

Git IzPack workflow

Git IzPack workflow

Branches and tags conventions

For the Codehaus repository:

- `master` refers to the main development branch (equivalent to a Subversion trunk)
- `a.b` refers to a branch for the `a.b` releases (e.g., `5.1` is the parent branch to prepare the `5.1.0`, `5.1.1` and so on releases)
- `va.b.c` refers to a release tag (e.g., `v5.0.0`)

Each developer is free to name his/her topic branches as they want as long as the name remains meaningful. We strongly advise to use the related JIRA issue name if possible (e.g., name your branch `IZPACK-123` if it refers to `IZPACK-123`).

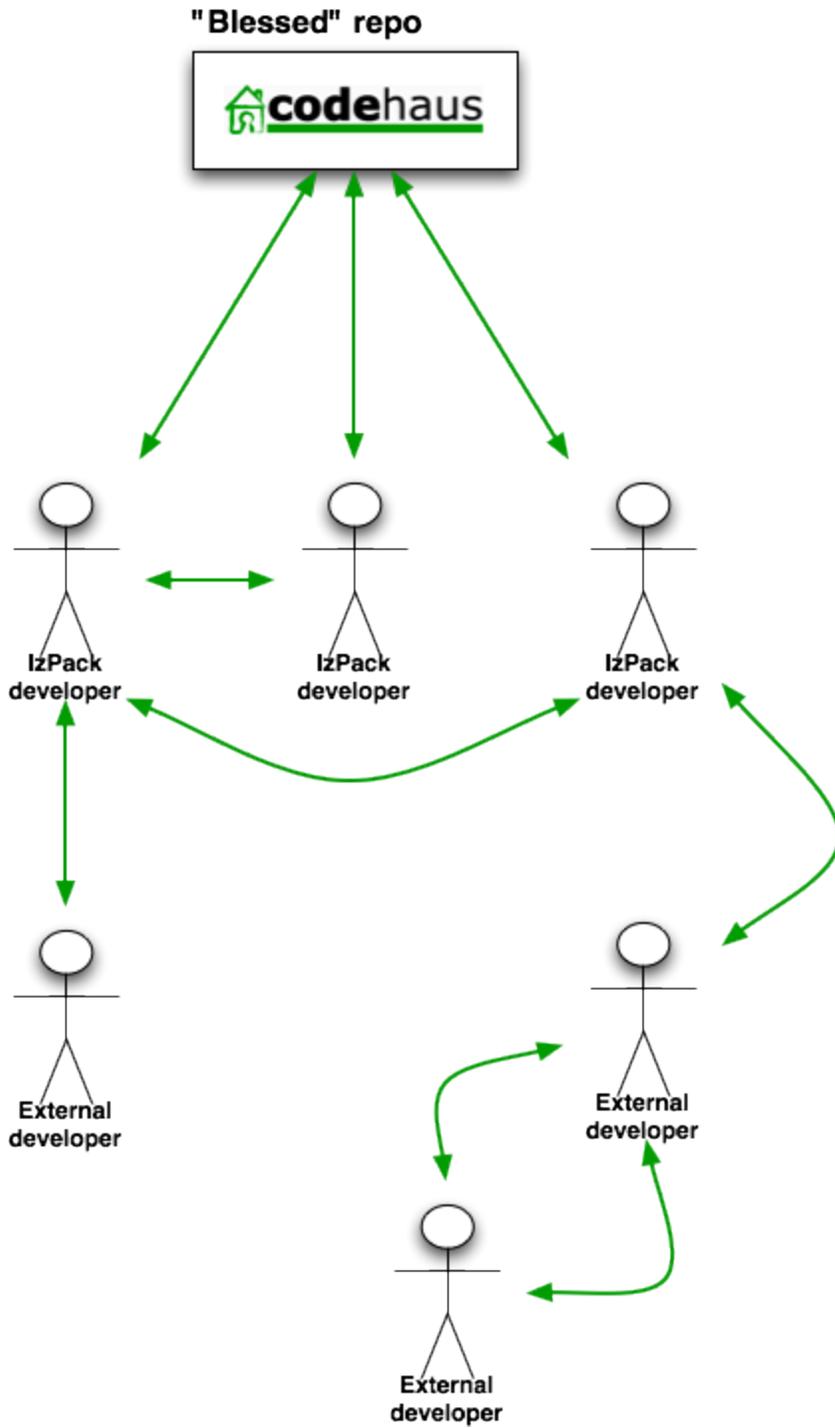
Peer-to-peer interactions

Given that Git is a distributed system, anyone can clone a Git repository and maintain changes locally. This is fundamentally different compared to a centralized system like Subversion where only committers can put their changes under version control.

We distinguish 3 types of Git repositories:

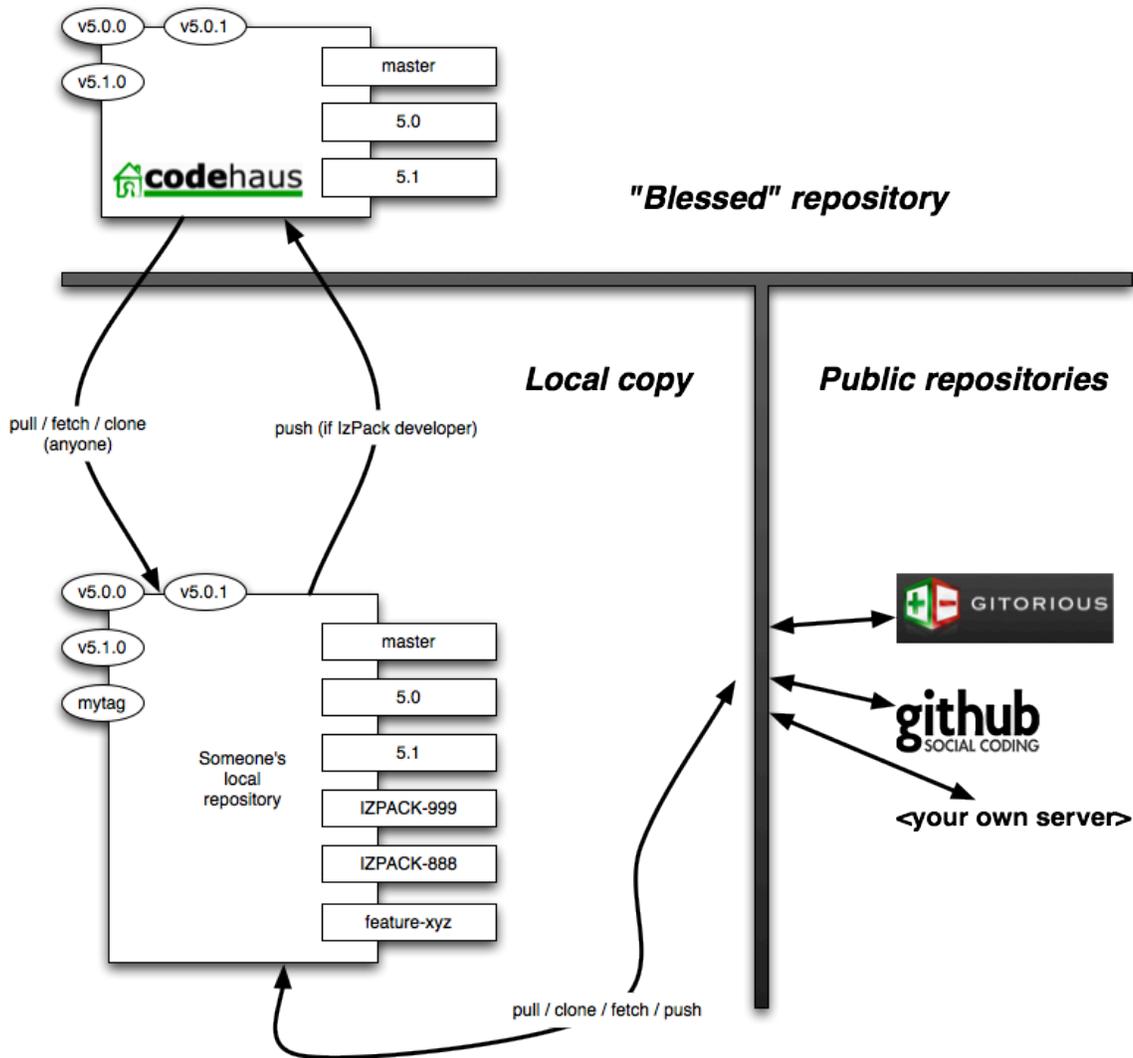
- the blessed repository where only IzPack developers have write access (like with any centralized system)
- IzPack developers repository which allows them to collaborate and publish changes before they can make it to the blessed repository
- external people repositories that can be used to maintain customized versions against upstream changes, and that can also be used to offer new contributions to the IzPack developers that can easily pull them from their repositories.

The following figure shows how the peer-to-peer interactions happen in the IzPack Git workflow:



As you can see, a distributed system like Git puts various types of people of the IzPack community on equal foot.

Repositories workflow



Anyone can get and maintain his/her very own copy of the IzPack source code with the whole history.

The blessed repository must only feature the development branch (**master**), the stable release branches and the release tags (an exception is made only for those recovered from the CVS to Subversion then Subversion to Git conversions).

IzPack developers willing to share work in progress and experiments should have a public repository to push their own branches to. [GitHub](#) and [Gitorious](#) are excellent public repository hosting platforms.

i Rule of thumb for IzPack developers

Do not directly work on **master** or stable version branches that match those of the blessed repository.

Instead:

1. create branches for each task
2. polish them
3. publish them to your own repository if you need to collaborate with somebody else
4. once the task is complete, merge back to the branches that match the blessed repository
5. pull/push to the blessed repository.

Tip: do not hesitate to have a staging / integration branch between your master and feature branches, as this can sometimes be helpful in complex merge situations.

What you can do... or not

