Workflow Git

Equipe Nano
Aldísio Medeiros
Átila Camurça
Leandro Souza
Lucas Silva
Manuel Paulo
Péricles Henrique
Murilo Barata

17/02/2015

1 Tarefas (Issues)

Veja as tarefas disponíveis em: https://github.com/ifce-gp-20151/saa/issues

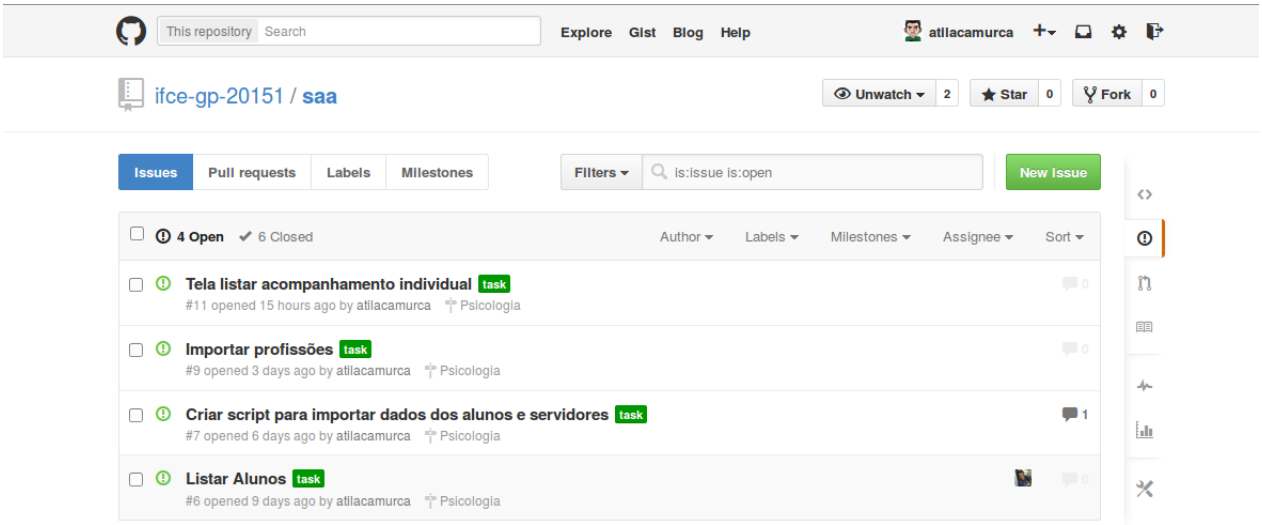

Figure 1: Página de issues

Clique na tarefa que deseja resolver e indique que você irá resolver.

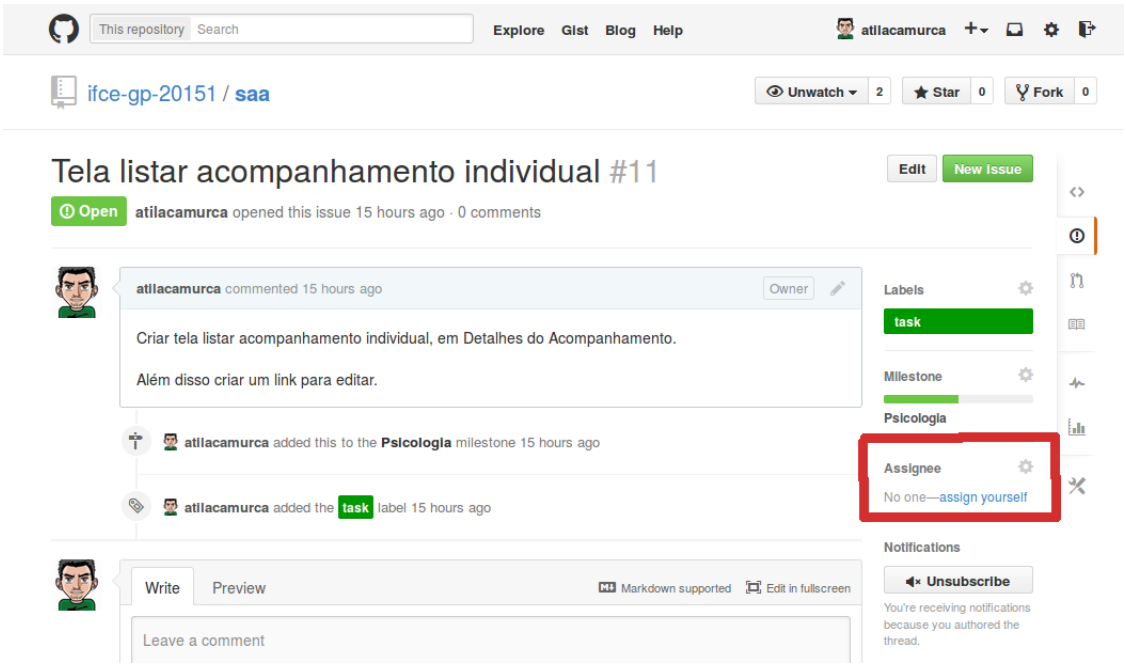

Figure 2: Resolver issue

Verifique o número da issue. Ela vai ser o nome do seu *branch* local.

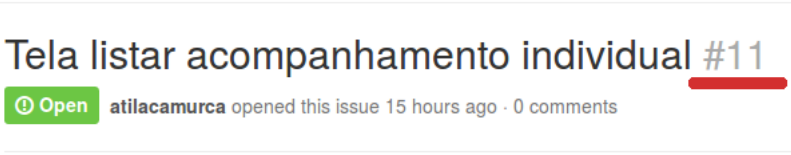

Figure 3: Número da issue

2 Branch no repositório local

Entre em seu repositório local pelo terminal e crie o branch de acordo com o número da issue.

```
git checkout master
git fetch origin
git merge origin/master
git checkout -b issue-11
```

Começe a trabalhar normalmente.
Verifique a pasta `./docs/manual/workflow-zend`.

3 Finalizar tarefa

Ao finalizar a tarefa você deve fazer *commit* de suas alterações. Para facilitar use o `gitg`. Para utilizar basta executar no seu repositório local `gitg`.

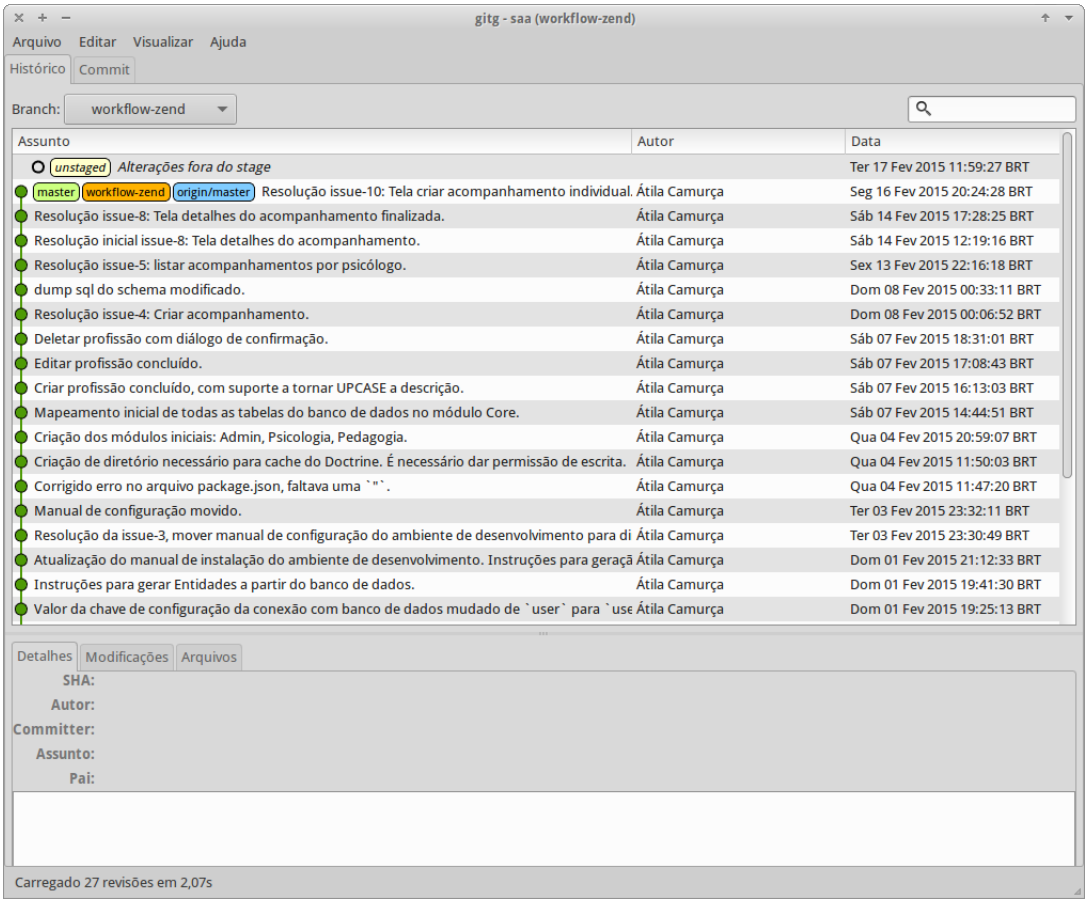

Figure 4: gitg

Verifique se o *branch* está correto:

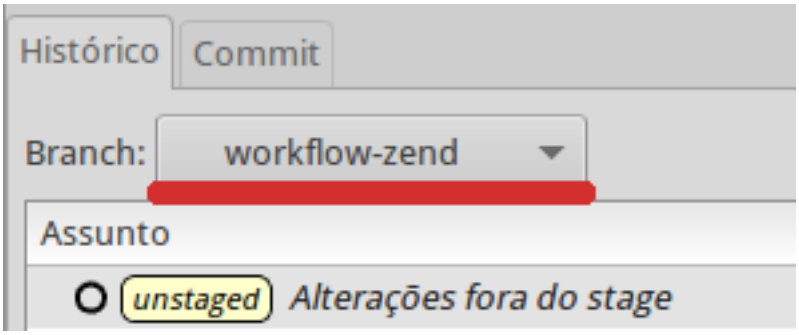

Figure 5: Branch

Obs.: no exemplo está com outro nome, mas no caso deveria ter o valor **issue-11**.

3.1 Commit das alterações

Para fazer o *commit* clique na aba Commit. Coloque os arquivos necessários de *Unstaged* para *Staged*. Por fim escreva a mensagem de *commit*, e clique no botão **Commit**:

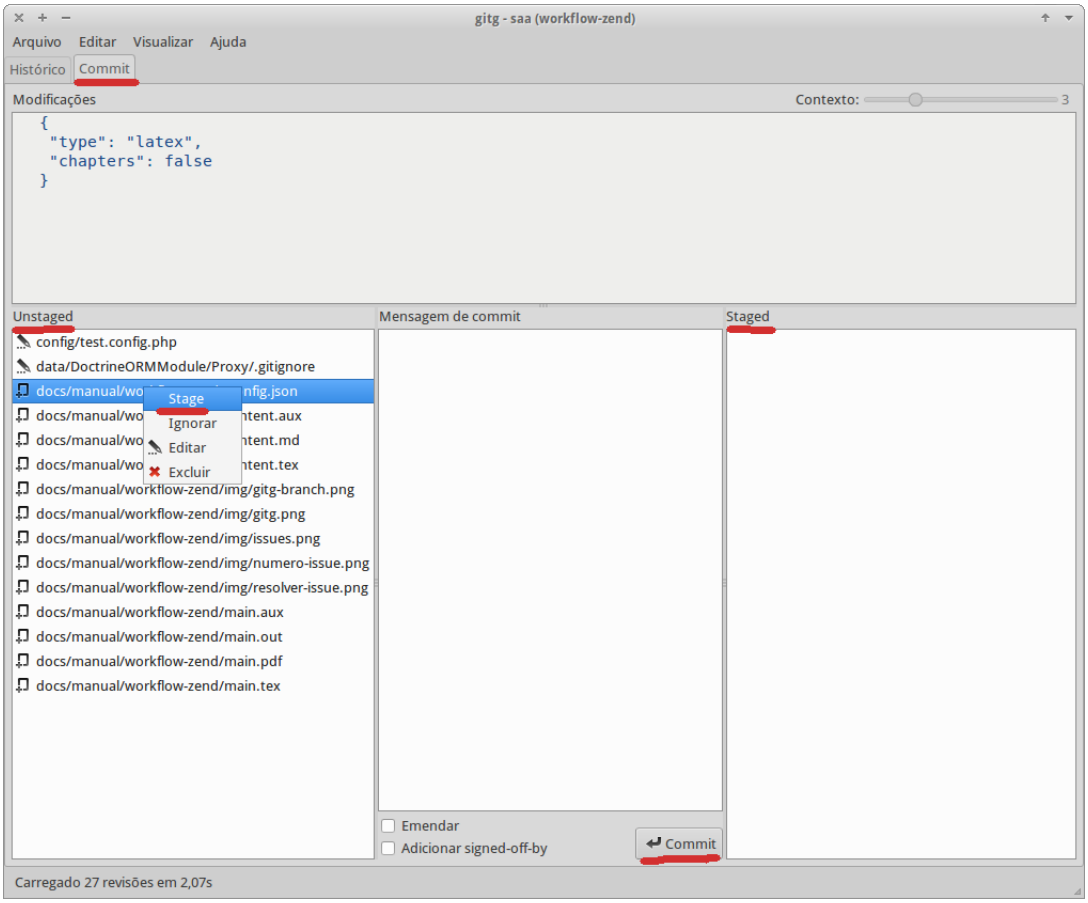

Figure 6: Commit

Feche o `gitg`.

4 Atualizar com o código Remoto

Antes de subir as alterações é necessário atualizar a base com o repositório remoto. Para isso faça:

```
git checkout master
git fetch origin
git merge origin/master
```

Logo depois faça o *merge* com o seu *branch*:

```
git checkout master
git merge issue-11
```

A saída será algo como:

```
Updating f42c576..3a0874c
Fast-forward
 index.html | 2 ++
 1 file changed, 2 insertions(+)
```

Verifique se `Fast-foward` aparece na mensagem. Isso quer dizer: "Ok, tudo certo para subir o código!".

```
git push -u origin master
```

4.1 Conflitos!

Podem acontecer conflitos ao fazer o *merge*, ou seja, seu código possui linhas de código modificadas no mesmo local que as do repositório remoto. Quando isso acontecer ao fazer o *merge* a saída será algo como:

```
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
```

A frase final indica:

> Merge automático falhou; resolva os conflitos e então faça commit do resultado.

Para resolver os conflitos use o `meld`. Instalação:

```
sudo apt-get install meld
```

Em seguida dento do seu repositório local execute:

```
git mergetool
```

Uma mensagem semelhante a essa irá aparecer, apenas dê `Enter`.

```
This message is displayed because 'merge.tool' is not configured.
See 'git mergetool --tool-help' or 'git help config' for more details.
'git mergetool' will now attempt to use one of the following tools:
opendiff meld tortoisemerge bc3 codecompare vimdiff emerge
Merging:
index.html
```

```
Normal merge conflict for 'index.html':
  {local}: modified file
  {remote}: modified file
Hit return to start merge resolution tool (meld):
```

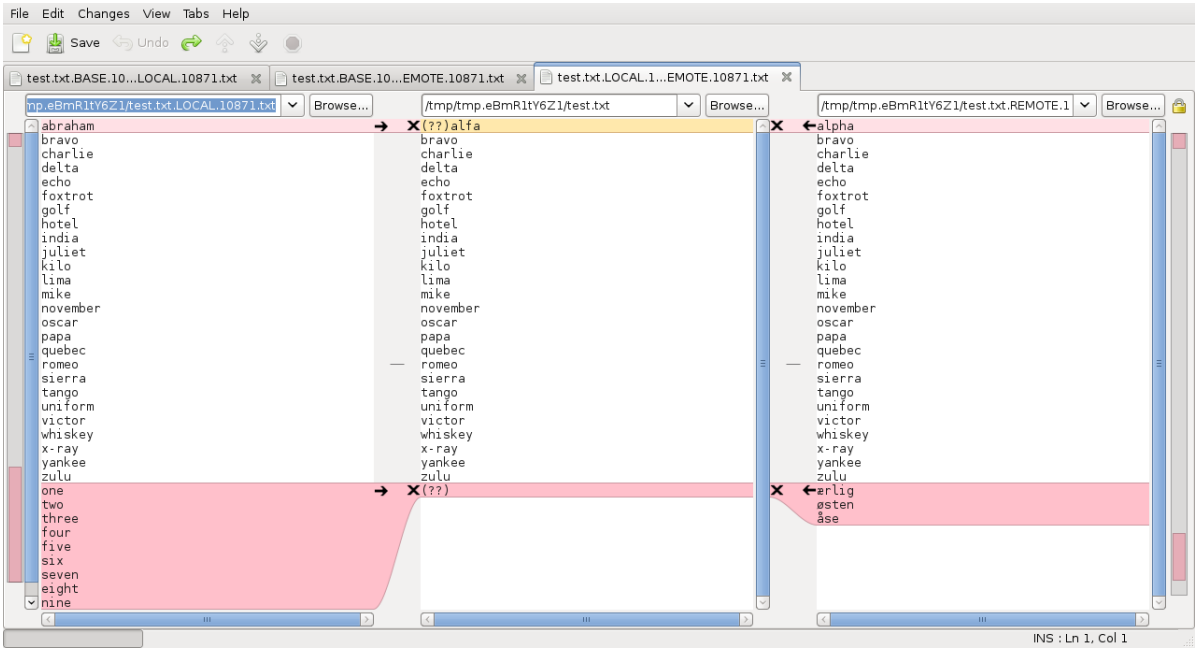

Figure 7: Meld

Esse é o chamado *Three way git merging*, ou merge de três vias. O arquivo a esquerda é seu arquivo local, o do meio é o arquivo que é ancestral, e o da direita o arquivo remoto.

Coloque todas as alterações para o arquivo do meio, clicando na `seta`. Clique no `X` para apagar. Segure `Ctrl` para colocar o código abaixo ou acima do indicado, assim é possível juntar algo do seu *branch* com o do *branch* remoto.

Caso tenha acontecido conflitos em mais de um arquivo, ao fechar o `meld` basta continuar dando `Enter`.