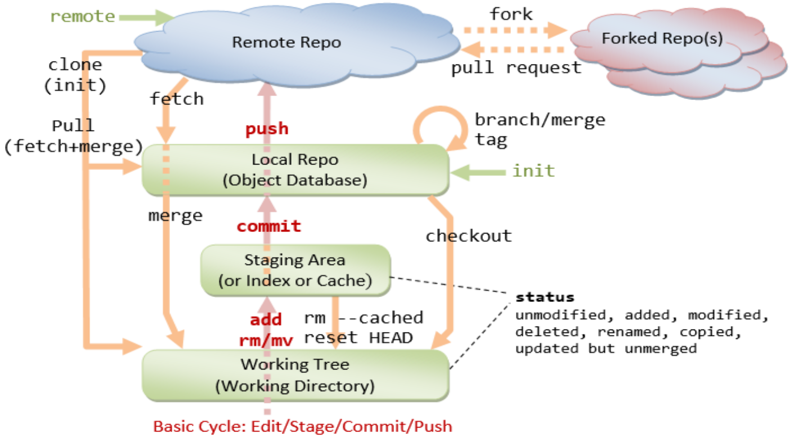
1. Centralized version control system (CVCS).
2. Distributed/Decentralized version control system (DVCS).

Git is a distributed version-control system for tracking changes in source code during software development.

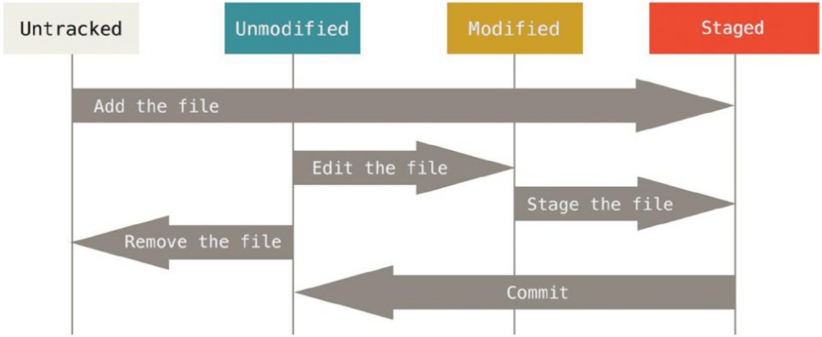


INSTALLATION

- **Fedora :** yum install git
- **Ubuntu :** sudo apt-get install git
- **Windows :** Just go to http://git-scm.com/download/win
- Set your username and password.
  - $ git config –global user.name ”John Doe”
  - $ git config –global user.email johndoe@example.com


remote
Remote Repo
fork
Forked Repo(s)
pull request
clone
(init)
fetch
Pull
(fetch+merge)
push
branch/merge
tag
Local Repo
(Object Database)
init
merge
commit
checkout
Staging Area
(or Index or Cache)
status
unmodified, added, modified,
deleted, renamed, copied,
updated but unmerged
add
rm/mv
rm --cached
reset HEAD
Working Tree
(Working Directory)
Basic Cycle: Edit/Stage/Commit/Push

THE LIFECYCLE OF THE STATUS OF YOUR FILES



COMMANDS

git init [repository name]
This command is used to start a new repository.

git clone [url]
This command is used to obtain a repository from an existing URL.

git add [file]
This command adds a file to the staging area.

git add *
This command adds one or more to the staging area.

git commit -m ”[Type in the commit message]”
This command records or snapshots the file permanently in the version history.

git commit -a
This command commits any files you’ve added with the git add command and also commits any files you’ve changed since then.

COMMANDS

git diff
This command shows the file differences which are not yet staged.

git diff -staged
This command shows the differences between the files in the staging area and the latest version present.

git diff [first branch] [second branch]
This command shows the differences between the two branches mentioned.

git reset
This command unstages the file, but it preserves the file contents.

git reset [commit]
This command undoes all the commits after the specified commit and preserves the changes locally.

git reset -hard [commit]
This command discards all history and goes back to the specified commit.

Commands

git status
This command lists all the files that have to be committed.

git rm [file]
This command deletes the file from your working directory and stages the deletion.

git log
This command is used to list the version history for the current branch.

git log -follow[file]
This command lists version history for a file, including the renaming of files also.

git show [commit]
This command shows the metadata and content changes of the specified commit.

Commands

git tag [commitID]
This command is used to give tags to the specified commit.

git branch
This command lists all the local branches in the current repository.

git branch [branch name]
"This command creates a new branch."

git branch -d [branch name]
This command deletes the feature branch.

git checkout [branch name]
This command is used to switch from one branch to another.

git checkout -b [branch name]
This command creates a new branch and also switches to it.

git merge [branch name]
This command merges the specified branch?s history into the current branch.

COMMANDS

git remote add [variable name] [Remote Server Link]
This command is used to connect your local repository to the remote server.

git push [variable name] master
This command sends the committed changes of master branch to your remote repository.

git push [variable name] [branch]
This command sends the branch commits to your remote repository.

git push -all [variable name]
This command pushes all branches to your remote repository.

git push [variable name] :[branch name]
This command deletes a branch on your remote repository.

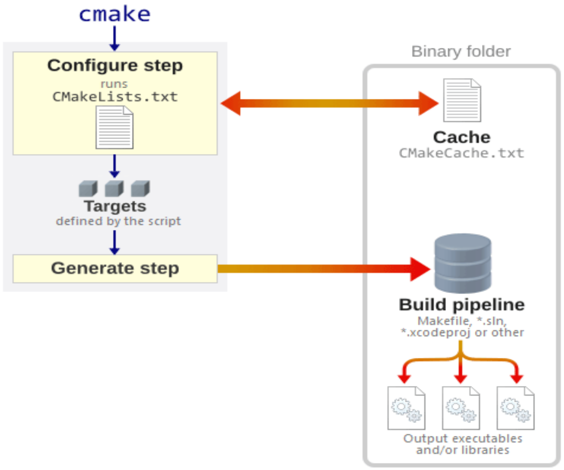
git pull [Repository Link]
This command fetches and merges changes on the remote server to your working directory.

CMAKE

- It is an extensible, open-source system that manages the build process in an operating system and in a compiler-independent manner.
- It can generate a native build environment that will compile source code, create libraries, generate wrappers and build executables in arbitrary combinations
- Users can configure builds through a GUI
- It supports static and dynamic library builds.
- It helps to manage and build your source codes effectively

SOURCE AND BUILD TREES

- The Source Tree contains:
  - CMake input files (CMakeLists.txt)
  - Program source files (hello.cxx)
- The Binary Tree (build tree) contains:
  - Native build system files (hello.dsp)
  - Program libraries and executables (hello.exe)
- Source and Binary trees may be:
  - In the same directory (in-source build)
  - In different directories (out-of-source build)


cmake
Configure step
runs
CMakeLists.txt
Targets
defined by the script
Generate step
Binary folder
Cache
CMakeCache.txt
Build pipeline
Makefile, *.sln,
*.xcodeproj or other
Output executables
and/or libraries

Installation

```
# For Ubuntu

$ sudo apt-get install cmake

# For Redhat

$ yum install cmake

# For Mac OS X with Macports

$ sudo port install cmake
```

THE CMAKE CACHE

- Represents build configuration
- Populated by CMake code
- Stored in CMakeCache.txt at top of build
- Entries have a type to help the GUI
- Holds global information for CMake code
- Updated by CMake configuration phase

CMAKE FILES IN PARMOON

/ParMooN/ParMooN/CMakeLists.txt
/ParMooN/ParMooN/UserConfig.cmake

DIFFERENT USES OF CMAKE IN PARMOON

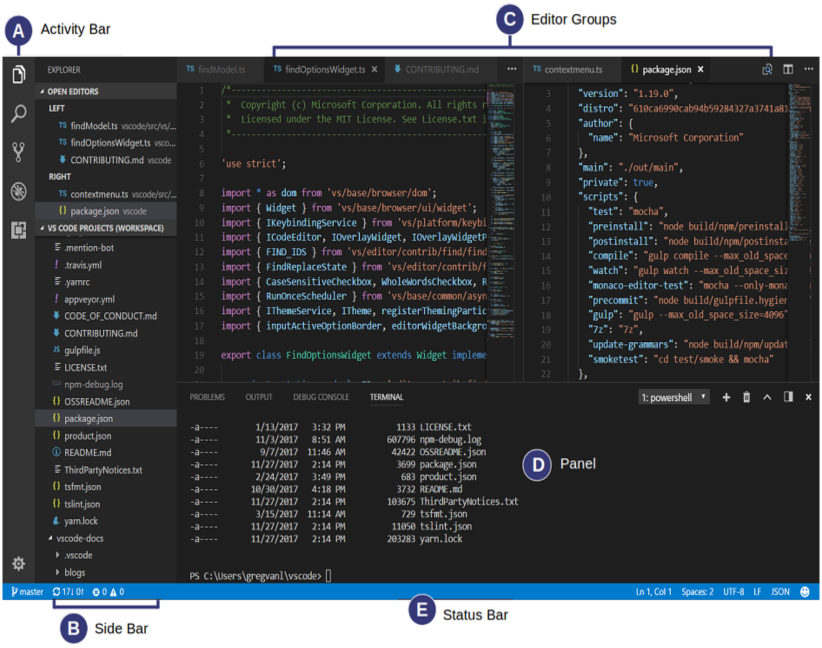
- Include main program
- Mention the output directory
- Select the architecture type
- Select the parallel type
- Select the Operating system
- Link static and dynamic libraries

VS CODE

- Developed by Microsoft for Windows, Linux and macOS.
- Includes support for debugging, embedded Git control and GitHub, syntax highlighting, intelligent code completion, snippets, and code refactoring.
- It is highly customizable, allowing users to change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.
- It is free and open source and released under the permissive MIT License.

Basic Layout

1. Editor - The main area to edit your files.
2. Side Bar - Contains different views like the Explorer to assist you while working on your project.
3. Status Bar - Information about the opened project and the files you edit.
4. Activity Bar - Located on the far left-hand side, this lets you switch between views and gives you additional context-specific indicators, like the number of outgoing changes when Git is enabled.
5. Panels - You can display different panels below the editor region for output or debug information, errors and warnings, or an integrated terminal. Panel can also be moved to the right for more vertical space.



```
/*---------------------------------------------------------------
 *  Copyright (c) Microsoft Corporation. All rights r
 *  Licensed under the MIT License. See License.txt i
 *--------------------------------------------------------------

'use strict';

import * as dom from 'vs/base/browser/dom';
import { Widget } from 'vs/base/browser/ui/widget';
import { IKeybindingService } from 'vs/platform/keybi
import { ICodeEditor, IOverlayWidget, IOverlayWidgetP
import { FIND_IDS } from 'vs/editor/contrib/find/find
import { FindReplaceState } from 'vs/editor/contrib/f
import { CaseSensitiveCheckbox, WholeWordsCheckbox, R
import { RunOnceScheduler } from 'vs/base/common/asyn
import { IThemeService, ITheme, registerThemingPartic
import { inputActiveOptionBorder, editorWidgetBackgro

export class FindOptionsWidget extends Widget impleme
```

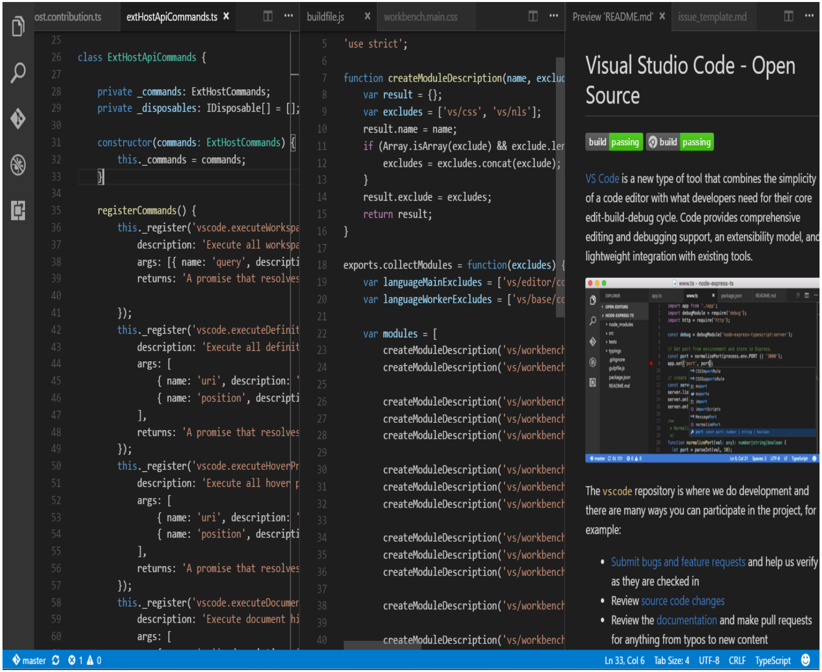
```
"version": "1.19.0",
"distro": "610ca6990cab94b59284327a3741a81
"author": {
  "name": "Microsoft Corporation"
},
"main": "./out/main",
"private": true,
"scripts": {
  "test": "mocha",
  "preinstall": "node build/npm/preinstall
  "postinstall": "node build/npm/postinsta
  "compile": "gulp compile --max_old_space
  "watch": "gulp watch --max_old_space_siz
  "monaco-editor-test": "mocha --only-mona
  "precommit": "node build/gulpfile.hygier
  "gulp": "gulp --max_old_space_size=4096"
  "7z": "7z",
  "update-grammars": "node build/npm/updat
  "smoketest": "cd test/smoke && mocha"
},
```

```
-a----       1/13/2017   3:32 PM        1133 LICENSE.txt
-a----       11/3/2017   8:51 AM      607796 npm-debug.log
-a----        9/7/2017  11:46 AM       42422 OSSREADME.json
-a----      11/27/2017   2:14 PM        3699 package.json
-a----       2/24/2017   3:49 PM         683 product.json
-a----      10/30/2017   4:18 PM        3732 README.md
-a----      11/27/2017   2:14 PM      103675 ThirdPartyNotices.txt
-a----       3/15/2017  11:14 AM         729 tsfmt.json
-a----      11/27/2017   2:14 PM       11050 tslint.json
-a----      11/27/2017   2:14 PM      203283 yarn.lock

PS C:\Users\gregvanl\vscode> 
```

SIDE BY SIDE EDITING

1. Alt click on a file in the Explorer.
2. Ctrl+ to split the active editor into two.
3. Open to the Side (Ctrl+Enter) from the Explorer context menu on a file.
4. Click the Split Editor button in the upper right of an editor.
5. Drag and drop a file to any side of the editor region.
6. Ctrl+Enter (macOS: Cmd+Enter) in the Quick Open (Ctrl+P) file list.



ost.contribution.ts | extHostApiCommands.ts × | buildfile.js × | workbench.main.css | Preview 'README.md' × | issue_template.md

```
class ExtHostApiCommands {

    private _commands: ExtHostCommands;
    private _disposables: IDisposable[] = [];

    constructor(commands: ExtHostCommands) {
        this._commands = commands;
    }

    registerCommands() {
        this._register('vscode.executeWorkspa
            description: 'Execute all workspa
            args: [{ name: 'query', descripti
            returns: 'A promise that resolves

        });
        this._register('vscode.executeDefinit
            description: 'Execute all definit
            args: [
                { name: 'uri', description: '
                { name: 'position', descripti
            ],
            returns: 'A promise that resolves
        });
        this._register('vscode.executeHoverPr
            description: 'Execute all hover p
            args: [
                { name: 'uri', description: '
                { name: 'position', descripti
            ],
            returns: 'A promise that resolves
        });
        this._register('vscode.executeDocumer
            description: 'Execute document hi
            args: [
```

```
'use strict';

function createModuleDescription(name, excluc
    var result = {};
    var excludes = ['vs/css', 'vs/nls'];
    result.name = name;
    if (Array.isArray(exclude) && exclude.ler
        excludes = excludes.concat(exclude);
    }
    result.exclude = excludes;
    return result;
}

exports.collectModules = function(excludes) {
    var languageMainExcludes = ['vs/editor/cc
    var languageWorkerExcludes = ['vs/base/cc

    var modules = [
        createModuleDescription('vs/workbench
        createModuleDescription('vs/workbench

        createModuleDescription('vs/workbench
        createModuleDescription('vs/workbench
        createModuleDescription('vs/workbench

        createModuleDescription('vs/workbench
        createModuleDescription('vs/workbench
        createModuleDescription('vs/workbench

        createModuleDescription('vs/workbench
        createModuleDescription('vs/workbench
        createModuleDescription('vs/workbench

        createModuleDescription('vs/workbench

        createModuleDescription('vs/workbench
```

Visual Studio Code - Open Source

build passing | build passing

VS Code is a new type of tool that combines the simplicity of a code editor with what developers need for their core edit-build-debug cycle. Code provides comprehensive editing and debugging support, an extensibility model, and lightweight integration with existing tools.

The vscode repository is where we do development and there are many ways you can participate in the project, for example:

- Submit bugs and feature requests and help us verify as they are checked in
- Review source code changes
- Review the documentation and make pull requests for anything from typos to new content

master ⊗ 1 ⚠ 0 Ln 33, Col 6 Tab Size: 4 UTF-8 CRLF TypeScript

LANGUAGE SPECIFIC EDITOR SETTINGS

Customize the editor by language:

- Run the global command Preferences: Configure Language Specific Settings (command id: workbench.action.configureLanguageBasedSettings) from the Command Palette (Ctrl+Shift+P) which opens the language picker.
- Selecting the required language, opens the Settings editor with the language entry where you can add applicable settings.