

UGit User Guide

Lite Version

Contents

[Introduction](#)

[Installation](#)

[User Interface](#)

[Initialization](#)

[Clone repository](#)

[Manage changes](#)

[Commit and push](#)

[Check remote changes](#)

[Pull remote changes](#)

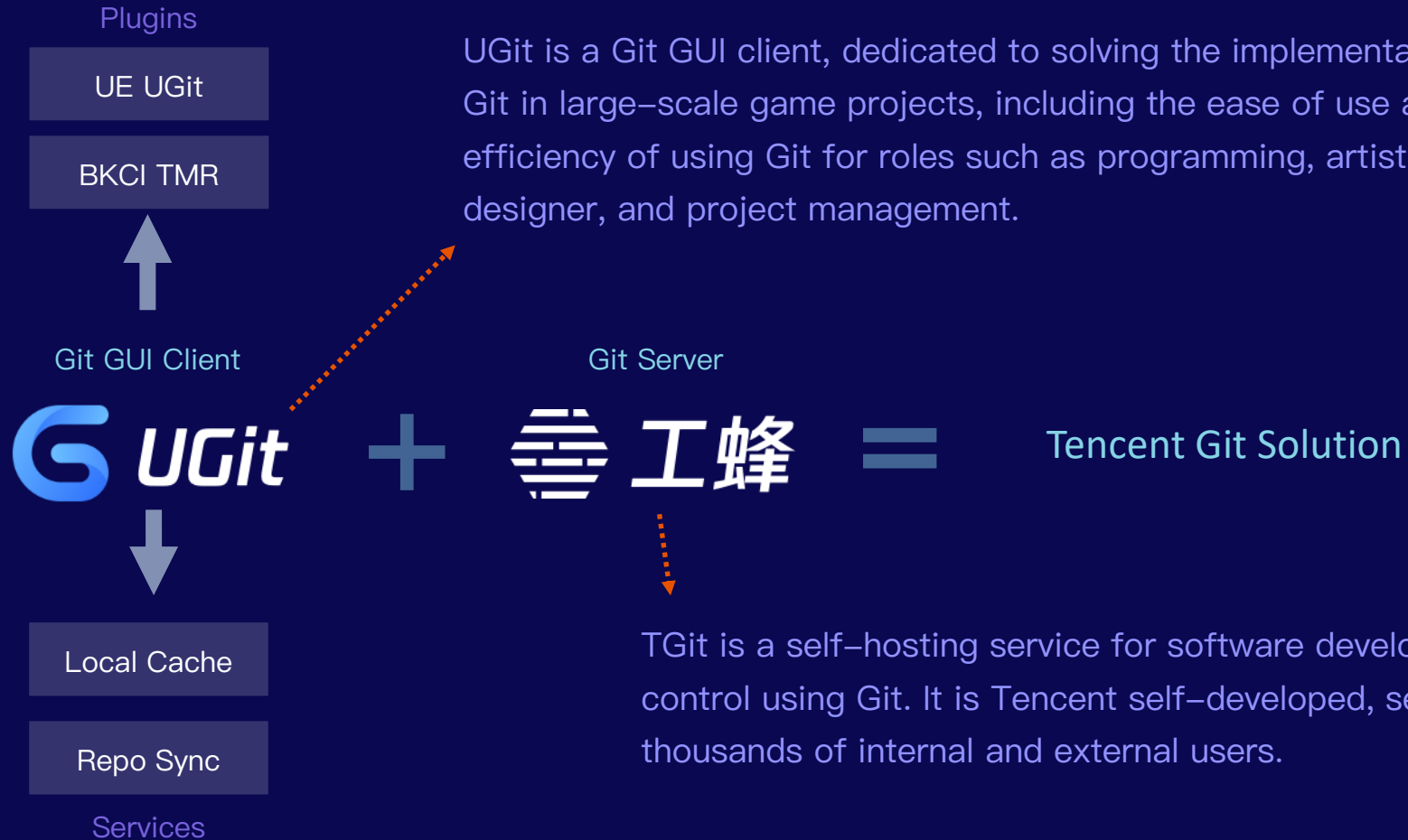
[View commit history](#)

[Resolving conflicts](#)

[Working with branches](#)

Introduction

About UGit



UGit is a Git GUI client, dedicated to solving the implementation of Git in large-scale game projects, including the ease of use and efficiency of using Git for roles such as programming, artist, designer, and project management.

TGit is a self-hosting service for software development and version control using Git. It is Tencent self-developed, serving hundreds of thousands of internal and external users.

Introduction

Extreme Versioning Problems Facing Larger Games



Installation

Download and install (Windows)

<https://ugit.qq.com>, download the Windows package, then double click the package to finish install.

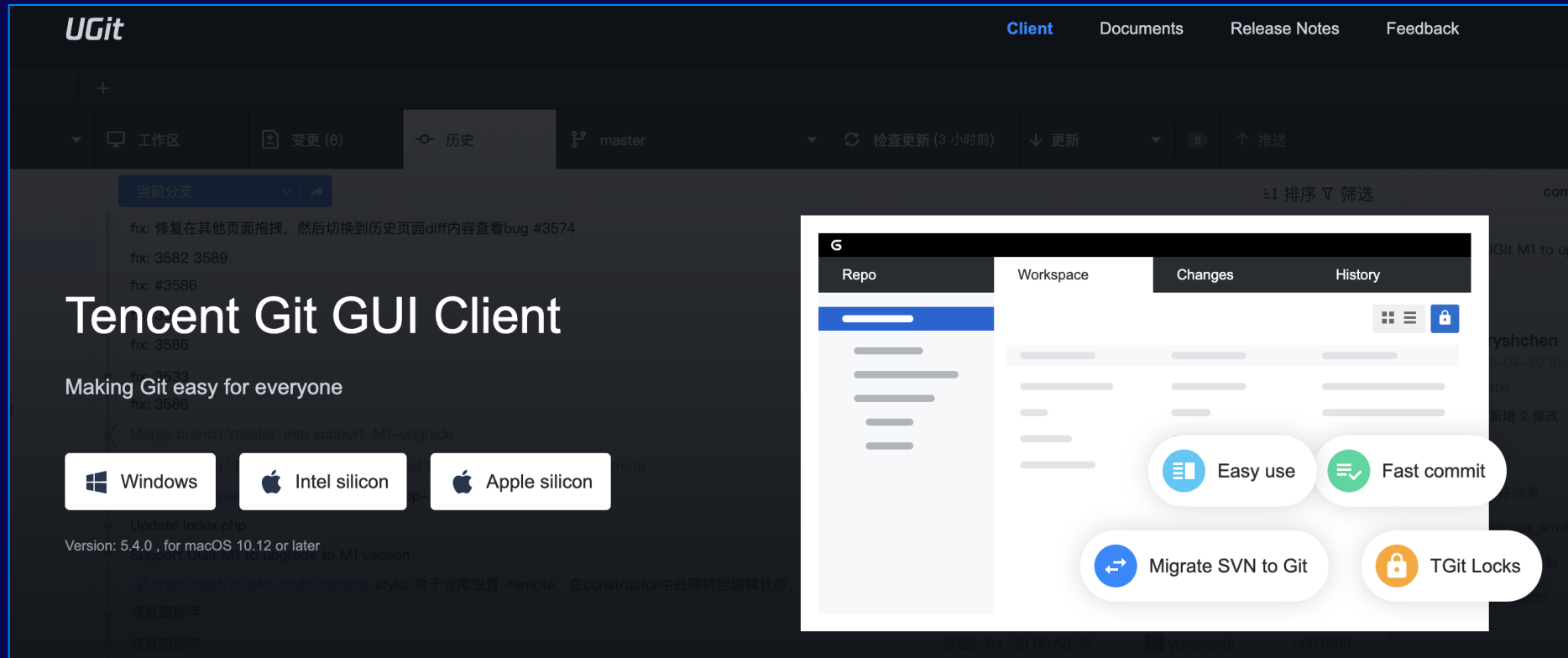
The image displays the Tencent Git GUI Client interface. The main window features a dark theme with a top navigation bar containing the 'UGit' logo and links for 'Client', 'Documents', 'Release Notes', and 'Feedback'. Below the navigation bar, there are tabs for '工作区' (Workspace), '变更 (6)' (Changes), and '历史' (History). The main content area shows a list of commits with details like 'fx: 修复在其他页面拖拽, 然后切换到历史页面diff内容查看bug #3574'. A large white text overlay reads 'Tencent Git GUI Client' and 'Making Git easy for everyone'. Below this, there are three buttons for 'Windows', 'Intel silicon', and 'Apple silicon'. A version string 'Version: 5.4.0, for macOS 10.12 or later' is also visible.

A floating window is overlaid on the main interface, showing a lighter theme with a top bar containing 'G' and tabs for 'Repo', 'Workspace', 'Changes', and 'History'. This window highlights four key features in callout boxes: 'Easy use', 'Fast commit', 'Migrate SVN to Git', and 'TGit Locks'.

Installation

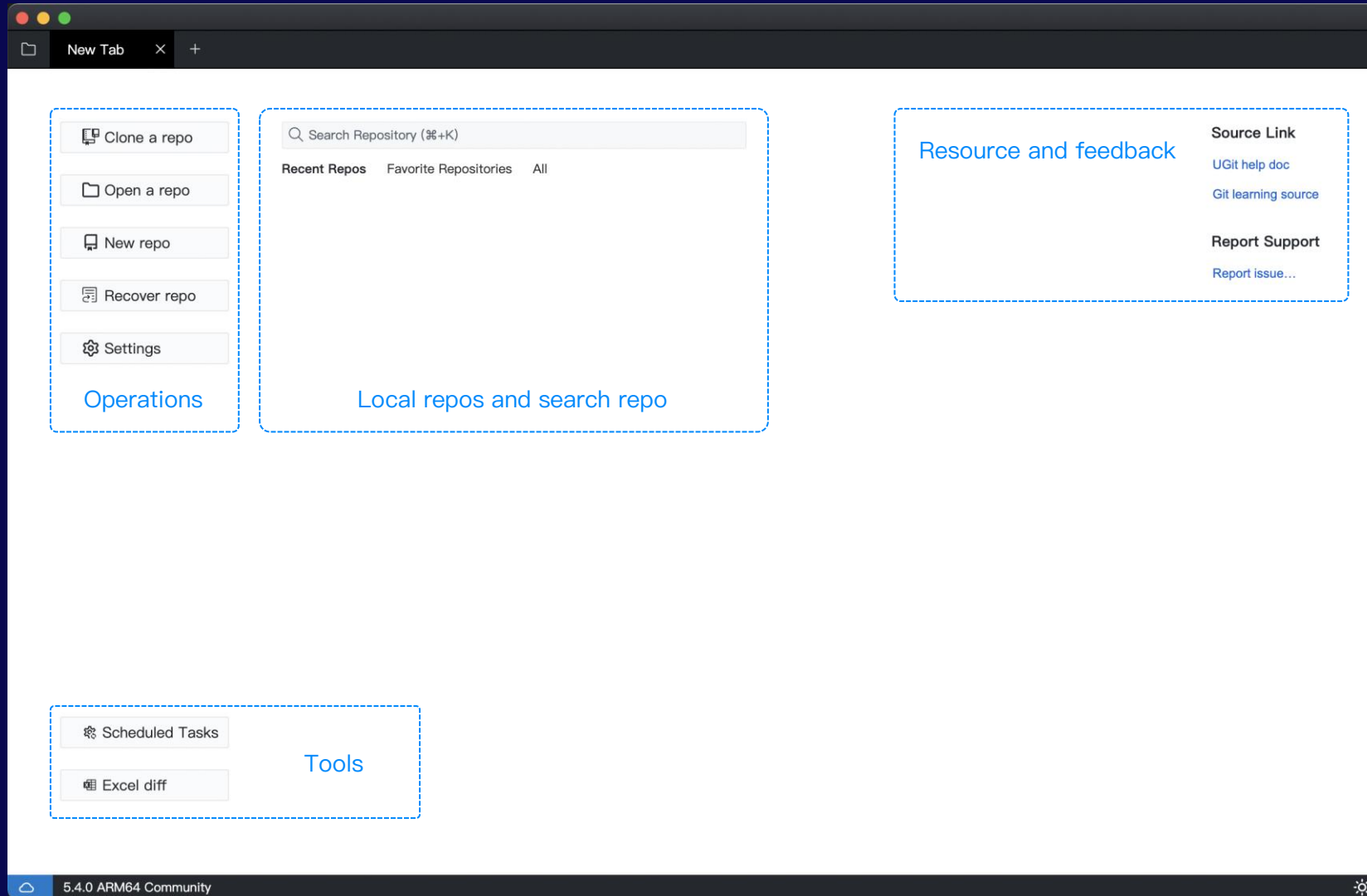
Download and install (macOS)

<https://ugit.qq.com>, download the macOS package, double click to extract, then drag UGit.app to /Application folder.



User Interface

New Tab



User Interface

Repository Tab

Current Repository

Activity Bar

Settings

Remote repository

View

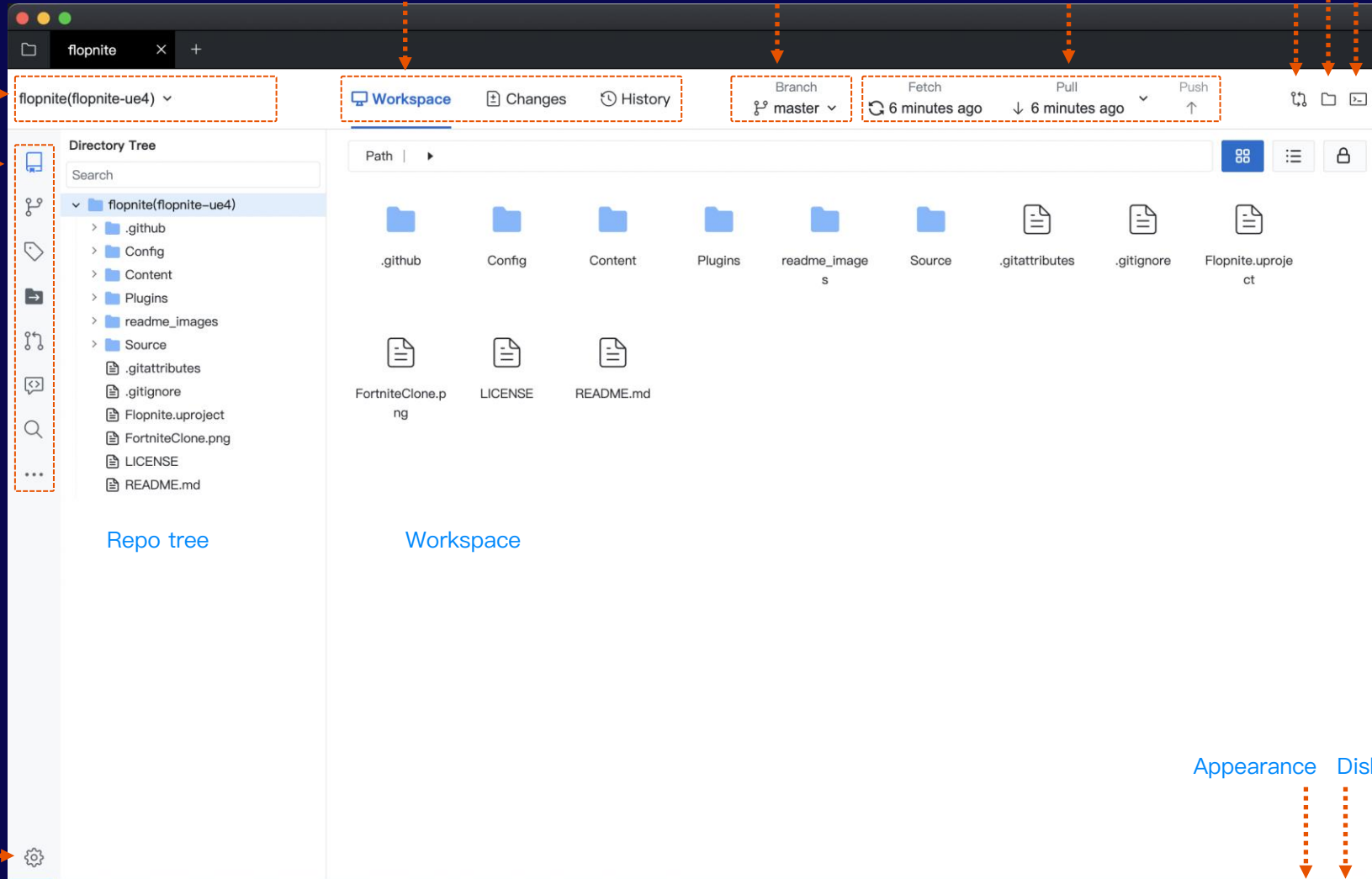
Current Branch

Operations

Branch Compare

Show in Explorer/Finder

Open Terminal



Appearance

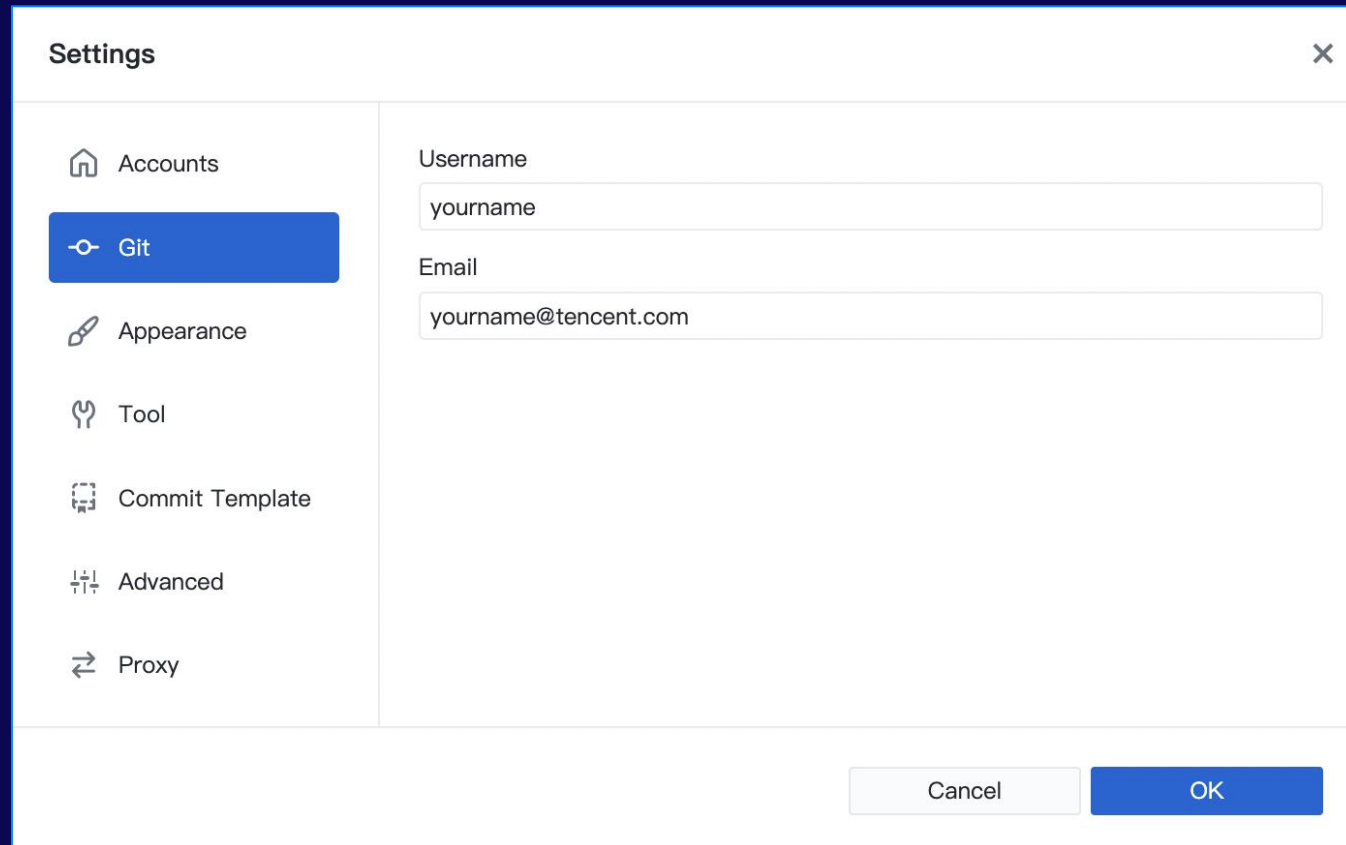
Disk Usage

Operation logs

Initialization

Config git user name and email

Config git user.name and user.email for the first time usage(That is git global config).



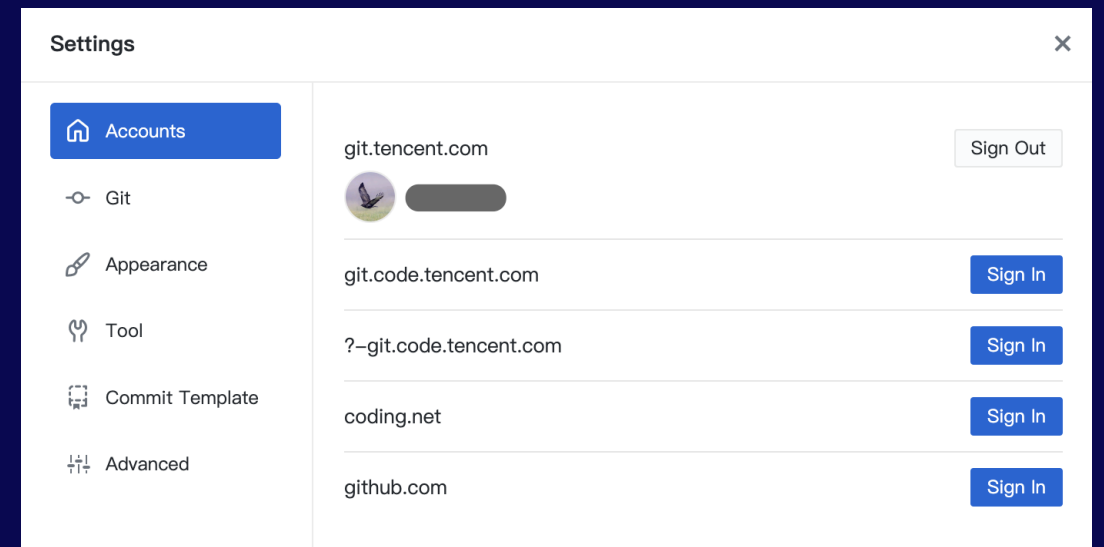
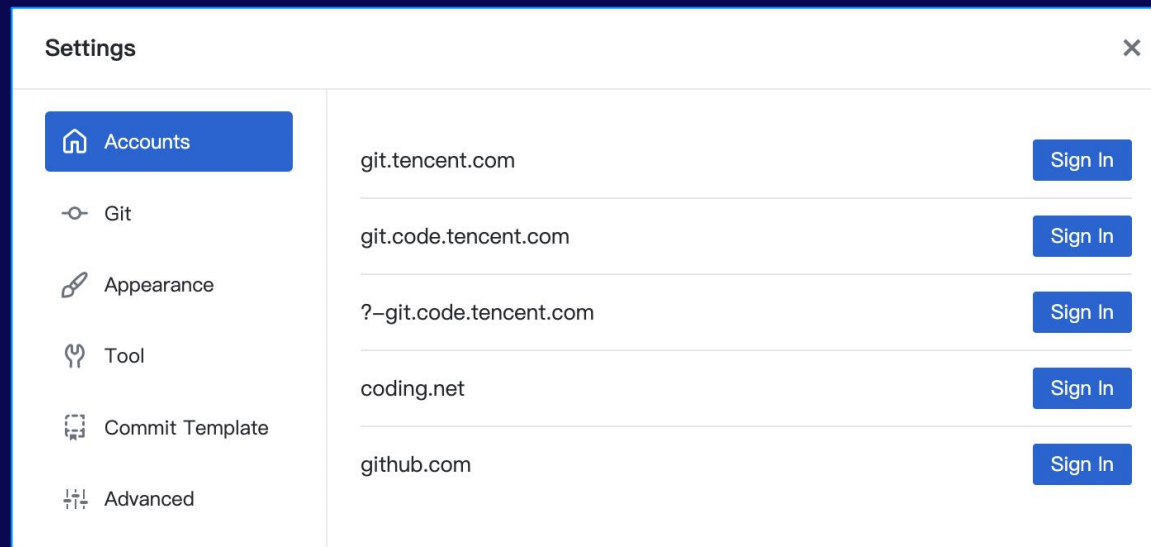
The image shows a 'Settings' dialog box with a sidebar on the left and a main content area on the right. The sidebar contains several menu items: 'Accounts', 'Git' (highlighted in blue), 'Appearance', 'Tool', 'Commit Template', 'Advanced', and 'Proxy'. The main content area has two input fields: 'Username' with the value 'yourname' and 'Email' with the value 'yourname@tencent.com'. At the bottom right, there are two buttons: 'Cancel' and 'OK'.

Category	Field	Value
Git	Username	yourname
	Email	yourname@tencent.com

Initialization

Authenticating git platform

From File>Preferences(macOS: UGit>Preferences), click 'Sign In' to authenticating the platform. After sign in, then you can clone repo from it.
p.s. The git.tencent.com require ip whitelist to access.



Initialization

Init git lfs and git ignore for new repo

The git lfs config is very important for game projects, especially when creating a new git repo, it needs to be properly configured.

Repository Management

Clone **New Repository**

Open New TMR

New

Name
repository name

Description

Local Path
/Users/yunshan/Work/01-workspace Choose...

Initialize this repository with a README

Repo Group
None

Git Ignore [View Template](#)
UnrealEngine

Git LFS [View Template](#)
ue5.all

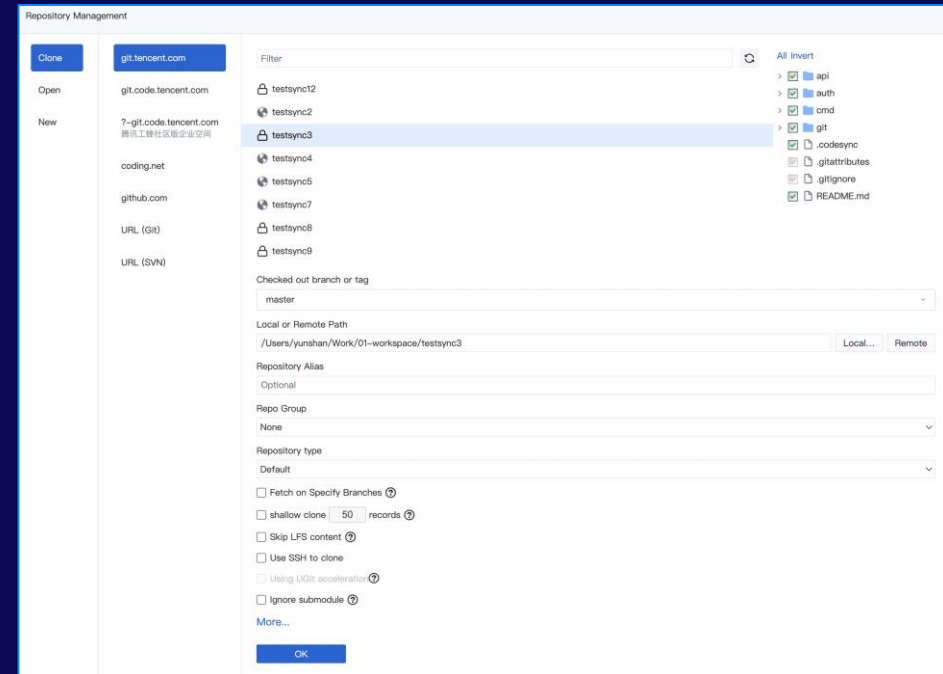
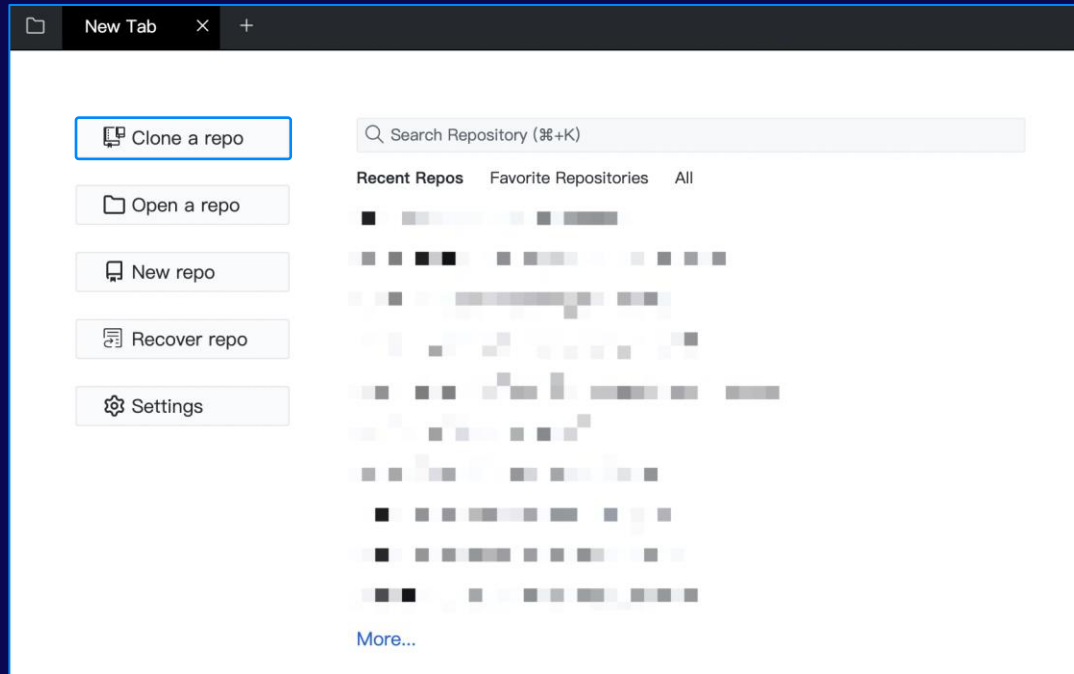
License
None

OK

Clone repository

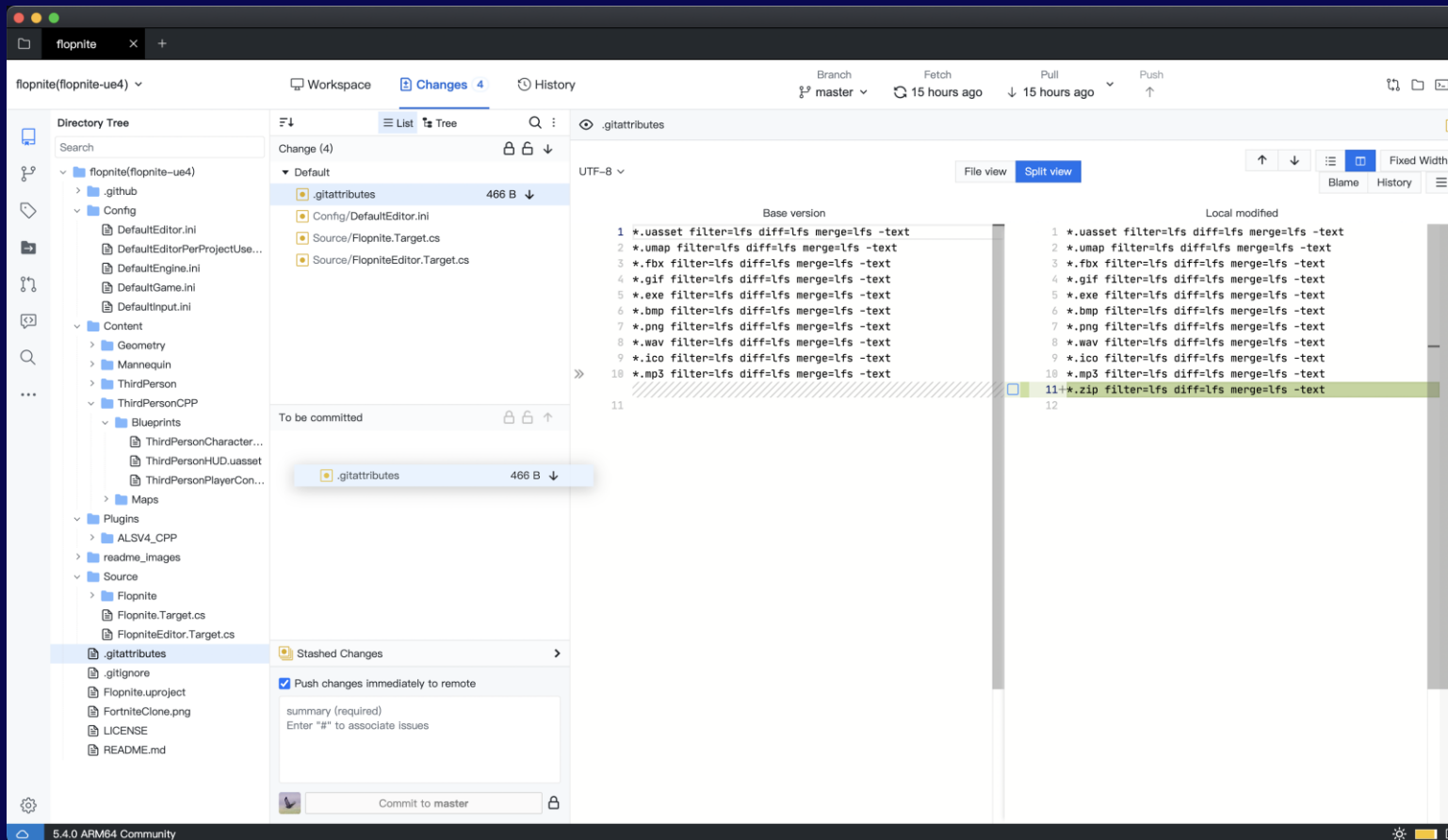
Clone repository to local disk

The git lfs config is very important for game projects, especially when creating a new git repo, it needs to be properly configured.



Manage changes

View changes

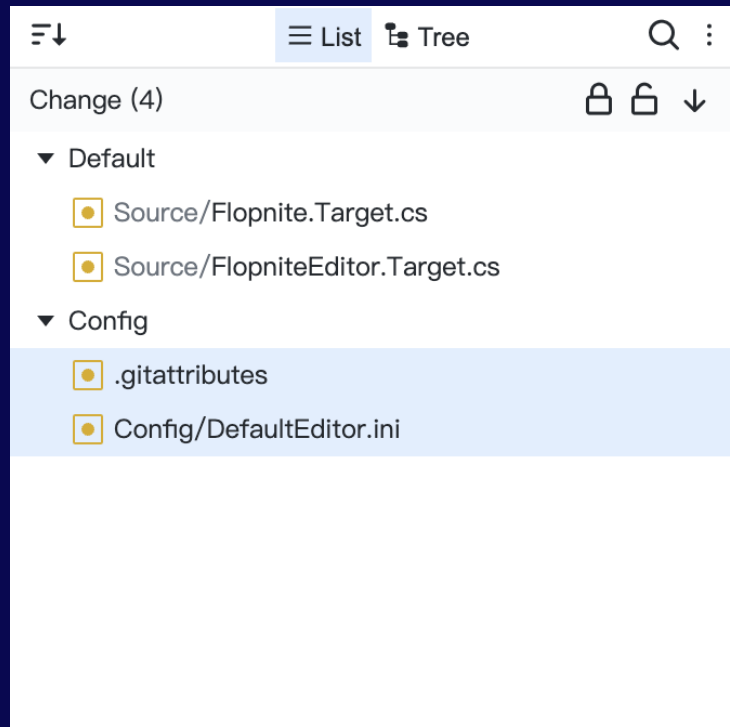
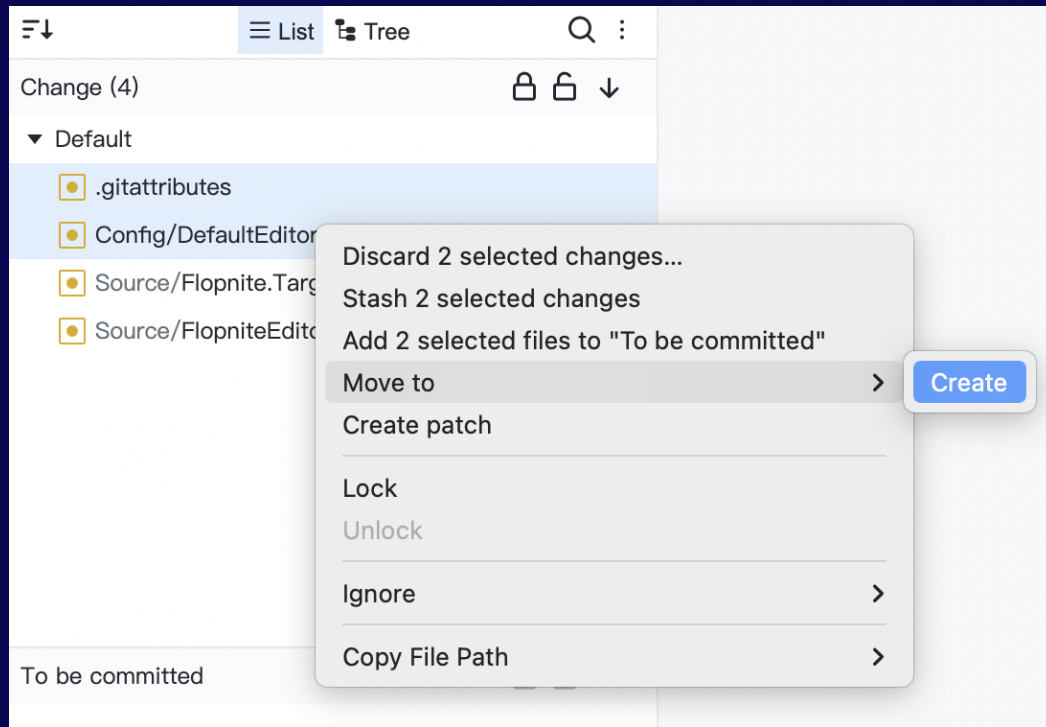


- UGit will detect local changes and put it in 'Change' list.
- Double click file or drag files into 'To be committed' list, then you are ready to commit.

Manage changes

Change group

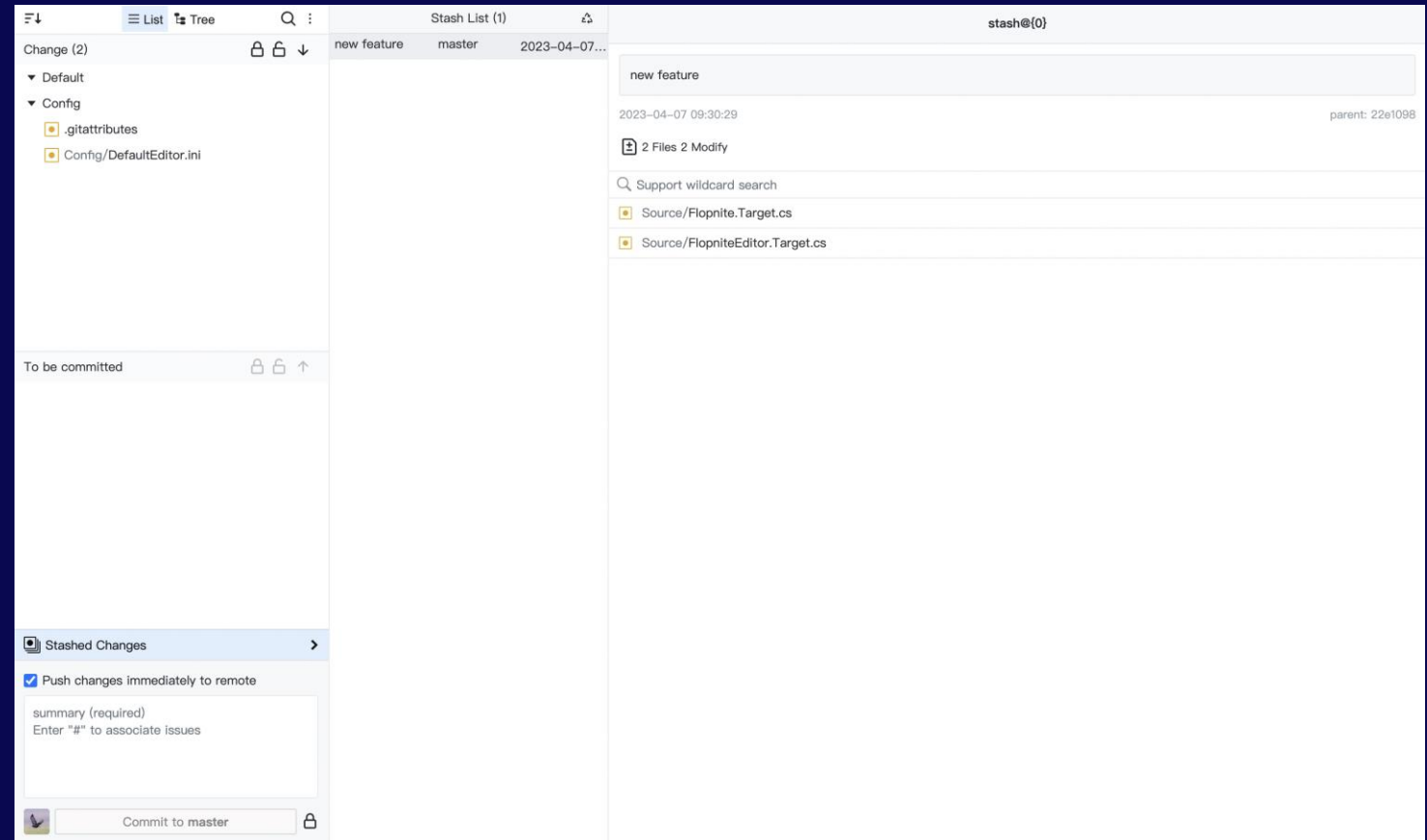
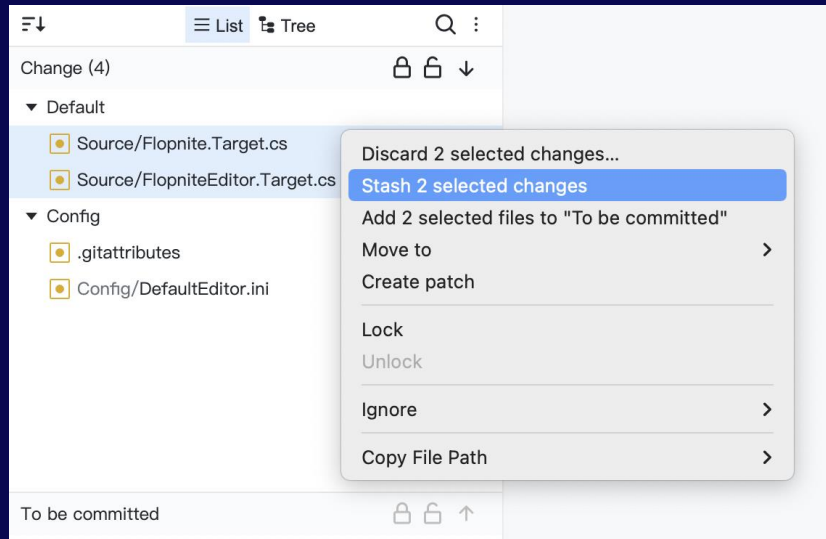
You can better manage changes by creating groups.



Manage changes

Stash changes

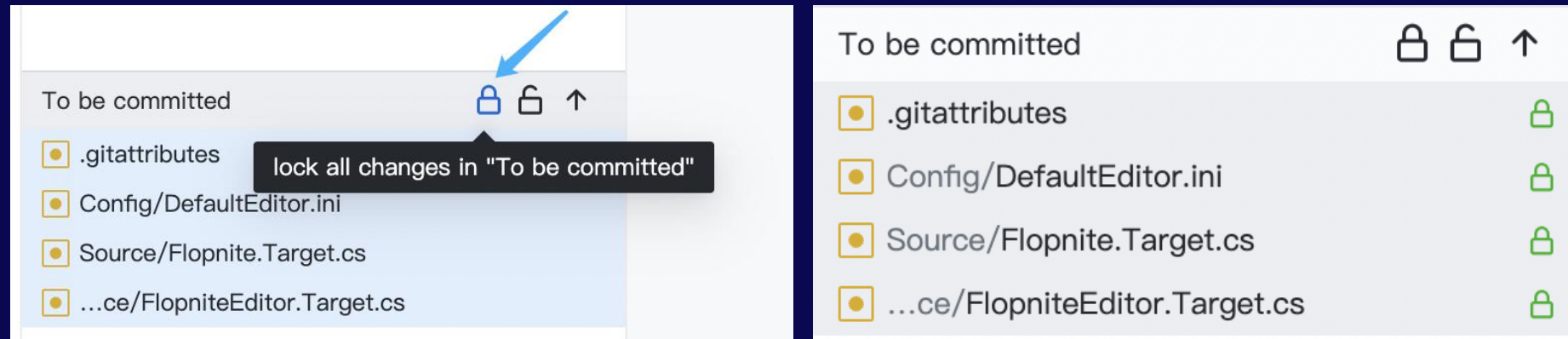
Stash allows you to save temporary changes so that they are not affected by actions such as pulling or switching branches.



Manage changes

Lock changes

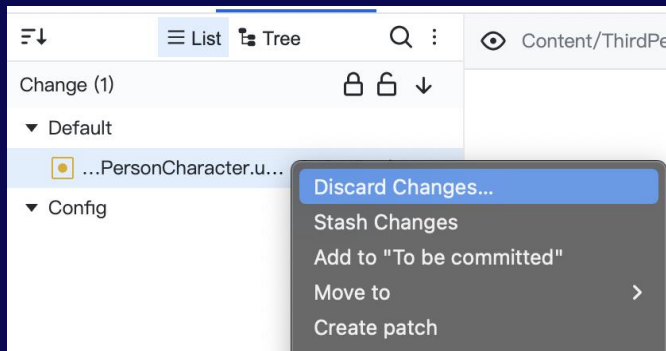
In order to change the binary file more safely, you can select the file you want to change and lock it with one click. Of course, you can also lock the file or directory in the directory tree.



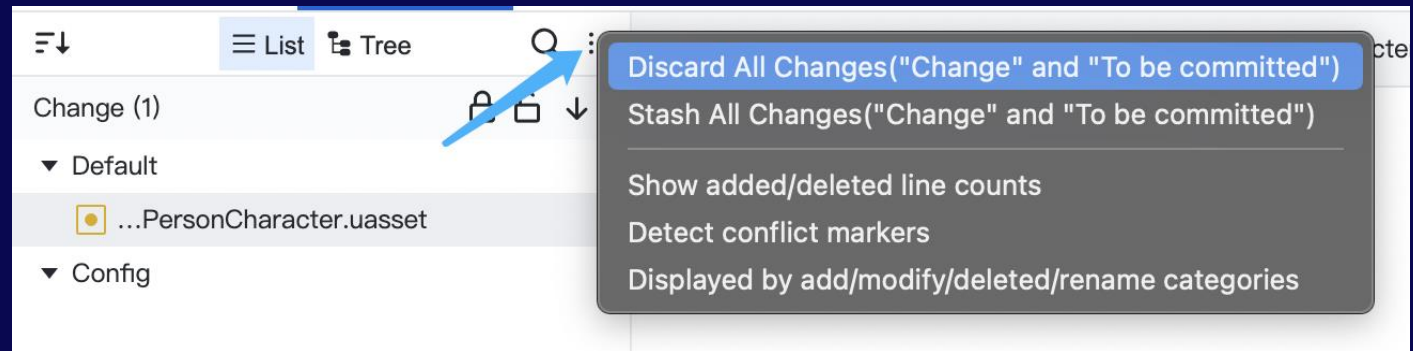
Manage changes

Discard changes

Discard changes of specific file(s).

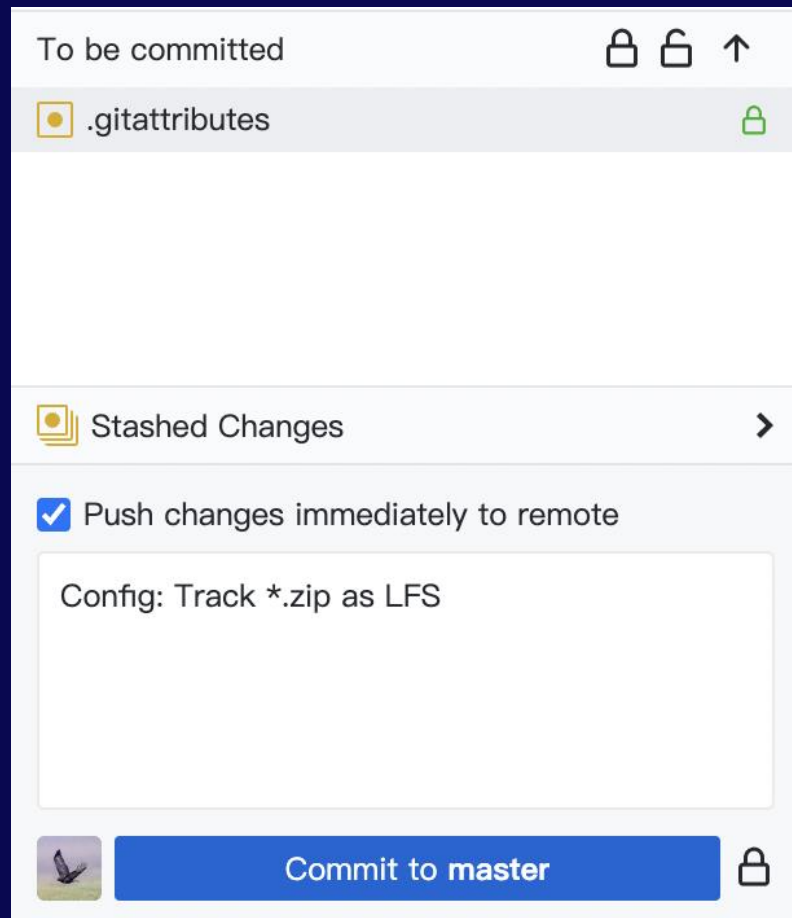


Discard all local changes (Dangerous operation, need attention).



Commit and push

Submit changes



After moving the files from 'Change' to the 'To be committed', fill in the commit message and click 'Commit'.

Note that after checking the 'Push changes immediately to remote' option, the commit and push are complete at one time.

Check remote changes

Fetching



In order to discover remote changes, you need to click the 'Fetch' button, which will allow UGit to synchronize changes from the remote repository to the local repository, but not modify the content of the workspace.

Check remote changes

Change details of remote commits

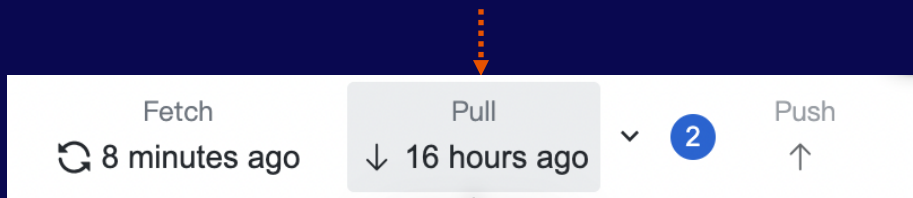
New commits from remote branch

Fetch just now Pull ↓ 16 hours ago **2** Push ↑

The screenshot shows a code editor with a diff view. The left pane shows the original file with a red line indicating a change. The right pane shows the updated file with a green line indicating a change. The change is in the copyright notice, updating the year from 2019 to 2023.

```
1 MIT License
2
3 -Copyright (c) 2019 - 2020 Chris Gong
4
5 Permission is hereby granted, free of charge, to any person obtaining
6 of this software and associated documentation files (the "Software"),
7 in the Software without restriction, including without limitation the
8 to use, copy, modify, merge, publish, distribute, sublicense, and/or
9 copies of the Software, and to permit persons to whom the Software is
10 furnished to do so, subject to the following conditions:
11
12 The above copyright notice and this permission notice shall be includ
13 copies or substantial portions of the Software.
14
15 THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRE
16 IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY
17 FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHA
18 AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHE
19 LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING
20 OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALING
21 SOFTWARE.
```

Pull remote changes



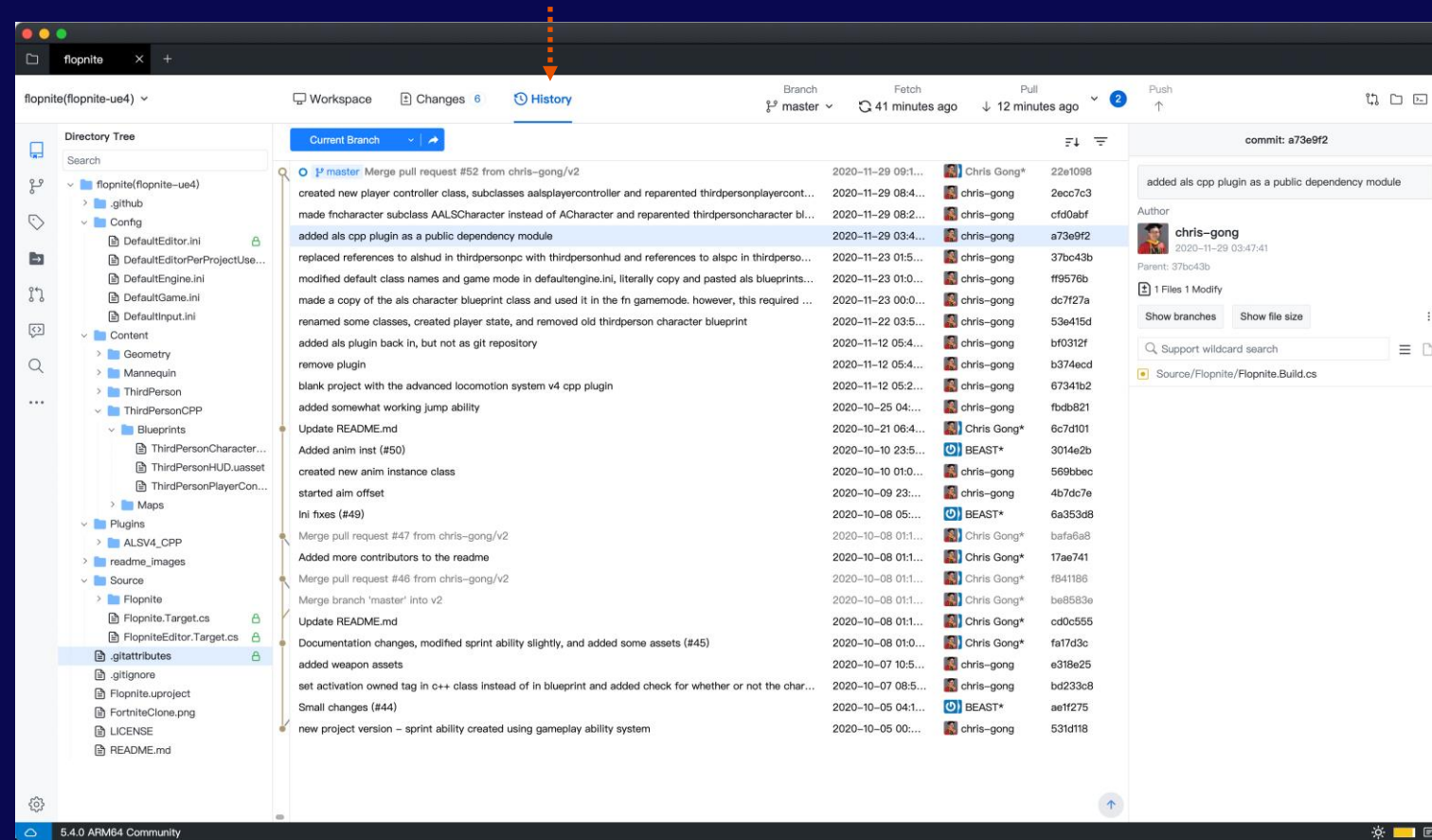
Unlike SVN and P4, Git updates the entire workspace, while SVN and P4 can support directory or file-level updates.

In fact, by default, a git pull operation is equal to a fetch operation plus a merge operation, and the pull operation may cause conflicts, that is, conflicts between local branches and remote branches.

```
✓ Pull
✓ Pull
[10:38:07] Start: git -c credential.helper= pull --progress origin master
[10:38:07] From https://git.tencent.com/game/flopnite-ue4
[10:38:07] * branch master -> FETCH_HEAD
[10:38:07] Updating 22e1098..cad1475
[10:38:07] Fast-forward
[10:38:07] LICENSE | 2 +-
[10:38:07] README.md | 2 +-
[10:38:07] 2 files changed, 2 insertions(+), 2 deletions(-)
[10:38:07] Finish: git -c credential.helper= pull --progress origin master
[10:38:07] Update directory tree...
[10:38:07] Load status...
```

View commit history

Change details of remote commits



By viewing the commit history, you can easily trace who, when, and what you did. Of course, if you do something wrong, you can also quickly roll back through the history.

Resolving conflicts

Binary conflict

For binary conflicts, you can easily choose one version to resolve the conflict.

The screenshot displays a Git GUI interface with a conflict resolution window. At the top, a notification states: "There are file conflicts in current repository, please resolve conflicts [Abort](#) [View conflicts](#)".

Directory Tree: Shows the project structure with the file `Content/ThirdPersonCPP/Blueprints/ThirdPersonCharacter.uasset` selected.

Change (0): Shows the current state with no changes.

Conflict Resolution: A message states: "The following file is modified locally and remotely, and need to be resolved manually".

Conflict Details:

- origin/master(Remote):** Contains "new character" (commit: `yunshandi 2023-04-07 11:57:27 ->7928717`). A blue checkmark is next to this entry, and a blue button "Use origin/master(R...)" is highlighted.
- master(Local):** Contains "test conflict" (commit: `yunshandi 2023-04-07 11:58:54 ->a60ecf6`). A greyed-out button "Use origin/master(Remote)" is next to this entry.

Stashed Changes: Shows "Commit changes immediately after conflict resolved" and "Push changes immediately to remote" both checked.

Commit Summary: A text box for a summary (required) with the instruction "Enter '#' to associate issues".

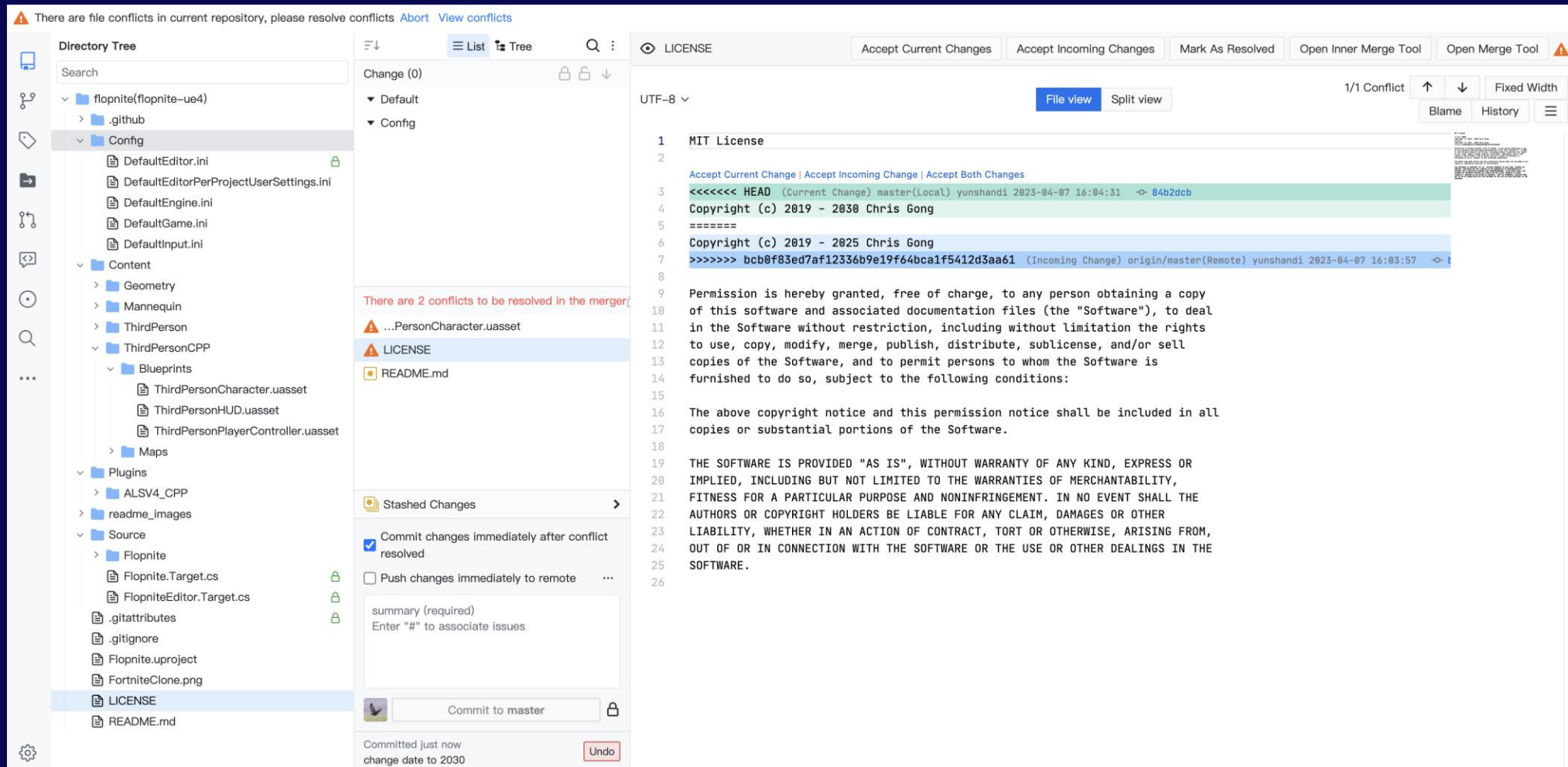
Commit: A "Commit to master" button is visible.

Status: At the bottom, it says "Committed just now test conflict" with an "Undo" button.

Resolving conflicts

Text conflict

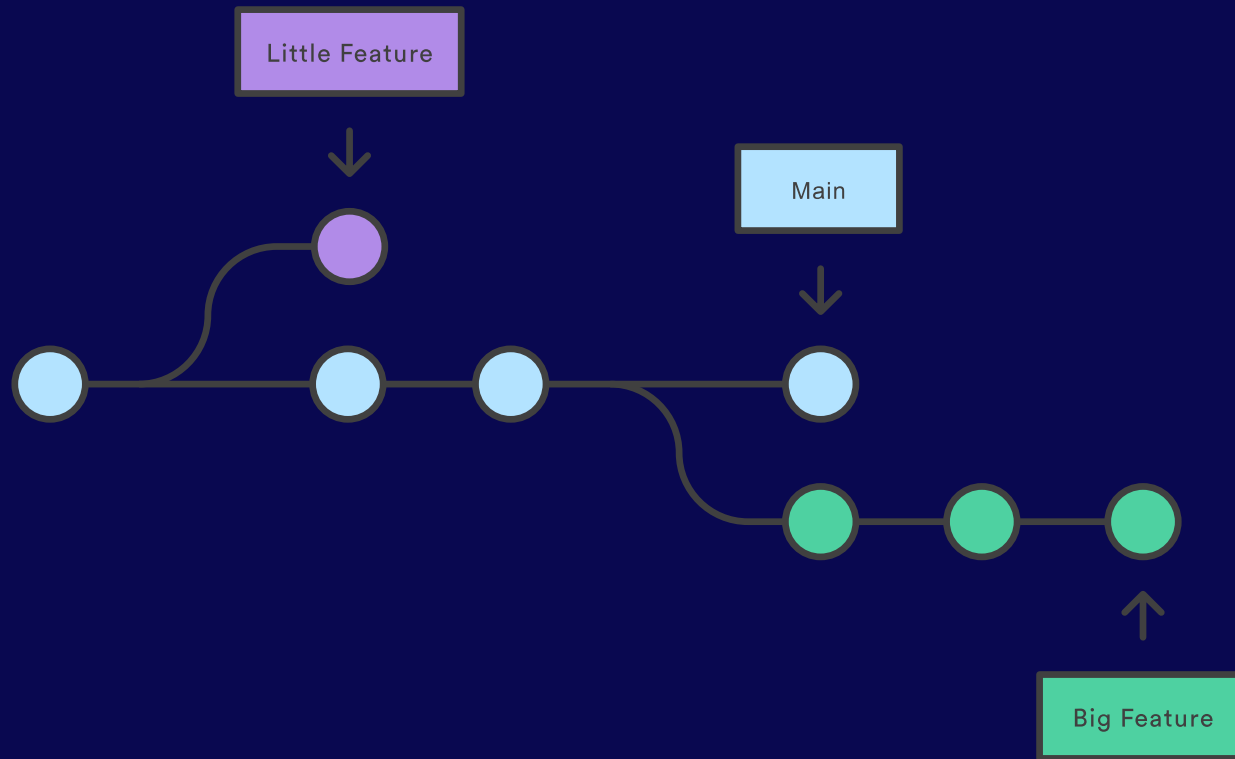
For text conflicts, you can resolve conflict via vscode style interaction.



Working with branches

Branch overview

Branching is a means of isolating changes. When modifying files, it can avoid being affected by other users' commits. Through branches, multiple changes can be developed in parallel.



Working with branches

Creating branch

By selecting a commit in the history, right click to create a branch.

The screenshot shows a version control interface with a top navigation bar containing 'Workspace', 'Changes', and 'History' tabs. The 'History' tab is active. On the right side of the top bar, there are buttons for 'Branch' (set to 'master'), 'Fetch' (4 hours ago), 'Pull' (4 hours ago), and 'Push'. Below the top bar, a 'Current Branch' dropdown is set to '...tent/ThirdPersonCPP/Blueprints'. The main area displays a list of commit messages. The top commit, 'test conflict', is selected, and a context menu is open over it. The context menu includes the following options: 'Create Branch' (highlighted), 'Create Tag', 'Create patch', 'Revert Commit...', 'Reset master to Here', 'Checkout Commit', 'Modify submission information', 'Squash commits', 'Cherry-pick to current branch', 'Cherry-pick to remote branch', 'Copy files in changeset to workspace', 'Historical version rollback / download', and 'Roll back changes to this version'.

Workspace Changes History Branch master Fetch 4 hours ago Pull 4 hours ago 8 Push

Current Branch ...tent/ThirdPersonCPP/Blueprints

test conflict

created new player controller class, subclasses aalsplayercontroller and reparented thirdpersonplayerco...

made fncharacter subclass AALSCharacter instead of ACharacter and reparented thirdpersoncharacter...

replaced references to alshud in thirdpersonpc with thirdpersonhud and references to alspsc in thirdper...

modified default class names and game mode in defaultengine.ini, literally copy and pasted als blueprin...

made a copy of the als character blueprint class and used it in the fn gamemode. however, this require...

renamed some classes, created player state, and removed old thirdperson character blueprint

blank project with the advanced locomotion system v4 cpp plugin

added somewhat working jump ability

started aim offset

new project version – sprint ability created using gameplay ability system

Create Branch

Create Tag

Create patch

Revert Commit...

Reset master to Here

Checkout Commit

Modify submission information

Squash commits

Cherry-pick to current branch

Cherry-pick to remote branch

Copy files in changeset to workspace

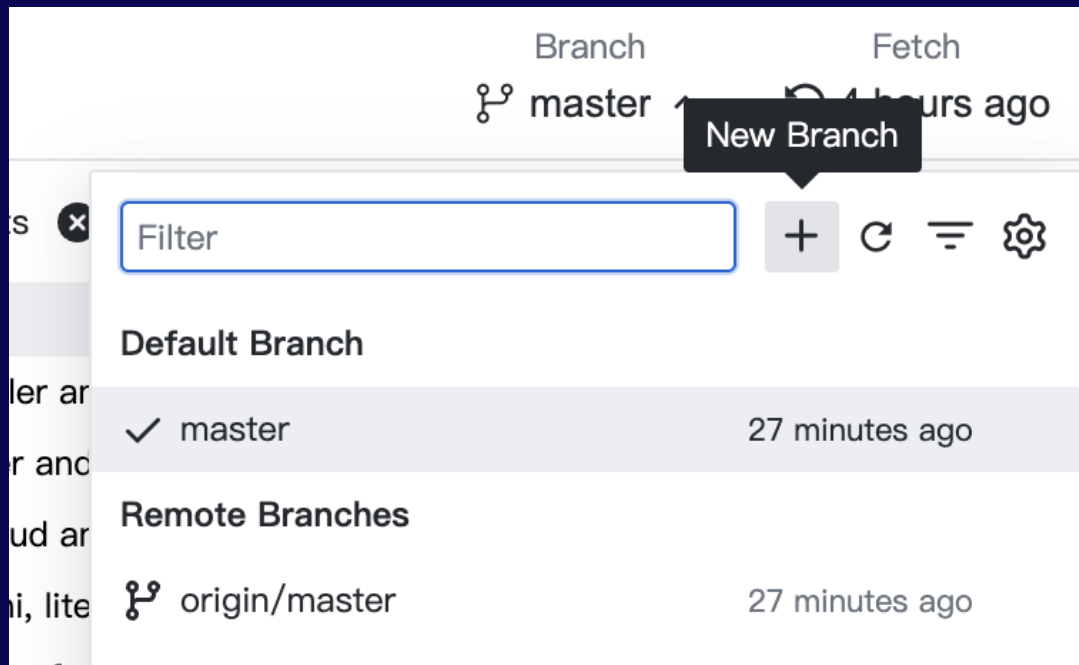
Historical version rollback / download

Roll back changes to this version

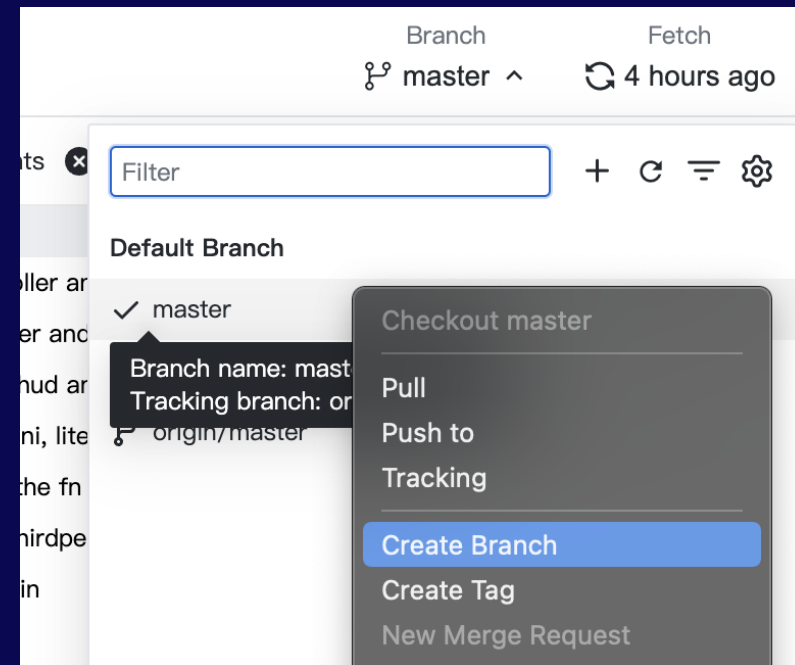
Working with branches

Creating branch

Create branch from the branch list panel, by click '+' button.



Create a new branch from a existing branch.



Working with branches

Merging branch

Before merging branch feat/test_conflict to master, you should with branch master checkedout. Then you can merge feat/test_conflict to master from branch list panel.

