6th DualSPHysics Workshop

25th – 27th October 2022, Campus Nord UPC

© J.Sospedra

Contribute to the DualSPHysics project through our repository

George Fourtakas

Department of Mechanical, Aerospace and Civil Engineering
Faculty of Science and Engineering
The University of Manchester,
Manchester, UK

DualSPHysics software

MANCHESTER 1824
The University of Manchester
UNIVERSITÀ DEGLI STUDI DI PARMA
UniversidadeVigo
NJIT New Jersey Institute of Technology
TÉCNICO LISBOA
UPC

OPEN-SOURCE CODE

AVAILABLE FOR FREE

COLLABORATIVE PROJECT

LGPL LICENSE

HIGHLY PARALLELISED

PRE- & POST-PROCESSING

REAL-LIFE PROBLEMS

JOURNAL PUBLICATIONS

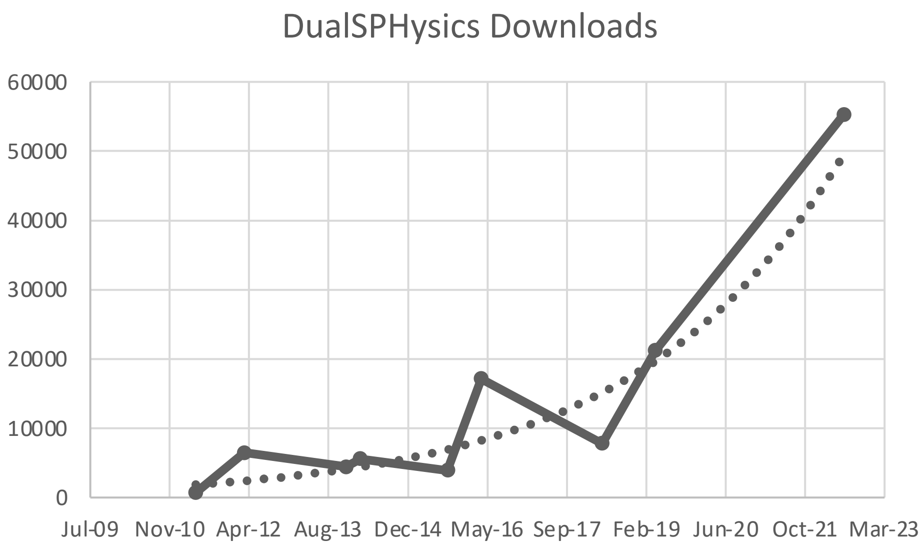
Current Developers:

- **Dr José M. Domínguez (jmdominguez@uvigo.es). Universidade de Vigo, Spain**
- **Dr Georgios Fourtakas (georgios.fourtakas@manchester.ac.uk). The University of Manchester, UK**
- **Dr Alejandro J.C. Crespo (alexbexe@uvigo.es). Universidade de Vigo, Spain**
- **Dr Benedict D. Rogers (benedict.rogers@manchester.ac.uk). The University of Manchester, UK**
- **Dr Renato Vacondio (renato.vacondio@unipr.it). Università degli studi di Parma, Italy**
- **Dr Corrado Altomare (corrado.altomare@upc.edu). Universitat Politecnica de Catalunya – BarcelonaTech, Spain**
- **Dr Angelo Tafuni (atafuni@njit.edu). New Jersey Institute of Technology, US**
- **Dr Orlando García Feal (orlando@uvigo.es). Universidade de Vigo, Spain**
- **Dr Joseph O'Connor (joseph.oconnor@imperial.ac.uk) Imperial College London, UK**
- **Iván Martínez Estévez (ivan.martinez.estevez@uvigo.es). Universidade de Vigo, Spain**
- **Professor Peter Stansby (p.k.stansby@manchester.ac.uk). The University of Manchester, UK**
- **Professor Moncho Gómez Gesteira (mggesteira@uvigo.es). Universidade de Vigo, Spain**

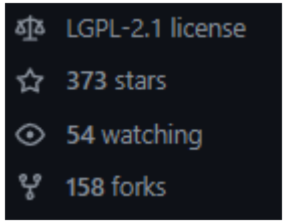
Current state of DualSPHysics

The DualSPHysics project has grown

- 100+k downloads* through the web page
- Dedicated forum with:
  - 23930 users
  - 2k discussion topics



- In 2017 a dedicated GitHub project was created which is extremely active with
  - 158 forks
  - Many GitHub “Issues” and ***pull requests***
  - Active community


LGPL-2.1 license
373 stars
54 watching
158 forks

*downloads (local clones) from GitHub are not being tracked

Collaborative development model

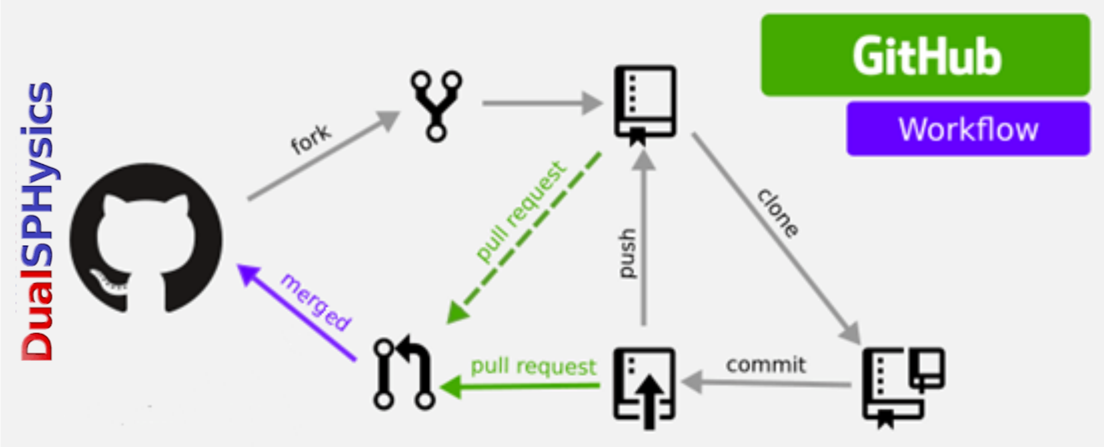
Fork and pull model

- The DualSPHysics GitHub project (https://github.com/DualSPHysics/DualSPHysics) contains repositories which are public
  - **DualSPHysics** and **DesignSPHysics** (more to be included in the near future)
- Access to the project and repos are restricted to maintainers (push access)
- Anyone can fork the existing repository (requires GitHub account) and push changes to their personal repo (fork)
- Changes can be pulled to the upstream repo (DualSPHysics) by opening a "pull request"
  - User-own fork to upstream branch
  - Allow push access to maintainers to make changes to your pull request

Collaborative development model

About **Forks:**

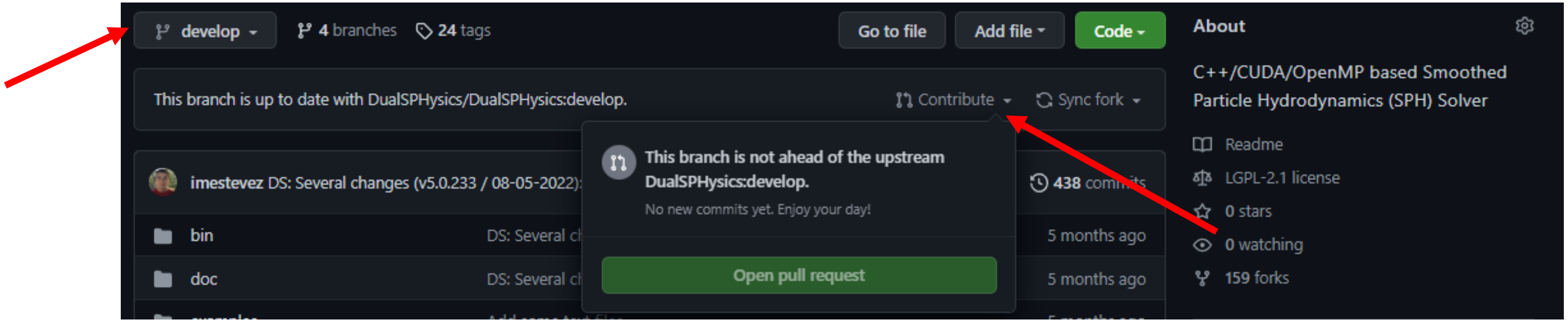
- A fork is your own copy of the repo
- You do not affect the upstream repo (DualSPHysics)
- You can “fetch” updates form the upstream repo
- You can use a **pull request** to suggest changes
  - Configure different remotes for the upstream repo and your own *origin*
  - Sync with upstream (git fetch upstream)
  - Create pull requests



Collaborative development model

Creating a pull request from a fork

- We are accepting pull requests on the *develop* and *develop_nn* branch of our repo **only**
  - Pull requests to other branches may be denied without reviewing



This branch is up to date with DualSPHysics/DualSPHysics:develop.

imestevez DS: Several changes (v5.0.233 / 08-05-2022):

This branch is not ahead of the upstream DualSPHysics:develop.

No new commits yet. Enjoy your day!

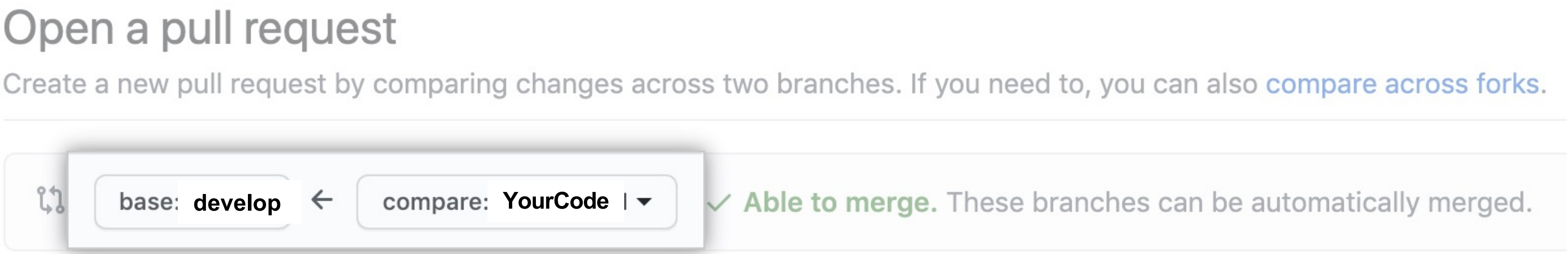
About

C++/CUDA/OpenMP based Smoothed Particle Hydrodynamics (SPH) Solver

Collaborative development model

Creating a pull request from a fork

- We are accepting pull requests on the *develop* and *develop_nn* branch of our repo **only**
  - Pull requests to other branches may be denied without reviewing



Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

base: develop ← compare: YourCode

✓ Able to merge. These branches can be automatically merged.

Collaborative development model

Creating a pull request from a fork

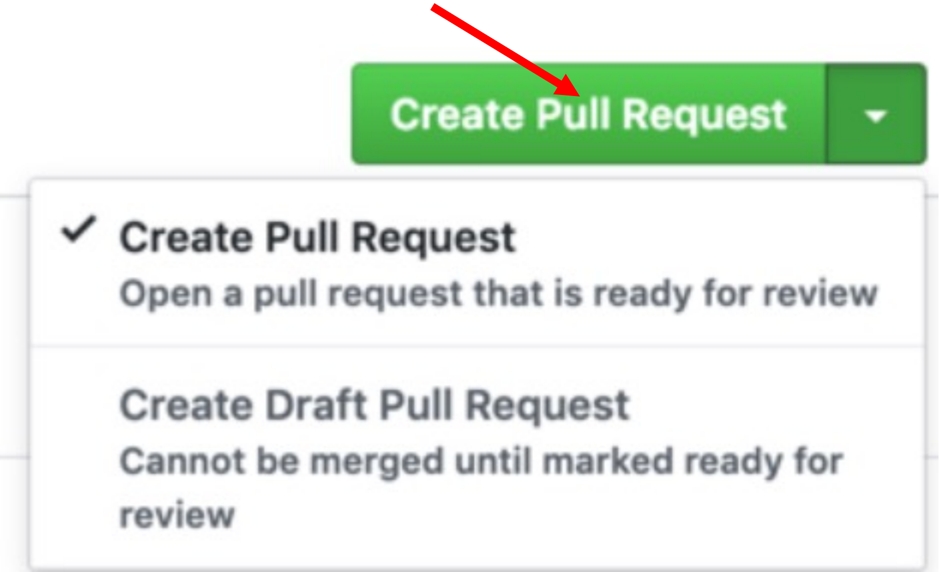
- Add your “Contributing.md” for the maintainer, we require a detailed description of the pull request:
  - Description: Summary of the changes and the related issue
  - Type of change
    - Bug fix (non-breaking change which fixes an issue)
    - New feature (non-breaking change which adds functionality)
    - Breaking change (fix or feature that would cause existing functionality to not work as expected)
    - This change requires a documentation update
  - Testing/validation: test and reproducibility of results
  - Checklist:
    - My code follows the style guidelines of this project
    - I have performed a self-review of my code
    - I have made corresponding changes to the documentation
    - My changes generate no new warnings
    - I have added tests that prove my fix is effective or that my feature works
    - My changes do not alter results from other cases/examples

A contribution.md template will be added to our GitHub soon

Collaborative development model

Creating a pull request from a fork

- We are accepting pull requests on the *develop* and *develop_nn* branch of our repo **only**
  - Pull requests to other branches may be denied “without reviewing them“



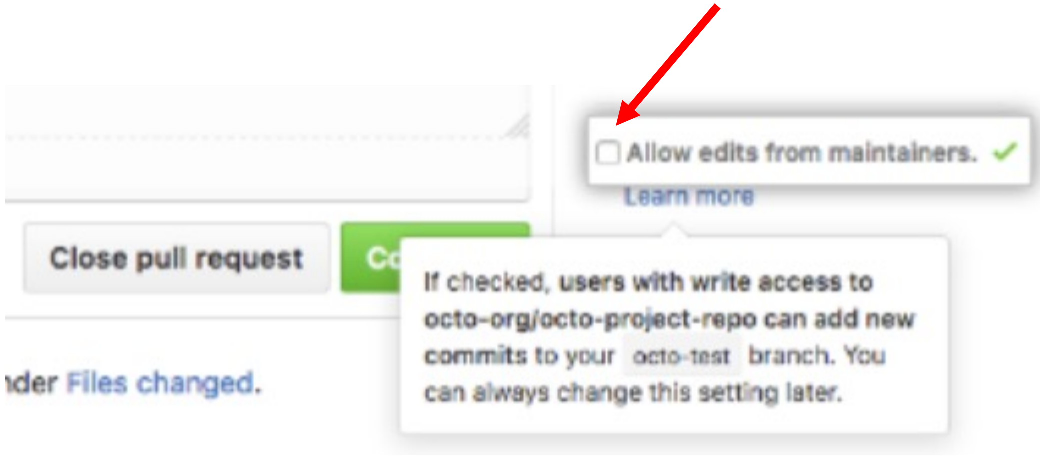
Create Pull Request ▾

✓ Create Pull Request

Open a pull request that is ready for review

Create Draft Pull Request

Cannot be merged until marked ready for review



□ Allow edits from maintainers. ✓

Learn more

Close pull request

Co

nder Files changed.

If checked, users with write access to octo-org/octo-project-repo can add new commits to your octo-test branch. You can always change this setting later.

Contribution models

Minor contribution - GitHub:

- Small code changes which do not impact on the general structure of the solver (i.e., bug fixes, 1- 10 lines of features, etc)
- Minor improvements in the formulation or implementation which do not change the results significantly
- Bugs which are beyond the GitHub “Issues“ scope

Major contribution - GitHub*:

- Code changes are significant and span beyond one file, introduction of new functions and calls, hardware acceleration
- Major improvements/reformulation of the scheme and/or models which improve or add extra functionality to the solver (i.e., higher accuracy, a new phases, new coupling techniques, etc)

*Major contribution tend to be already published in peer review journals

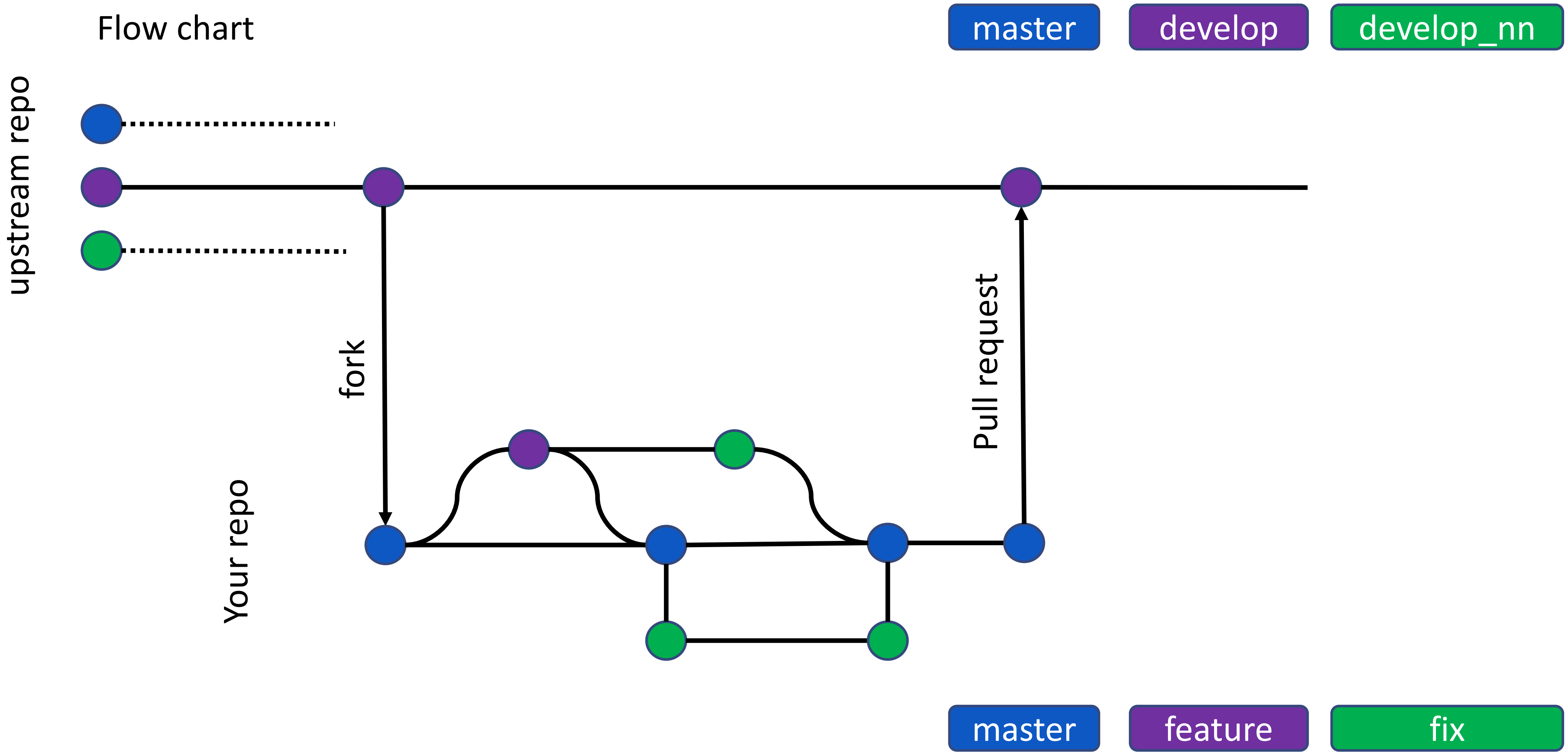
Contribution models

Minor/Major contribution as collaborator through GitHub/GitLab:

- Mostly reserved for major contributions as collaborators to the DualSPHysics project
- Improvements may be of computational or numerical nature
- At least one of the core developers must be associated with the contributions
- The contributions can be published or unpublished but must be published when merged to a release package
- If you are interested contributing through tis model, speak to a developer
- Advantage: We will provide access to your own private GitHub/GitLab repo and the latest DualSPHysics version (if required)!!!
- “Caveat”: **Requires large time commitment** (usually reserved for MPhil/PhD student or similar)

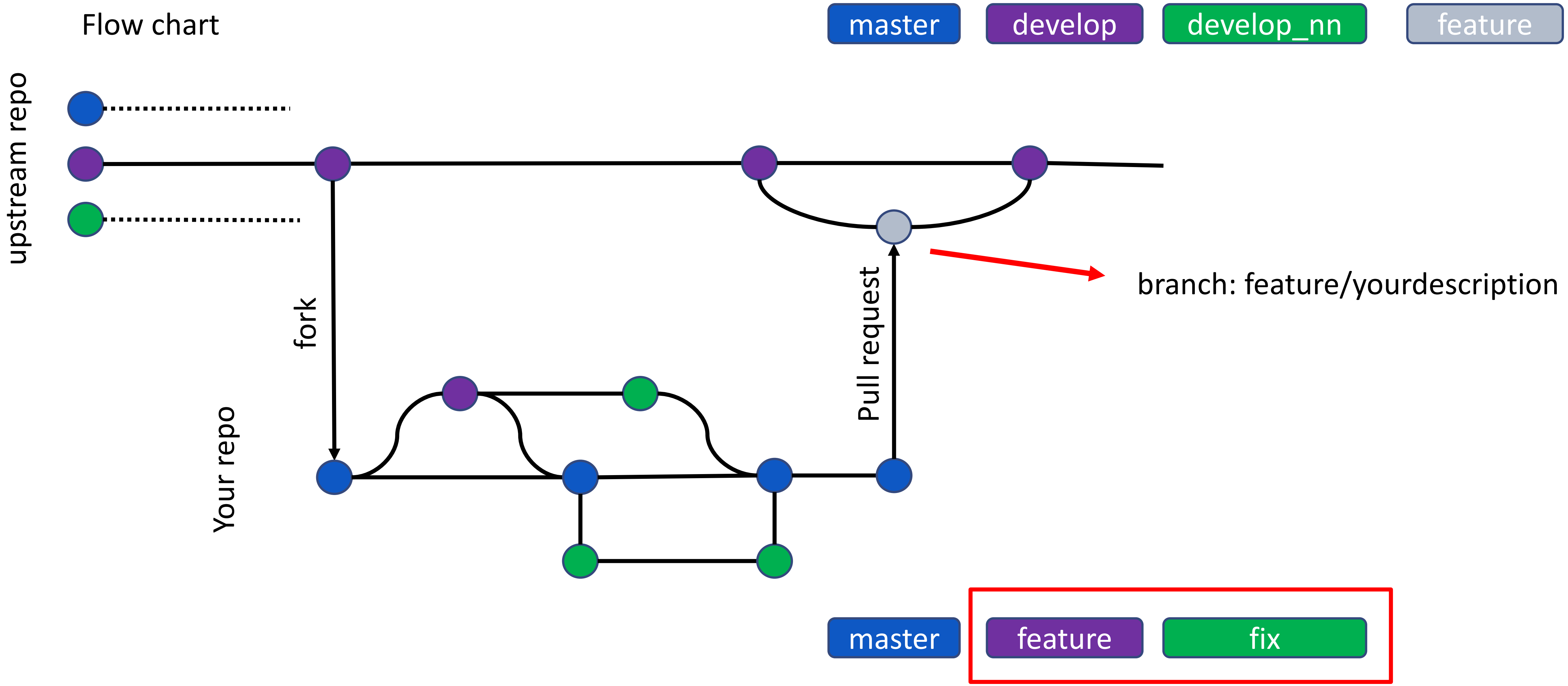
Minor contribution to GitHub

Flow chart



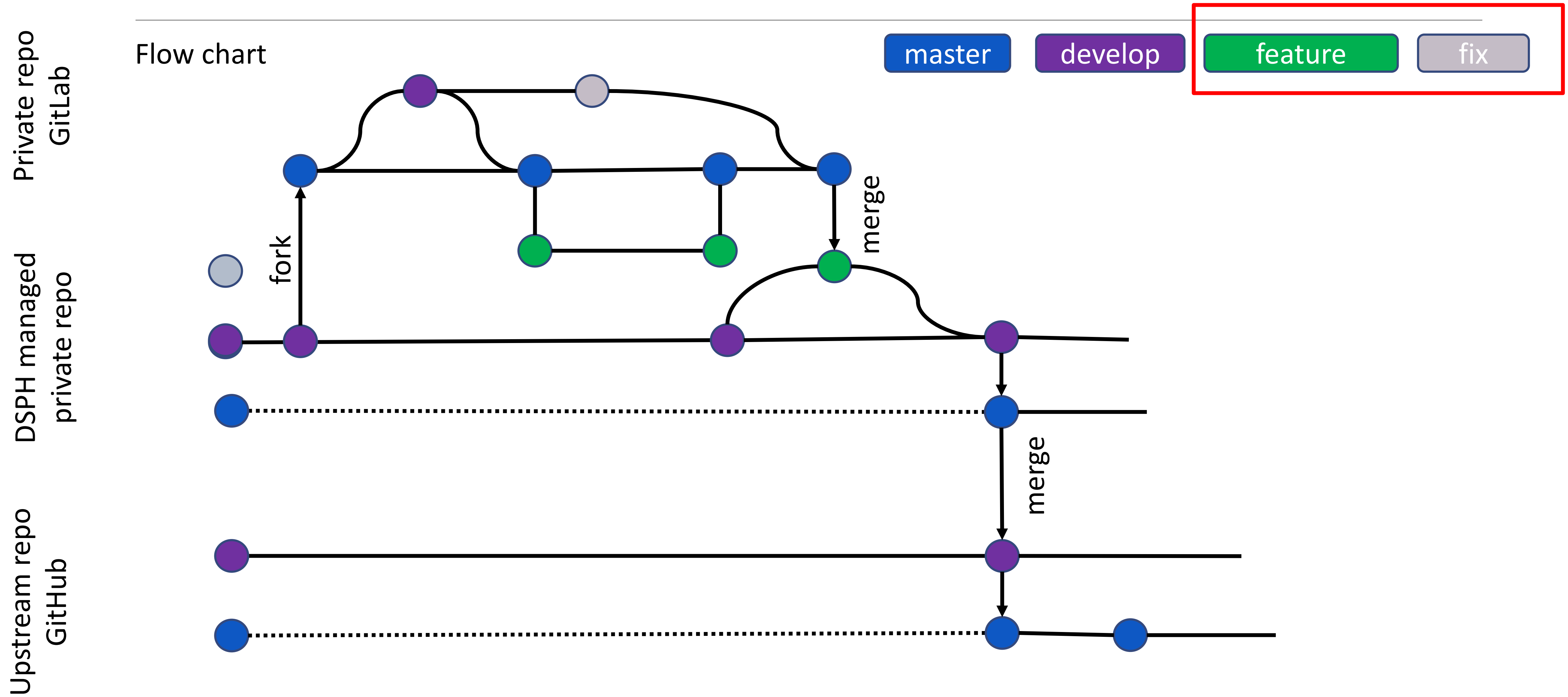
Major contribution to GitHub

Flow chart



branch: feature/yourdescription

Major contribution to GitL(H)ab



Code requirements and tests

A code structure and format guide will be added to our GitHub soon

Your implementation must conform with:

- Code structure and format (UseOurVariableNames please)
- XML switches/options (***no hard coding***)
- 2-D and 3-D
- CPU and **GPU**
- **Warnings** for features your modifications **are not compatible with** (see JSph.cpp for examples)

Code requirements and tests

- Full compatibility with at least one fluid solver (single phase, multiphase, flexstructures, etc)
  - Time stepping: Verlet & Predictor-Corrector
  - Wall boundaries: DBC and mDBC
  - Moving wall boundaries
  - Floating objects
  - Density diffusion terms
  - Shifting algorithms

Code requirements and tests

Your pull request must include

- Tests case(s) that show fix/feature are improving the results
  - at folder “*./examples/main/feature*”
  - with a batch (including pre- and post-processing) and xml file
- Documentation *“./doc”*
  - fix: a short pdf document highlighting the issue and fix/solution
  - feature: a pdf which discusses the computational/numerical advances and implementation (or journal paper), functionality and options (i.e., XML)

Maintainer's checks

- Review “pull request”
- Feature (or fix) documentation
- Test cases and validation of fix/feature (including vanilla cases)
- Code checking
  - **Breaking** or non-breaking change
  - Pull request **requirements**
  - **Structure** of code
- Maintainer -> Communicate with core developers
- Accept/reject pull request

Take away message

https://github.com/DualSPHysics/DualSPHysics

- DualSPHysics is an open-source solver with LGPL
- It is a "collaborative project"
- The developers and users pool is increasing continuously
- Our resources are limited
  - Community resources are (almost) unlimited
  - Code developers have application specific code improvements (computational or numerical)
  - Fork and contribute to the project

Contribute to the project through our repo