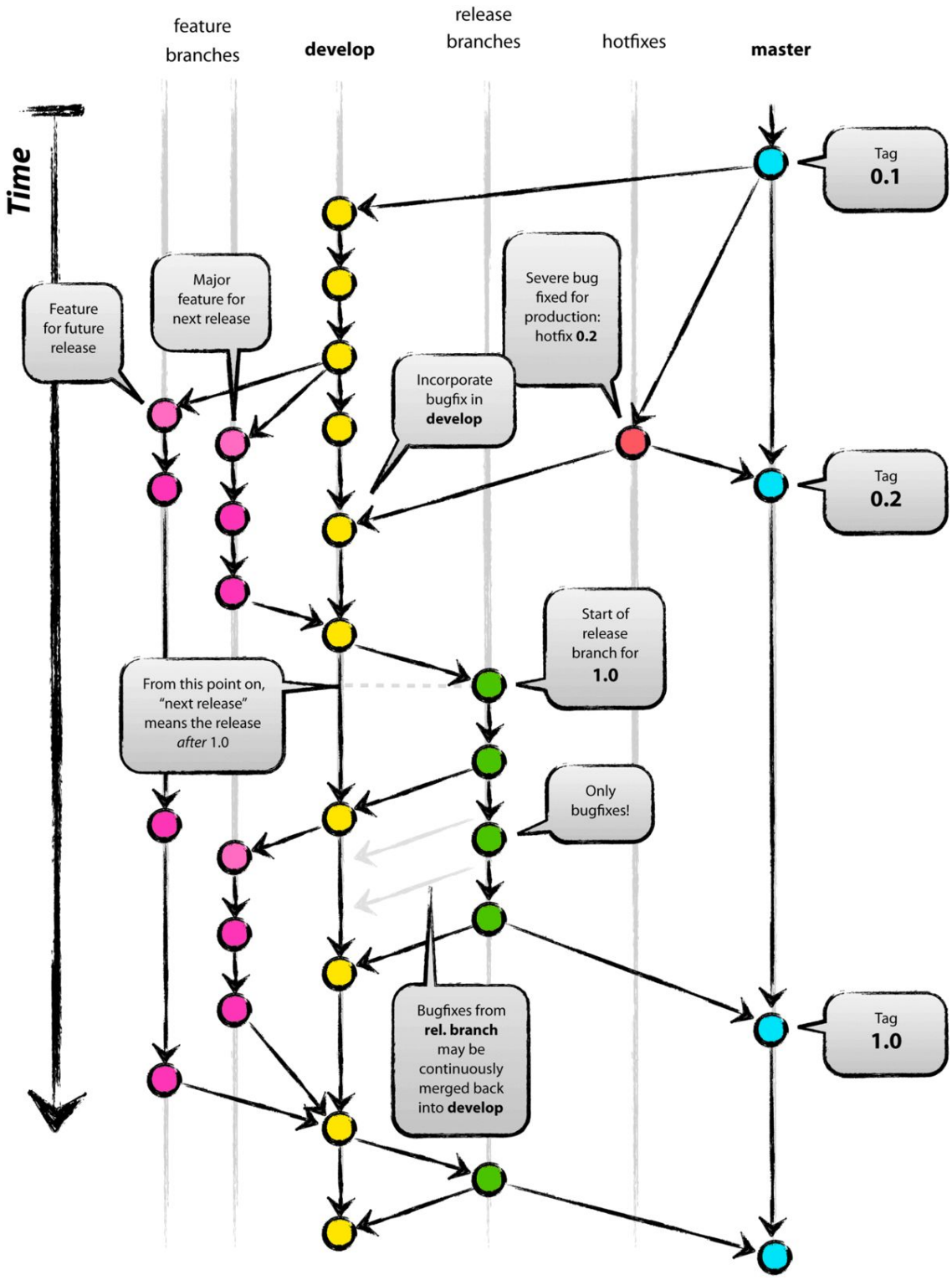
Git

Why Git

Git is the most commonly used version control system. Git tracks the changes you make to files, so you have a record of what has been done, and you can revert to specific versions should you ever need to. Git also makes collaboration easier, allowing changes by multiple people to all be merged into one source.
For detailed please visit runoob



Register on Github

Enter github and Register your account.

Create a new repo

Use current directory as your Git Repository

```
$ git init
```

Refer a directory to your Git Repository

```
 $ git init <repo>
```

Clone a repository from Github

```
$ git clone <repo>
eg: $ git clone https://github.com/XJTURobocupTurtorial/git-use
```

Commit your codes

```
$ git add .
$ git commit -m 'first commit'
```

By the way, you can write a .gitignore to ignore files that you don't want to commit such as experiment data, temp files.

To see your commits, use this command

```
$ git log
```

Basic Concepts

- Workspace
- Staging area
- Local repository

- Remote repository

Reset

Different from revert

git reset --hard is **DANGEROUS**

Branch management

Display branches:

```
$ git branch
```

Create <branchname>：

```
$ git branch <branchname>
```

Swirch：

```
$ git checkout <branchname>
```

(Create & Swirch)：

```
$ git checkout -b <branchname>
```

Updates data in master to the current branch：

```
$ git rebase master
```

Tasks

1. Register an account on Github and Install Git on your Computer

For Ubuntu, use apt install

```
$ sudo apt install git
```

2. Cloning

Clone the repo on https://github.com/XJTURobocupTurtorial/git-use

 # DO NOT DELETE this repo, tasks following next week will be baesd on this one.

3. Try to create your fisrt repository on Github

Pleased named it as "first-try"

4. Establish a local repository.

Create a folder on your local computer and enter it.

```
$ mkdir first-try && cd first-try
```

Enter this folder and initialize your repo.

```
$ git init
```

Create a python file in your repo

```
$ touch HelloGit.py
$ echo "print(\"Hello git\")" >> HelloGit.py
```

Commit your codes

```
$ git add .
$ git commit -m 'my first commit'
```

This operation automatically create a branch named "master".

5. Add Remote

Pleased named it as "first"

```
$ git remote add origin git@github.com:<account name>/first-try.git
$ git push -u origin master
```

Then your codes will be seen on Github.

6. Try again

Make changes on your codes.

```
$ echo "print(\"git so useful\")" > HelloGit.py
```

And commit it again

Then push your code to Github.

```
$ git push
```

7. Version control.

If you want to use a former version of your code that prints "Hello git"

```
$ git log
$ git checkout <commit>
eg: $ git checkout  bb8de8f2ccf3b9dec4d41d9423feab145e772448
    $ git checkout  HEAD~1 (last version)
$ python HelloGit.py
```

If you want to switch back and prints "git so useful"

```
$ git checkout master
$ python HelloGit.py
```

Be aware that all those commands below can achieve this purpose, but lead to different consequences, and some may be **DANGEROUS**:

```
$ git checkout <commit>
$ git reset [--soft/hard/mixed] <commit>
$ git revert <commit>
```

All these operations can be operated on single or several specefic files.

8. (Optional) Understand the basic concepts of Git

- Workspace
- Staging area
- Local repository
- Remote repository

9. (Optional) Preview Branch Operations

Figure out the meaning of HEAD, master, main?