

MARCELLO MISSIROLI
**Tecnologia
e Progettazione**
per il mondo *digitale*
e per il *web* II



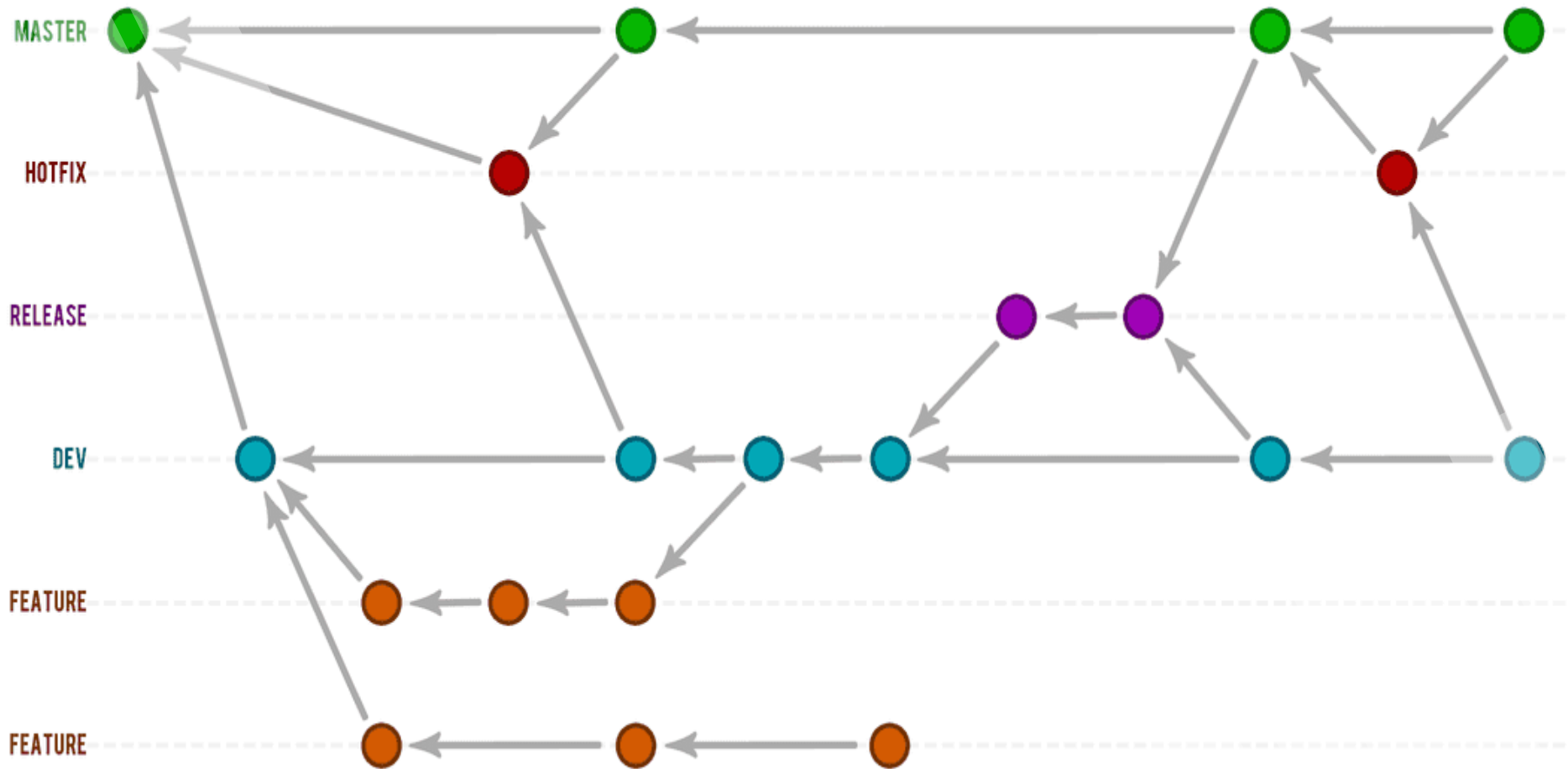
000-000

5-0000000, 000000, 00000

0000. 00000000 0000000000

මම මගේ මම මගේම?

මම මගේම (මගේම) මගේම, මගේම මම මම' මම මගේම මගේම මගේම මම මම. ම මම මම මගේම මම මම මගේම මම මගේම/මගේම, මම මගේම මගේම මම මගේම මම මගේම ම මගේම මගේම. මගේම මගේම ම මගේම ම මගේම මගේම “මගේම මම මගේම” මම ම මගේම මගේම මම මම. මගේම ම මගේම මගේම මගේම ම මගේම ම “මගේම මගේම”.

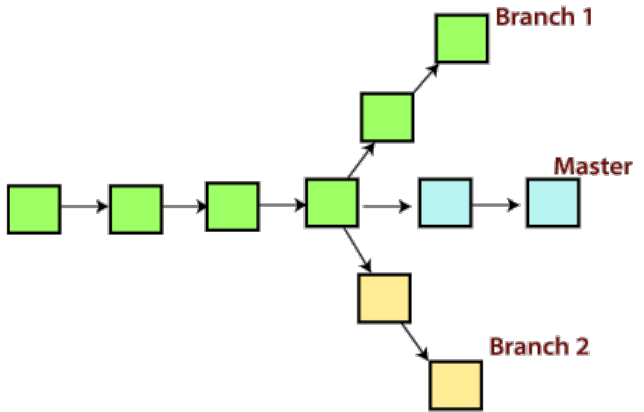


1.

□□□□□□

Git

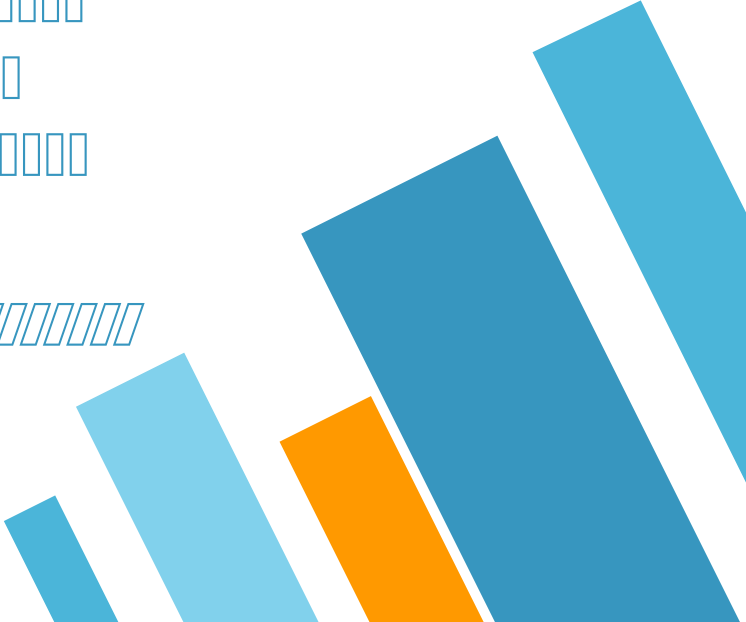
Git 是分布式版本控制系统，它允许开发者在本地计算机上保存项目的完整历史记录。与传统的集中式版本控制系统（如 SVN）不同，Git 的仓库是分散的，每个开发者都可以拥有完整的仓库副本。这使得 Git 非常适合团队协作和分布式开发环境。此外，Git 还支持分支管理，方便开发者在开发新功能或修复bug时创建分支，避免影响生产环境。最后，Git 的提交历史清晰可见，可以轻松追踪代码的变更和责任人。





“**በጣም ጣም ጣም ጣም ጣም ጣም**
ጣም ጣም ጣም ጣም ጣም
ጣም ጣም ጣም ጣም ጣም ጣም
ጣም ጣም ጣም ጣም
ጣም ጣም ጣም ጣም
ጣም ጣም ጣም ጣም ጣም
ጣም.”

ጣም ጣም ጣም ጣም ጣም



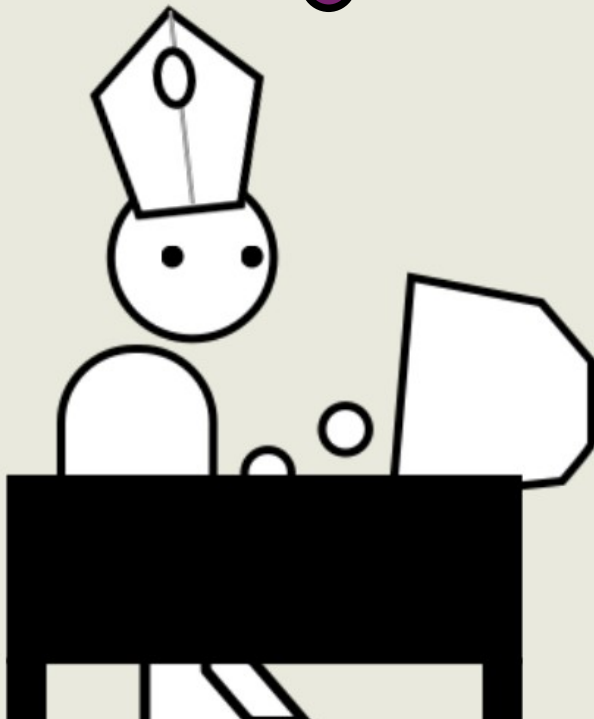
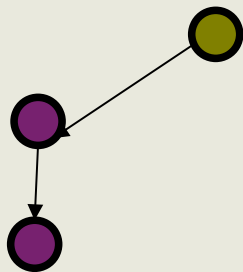
2.

□□□□□□ □□□□□□

□□□□□□□□

□□□□

“□□□□□□□□” □ □□□ □□ □□□□□□□□
□□□□□□□□. □ □□□□□□□ □□□□□□
□□□□□□□□ □□□□ □□ □□□□□□□□
□□□□□□□□



□ □□□ □□□□□□□□ -□ □□□□□□□□

□ □□□ □□□□□□□□ -□□

“□□□□□□□□□□”

0 000 00000000 -0 00000000000000

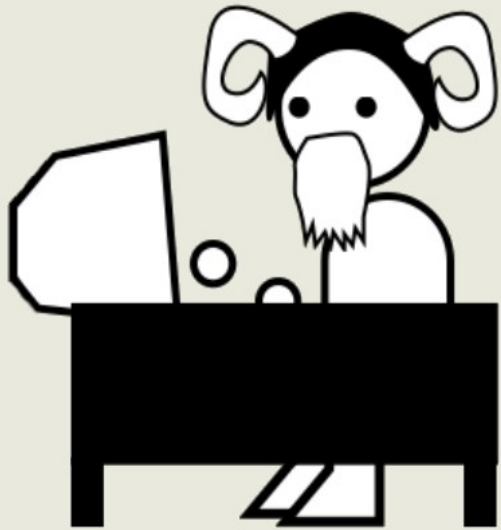
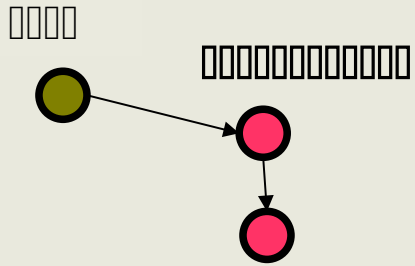
0 000 000 00000000000000.000

0 000 000000 -0 “000000”

0 000 000000

0000

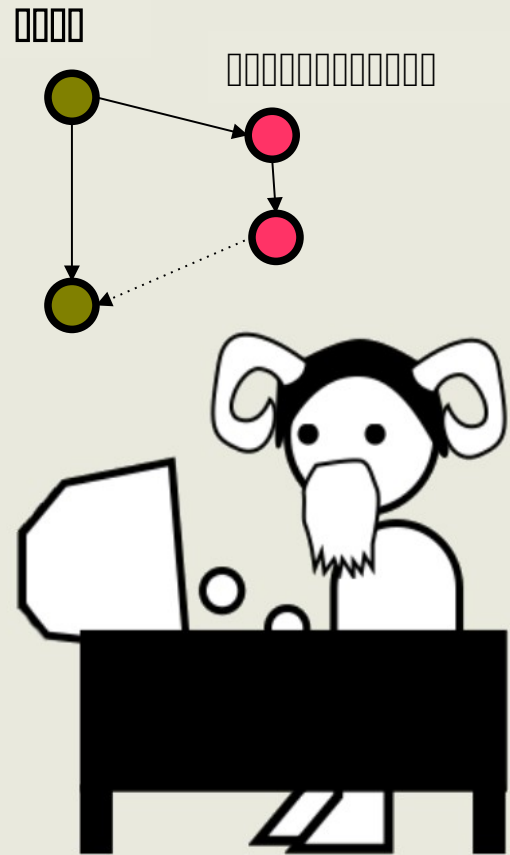
* 000000000000



□□□□□

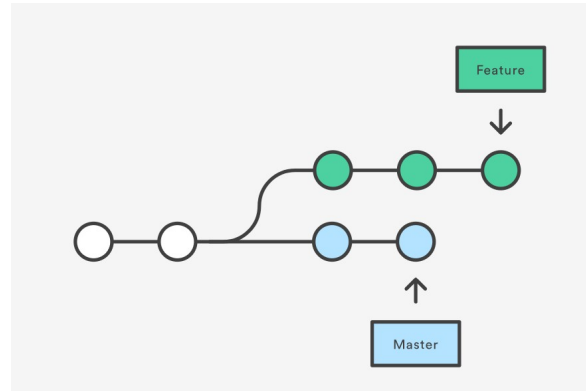
□□ □□□□□□ □□ □□□ □□□□□□□□□□□□ □□□ □□□□□ □□□□□□, □ □□□□□□□□ □□□□□□□□□□□□ □ □□□□. □'□□□□□□□□□□ □ □□□□□ □□□□□ (□□□□□□□□). □□□□ □□□ □□□□ □□□ □□□□ (□□□□ □ □□ □□□□□ □□ □□□□□), □□ □□□□□□□ □□□□□□□□□ □□□ □□□□□□□□□□.

```
0 000 00000000 0000
0 000 00000 000000000000
```



Git: 分支?

Git 分支功能 是 分支 分支
 分支功能 是 分支功能 分支:
 分支 是 分支。 是 分支
 是 分支 分支功能 是 分支。



Git

Git 是分布式版本控制系统

Git 是分布式版本控制系统，每个开发人员都有自己的本地仓库，可以离线工作。Git 使用本地仓库和远程仓库（如 GitHub）来管理代码。Git 使用分支来管理不同的开发任务，可以方便地进行代码合并和回滚。

Git 使用分支来管理不同的开发任务，可以方便地进行代码合并和回滚。Git 使用分支来管理不同的开发任务，可以方便地进行代码合并和回滚。Git 使用分支来管理不同的开发任务，可以方便地进行代码合并和回滚。

Git

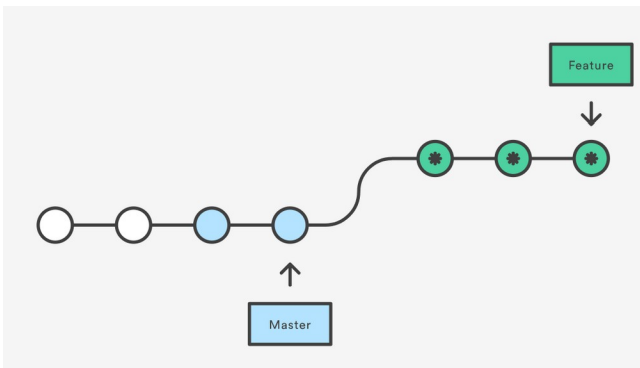
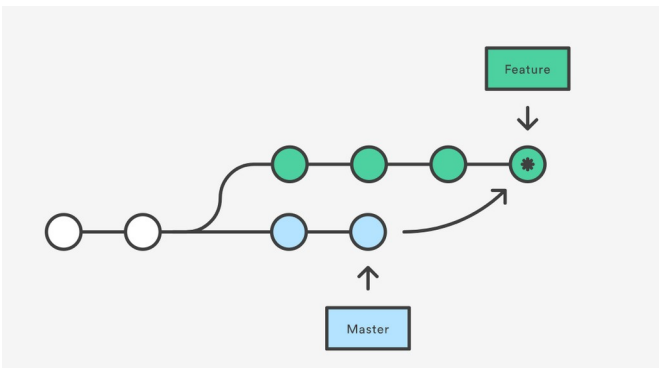
Git 是分布式版本控制系统

Git 是分布式版本控制系统

Git 是分布式版本控制系统，每个开发人员都有自己的本地仓库，可以离线工作。Git 使用本地仓库和远程仓库（如 GitHub）来管理代码。Git 使用分支来管理不同的开发任务，可以方便地进行代码合并和回滚。

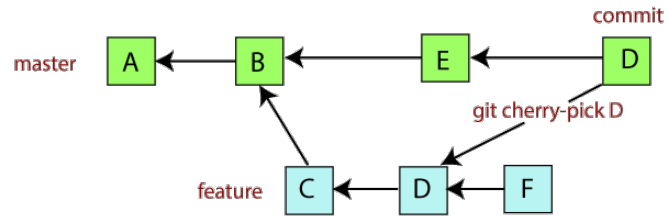
Git 使用分支来管理不同的开发任务，可以方便地进行代码合并和回滚。Git 使用分支来管理不同的开发任务，可以方便地进行代码合并和回滚。Git 使用分支来管理不同的开发任务，可以方便地进行代码合并和回滚。

Git 是分布式版本控制系统，每个开发人员都有自己的本地仓库，可以离线工作。



Git 分支管理

Git 分支管理 (Git Branching Model) 是一种用于管理代码分支的策略。它允许开发者在开发新功能、修复bug或进行实验时，从主分支（通常是 master）创建新的分支。当开发完成或需要回滚时，可以轻松地将更改合并回主分支。这种策略有助于保持代码库的稳定性和可追溯性。





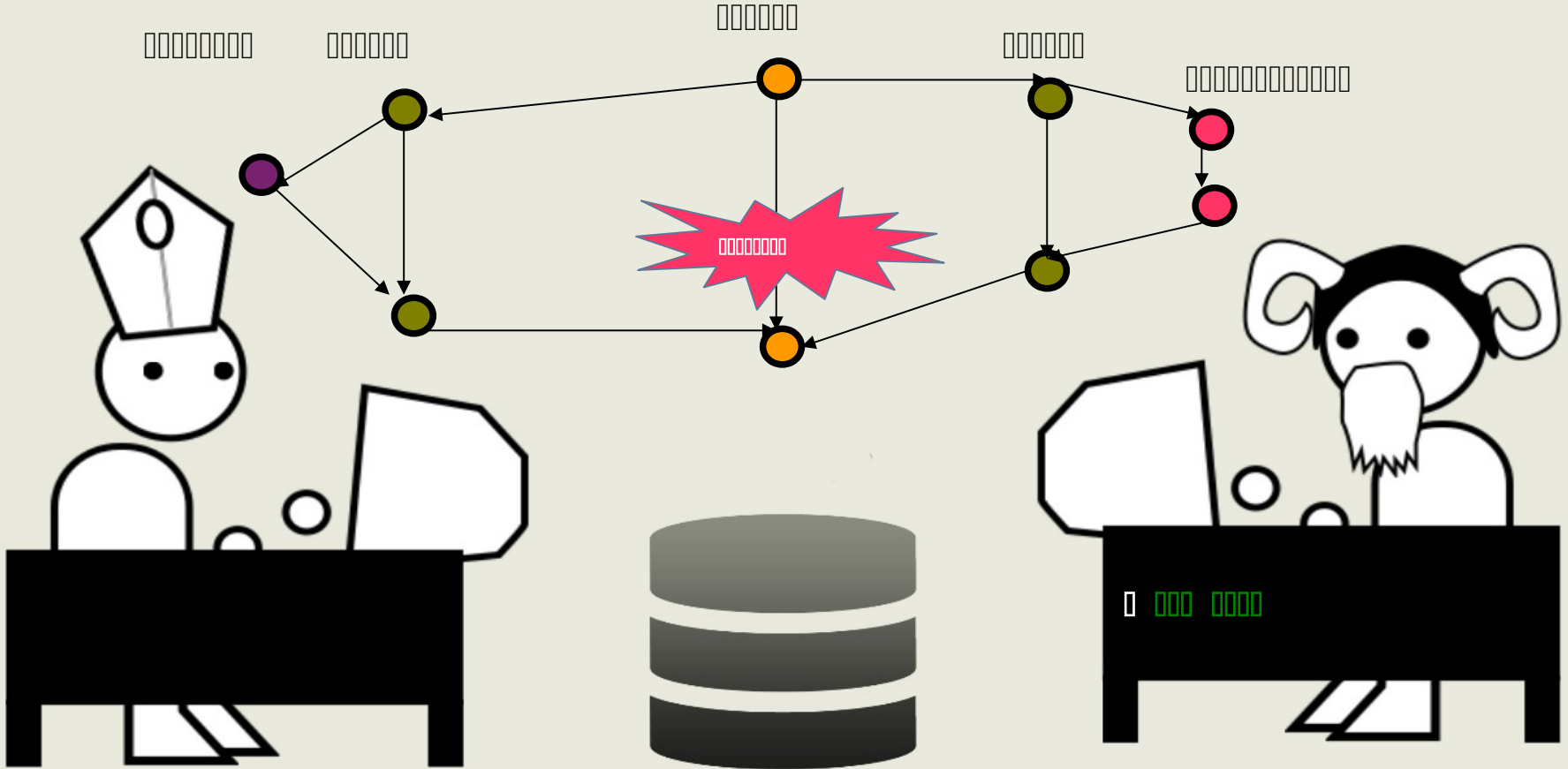
4.

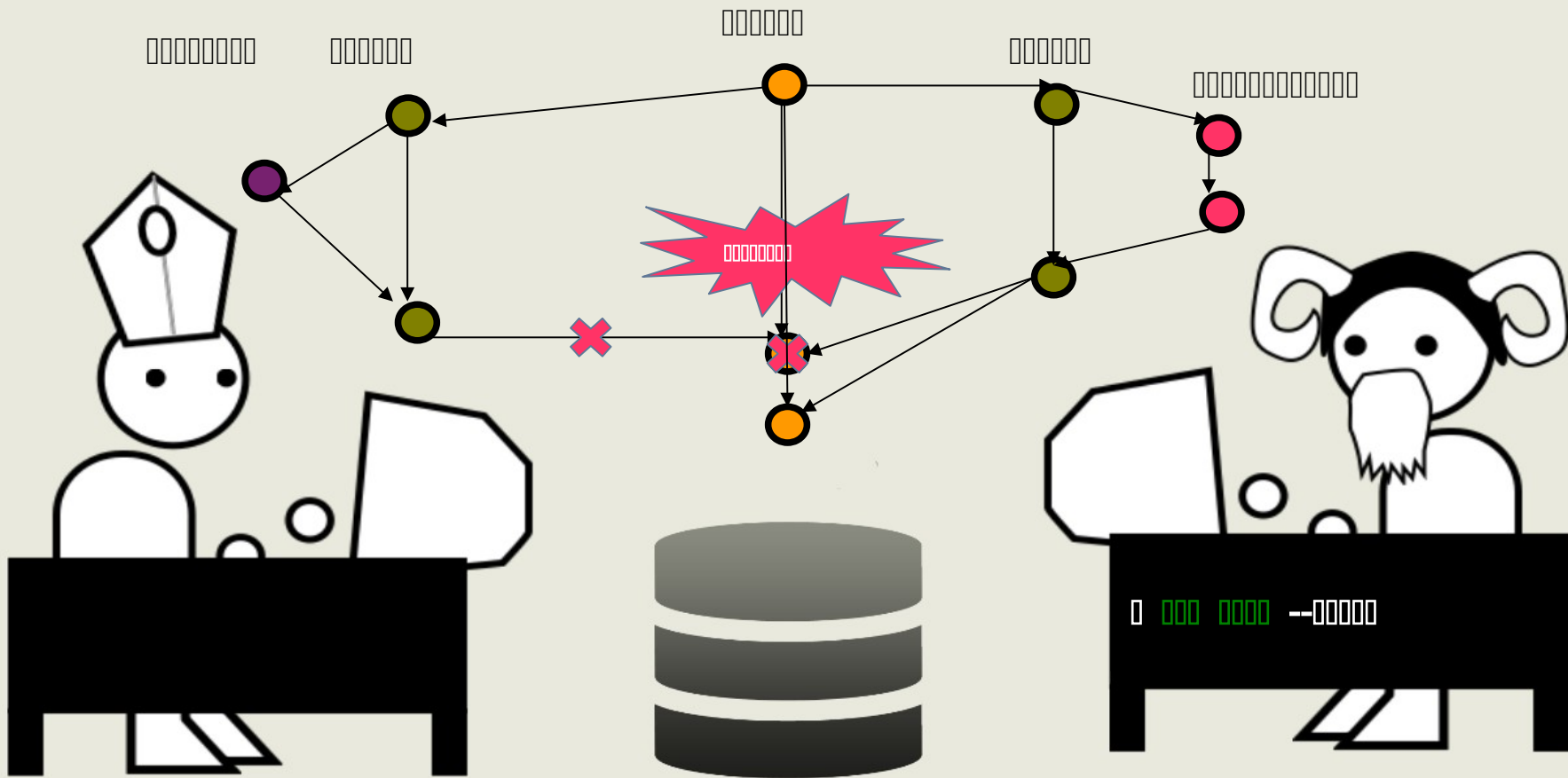
□□□□□□□□ □□□ □ □□□□□□□

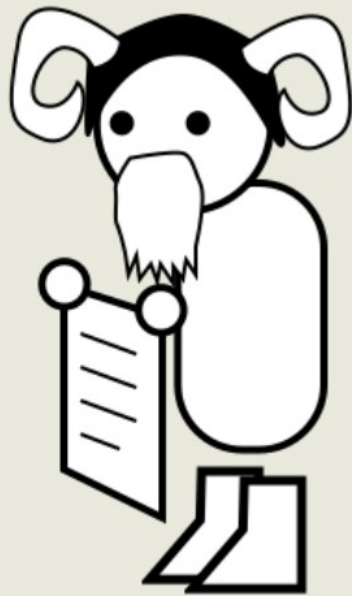


အိတ်အိတ်အိတ် အိတ် အိတ်အိတ်

အိတ်အိတ် အိတ်အိတ် အိတ်အိတ်, ဝ အိတ်အိတ်အိတ်အိတ် အိတ်အိတ်အိတ် အိတ်အိတ် အိတ် အိတ်အိတ်အိတ် အိတ်အိတ်အိတ်အိတ် အိတ်အိတ်အိတ်. အိတ်အိတ် အိတ် အိတ်အိတ် အိတ်အိတ် အိတ် အိတ်အိတ်အိတ်အိတ် (အိတ်အိတ် အိတ် အိတ်အိတ်အိတ်အိတ်, အိတ် အိတ်အိတ် အိတ်အိတ်)

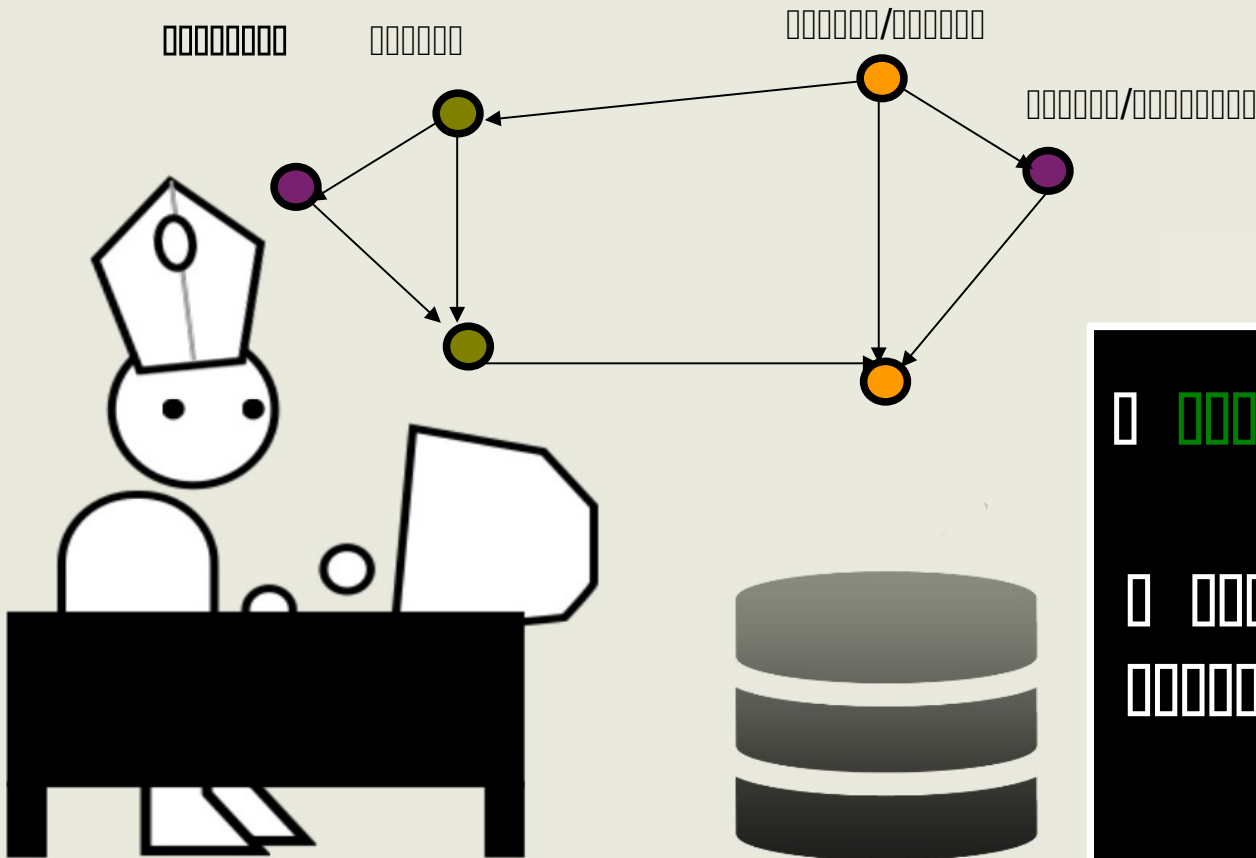






□□□□□□ □□□□□□

□□□□□□□□□□□□, □□ □□□□ □□□□□□ □□□□□□ □□ □□□□□□ □□□□□□ (□□□□□□□□□□□□). □□□□□□ □□□□□□ □□ □□□□ □□□□□□□□□□□□ □□□□□□ □□□□□□□□□□□□.



```
□ □□ □□□□□□ □□□□□□  
□ □□ □□□ -□ □□□□□  
□□□□□□□□
```

□□□□□□□□□□

- » □□□□□□□□□□ □□ □□□□□□ □□□□□□ : □□□ □□□□□□ -□ [□□□□□]
- » □□□□□□□□□□ □□ □□□□□□ □□□□□□: □□□ □□□□ □□□□□□ --
□□□□□□ [□□□□□]
- » □□□□□□□□□□ □□ □□□□□□ □□□□□□: □□□ □□□□□□ -□ [□□□□□]
- » □□□ □□□ □□□ □□□□□ □□□□□□□, “□□□□□”: □□□ □□□ --□□□□□□□□□□
--□□□□□□□□□□ --□□□□□□ □□□□□□

5.

□□□□□□ □□ □□□□□□

□□□□ □□□□ □□ □□□□□□□□ □□ □□□□□□

‘0 0000000000?

00000000 00000 000000 0
 000000000000 0000 00
 000000000. 000 0
 000000000000000000
 00000000000 0 00 000000
 0000 00000 00 0000000 00
 000000000 (0 000
 0000000000). 0000000000
 0000000.





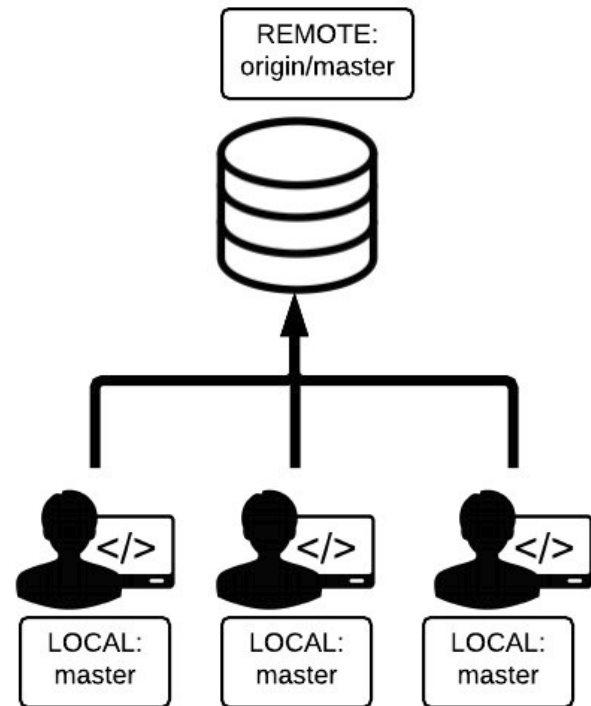
Git 本地仓库与远程仓库

Git 本地仓库 (即 .git) 与远程仓库 (即“代码托管平台”)。

本地仓库与远程仓库之间通过“推送”与“拉取”操作进行同步。

本地仓库与远程仓库之间通过“推送”与“拉取”操作进行同步。本地仓库与远程仓库之间通过“推送”与“拉取”操作进行同步。

本地仓库与远程仓库之间通过“推送”与“拉取”操作进行同步。



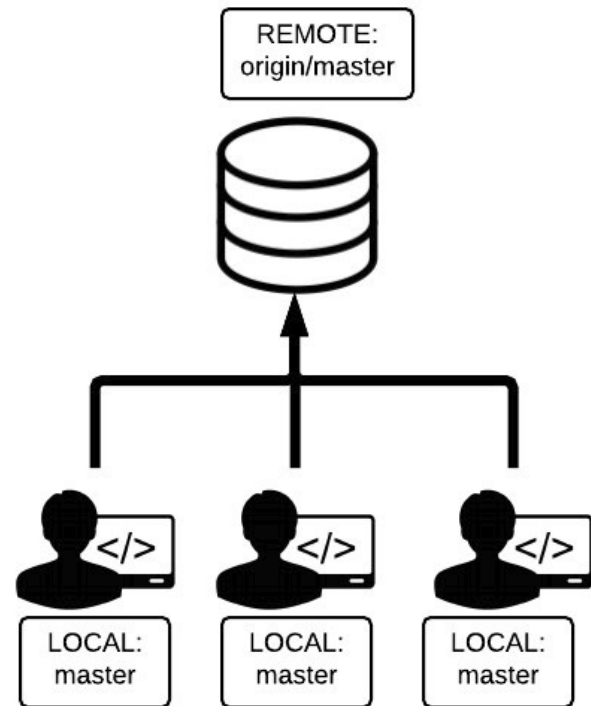
分布式数据库

分布式数据库

- » 数据库 库
- 数据库
- » 数据库 库 库
- 数据库

分布式数据库

- » 数据库 数据库
- 数据库 库
- 数据库
- 数据库
- 数据库
- » 库 数据库 库
- 数据库
- 数据库
- 库

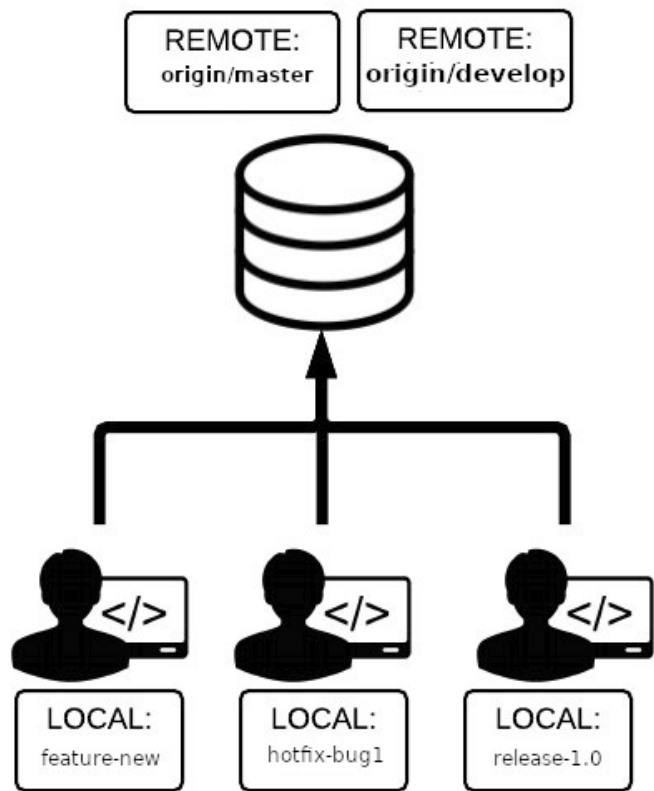




Git 2010

Git 2010 年推出，是分布式版本控制系统。它的设计理念是“去中心化”，每个开发者的本地机器上都有一个完整的代码库副本。这允许开发者在没有网络连接的情况下进行开发，并且可以随时随地提交代码。

- » 本地分支/工作区 — 本地开发环境，用于开发新功能（如新功能，“新功能”）。
- » 远程仓库 — 用于存储代码的中央位置。

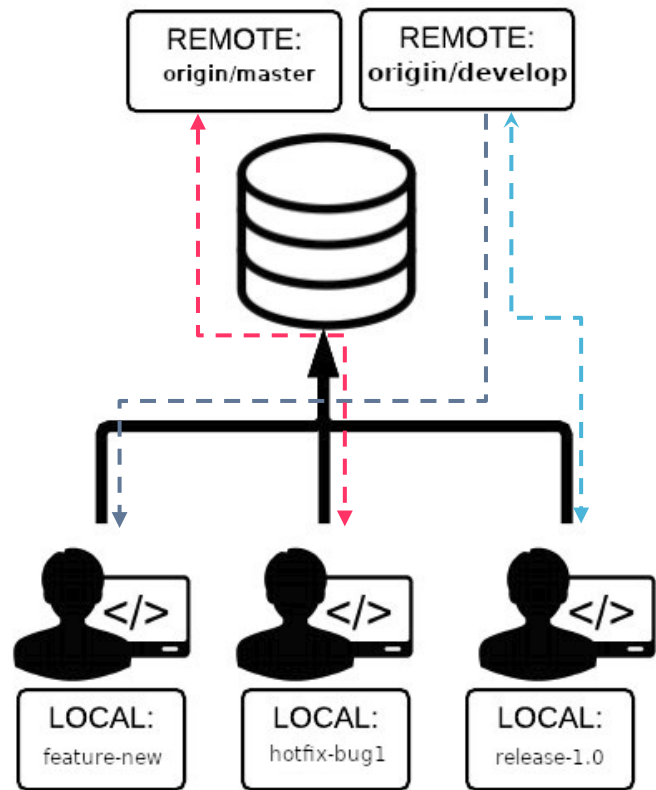


Git (2)

Git 的分支管理策略，通常是指 Git 的分支管理策略，即如何管理分支。

- » 分支名-* — 用于开发新功能、修复 bug 的分支。
- » 分支名-* — 用于发布生产环境的分支。
- » 分支名-* — 用于测试的分支。

• [Git 分支管理策略](#)



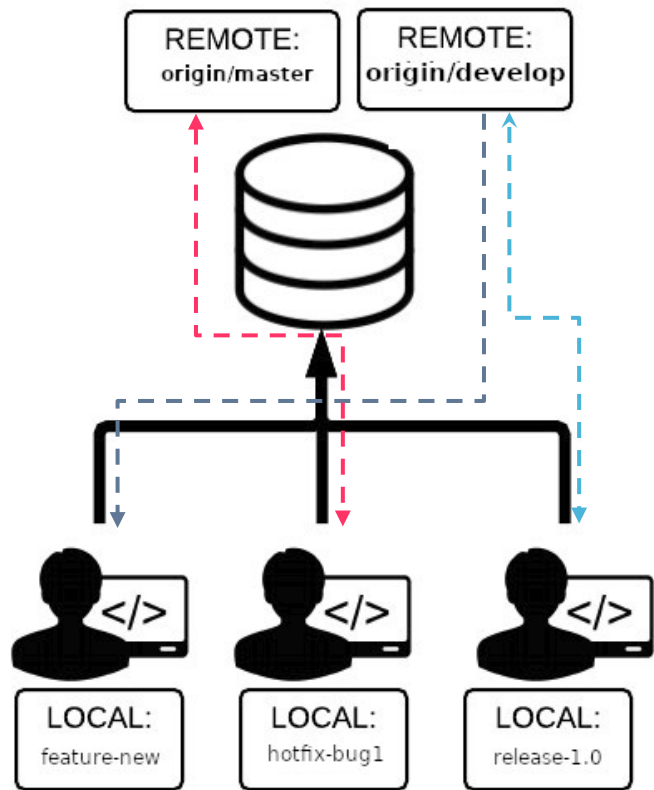
Git 分支管理

Git 分支管理

- » 分支管理是 Git 中非常重要的功能。
- » 分支管理可以帮助我们管理不同的开发任务。
- » 分支管理可以帮助我们管理不同的版本。

Git 分支管理

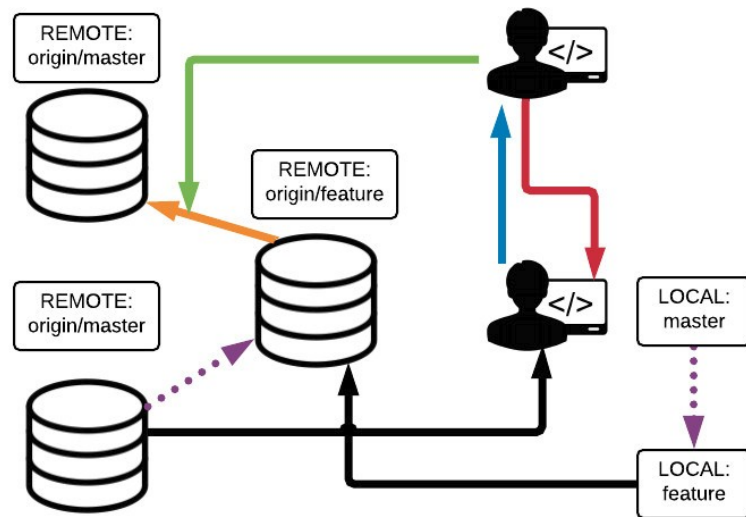
- » 分支管理可以帮助我们管理不同的开发任务。
- » 分支管理可以帮助我们管理不同的版本。
- » 分支管理可以帮助我们管理不同的版本。 1 (分支)。





Git 本地分支 本地分支 本地分支

Git 本地分支 本地分支 本地分支。本地分支 本地分支 本地分支，本地分支 本地分支；本地 本地分支 本地分支 本地分支 本地分支 本地分支。本地 本地分支 本地分支 本地分支 本地分支 本地分支 (本地分支 本地分支 本地分支 本地分支)。本地分支 本地分支 本地分支 本地分支 本地分支 本地分支 本地分支 本地分支 本地分支 本地分支。本地分支 本地分支 本地分支 本地分支 本地分支 本地分支 本地分支 本地分支 本地分支 本地分支。



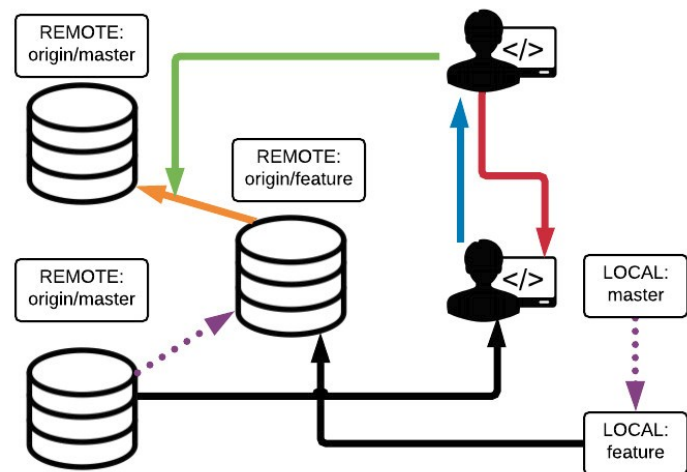
Git 分支管理

分支管理

- » 分支管理是 Git 中最重要的功能之一，它允许你在不同的分支上并行开发新功能，而不会影响生产环境。
- » 分支管理可以帮助你更好地组织代码，提高开发效率。

分支管理

- » 分支管理可以帮助你更好地组织代码，提高开发效率。
- » 分支管理可以让你在开发新功能时，可以随时切换到主分支，确保生产环境的稳定性。





□□□□□□□□ &

□□□□/□□□□□□

□□□□□□□□



□ 0000000?

0000000 0000000000 0 0000 0000 000 00000000000 00000000 00
0000 00000000 (0000000, 00000000000)/000000 00000000 (00000000).

0000 0000 00000000 00 000000000000 0 0000 00000000000 000000
00000000000, 00 000000 000 00000000000 00 0000000000 00 0000000000
00000000 00 000000 000000000000000. 00000000 000000 00000000000 00
00000000000 00000000 0000000000000, 00000000000 0000000 00 00000000000.



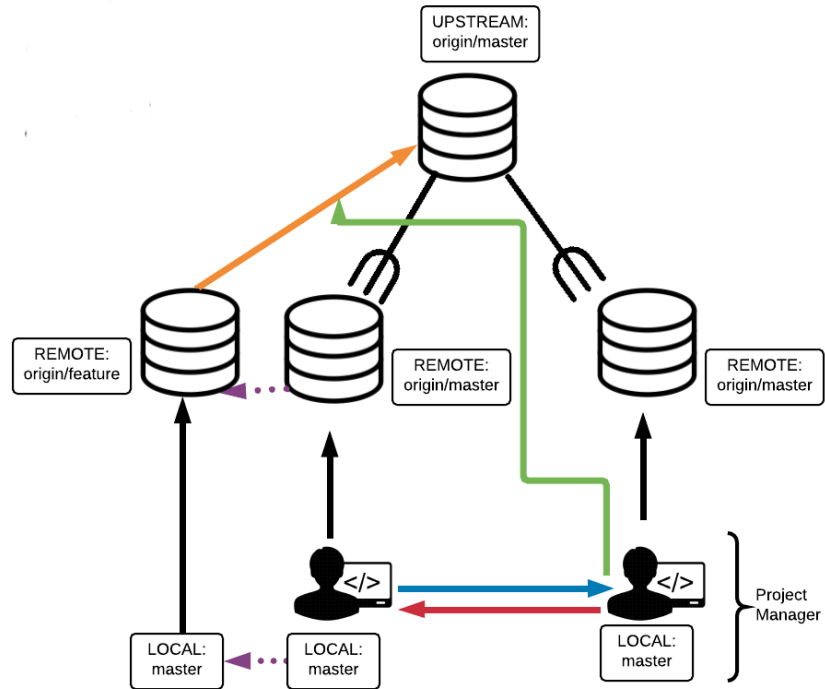
Git 工作流

Git 工作流 (Workflow) 是指团队在开发过程中遵循的一套规则和流程。

不同的团队可能会采用不同的工作流，但有一些常见的模式。了解这些模式可以帮助你更好地融入团队，并提高开发效率。

本文将介绍几种常见的 Git 工作流，包括分支策略、提交规范以及合并流程。这些工作流旨在减少冲突、提高代码质量，并确保生产环境的稳定性。

在开始之前，我们需要明确一些基本概念。Git 工作流的核心在于如何管理分支和提交。不同的分支通常用于不同的开发阶段，如开发、测试和生产。





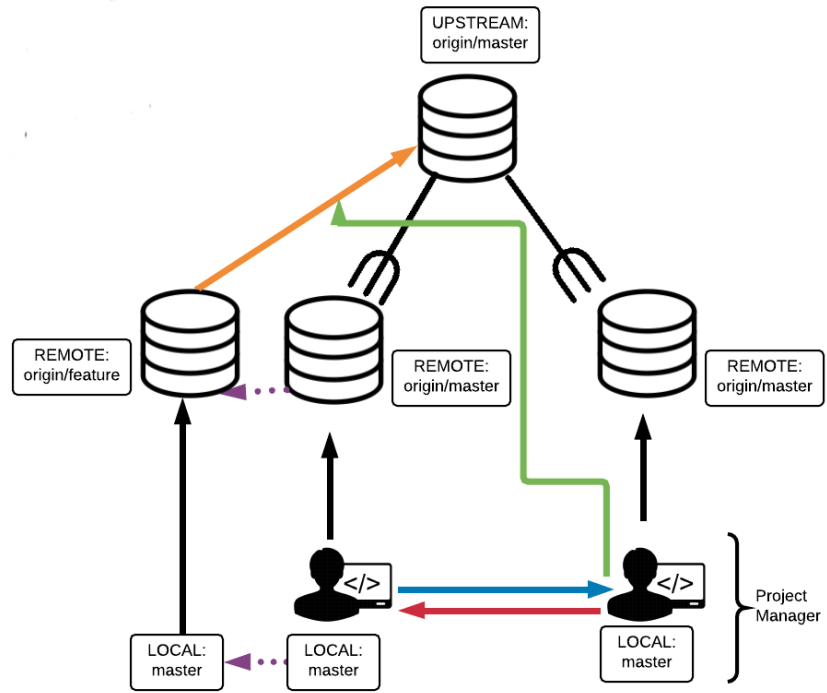
Git 專案管理

Git 專案管理

- » 專案管理 專案管理
- 專案管理 專案管理, 專案管理 專案管理
- 專案管理 專案管理
- 專案管理 專案管理
- 專案管理 專案管理

Git 專案管理

- » 專案管理 專案管理
- » 專案管理 專案管理
- 專案管理 專案管理
- 專案管理 專案管理



በግንባታ ላይ የሚደረግ ጥናት

ግንባታ

ግንባታ ለማከናወን የሚያስፈልጉትን ጥንቃቄ ለማድረግ የሚያስፈልገውን ጥናት ያደርጋል።

ጥናት

ጥናት ለማከናወን የሚያስፈልጉትን ጥንቃቄ ለማድረግ የሚያስፈልገውን ጥናት ያደርጋል።

ጥናት

ጥናት ለማከናወን የሚያስፈልጉትን ጥንቃቄ ለማድረግ የሚያስፈልገውን ጥናት ያደርጋል።

ጥናት

ጥናት ለማከናወን የሚያስፈልጉትን ጥንቃቄ ለማድረግ የሚያስፈልገውን ጥናት ያደርጋል።

ጥናት

ጥናት ለማከናወን የሚያስፈልጉትን ጥንቃቄ ለማድረግ የሚያስፈልገውን ጥናት ያደርጋል።

ጥናት/ጥናት ጥናት

ጥናት ለማከናወን የሚያስፈልጉትን ጥንቃቄ ለማድረግ የሚያስፈልገውን ጥናት ያደርጋል።



6.

□□□□□□□□

□□□□□□□□□□□□□□□□



git-quick-stats

```

git-quick-stats --help
git-quick-stats --generate
git-quick-stats --list
git-quick-stats --suggest
git-quick-stats --generate --branch=main
git-quick-stats --generate --author=johndoe
git-quick-stats --generate --format=csv
git-quick-stats --generate --format=json
git-quick-stats --list --branches
git-quick-stats --list --contributors
git-quick-stats --list --commits-per-author
git-quick-stats --list --commits-per-date
git-quick-stats --list --commits-per-month
git-quick-stats --list --commits-per-weekday
git-quick-stats --list --commits-per-hour
git-quick-stats --list --commits-per-hour-by-author
git-quick-stats --list --commits-per-timezone
git-quick-stats --list --commits-per-timezone-by-author
git-quick-stats --suggest

```

```

Terminal — git-quick-stats
Generate:
 1) Contribution stats (by author)
 2) Contribution stats (by author) on a specific branch
 3) Git changelogs (last 10 days)
 4) Git changelogs by author
 5) My daily status
 6) Output daily stats by branch in CSV format
 7) Save git log output in JSON format

List:
 8) Branch tree view (last 10)
 9) All branches (sorted by most recent commit)
10) All contributors (sorted by name)
11) Git commits per author
12) Git commits per date
13) Git commits per month
14) Git commits per weekday
15) Git commits per hour
16) Git commits per hour by author
17) Git commits per timezone
18) Git commits per timezone by author

Suggest:
19) Code reviewers (based on git history)



Please enter a menu option or press Enter to exit.
>

```





■■■■■■■■■■■■ ■ ■■■■■■■■■■
 ■■■ ■■■■■ ■■■ ■■■■■■ ■■
 ■■■■■■ ■■■■■■■ ■■■■■■■■■■■■.

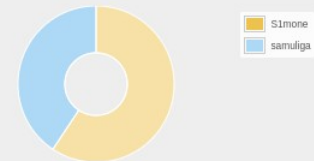
Le seguenti informazioni storiche sui commit, per autore, sono state trovate nel repository.

Autore v	Commit	Inserimenti	Rimoziioni	% dei cambiamenti
 S1mone	11	163	12	53.19
 samuliga	5	149	5	46.81







Di seguito il numero di righe da ogni autore che sono sopravvissute a sono ancora intatte nella versione corrente.

Autore v	Righe	Stabilità	Età	% in commenti
 S1mone	175	107.4	0.1	16.00
 samuliga	120	80.5	0.0	12.50



La seguente timeline storica è stata ricavata dal repository.

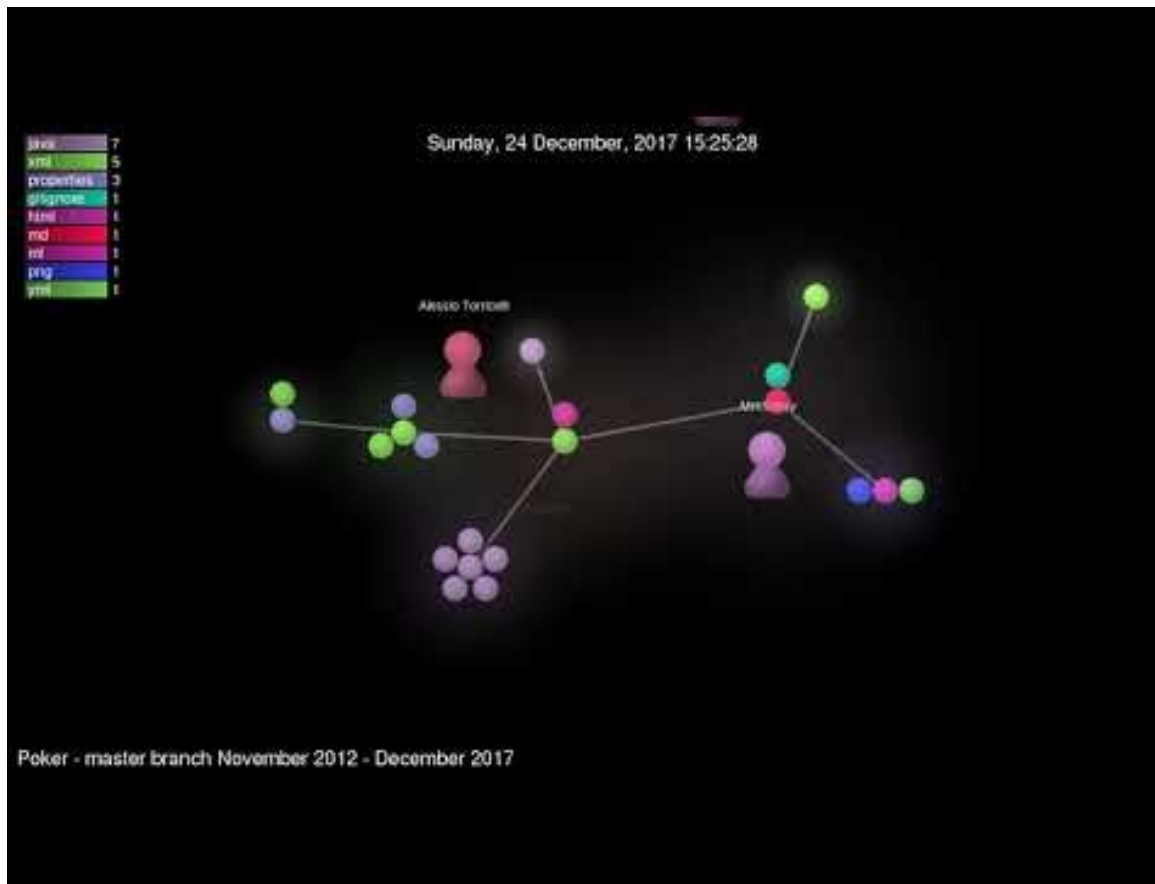
Autore	2021W12
 S1mone	
 samuliga	
Righe modificate:	329

Non sono state rilevate violazioni delle metriche nel repository.

Git

Git er et distribueret system til at håndtere projekter, der består af mange filer og undermapper. Det er et decentraliseret system, hvilket betyder, at hver bruger har en lokal kopi af hele projektets historie og kan arbejde offline. Git er meget hurtig og effektiv, og det er meget brugt i softwareudvikling.

Git er et decentraliseret system, hvilket betyder, at hver bruger har en lokal kopi af hele projektets historie og kan arbejde offline. Git er meget hurtig og effektiv, og det er meget brugt i softwareudvikling.



[[[[[[[[[[[[[[[[[[[[[[
[[[[: [[[[, [[[[[[[[[[, [[[[
[[[[[[[[[[[[

-- [[[[[[[[[[(1804-1881)



HTML 2 (DOM-Tree)

HTML 2 DOM, HTML
 DOM-Tree HTML DOM
 HTML DOM HTML DOM
 HTML DOM HTML DOM
 HTML DOM HTML DOM HTML DOM,
 HTML DOM HTML DOM HTML DOM
 HTML (HTML DOM HTML DOM HTML
 HTML' HTML DOM).

HTML DOM - "HTML DOM HTML DOM.

>

>

HTML-DOM-Tree: HTML

<HTML-DOM-Tree>

HTML-DOM-Tree: HTML-DOM-Tree

<HTML-DOM-Tree>

HTML DOM

በግንባታው ላይ ለሚከተሉት ሰነድ (ገንዘብ - ገንዘብ)

በግንባታው ላይ ለሚከተሉት ሰነድ,
በግንባታው ላይ ለሚከተሉት ሰነድ
በግንባታው ላይ ለሚከተሉት ሰነድ

በግንባታው ላይ ለሚከተሉት ሰነድ.ገንዘብ ገንዘብ.ገንዘብ
በግንባታው ላይ ለሚከተሉት ሰነድ "ገንዘብ ገንዘብ"

□□□□□□!

...□ □□□□□□ □□□□

□□□□□□□□...



□□□□□□

□□□□□□□□□□□□ □ :

- » □□□□□□□□□□□□ □□□□□□ □□ □□□□□□□□□□□□□□
- » □□□□□□□□□□□□ □□ □□□□□□□□
- » □□□□ □□□□□□ (□□□□.□□□□□□□□□□□□.□□□□)
- » □□□□ □□□□□□□□□□ (□□□□□□□□.□□□□/□□□□□□□□)
- » □□□□□□□□ □□□□

□□□□□□□□ □□□□□□□□□□ □□□□□□□□□□□□ □□□□

□□□□□□□□□□□□□□□□□□□□ □□-□□ 3.0