



# Data Science

Session 2 - Collaborative development



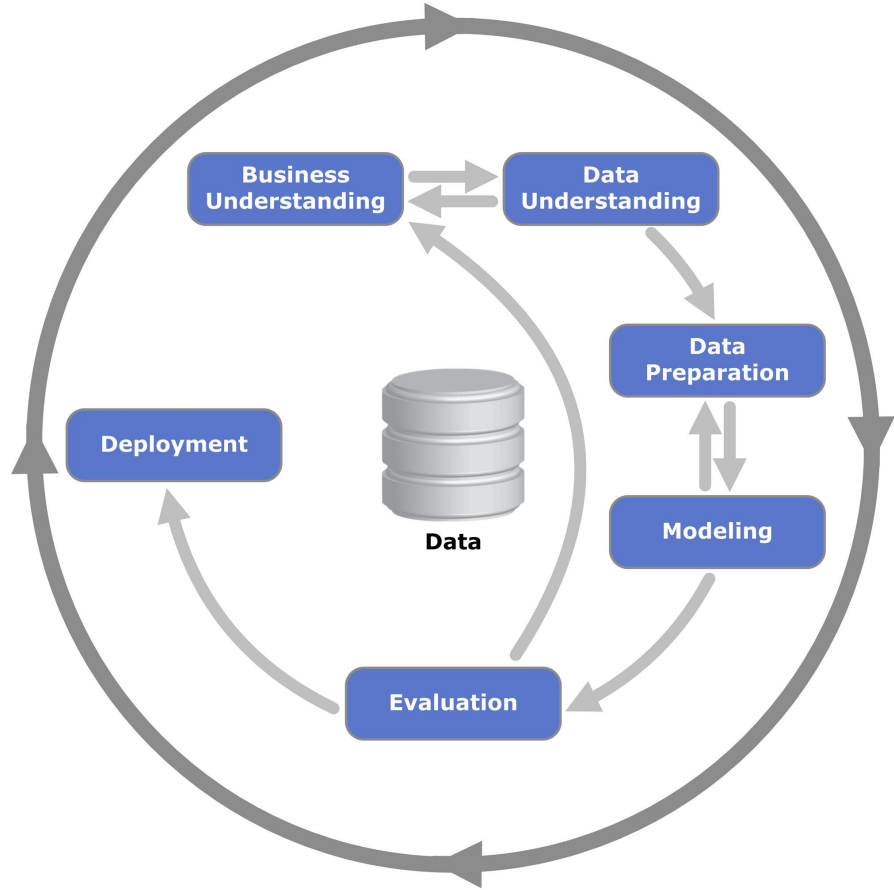
hadrien.salem@centralelille.fr



[introduction-to-data-science](#)

# Introduction

What did we do last time?



## The CRISP-DM method

**Cross-Industry Standard Process for Data Mining**

- Published in 1999
- Common in the industry
- Still relevant today

# Course outline

## Data science course

**Session 1:** Understanding data

**Session 2:** Collaborative development

**Session 3:** Preparing data - Managing missing data

**Session 4:** Preparing data - Dimensionality reduction

**Session 5:** Imbalanced data and deidentification

**Session 6:** Working with text



## Machine learning course

What the f\*\*\* does this have to do with Data Science?

# Why learn about collaborative development in a Data Science course?

## **Data scientists produce code**

- Research is pointless if you are not able to share your code with the people who will deploy it

## **Data scientists do not work alone**

- Research is pointless if you are the only one who can understand it

## **Version control systems are everywhere in the industry**

- Even as a manager, it is good to understand how your tech team manages their code

## **Open source projects are more and more popular**

- Platforms like GitHub and GitLab are what allow open source projects to grow

## **It's a good way to collect your works in class :)**

- I do need to evaluate your practicals. I'm sorry.



# What are version control systems?

# What are version control systems?

They are systems that allow the management of different versions for one or several files.

**Simplify code storage**

**Simplify code versioning**

**Keep a history of changes**

**Parallelize work**

# Vocabulary : Git, GitHub, GitLab and more



What is the difference?



# Vocabulary : Git, GitHub, GitLab and more



**git**



mercurial



FOSSIL

**Version control  
systems**



GitLab

ATLASSIAN



Bitbucket

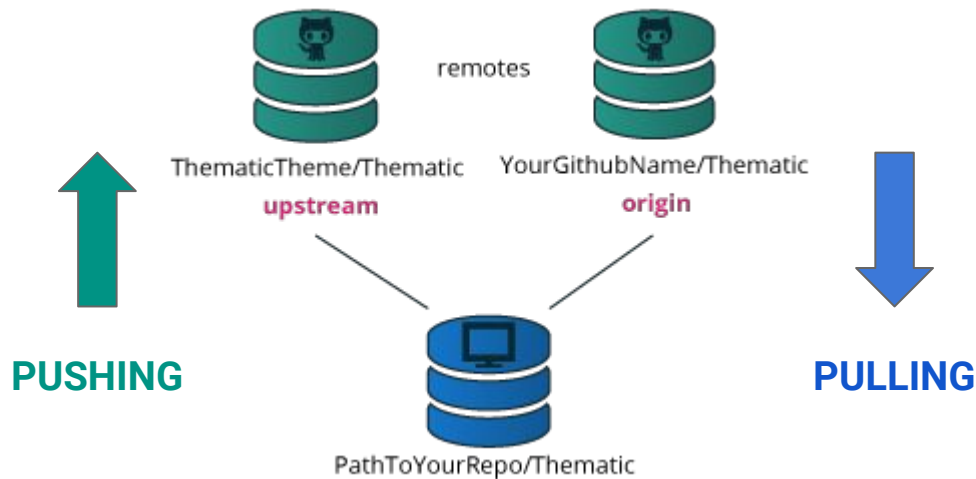


Space

**Services to store  
git projects**

# Fundamental git concepts

# Repositories and remotes



A **git repository** contains a `.git` folder

A **remote** is a **repository** hosted on the cloud.

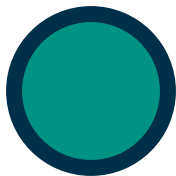
**NOTHING IS UPLOADED TO THE REMOTE  
UNTIL YOU PUSH IT YOURSELF!**

**You do not need a remote** to work with git.

Your local code can have **several remotes**.

A **fork** is an **independent copy** of a repository.

# Histories, commits and branches



Commit 000000000000000000000001

Author: Joe Mama <[joe.mama@centralegille.fr](mailto:joe.mama@centralegille.fr)>

Date: Fri Jul 27 15:29:25 2023

Message: Create slide 15 for GitHub session

+ slide15

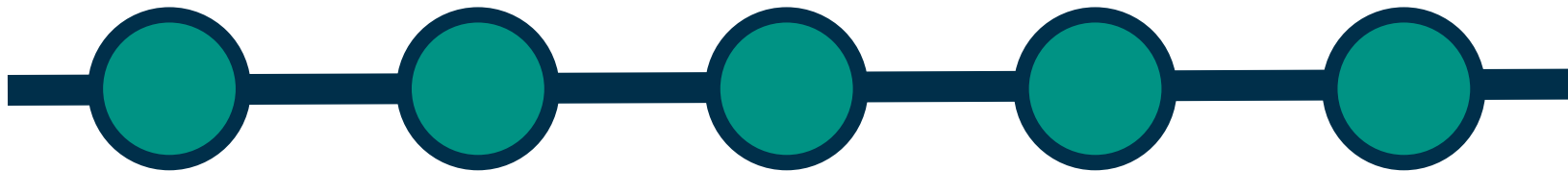
- slide25

renamed slide11 > slide 12

A **commit** is a list of changes.

Committing your work is **saving a version of the repository**.

# Histories, commits and branches

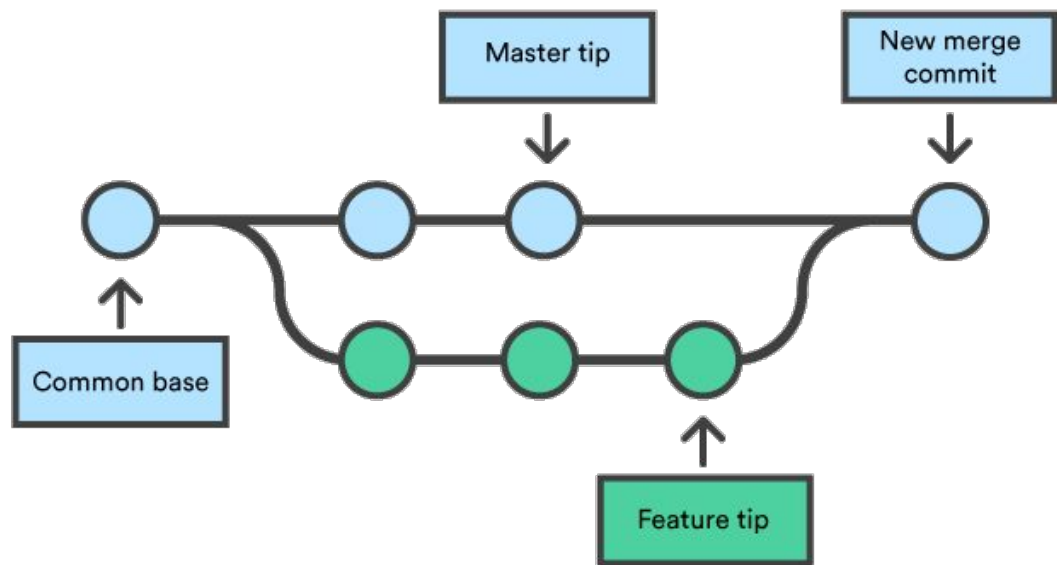


A **branch** is a sequence of commits.

They are a convenient way to manage your commits and their history.



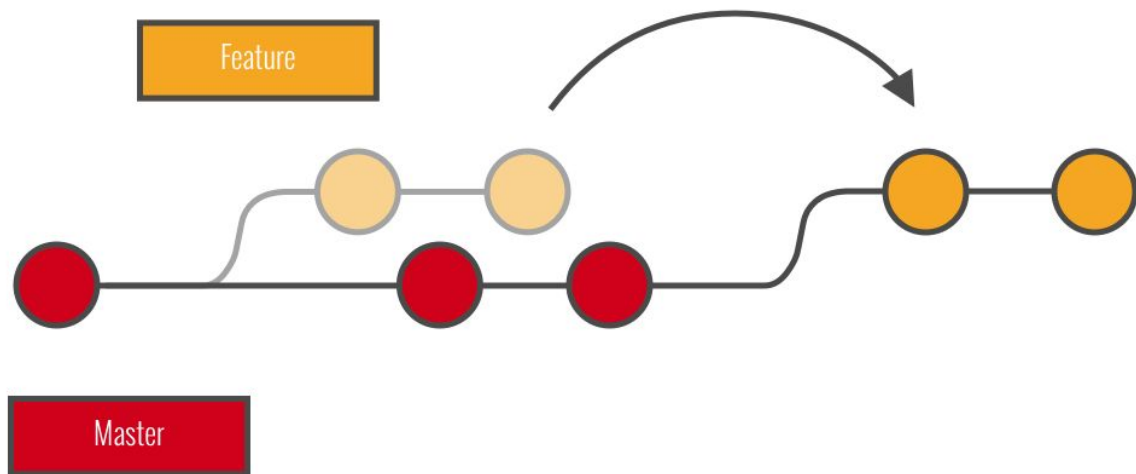
# Managing branches



Branches can be created from other branches to work in parallel.

Bringing back commits from a branch to the other is called **merging**.

# Managing branch histories



**Rebasing** a branch is **redefining its origin** by rewriting the commit history.

It can help having a **clean history**.

It can however generate **conflicts**.

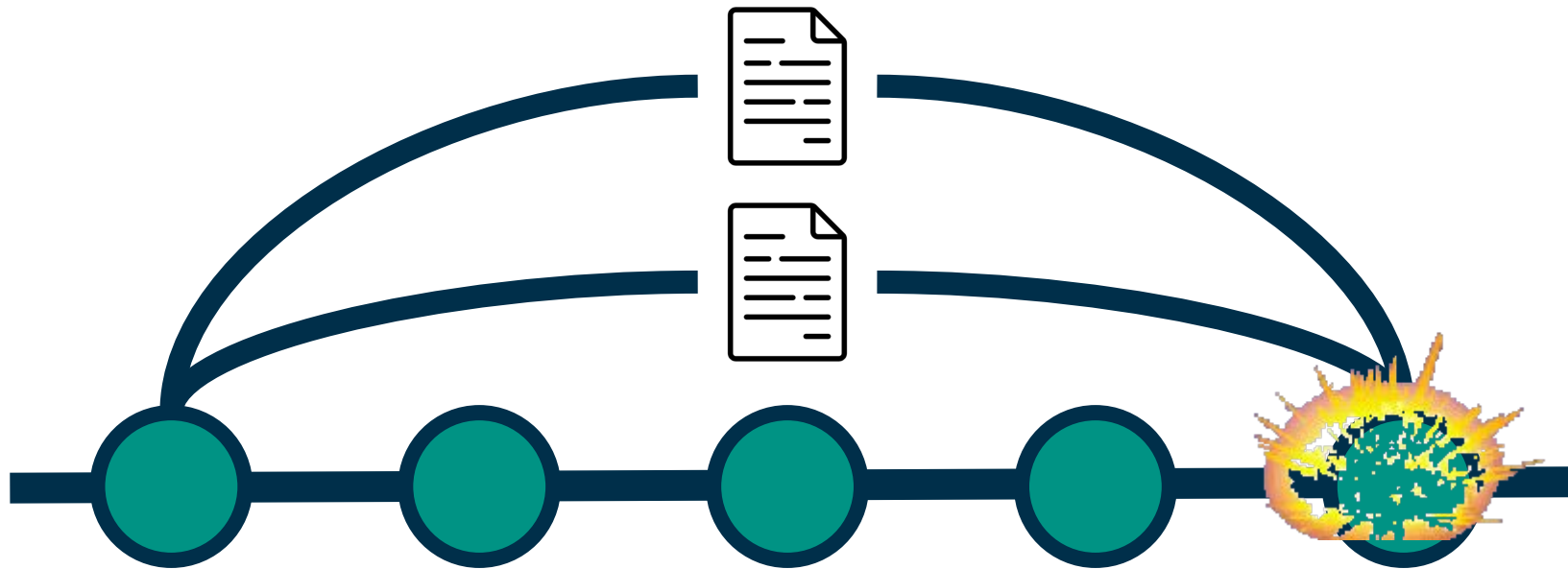


# NEVER CHANGE A SHARED BRANCH'S HISTORY

You WILL regret it. Force push responsibly.



# Managing conflicts



Conflicts happen when **histories do not match**.

This happens when **several people change the same file** and try to merge their work.

# Using git collaboratively

# Issues

Filters

Q is:issue is:open

Labels 26

Milestones 1

New issue

129 Open 509 Closed

Author Label Projects Milestones Assignee Sort

A shuffled drop's textbox might not display if you collect another item at the same time? bug

#2052 opened 16 hours ago by r0bd0g

Multiworld on EverDrive Component: ASM/C Component: Documentation enhancement

#2042 opened 2 weeks ago by fenhl

Recursion Error bug Component: Randomizer Core

#2032 opened 3 weeks ago by r0bd0g

2

Make grotto entrances act more like normal entrances bug Component: ASM/C

#2023 opened on Jun 25 by fenhl

1

Gossip Stone text boxes break for really long text (1200+ characters) bug Component: Plandomizer

#2017 opened on Jun 22 by ETR-BTF

2

Offline generator sometimes fails to generate WADs bug Component: Patching

#2016 opened on Jun 22 by ETR-BTF

2

When generating a WAD offline, clicking the "Ok" button after the ROM has been created but before the WAD has been created will prematurely end the WAD creation bug Component: GUI/Website

#2015 opened on Jun 22 by ETR-BTF

1

Going through loading zones on Epona in Entrance Randomizer can cause Link to be stuck on Epona, forcing the player to Save + Quit bug

#2012 opened on Jun 19 by ETR-BTF

3

ER logic bug with shops behind the GTG entrance bug

#2011 opened on Jun 19 by ETR-BTF

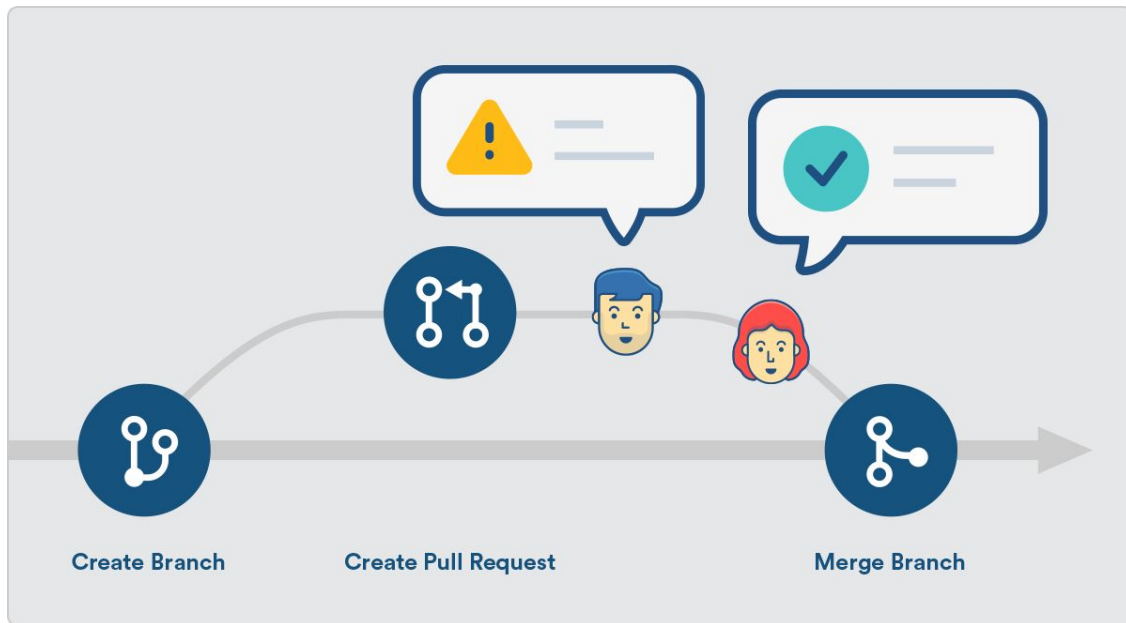
2

Executing the "Collection Delay Glitch" can softlock the game bug

#2010 opened on Jun 18 by ETR-BTF

Issues are akin to tasks.

# Pull requests

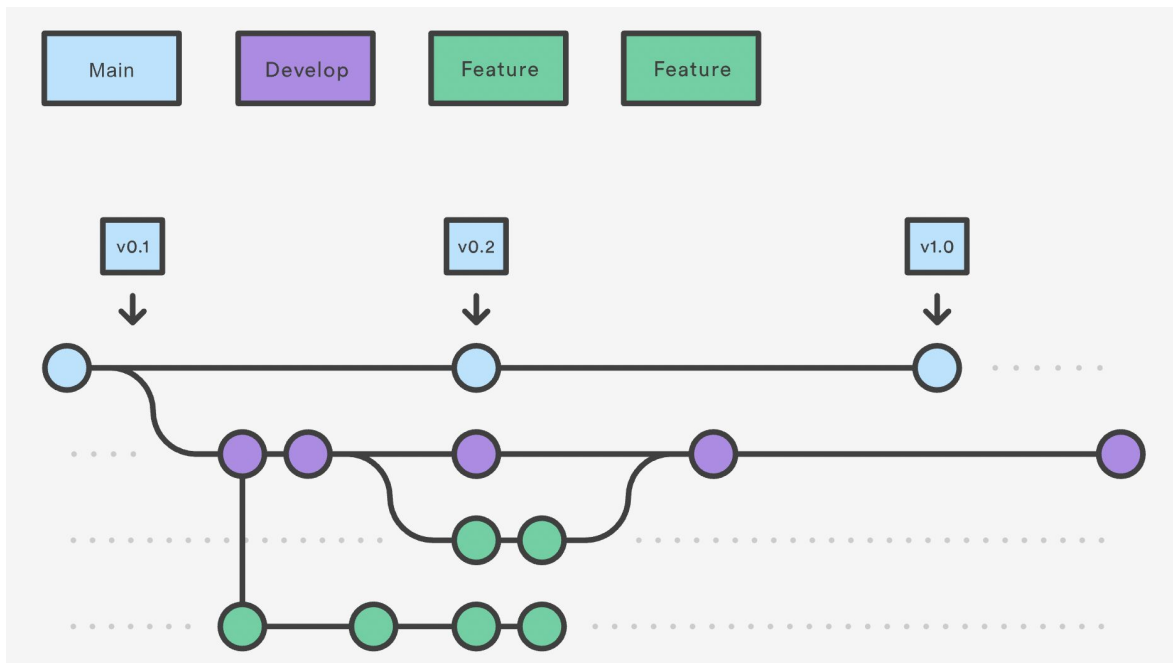


Pull requests are a **communication tool** between developers.

They allow for:

- Better code quality
- Knowledge sharing
- Communication

# Defining a git flow

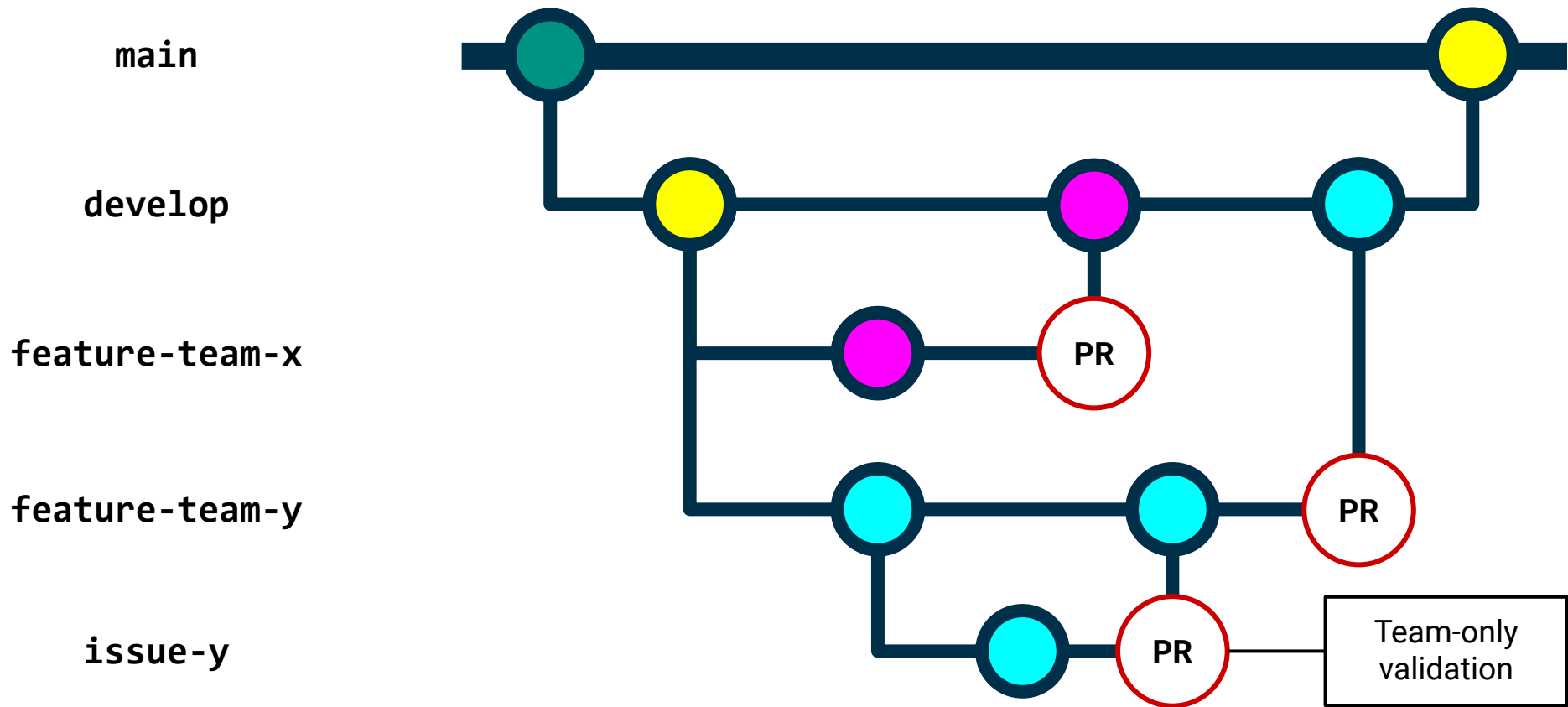


Git flows are **conventions** within development teams to manage their repository.



# Practical work

Let's build a mock app with Git!



How we will use GitHub as a class

practicals-2023-2024

main

session\_name-name1-name2



1. **Create a branch** from main with name `session-x-name1-name2` (e.g. `01_understanding_data-salem-salem`)
2. **Create a folder** with name `name1-name2` within the relevant folder (e.g. `01_understanding_data/salem-salem`)
3. Upload your work within the folder you created
4. Create a pull request for the practical

**Respecting these conventions is part of the evaluation!**

Don't forget to  
upload your work!

# Debrief

# Debrief



<https://forms.gle/z2zarAiDR8E5verp6>