Curso de Introdução ao Git

USPGameDev

15 de agosto de 2012

Introdução

O que é Controle de Versão?

Softwares de controle de versão é um sistema que grava todas as modificações em um conjunto de arquivos, criando um histórico deles, criando um ambiente onde o usuário possa recriar versões antigas dos arquivos.
Eles são separados em 3 conjuntos:

- Controle de versão local. (Ex: rcs)
- Controle de versão centralizado. (Ex: SVN, CVS)
- Controle de versão distribuído. (Ex: Git, Mercurial)

Fundamentos

Snapshots, não diferenças

O Git, diferente de outros controles de versões que salvam os arquivos e as modificações entre os commits, o Git salva, a cada commit, o estado atual de cada arquivo.

Trabalho Local

Quase todo trabalho do Git é local. Isso significa que ele não precisa de comunicação constante com algum servidor remoto, como é normal em outros CVs. Essa liberdade garante que os usuários não fiquem dependentes de internet ou intranet para trabalhar.

Integridade

Todo arquivo que está sob os cuidados do Git recebe um checksum (soma de verificação). Olhando para esse checksum, o Git sabe se algum arquivo ou diretório foi modificado, tornando impossível modificações obscuras.

Adição de dados (geralmente...)

Quase todas as ações do Git tem como finalidade adicionar dados na database do projeto. Depois que algo foi incorporado em um repositório, é quase impossível tirá-lo de lá.

3 Estados

Todo e qualquer arquivos que está no Git tem 3 possíveis estados:

- Committed - Significa que as modanças feitas no arquivo estão salvas na database do Git.
- Modificado - Significa que o arquivo foi modificado.
- Staged - Significa que as mudanças atuais no arquivos seram inclusas no próximo commit.

3 Estados

Esses 3 estados fazem com que um repositório local do Git seja separado em 3 seções:

- Git directory - Onde todos os arquivos vitais e a database do Git estão.
- Working directory - É uma instância de uma versão do seu projeto.
- Staging area - Um arquivo que contêm todas as modificações que entrarão no próximo commit.

3 Estados

O processo de trabalho com o Git pode ser resumido a:

1. Você modifica os arquivos no seu working directory.
2. Você adiciona as modificações na staging area.
3. Você commita, aicionando permanentemente as modificações no seu Git directory.

Instalação

Para instalar o Git, vamos seguir o tutorial do GitHub =D

Configuração

O Git permite uma variedade de configurações. Podemos modificá-las usando o comando:

```
$ git config
```

Configuração

O Git permite uma variedade de configurações. Podemos modificá-las usando o comando:

```
$ git config
```

Duas configurações que temos que modificar depois de instalar o Git são:

```
$ git config --global user.name "José da Silva"
$ git config --global user.email josé@silva.com.br
```

Temos que mudar essas configurações pois o Git usam elas para gerar as mensagens de commit.

Podemos pegar um projeto que usa o Git de duas maneiras distintas:

- Portanto um projeto existente para o Git, ou
- Clonando um projeto que já usa o Git.

Para iniciar o Git num projeto já existente basta ir na pasta raiz do projeto e usar o comando:

```
$ git init
```

Para iniciar o Git num projeto já existente basta ir na pasta raiz do projeto e usar o comando:

```
$ git init
```

Para clonar um projeto que já usa o Git usamos o comando:

```
$ git clone <url-do-repositorio> <nome-da-pasta>
```

Esse comando cria uma pasta com o nome do projeto no local em que foi chamado. Podemos passar um nome para a pasta que vai seer criada.

a