Git, Trac, and Sage Development

Alyson Deines

CCR La Jolla

March 20, 2016

What is Git?

- Git is open source distributed version control software, i.e., source code management system for software development. In a nutshell, it allows multiple people to safely work on the same project.
- Git takes "snapshots" of files and if a file is unchanged, it just links to previous versions instead of taking more "snapshots".
- Distributed: not just kept in a remote database, each user's copy mirrors the entire development history.

What is GitHub?

- GitHub is a web-based Git repository hosting service
- graphical web interface for managing git projects
- free for up to 5 public repositories

How do you use Git?

- Have a central repository with a main branch called master. Sage has another branch which we will develop called develop.
- Developers clone the central repository and modify the branch called develop.
- In their own local copy, they edit and write new code.
- When satisfied, developers push their code, their local copy of develop, back to the central repository.

`https://training.github.com/kit/downloads/github-git-cheat-sheet.pdf`

What is Trac?

GitHub hosts projects, Git controls the flow of code, but how do we deal with bugs and reviewing what code actually goes into Sage? We use Trac.

Trac is an enhanced wiki and issue tracking system for software development projects.

`http://trac.sagemath.org/`

How to set up tools for developing

There's a bit of set up you must do to get all these tools to play well together.

- Get a Trac account. `http://trac.sagemath.org/`
- Generate and upload ssh key. `http://doc.sagemath.org/html/en/developer/walk_through.html`
- Install and set up Git. `http://doc.sagemath.org/html/en/developer/git_setup.html#chapter-git-setup`
- Get and setup a Github account. `https://help.github.com/articles/set-up-git/`
- Finally, install git-trac-command. `http://doc.sagemath.org/html/en/developer/git_trac.html#chapter-git-trac`

How to set up Sage for developing

- Get source code:

```
$ git clone git://github.com/sagemath/sage.git
Cloning into 'sage'...
[...]
Checking connectivity...  done.
```

- Switch to the development branch:

```
$ cd sage
[sage]$ git checkout develop
[sage]$ make
```

- Now go do something else for quite a while!

An example

The instructions for developing can be found: `http://doc.sagemath.org/html/en/developer/walk_through.html`

- Make sure you are using an up-to-date development branch

```
[sage]$ git checkout develop
[sage]$ ./sage -upgrade develop
```

- Pull the remote ticket

```
[sage]$ git checkout develop -b ticket/12345
[sage]$ git trac pull 12345
[sage]$ git diff develop # check changes against the
latest develop branch
```

Reviewing a Patch

After looking at the source code, if it looks good, re-build Sage.

```
[sage]$ ./sage -b
```

If Sage builds cleaning, try out the code and play with it. Try to break it. Make sure that there are enough test cases to check every line of code. Now run the doc tests.

```
[sage]$ ./sage -t src/sage/path_to_code
```

Finally, build the documentation and make sure it is correctly formatted.

```
[sage]$ ./sage --docbuild reference html
```

Editing Sage Code

When you want to write your own Sage code, first make a new branch. You can make a ticket on Trac and switch to that branch or

```
[sage]$ git branch new_branch_name
[sage]$ git checkout new_branch_name
```

When you want to make changes, just commit them.

```
[sage]$ git commit -m "Short description of commit."
```

Finally, when you want to upload it to Trac, if it is already associated to a ticket

```
[sage]$ git trac push
```

or

```
[sage]$ git trac push TICKET_NUMBER
```

Sage conventions

When reviewing or writing your own code, make sure it follows proper python conventions and is documented with doctests. `http://doc.sagemath.org/html/en/developer/coding_basics.html`

Developing in the Cloud

You can totally develop Sage in the SageCloud!
`https://github.com/sagemathinc/smc/wiki/SageMath-Development-on-SageMathCloud`

The steps are similar, but you never go directly through GitHub, as you can just copy a pre-compiled development version.

Highly recommended for windows machines.

Downside: must be connected to the internet.