

# 软件开发辅助工具及使用

---

第三届高能物理计算暑期学校

杜然

中国科学院高能物理研究所计算中心

2022-08-18



# 目录

## ● 代码管理工具Git及托管平台

- 什么是代码管理工具？
- 如何安装Git？
- 如何操作Git？
- 如何处理冲突？
- 高能所代码托管平台code.ihep.ac.cn

## ● Markdown语言及编辑平台

- 什么是Markdown？
- 如何写Markdown文档？
- 高能所Markdown文档编辑平台note.ihep.ac.cn
- 本地Markdown文档编辑软件



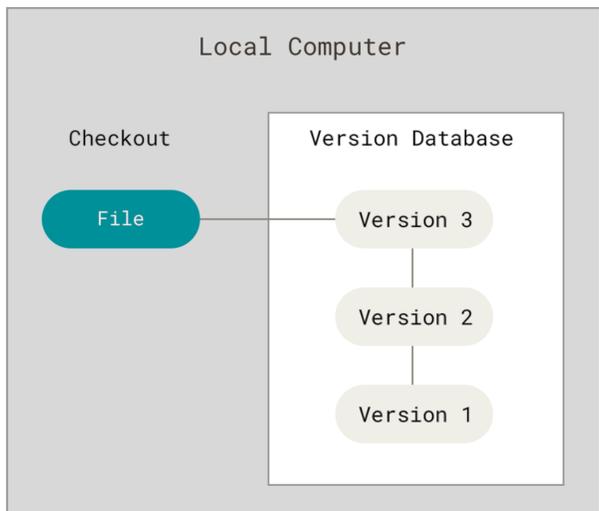
# 什么是代码管理工具？

---

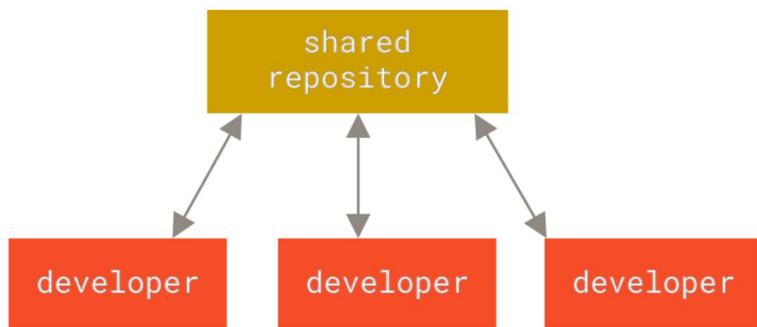


# 代码管理工具

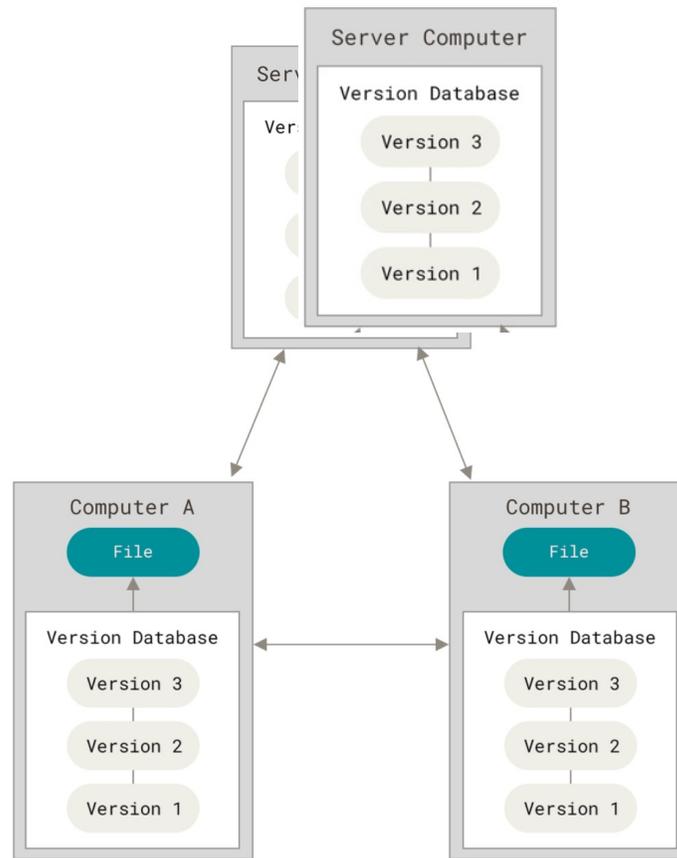
- 功能：记录并管理多个版本的代码文件、多人协作
- 三种模式：开发端和托管端的架构
  - 本地式：RCS
  - 集中式：CVS, Subversion, Perforce
  - 分布式：Git, Mercurial, Bazaar, Darcs



本地版本控制



集中式版本控制

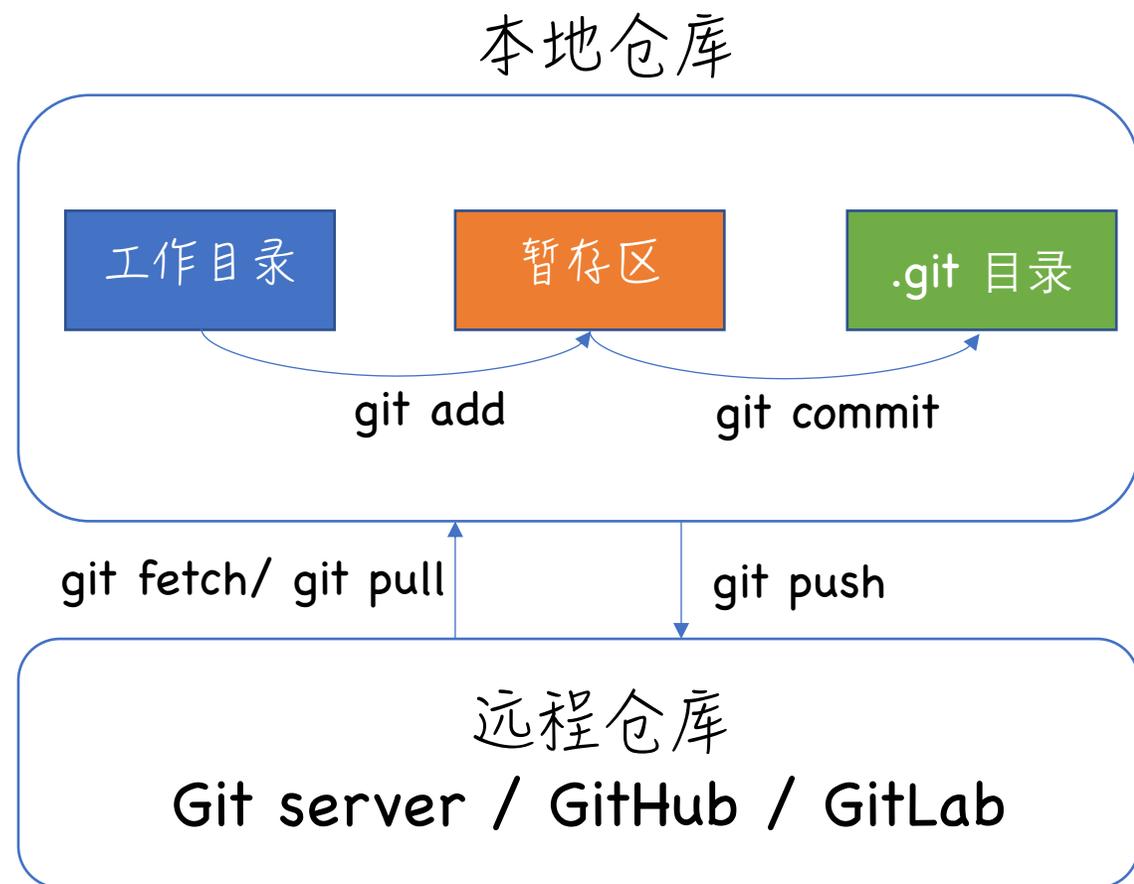


分布式版本控制



# Git的一些概念

- 使用Git管理的代码：本地仓库 + 远程仓库
- 几乎所有的操作都可以在本地仓库完成
- 发布代码时，推送到远程仓库
- Git本地仓库的三个区域
  - 工作目录：待操作的文件（代码）
  - 暂存区：待提交的文件列表
  - .git仓库目录：元数据+数据库
- Git本地仓库的三种文件状态
  - 已修改
  - 已暂存
  - 已提交



# 如何安装Git？

---



# 安装Git的简单说明

## ● Linux

```
# Debian based system : ubuntu
```

```
$ sudo apt-get install git-all
```

```
# Fedora / CentOS / RHEL
```

```
$ sudo dnf install git-all
```

```
Or : sudo yum install git-all
```

## ● Mac OS

```
# 使用homebrew安装
```

```
$ brew install git
```

```
# 使用二进制安装文件安装
```

```
https://git-scm.com/download/mac
```

## ● Windows

```
# 使用二进制安装文件
```

```
https://git-scm.com/download/win
```

```
# 或者windows自带的linux虚拟系统安装
```



# 如何操作Git？

---



# 操作前的一些配置

## ● 配置用户信息

```
$ git config --global user.name "John Smith"
$ git config --global user.email johnsmith@example.com
```

## ● 配置编辑器

```
$ git config --global core.editor vim
```

## ● 查看所有配置及所在文件

```
$ git config --list
```

## ● 查看git config详细用法

```
$ man git config
```

## ● git使用三级配置文件

- /etc/gitconfig, --system
- ~/.gitconfig, --global
- .git/config, --local

## ● 自上而下起效

Given a `.git/config` like this:

```
#
# This is the config file, and
# a '#' or ';' character indicates
# a comment
#

; core variables
[core]
; Don't trust file modes
filemode = false

; Our diff algorithm
[diff]
    external = /usr/local/bin/diff-wrapper
    renames = true

; Proxy settings
[core]
    gitproxy=proxy-command for kernel.org
    gitproxy=default-proxy ; for all the rest
```

you can set the filemode to true with

```
% git config core.filemode true
```



# Git操作-准备本地仓库

准备本地仓库

工作目录操作

暂存区操作

.git目录操作

远程仓库操作

标签操作

分支操作

- 使用已有的目录作为本地仓库

```
$ cd my_project  
$ git init
```

- 克隆现有仓库

```
# 克隆本地的另一个目录作为本地仓库  
$ git clone /path/to/my_project  
# 克隆某台服务器上的目录作为本地仓库  
$ git clone username@host:/path/to/my_project
```



# Git操作-工作目录操作

准备本地仓库

工作目录操作

暂存区操作

.git目录操作

远程仓库操作

标签操作

分支操作

- 根据需求修改工作目录下的文件
- 查看文件状态

```
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
nothing to commit, working directory clean
```

```
$ git status -s
M README
MM Rakefile
A lib/git.rb
M lib/simplegit.rb
?? LICENSE.txt
```

状态为两列：

- ?? : 未被跟踪
- M : 已修改但未被暂存
- M : 已修改已暂存
- MM : 已暂存又被修改过
- A : 已暂存的新文件



# Git操作-暂存区操作

准备本地仓库

工作目录操作

暂存区操作

.git目录操作

远程仓库操作

标签操作

分支操作

## ● 暂存已修改的文件

```
$ git add README
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)

    new file:   README
```

## ● 可使用.gitignore文件跳过某些文件：

- 密码文件：数据库连接
- 编译中间文件：\*.o



# Git操作-.git目录操作

准备本地仓库

工作目录操作

缓存区操作

**.git目录操作**

远程仓库操作

标签操作

分支操作

## ● 提交更新

```
$ git commit
Add README; Update CONTRIBUTING.MD
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch master
# Your branch is up-to-date with 'origin/master'.
#
# Changes to be committed:
#   new file:   README
#   modified:  CONTRIBUTING.MD
#
~
~
~
README CONTRIBUTING.md
```

```
$ git commit -m 'Add README; Update CONTRIBUTING.MD'
```



# Git操作-远程仓库操作

准备本地仓库

工作目录操作

缓存区操作

.git目录操作

远程仓库操作

标签操作

分支操作

## ● 查看远程仓库

```
$ git remote -v  
origin https://github.com/schacon/ticgit (fetch)  
origin https://github.com/schacon/ticgit (push)
```

## ● 本地仓库发布到远程仓库

若使用git clone获取本地仓库

```
$ git push origin master
```

```
Or : git push origin main
```

若使用本地非空白目录作为本地仓库，并且第一次推送；之后可使用git push命令

```
$ git remote add origin <server>
```



# Git操作-标签操作

准备本地仓库

工作目录操作

缓存区操作

.git目录操作

远程仓库操作

标签操作

分支操作

- 标签可用于发布代码稳定版本，2种
  - 轻量标签：某个特定提交的引用
  - 附注标签：推荐，包含操作信息完整，可供后续查看

## ● 创建标签

```
$ git tag -a <tag_name> -m "<some_comments>"
```

```
$ git tag -a v1.4 -m "my version 1.4"
$ git tag
v0.1
v1.3
v1.4
```

## ● 查看标签

```
$ git tag -l
```

```
$ git tag -l "v1.8.5*"
v1.8.5
v1.8.5-rc0
v1.8.5-rc1
v1.8.5-rc2
v1.8.5-rc3
v1.8.5.1
```

## ● 共享标签

```
$ git push <remote> <tag_name>
```

```
$ git push origin v1.5
Counting objects: 14, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (12/12), done.
Writing objects: 100% (14/14), 2.05 KiB | 0 bytes/s, done.
Total 14 (delta 3), reused 0 (delta 0)
To git@github.com:schacon/simplegit.git
* [new tag]          v1.5 -> v1.5
```



# Git操作-分支操作

准备本地仓库

工作目录操作

缓存区操作

.git目录操作

远程仓库操作

标签操作

分支操作

- **master**或**main**是**git clone**时系统默认的主分支，可用于保存代码稳定版
- 其他分支可用于实现某单一功能，**Debug**等。分支代码稳定后，可并入主分支
- 创建分支

```
$ git branch <branch_name>
$ git checkout <branch_name>
# 等于上面两条命令
$ git checkout -b <branch_name>
```

```
$ git branch testing
$ git checkout testing

$ git checkout -b testing
```

- 合并分支

```
$ git checkout master
$ git merge <branch name>
$ git branch -d <branch name>
```

```
$ git checkout main
$ git merge testing
$ git branch -d testing
```



# 发生冲突时如何处理？

- 发生冲突的文件内容有特殊标记

```
<<<<<< HEAD:index.html
<div id="footer">contact : email.support@github.com</div>
=====
<div id="footer">
  please contact us at support@github.com
</div>
>>>>>> iss53:index.html
```

- 可手动修改有冲突的文件，如本例中的index.html

```
<div id="footer">
please contact us at email.support@github.com
</div>
```

- 使用git add标记已解决冲突文件
- 使用git commit暂存，并使用git push提交到远程仓库

```
git add index.html
```

```
git commit -m 'conflict solved - support email modified'
git push origin master
```



# 高能所代码托管平台使用

---



# 登录

- 平台地址：<https://code.ihep.ac.cn/>
- 可使用高能所SSO账号登录，所内所外皆可使用

## 欢迎使用中科院高能所GitLab

1, IHEP SSO Account sign in/高能所统一认证帐号，可以直接登录。

2, Others, apply for IHEP SSO Account /其他人需要申请统一认证帐号：

<https://login.ihep.ac.cn>

3, IHEP Gitlab Manual / 用户指南：

<http://code.ihep.ac.cn/codeguide.pdf>

4, Helps/帮助平台：<http://helpdesk.ihep.ac.cn> Tel./电话：  
88236855

高能所计算中心负责本系统的可靠、稳定运行，并会对托管代码及其数据进行定期备份。

您在使用过程中如果有任何问题，请联系：

[helpdesk@ihep.ac.cn](mailto:helpdesk@ihep.ac.cn)

### IHEP SSO Account

IHEP SSO Account Username

Password

Remember me

Sign in



# 创建project

The screenshot shows the GitLab web interface. On the left, a sidebar menu is open, displaying various navigation options. The 'Create new project' button at the bottom of this menu is highlighted with a red rectangular border. The main content area shows a list of existing projects, including 'slurm\_maint', 'slurm\_testbed', 'ihpboinc', 'slurm\_knowledge\_base', and 'manual\_source'. Each project entry includes a user icon, the project name, a lock icon, an 'Owner' label, and statistics for stars, forks, and issues.

New project > Create blank project

## Project name

summer\_school

## Project URL

https://code.ihep.ac.cn/ duran

## Project slug

summer\_school

Want to organize several dependent projects under the same namespace? [Create a group.](#)

## Project description (optional)

Documents and related files for summer school of IHEP-CC.

## Visibility Level ?

Private

Project access must be granted explicitly to each user. If this project is part of a group, access is granted to members of the group.

Internal

The project can be accessed by any logged in user except external users.

Public

The project can be accessed without any authentication.

## Project Configuration

Initialize repository with a README

Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.



# 增加project成员

Menu

- summer\_school
- Project information
  - Activity
  - Labels
  - Members**
- Repository
- Issues 0
- Merge requests 0
- CI/CD
- Security & Compliance
- Deployments
- Packages & Registries
- Infrastructure
- Monitor
- Analytics
- Wiki
- Snippets
- Settings
- Collapse sidebar

duran@ihep.ac.cn > summer\_school > Members

## Project members

You can invite a new member to **summer\_school** or invite another group.

Import from a project

Invite a group

Invite members

### Invite members



You're inviting members to the **summer\_school** project.

#### Username or email address

Select members or type email addresses

#### Select a role

Guest

- ✓ Guest
- Reporter
- Developer
- Maintainer
- Owner

Cancel

Invite

# 在集群登录节点生成ssh public key文件

```
(base) [duran@lxslc706:205 .ssh]$ cd ~/.ssh
(base) [duran@lxslc706:205 .ssh]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/cc/duran/.ssh/id_rsa): lxslc7_id_rsa
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in lxslc7_id_rsa.
Your public key has been saved in lxslc7_id_rsa.pub.
The key fingerprint is:
SHA256:u6beIE8HpE1cQ8JvWYlORbr4cWscmk67Su+ClTkPwkY duran@lxslc706.ihep.ac.cn
The key's randomart image is:
+---[RSA 2048]-----+
|    ...++o.    |
|    ..oooo    |
|    ++.o      |
|    E= .=.    |
|    o. ++S o   |
|    + *o B o   |
|    ..+++B +   |
|    .=.Bo+    |
|    .=*0.     |
+-----[SHA256]-----+
(base) [duran@lxslc706:205 .ssh]$ cat lxslc7_id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDSn9aKkr0j/RjwdUmtiIWE0jcxwvog4rmojfEUgudKyAH0uLxsfSMsRjIH7JrcBGmfoGtBidTBBYwQ0NnOr/L8uGRc
/r+SjJxbmS3cTQitrdJLP9oVwLNW8WUzjxfSUQZiDNNd2Bf10qnpR8r481Hvg+k5a14kiCTDqZmI8aIstn+sq8r3yw8kNs+axNywukMFfxLZS+6HUjLAkJ9xn14HkHRZ
+Rw50ke7kCTsf6CQvdjdamQ08rol2fSHvjVkpY97dQheP7ka2AqPA3CibW/UXnttJmiZjN6SjENra3E2NiPJH3Dr0ez0srfffyfpr3b0UuEyhKY20l+QjBuruBJFN dur
an@lxslc706.ihep.ac.cn
(base) [duran@lxslc706:205 .ssh]$
```



# 上传ssh public key

User Settings > SSH Keys

Search page

### SSH Keys

SSH keys allow you to establish a secure connection between your computer and GitLab.

**Add an SSH key**  
Add an SSH key for secure access to GitLab. [Learn more.](#)

**Key**

```
ssh-rsa  
AAAAB3NzaC1yc2EAAAADAQABAAQDSn9aKkrOj/RjwdUmtilWE0jcxwvog4rmojfEUgudKy  
AHOulxsfSMsRjIH7JrcBGmfoGtBidTBBYwQ0NnOr/l8uGRc/r+SjJxbmS3cTQitrdJIP9oVwINW8  
WUzjxfSUQZiDNNd2Bf1OqnpR8r481Hvg+k5al4kiCTDqZml8alstn+sq8r3yw8kNs+axNywukM  
FfxLZS+6HUjIAKJ9xnl4HkHRZ+Rw50ke7kCTsf6CQvdjdamQ08rol2fSHvjVkpY97dQheP7ka2A  
qPA3CibW/UXnttJmiZjN6SjENra3E2NiPJH3Dr0ez0srffypr3bOUuEyhKY20I+QjBuruBJFN  
duran@lxslc706.ihep.ac.cn
```

begins with 'ssh-rsa', 'ssh-dss', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', 'ecdsa-sha2-nistp521', 'ssh-ed25519', 'sk-ecdsa-sha2-nistp256@openssh.com', or 'sk-ssh-ed25519@openssh.com'.

**Title**  
duran@lxslc706.ihep.ac.cn

**Expiration date**  
年 / 月 / 日

Key titles are publicly visible. Key becomes invalid on this date.

[Add key](#)

User menu: duran@ihep.ac.cn @duran  
Set status  
Edit profile  
**Preferences (1)**  
Sign out

Sidebar: Profile, Account, Applications, Chat, Access Tokens, Emails, Notifications, **SSH Keys (2)**, GPG Keys, Preferences, Active Sessions, Authentication log

# 在集群中使用code托管平台

## ● 获取project链接

The screenshot shows a code hosting interface for a project named 'summer\_school'. The project is owned by 'S' and has a Project ID of 1861. It features 1 commit, 1 branch, 0 tags, and 31 KB of project storage. The description reads: 'Documents and related files for summer school of IHEP-CC.' The current branch is 'main' within the 'summer\_school' repository. A recent commit is shown: 'Initial commit' by 'duran@ihep.ac.cn' authored 35 minutes ago. The interface includes buttons for 'Find file', 'Web IDE', 'Download', and 'Clone'. The 'Clone' dropdown menu is open, showing options for cloning with SSH and HTTPS, and opening in an IDE (Visual Studio Code or IntelliJ IDEA).

**Clone with SSH**

```
git@code.ihep.ac.cn:duran/summe
```

**Clone with HTTPS**

```
https://code.ihep.ac.cn/duran/s
```

**Open in your IDE**

- Visual Studio Code (SSH)
- Visual Studio Code (HTTPS)
- IntelliJ IDEA (SSH)
- IntelliJ IDEA (HTTPS)

Name	Last commit
README.md	Initial commit



# 在集群中使用code托管平台

```
(base) [duran@lxslc706:205 duran]$ git clone git@code.ihep.ac.cn:duran/summer_school.git
正克隆到 'summer_school'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
接收对象中: 100% (3/3), done.
```

## (1) 克隆代码

```
(base) [duran@lxslc706:205 summer_school]$ mkdir -p 2022
(base) [duran@lxslc706:205 summer_school]$ cd 2022/
(base) [duran@lxslc706:205 2022]$ touch markdown_examples.md
(base) [duran@lxslc706:205 2022]$ vim markdown_examples.md
(base) [duran@lxslc706:205 2022]$ ll -lht
总用量 4.0K
-rw-r--r-- 1 duran u07 695 8月 17 22:35 markdown_examples.md
```

## (2) 修改代码

```
(base) [duran@lxslc706:205 2022]$ git push origin main
Counting objects: 9, done.
Delta compression using up to 28 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (8/8), 1.13 KiB | 0 bytes/s, done.
Total 8 (delta 1), reused 0 (delta 0)
To git@code.ihep.ac.cn:duran/summer_school.git
db5f19f..8fa9980 main -> main
```

## (4) push

```
(base) [duran@lxslc706:205 2022]$ git status
# 位于分支 main
# 您的分支领先 'origin/main' 共 1 个提交。
#   (使用 "git push" 来发布您的本地提交)
#
# 尚未暂存以备提交的变更:
#   (使用 "git add <file>..." 更新要提交的内容)
#   (使用 "git checkout -- <file>..." 丢弃工作区的改动)
#
#       修改:       markdown_examples.md
#
修改尚未加入提交 (使用 "git add" 和/或 "git commit -a")
(base) [duran@lxslc706:205 2022]$ git add -A ./
(base) [duran@lxslc706:205 2022]$ git commit -m "add file - markdown_example.md"
[main 8fa9980] add file - markdown_example.md
 1 file changed, 9 insertions(+)
(base) [duran@lxslc706:205 2022]$ git branch
* main
```

## (3) add + commit



# 什么是Markdown？

---



# Markdown简介

- 轻量化标记性语言
- 使用特殊标记提供富文本格式
  - 多级标题
  - 文字列表
  - 代码段
  - 表格
  - 图片
  - ...
- 通常用于代码开发文档、即时通讯富文本以及Blog
  - Git project中的README.md，计算中心提供的集群使用手册
  - 可在即时通信软件如mattermost中使用Markdown文本
  - 可转化为pdf, html等格式

The screenshot shows a GitLab project page for 'summer\_school' (Project ID: 1861). It displays project statistics (3 Commits, 1 Branch, 0 Tags, 61 KB Project Storage) and a list of documents. A commit titled 'add file - markdown\_example.md' by 'duran@ihep.ac.cn' is highlighted with a red box. Below the commit list, the project's README content is visible, including a 'Getting started' section with instructions for new users and a link to a template.

Name	Last commit	Last update
2022	add file - markdown_example.md	11 hours ago
README.md	Initial commit	12 hours ago



# Markdown的基本语法

---

以<https://note.ihep.ac.cn> 编辑平台为例



# 标题

- 共六级标题，使用#标识，#的数量表示标题的级别
- 通常用于构筑文档章节，类似于latex的section、html的tag <h1>~<h6>

The screenshot shows the HedgeDoc web editor interface. At the top, there's a navigation bar with 'HedgeDoc' logo, view, edit, and help icons, and buttons for '+ 新建', '发表', '菜单', and '1 ONLINE'. Below the navigation bar is a rich text toolbar with icons for bold (B), italic (I), link, header (H), code (</>), quote, list, check, link, image, upload, grid, and chat. The main editor area is split into two panes. The left pane shows a list of heading levels with their corresponding hash symbols: 1 | 标题, 2 |, 3 | # 一级标题, 4 | ## 二级标题, 5 | ### 三级标题, 6 | #### 四级标题, 7 | ##### 五级标题, 8 | ##### 六级标题, 9 |. The right pane shows the rendered output of these headings: '标题', '标题', '一级标题', '二级标题', '三级标题', '四级标题', '五级标题', and '六级标题'. The rendered headings are displayed in a descending order of size, with '一级标题' being the largest and '六级标题' being the smallest. The right pane also shows a status bar with '已修改 3 分钟前' and 'PRIVATE'.



# 列表

- 无序列表使用“-”
- 有序列表使用“1.”
- 清单列表使用“- [ ]”

The screenshot displays the HedgeDoc web interface. On the left, a code editor shows the following list structure:

```
1 | 列表
2 |
3 | - 今日待做事项 - 无序列表
4 |   - cosmos记账系统增加新功能
5 |   - 同步最新history文件
6 |   - 测试用例
7 |   - 更新文档说明
8 |
9 | - 今日待做事项 - 有序列表
10 |   1. cosmos记账系统增加新功能
11 |   2. 同步最新history文件
12 |   3. 测试用例
13 |   4. 更新文档说明
14 |
15 | - 今日待做事项 - 清单列表
16 |   - [ ]cosmos记账系统增加新功能
17 |   - [ ]同步最新history文件
18 |   - [ ]测试用例
19 |   - [ ]更新文档说明
```

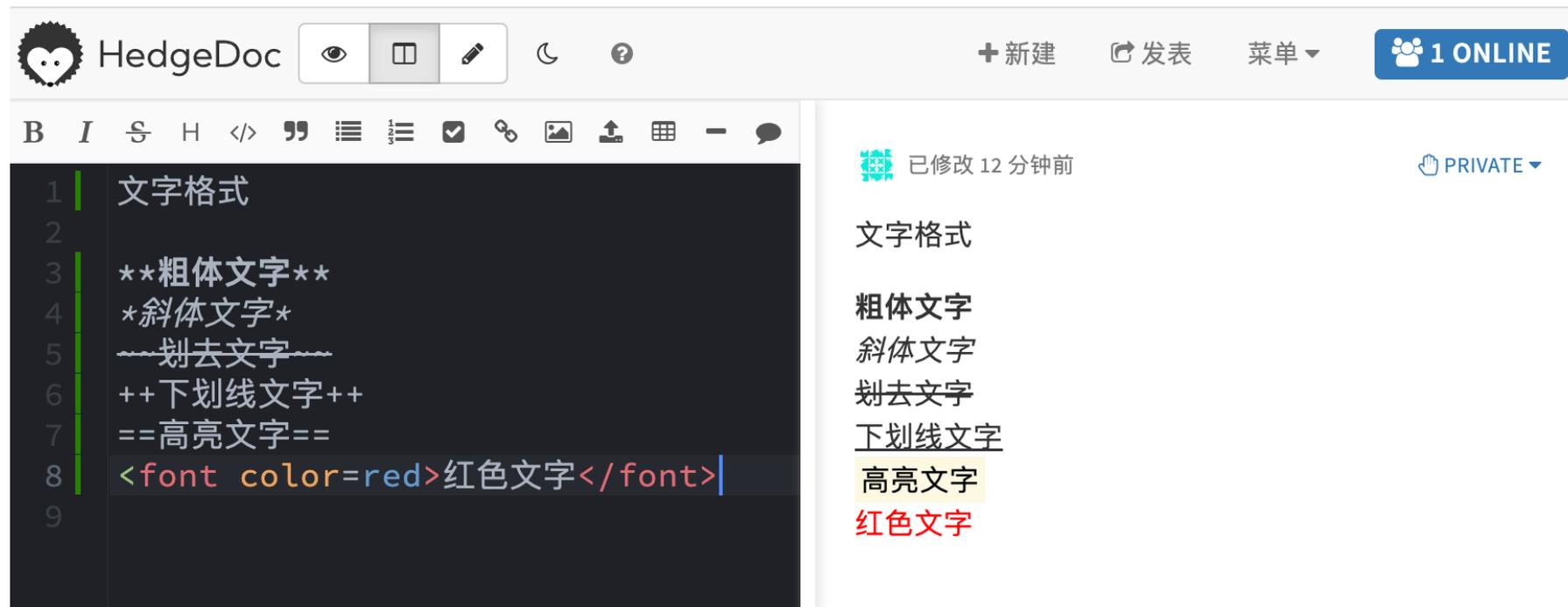
On the right, a preview window shows the rendered output:

- 列表
- 今日待做事项 - 无序列表
  - cosmos记账系统增加新功能
  - 同步最新history文件
  - 测试用例
  - 更新文档说明
- 今日待做事项 - 有序列表
  1. cosmos记账系统增加新功能
  2. 同步最新history文件
  3. 测试用例
  4. 更新文档说明
- 今日待做事项 - 清单列表
  - cosmos记账系统增加新功能
  - 同步最新history文件
  - 测试用例
  - 更新文档说明



# 文字格式

- 粗体使用\*\*
- 斜体使用\*
- 划去使用~~
- 下划线使用++
- 高亮使用==
- 文字颜色<font color=red></font>



The screenshot shows the HedgeDoc web editor interface. The top bar includes the HedgeDoc logo, navigation icons (eye, window, edit, moon, help), and user controls (+新建, 发表, 菜单, 1 ONLINE). The main editor area displays a list of text formatting examples with line numbers 1 through 9. The examples are: 1. 文字格式; 2. (blank); 3. \*\*粗体文字\*\* (bold); 4. \*斜体文字\* (italic); 5. ~~~划去文字~~~ (strikethrough); 6. ++下划线文字++ (underline); 7. ==高亮文字== (highlight); 8. <font color=red>红色文字</font> (red text); 9. (blank). To the right of the editor, a preview pane shows the rendered output of these examples: 文字格式, 粗体文字, 斜体文字, 划去文字, 下划线文字, 高亮文字, and 红色文字. The interface also shows a 'PRIVATE' status indicator and a '已修改 12 分钟前' (modified 12 minutes ago) timestamp.





# 特殊格式文字片段

- 警告/说明文字片段
- 引用文字片段

The screenshot displays the HedgeDoc web interface. The top navigation bar includes the HedgeDoc logo, a toolbar with icons for view, edit, and search, and user information showing '1 ONLINE'. The main content area is split into two panes. The left pane is a code editor with a dark background, showing the following code:

```
1 警告/说明片段
2  -----
3  :::info
4  特别说明:
5  - 说明片段中依然可以使用Markdown语法
6  - 共四类: info、warning、danger、success, 背景颜色不同
7  :::
8
9  引用片段
10 -----
11
12 > Markdown is a lightweight markup language for
    creating formatted text using a plain-text editor.
    John Gruber and Aaron Swartz created Markdown in 2004
    as a markup language that is appealing to human
    readers in its source code form. Markdown is widely
    used in blogging, instant messaging, online forums,
    collaborative software, documentation pages, and
    readme files.
```

The right pane shows the rendered output. At the top, it indicates '已修改 3 分钟前' and 'PRIVATE'. The first section is titled '警告/说明片段' and contains a light blue background box with the following text:

特别说明:

- 说明片段中依然可以使用Markdown语法
- 共四类: info、warning、danger、success, 背景颜色不同

The second section is titled '引用片段' and contains a quote with a light gray background:

Markdown is a lightweight markup language for creating formatted text using a plain-text editor. John Gruber and Aaron Swartz created Markdown in 2004 as a markup language that is appealing to human readers in its source code form. Markdown is widely used in blogging, instant messaging, online forums, collaborative software, documentation pages, and readme files.



# 表格

- 点击红框内的图标，可出现表格的编辑模版

The screenshot shows the HedgeDoc interface. At the top, there's a navigation bar with 'HedgeDoc' logo, view, edit, and help icons, and buttons for '+ 新建', '发表', and '菜单'. A '1 ONLINE' indicator is on the right. Below the navigation bar is a rich text editor toolbar with icons for bold, italic, strikethrough, heading, code, quote, list, link, image, and a table icon (highlighted with a red box). The editor content shows a table with columns 'partition', 'qos', and 'account' and rows of data. To the right, the rendered table is displayed.

partition	qos	account
gpu	normal	lqcd
raq	raqregular	raq
heps	regular	heps
ali	regular	alicpt



# 图片

- 点击红框内的图标，可得到图片的基础编辑模版
  - 本地图片：可通过拖拽方式上传
  - 网络图片：提供图片链接地址

The screenshot displays the HedgeDoc web interface. The top navigation bar includes the HedgeDoc logo, a toolbar with icons for eye, window, edit, and search, and buttons for '+ 新建', '发表', '菜单', and '1 ONLINE'. The main editing area is a dark-themed text editor with a toolbar containing icons for bold, italic, strikethrough, heading, code, quote, list, link, image, and other functions. A red box highlights the image icon in the toolbar. The text editor contains the following content:

```
1 本地上传图片
2  ![本地图片]
   (https://note.ihep.ac.cn/uploads/1b51b5d1-6548-4d18-a44f-2dff724c7deb.png)
3
4 在线图片:
5  ![网络链接图片]
   (https://img2.baidu.com/it/u=3735799021,1953155234&fm=253&fmt=auto&app=138&f=JPEG?w=485&h=730)
6
7
8
```

Below the text editor, there is a section titled '本地上传图片' (Local Image Upload) which features a line chart titled 'Transferred data by Source' for the period '30 Days from 2022-06-30 to 2022-07-30'. The chart shows data for 'IHEP-DATA-SE' (red area) and 'IHEP-TMP-SE' (blue area). The y-axis is labeled 'TB' and ranges from 0 to 175. The x-axis shows dates from 2022-07-02 to 2022-07-29. The chart shows a steady increase in data transfer over time, with a significant spike at the end of the period. Below the chart, the following statistics are provided: Max: 165, Min: 2.09, Average: 79.8, Current: 165. A legend at the bottom indicates 'IHEP-DATA-SE 164.8' and 'IHEP-TMP-SE 0.6'.

Below the chart, there is a section titled '在线图片:' (Online Image) which displays a photograph of a dandelion seed head against a green background.



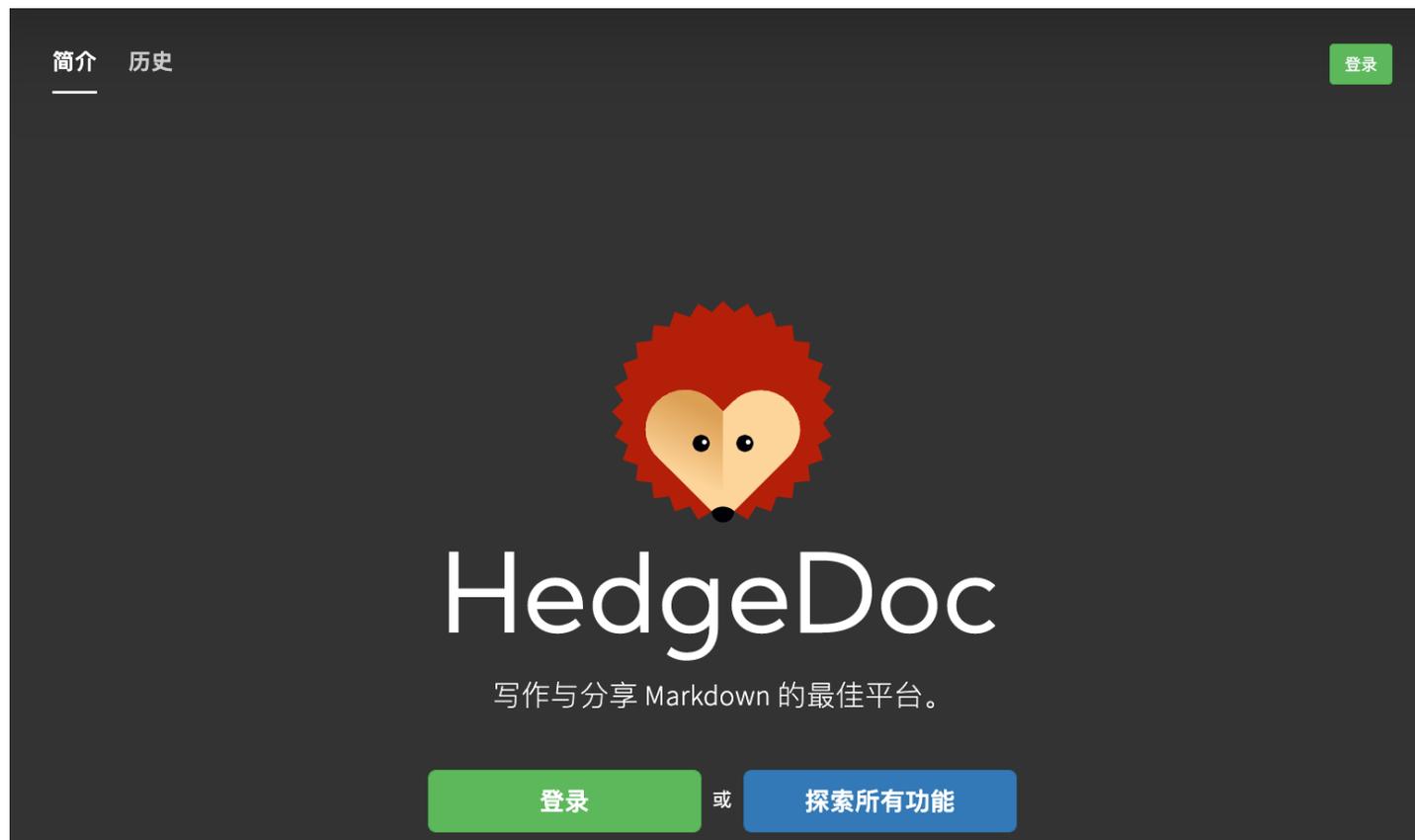
# 高能所Markdown文档编辑平台

---



# 登录

- 平台地址：<https://note.ihep.ac.cn>
- 通过代码托管平台<https://code.ihep.ac.cn> 授权登录



# 新建

简介 历史

+ 新建笔记  duran@ihep.ac.cn

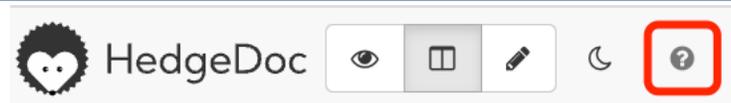
选择标签... 搜索关键字... 标题 时间    

- 文字格式**  
visited 11 分钟前  
2022年8月17日星期三 20:30
- 表格**  
visited 11 分钟前  
2022年8月17日星期三 20:29
- 标题**  
visited 12 分钟前  
2022年8月17日星期三 20:29
- 特殊文字片段**  
visited 12 分钟前  
2022年8月17日星期三 20:29
- 代码**  
visited 12 分钟前  
2022年8月17日星期三 20:28
- 列表**  
visited 13 分钟前  
2022年8月17日星期三 20:28
- 图片**  
visited 13 分钟前  
2022年8月17日星期三 20:27
- Features**  
visited 3 小时前  
2022年8月17日星期三 17:54  
features cool updated



# Markdown语法速查

- 新建或打开Markdown文件后，点击红框内图标
- 可得到如下语法速查表



## 帮助

### 联系我们

- 加入社区
- # 在 Matrix 上联系我们
- 报告问题
- 帮助我们翻译

### 文档

- 功能
- YAML 元数据
- 幻灯范例

### 速查表

范例	语法
标题	# 标题
• 无序列表	- 无序列表
1. 有序列表	1. 有序列表
<input type="checkbox"/> 清单	- [ ] 清单
引用	> 引用
粗体	**粗体**
斜体	*斜体*
删除线	~~删除线~~
19 <sup>th</sup>	19 <sup>^</sup> th <sup>^</sup>
H <sub>2</sub> O	H~2~O
下划线文字	++下划线文字++



# 编辑

- 若不记得语法，也可通过点击红框内的图标，快速得到相关模版

HedgeDoc

+ 新建 发表 菜单 1 ONLINE

B I U ABC </> ” ≡ ≡ ✓ 🔗 🖼️ ⬆️ 📄 - 💬

```
1 ← Start by entering a title here
===
Visit /features if you don't know what to do.
Happy hacking :)
```

已修改 3 分钟前 PRIVATE



# 设置共享权限

```
HedgeDoc [Icons]
B I S H <> [Icons]
1 # nvidia-smi 操作命令
2
3 [TOC]
4
5 ## 1 enable/disable ECC memory
6
7 - 需root权限
8 - 需重启后生效
9 - 命令
10
11 ~~~bash
12 $ nvidia-smi -i <gpu_card_index> --ecc-config=<0:1>
13 # <gpu_card_index> : 0,1,2...
14 # --ecc-config : 0代表enable, 1代表disable
15
16 # Example 1 : /dev/nvidia1 启动ecc memory
17 nvidia-smi -i 1 --ecc-config=1
18
19 # Example 2 : /dev/nvidia1 停用ecc memory
20 nvidia-smi -i 1 --ecc-config=0
21
22 # Example 3 : 所有GPU卡启动ecc memory
23 nvidia-smi --ecc-config=1
24 ~~~
25
26
Line 1, Columns 1 — 48 Lines Spaces: 4 SUBLIME Length 846
```

已修改 41 分钟前

## nvidia-smi 操作命令

- nvidia-smi 操作命令
  - 1 enable/disable ECC memory
  - 2 Reset Uncorretable ECC memory counter

### 1 enable/disable ECC memory

- 需root权限
- 需重启后生效
- 命令

```
$ nvidia-smi -i <gpu_card_index> --ecc-config=<0:1>
# <gpu_card_index> : 0,1,2...
# --ecc-config : 0代表enable, 1代表disable

# Example 1 : /dev/nvidia1 启动ecc memory
nvidia-smi -i 1 --ecc-config=1

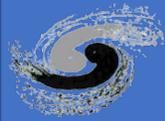
# Example 2 : /dev/nvidia1 停用ecc memory
nvidia-smi -i 1 --ecc-config=0

# Example 3 : 所有GPU卡启动ecc memory
nvidia-smi --ecc-config=1
```

PRIVATE

- 可编辑 - 已登录用户可编辑
- 限制 - 已登录用户可编辑 (禁止游客查看)
- 锁定 - 仅所有者可编辑
- 保护 - 仅所有者可编辑 (禁止游客查看)
- 私有 - 仅所有者可查看与编辑

Delete this note



# 保存并查看

- 点击红框内图标可保存并查看生成的文档



The screenshot shows the HedgeDoc web interface. At the top left, the HedgeDoc logo and name are visible. A red box highlights the eye icon, which is used for saving and viewing documents. To the right of the eye icon are icons for a document, a pencil, a moon, and a question mark. Further right, there are buttons for '+ 新建' (New), '发表' (Publish), and a '菜单' (Menu) dropdown. A blue badge indicates '1 ONLINE' user. The main content area shows a document titled 'nvidia-smi 操作命令' (nvidia-smi Operation Commands), which was last modified 5 minutes ago. The document content includes a list of commands: 'nvidia-smi 操作命令', '1 enable/disable ECC memory', and '2 Reset Uncorretable ECC memory counter'. Below this, a section titled '1 enable/disable ECC memory' lists requirements: root permissions, restart after changes, and the command 'nvidia-smi -i <gpu\_card\_index> --ecc-config=<0:1>'. The command line examples are: 'nvidia-smi -i 1 --ecc-config=1' for enabling ECC memory and 'nvidia-smi -i 1 --ecc-config=0' for disabling it. On the right side, there is a sidebar with 'PRIVATE' status, the document title, and a list of items: '1 enable/disable ECC...' and '2 Reset Uncorretable...'. Below the list are options to 'Expand all', 'Back to top', and 'Go to bottom'.



# 下载

```
HedgeDoc [Icons]
B I S H <> " " [Icons] -
1 # nvidia-smi 操作命令
2
3 [TOC]
4
5 ## 1 enable/disable ECC memory
6
7 - 需root权限
8 - 需重启后生效
9 - 命令
10
11 ~~~bash
12 $ nvidia-smi -i <gpu_card_index> --ecc-config=<0:1>
13 # <gpu_card_index> : 0,1,2...
14 # --ecc-config : 0代表enable, 1代表disable
15
16 # Example 1 : /dev/nvidia1 启动ecc memory
17 nvidia-smi -i 1 --ecc-config=1
18
19 # Example 2 : /dev/nvidia1 停用ecc memory
20 nvidia-smi -i 1 --ecc-config=0
21
22 # Example 3 : 所有GPU卡启动ecc memory
23 nvidia-smi --ecc-config=1
24 ~~~
25
https://note.ihep.ac.cn/# Spaces: 4 SUBLIME Length 846
```

已修改 40 分钟前

## nvidia-smi 操作命令

- nvidia-smi 操作命令
  - 1 enable/disable ECC memory
  - 2 Reset Uncorretable ECC memory counter

### 1 enable/disable ECC memory

- 需root权限
- 需重启后生效
- 命令

```
$ nvidia-smi -i <gpu_card_index> --ecc-config=<0:1>
# <gpu_card_index> : 0,1,2...
# --ecc-config : 0代表enable, 1代表disable

# Example 1 : /dev/nvidia1 启动ecc memory
nvidia-smi -i 1 --ecc-config=1

# Example 2 : /dev/nvidia1 停用ecc memory
nvidia-smi -i 1 --ecc-config=0

# Example 3 : 所有GPU卡启动ecc memory
nvidia-smi --ecc-config=1
```

附加功能

- 修订版本
- 幻灯模式

导出

- Snippet

导入

- Gist
- Snippet
- 剪贴板

下载

- Markdown
- HTML
- 原始 HTML

1 ONLINE

PRIVATE

# 本地Markdown编辑软件

---



# 两个本地编辑软件

## ● Typora(收费)



大纲

nvidia-smi 操作命令

- 1 enable/disable ECC memory
- 2 Reset Uncorretable ECC memory counter

### nvidia-smi 操作命令

nvidia-smi 操作命令

- 1 enable/disable ECC memory
- 2 Reset Uncorretable ECC memory counter

### 1 enable/disable ECC memory

- 需root权限
- 需重启后生效
- 命令

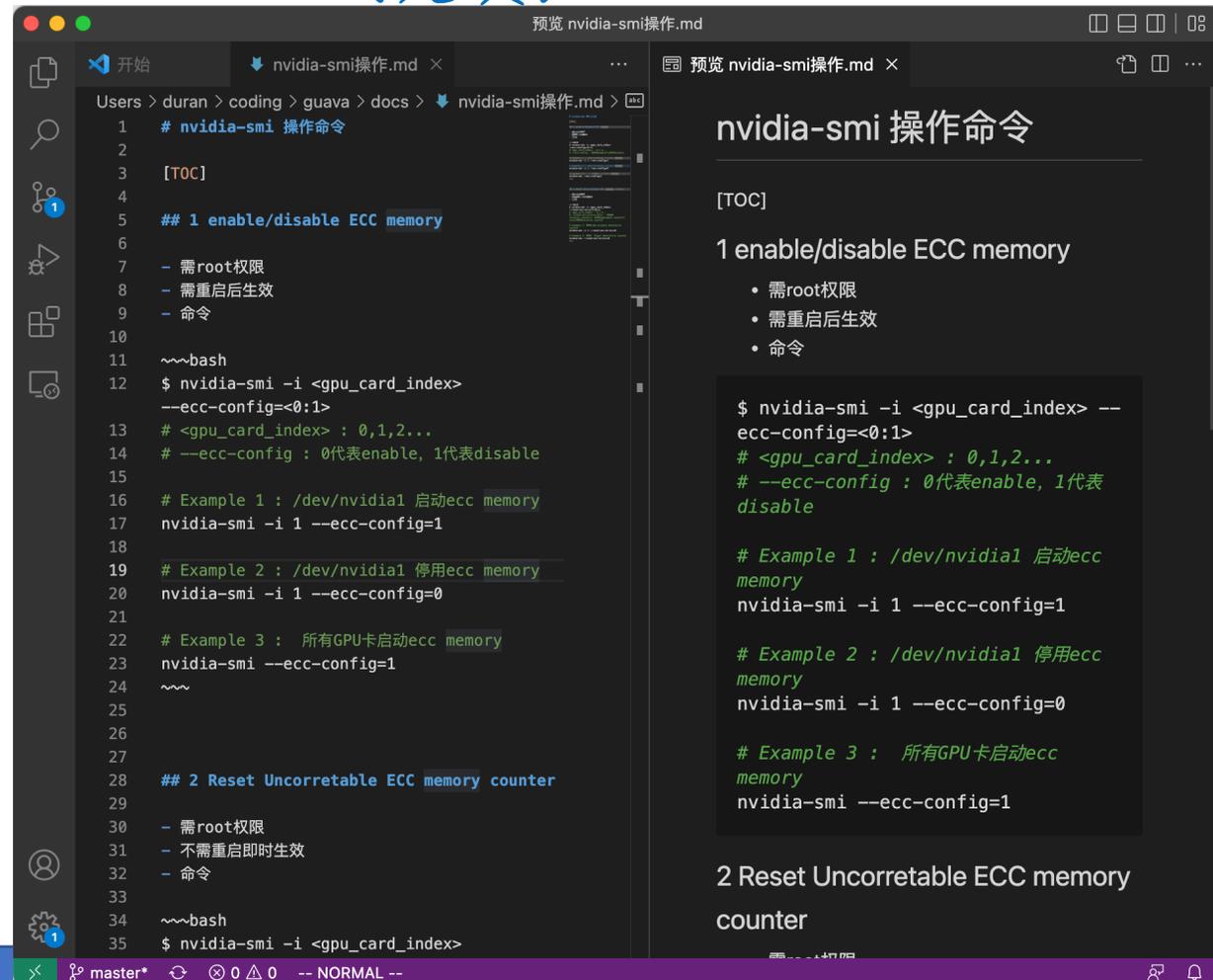
```
$ nvidia-smi -i <gpu_card_index> --ecc-config=<0:1>
# <gpu_card_index> : 0,1,2...
# --ecc-config : 0代表enable, 1代表disable

# Example 1 : /dev/nvidia1 启动ecc memory
nvidia-smi -i 1 --ecc-config=1

# Example 2 : /dev/nvidia1 停用ecc memory
nvidia-smi -i 1 --ecc-config=0

# Example 3 : 所有GPU卡启动ecc memory
nvidia-smi --ecc-config=1
```

## ● vscode(免费)



开始

nvidia-smi操作.md

Users > duran > coding > guava > docs > nvidia-smi操作.md

```
1 # nvidia-smi 操作命令
2
3 [TOC]
4
5 ## 1 enable/disable ECC memory
6
7 - 需root权限
8 - 需重启后生效
9 - 命令
10
11 ~~~bash
12 $ nvidia-smi -i <gpu_card_index>
13 --ecc-config=<0:1>
14 # <gpu_card_index> : 0,1,2...
15 # --ecc-config : 0代表enable, 1代表disable
16 # Example 1 : /dev/nvidia1 启动ecc memory
17 nvidia-smi -i 1 --ecc-config=1
18
19 # Example 2 : /dev/nvidia1 停用ecc memory
20 nvidia-smi -i 1 --ecc-config=0
21
22 # Example 3 : 所有GPU卡启动ecc memory
23 nvidia-smi --ecc-config=1
24 ~~~
25
26 ## 2 Reset Uncorretable ECC memory counter
27
28 - 需root权限
29 - 不需重启即时生效
30 - 命令
31
32 ~~~bash
33 $ nvidia-smi -i <gpu_card_index>
```

### nvidia-smi 操作命令

[TOC]

### 1 enable/disable ECC memory

- 需root权限
- 需重启后生效
- 命令

```
$ nvidia-smi -i <gpu_card_index> --ecc-config=<0:1>
# <gpu_card_index> : 0,1,2...
# --ecc-config : 0代表enable, 1代表disable

# Example 1 : /dev/nvidia1 启动ecc memory
nvidia-smi -i 1 --ecc-config=1

# Example 2 : /dev/nvidia1 停用ecc memory
nvidia-smi -i 1 --ecc-config=0

# Example 3 : 所有GPU卡启动ecc memory
nvidia-smi --ecc-config=1
```

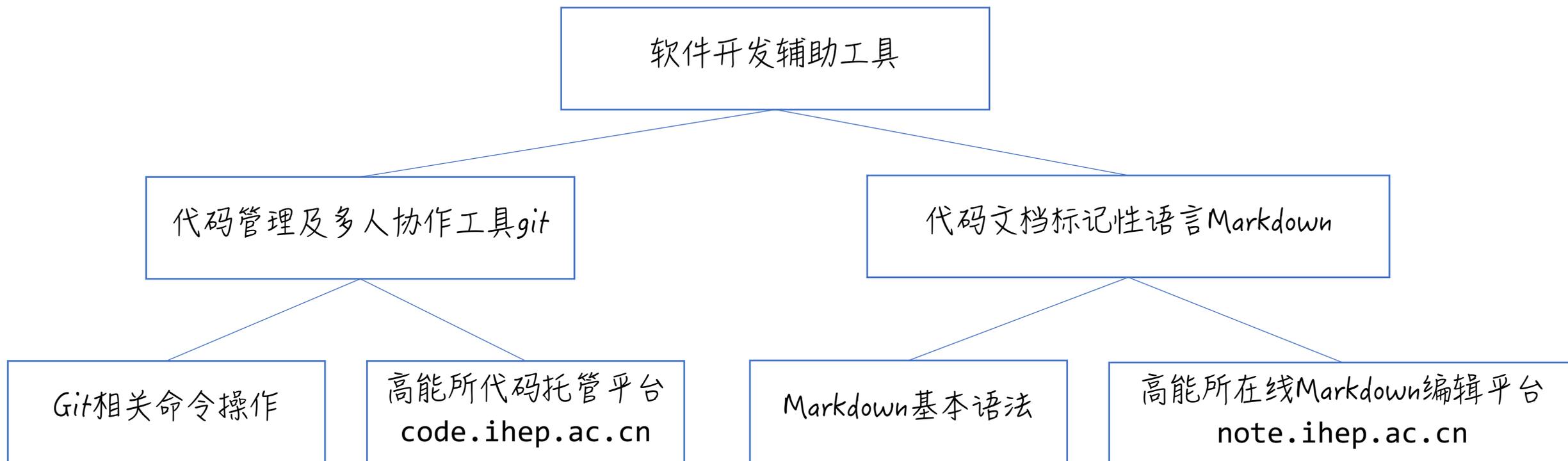
### 2 Reset Uncorretable ECC memory counter

# 总结

---



# 内容图谱



强调几点：1. 代码完整性 2. 代码分享 3. 合作开发 4. 文档非常重要

珍而重之，学以致用



就到这里。。。。

休息。。。。

休息一下吧！



# 附录

---



# Git操作小结

- 准备代码仓库

- git init
- git clone

- 操作配置文件

- git config

- 代码文件操作

- git status
- git diff
- git rm
- git mv

- 暂存区操作

- git add
- git commit
- git log

- 远程仓库操作

- git remote add/show/rename/remove
- git fetch
- git pull
- git push

- 标签操作

- git tag -l
- Git tag -a <tag\_name> -m "some comments"
- git show <tag\_name>
- Git push <remote> <tag\_name>
- Git push <remote> --tags
- Git tag -d <tag\_name>
- Git push <remote> --delete <tag\_name>

- 别名操作 (可选)

- Git config --global alias.<alias\_cmd> "<cmd>"

- 分支操作

- git branch <branch>
- git checkout <branch>
- git log --online --decorate --graph --all
- Git merge <branch>
- Git branch -d <branch>
- git checkout --track <remote>/<branch>
- git branch -vv
- git push origin --delete <branch>

- 变基操作 (可选)

- Git rebase <base\_branch> <topic\_branch>



# 参考资料

- Git入门
  - <http://rogerdudler.github.io/git-guide/index.zh.html>
- Git Pro 2 : 本次报告参考的主要书籍
  - <https://git-scm.com/book/zh/v2>
- GitHub help
  - <https://docs.github.com/en>
- 图解Git
  - <http://marklodato.github.io/visual-git-guide/index-zh-cn.html>
- Think Like a Git
  - <http://think-like-a-git.net/>
- Git社区
  - <https://book.git-scm.com/>
- Markdown语法
  - <https://www.markdownguide.org/basic-syntax/>

