Issues

Issues são um meio de comunicação dentro de um repositório. Toda decisão importante deve ser realizada em um issue correspondente, devidamente marcado com **um ou mais** labels e **um** milestone.

Um issue é o mesmo que um post num forum: quando alguém quer tirar uma dúvida ou decidir o que fazer com o projeto, faz-se um issue correspondente ao assunto.

Labels

Labels são "marcadores" para issues. Cada issue deve ter pelo menos um label, mas pode ter mais de um. Os labels são utilizados para facilitar na hora da procura por issues, como também para indicar de que se trata cada issue sem ter que abrir um por um.

É bom também criar labels de prioridade, indicando qual issue tem mais urgencia.

Para a Becosystems, há 3 labels obrigatórios que devem ser criados, mas só usados pelo professor:

- Task(amarelo)
- Urgent(vermelho)
- Late(azul)

Esses labels são utilizados em issues feitos pelo professor para indicar o nível de urgência da tarefa dada.

Milestones

Milestone é um "objetivo". Você marca um determinado ponto no desenvolvimento com uma milestone, que deve ser criada previamente, como uma expectativa do progresso na determinada data. Pode-se dizer que é um "prazo" dado ao desenvolvedor, ou por si mesmo ou por seu empregador.

Os nomes de milestones são nomes de versões. As versões MAJOR, dadas por números inteiros(antes do ponto), são versões com mudanças significativas. As versões MINOR, dadas por números após o ponto, indicam mudanças pequenas no projeto. Há ainda as versões FIX, indicadas por um número após um segundo ponto, indicam correções de erros não-detectados.

Uma milestone deve estar no seguinte formato, no mínimo:

- v1.0 (MAJOR.MINOR)
- v1.0.1 (MAJOR.MINOR.BUGFIX)

Releases e Tags

Ao cumprir uma milestone, você dá uma tag no commit final daquela milestone, indicando o cumprimento do objetivo.

O nome da tag deve ser o mesmo do nome da milestone.

Também há a forma mais organizada:

- v1.0.180418.172630 (MAJOR.MINOR.AAMMDD.HHMMSS)
- v1.0.1.180418.172720 (MAJOR.MINOR.BUGFIX.AAMMDD.HHMMSS)

No caso também dando a última data de compilação antes do commit final, informação escrita no arquivo VERSION, gerado utilizando o make para comilar, com os seguintes parâmetros:

```
make exemplo.x MAJOR=1 MINOR=0 DEBUG=0 FORTIFY=1
```

Caso queira ocultar avisos sobre scanf, utilize `FORTIFY=0`

Você pode também usar o comando `checkout` numa tag(veja abaixo).

Branches

Branches são ramificações no projeto. O ramo principal, "master", é a "versão comercial" do projeto, a versão estável mais recente. Ao desenvolver código, você não deve mandar novas mudanças ao master. Deve-se criar um ramo "develop" para trabalhar, e assim que constatada a estabilidade na versão atual do develop, é feito um merge, que passa todos os commits do ramo develop ao ramo master.

Para criar um branch, utiliza-se um dos seguintes comandos:

- `git checkout -b nomedoramo` para criar e trocar para o ramo, também abreviado `git co -b nomedoramo` no hydra.
- `git branch nomedoramo` para criar sem trocar.

Para enviar o novo branch criado localmente ao repositório remoto(do github), utiliza-se o comando `git push -u`.

Ver também: Dicionário de Git por Victor Sena Molero.