Controle de Versões com o Git

#gitday - 10/09/2011

Slides por Tiago "Myhro" Ilieve

O que é versionamento de código?

- É manter diversas versões dos seus arquivos, de forma que você possa acessar o conteúdo dos mesmos exatamente como era em qualquer ponto do tempo.

O que é versionamento de código?

- É manter diversas versões dos seus arquivos, de forma que você possa acessar o conteúdo dos mesmos exatamente como era em qualquer ponto do tempo.
- Você pode adicionar ou remover partes e até mesmo os próprios arquivos, sem ter de se preocupar com o que estas mudanças poderão acarretar.

O que é versionamento de código?

- É manter diversas versões dos seus arquivos, de forma que você possa acessar o conteúdo dos mesmos exatamente como era em qualquer ponto do tempo.
- Você pode adicionar ou remover partes e até mesmo os próprios arquivos, sem ter de se preocupar com o que estas mudanças poderão acarretar.
- Algo parecido com uma "máquina do tempo dos CTRL+Z"

Por que o Git?

- É rápido, muito rápido!

Por que o Git?

- É rápido, muito rápido!
- Tudo é feito localmente e se necessário pode-se adotar tanto o modelo centralizado quanto distribuído para compartilhamento de código.

Por que o Git?

- É rápido, muito rápido!
- Tudo é feito localmente e se necessário pode-se adotar tanto o modelo centralizado quanto distribuído para compartilhamento de código.
- Neste caso, cada pessoa com acesso ao repositório tem uma cópia local e completa do mesmo.

Por que o Git?

- É rápido, muito rápido!
- Tudo é feito localmente e se necessário pode-se adotar tanto o modelo centralizado quanto distribuído para compartilhamento de código.
- Neste caso, cada pessoa com acesso ao repositório tem uma cópia local e completa do mesmo.
- Praticamente todas as mudanças podem ser desfeitas (a possibilidade de se perder algo é ínfima).

Por que o Git?

- É largamente utilizado, estável e seguro.

Por que o Git?

- É largamente utilizado, estável e seguro.
- Projetos imensos, como o próprio Kernel do Linux (14 mi de linhas), utilizam o Git para versionar seus códigos.

Por que o Git?

- É largamente utilizado, estável e seguro.
- Projetos imensos, como o próprio Kernel do Linux (14 mi de linhas), utilizam o Git para versionar seus códigos.
- Cresceu em popularidade, grande parte graças ao GitHub, e hoje é possivelmente o VCS mais usado no mundo.

Por que o Git?

- É largamente utilizado, estável e seguro.
- Projetos imensos, como o próprio Kernel do Linux (14 mi de linhas), utilizam o Git para versionar seus códigos.
- Cresceu em popularidade, grande parte graças ao GitHub, e hoje é possivelmente o VCS mais usado no mundo.
- Não é muito complicado, basta entender bem seus conceitos e se acostumar com a sintaxe dos seus comandos.

História do Git

- É um projeto recente, tendo sido criado em 2005 por Linus Torvalds e se popularizado a partir de 2008.

História do Git

- É um projeto recente, tendo sido criado em 2005 por Linus Torvalds e se popularizado a partir de 2008.
- Não era fácil de ser utilizado no começo, mas sua usabilidade evoluiu muito graças a Junio Hamano, atual mantenedor do Git.

História do Git

- É um projeto recente, tendo sido criado em 2005 por Linus Torvalds e se popularizado a partir de 2008.
- Não era fácil de ser utilizado no começo, mas sua usabilidade evoluiu muito graças a Junio Hamano, atual mantenedor do Git.
- Foi criado após os criadores do BitKeeper revogarem a licença qual permitia sua utilização gratuita.

História do Git

- É um projeto recente, tendo sido criado em 2005 por Linus Torvalds e se popularizado a partir de 2008.
- Não era fácil de ser utilizado no começo, mas sua usabilidade evoluiu muito graças a Junio Hamano, atual mantenedor do Git.
- Foi criado após os criadores do BitKeeper revogarem a licença qual permitia sua utilização gratuita.
- O GitHub foi peça chave para sua popularização e hoje hospeda quase três milhões de repositórios de quase um milhão de desenvolvedores.

GitHub - Social Coding

- É como um "Facebook para programadores". Você pode seguir outras pessoas, acompanhar suas atividades e até mesmo colaborar com seus projetos.

GitHub - Social Coding

- É como um "Facebook para programadores". Você pode seguir outras pessoas, acompanhar suas atividades e até mesmo colaborar com seus projetos.
- Você pode criar infinitos repositórios (até 300MB de disco) gratuitos, desde que seus projetos sejam Open Source.

GitHub - Social Coding

- É como um "Facebook para programadores". Você pode seguir outras pessoas, acompanhar suas atividades e até mesmo colaborar com seus projetos.
- Você pode criar infinitos repositórios (até 300MB de disco) gratuitos, desde que seus projetos sejam Open Source.
- Caso tenha necessidade de utilizar repositórios privados, há planos de assinaturas mensais.

GitHub - Social Coding

- É como um "Facebook para programadores". Você pode seguir outras pessoas, acompanhar suas atividades e até mesmo colaborar com seus projetos.
- Você pode criar infinitos repositórios (até 300MB de disco) gratuitos, desde que seus projetos sejam Open Source.
- Caso tenha necessidade de utilizar repositórios privados, há planos de assinaturas mensais.
- O "GitHub Firewall" está disponível para empresas que não querem ou não podem armazenar seus projetos externamente.

Conceitos básicos do Git

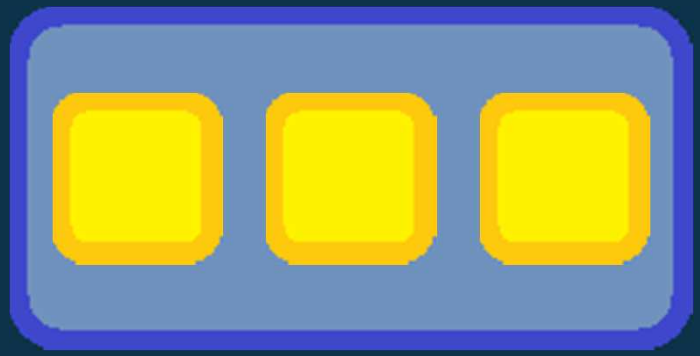
- Repositório
- Índice
- Área de Trabalho

Comando: git add <arquivo>

- Repositório
- Índice
- Área de Trabalho

Comando: git commit -m "Mensagem"

- Repositório



- Índice

- Área de Trabalho

Comando*: git rm --cached <arquivo>

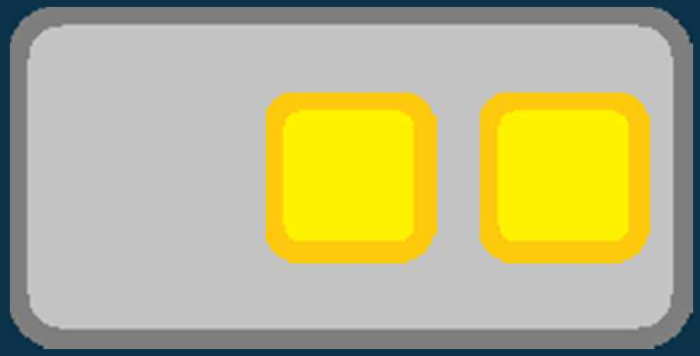
- Repositório
- Índice
- Área de Trabalho

* Se ainda não há nenhum commit no repositório.

Comando*: git reset HEAD <arquivo>

- Repositório

- Índice



- Área de Trabalho

* Retorna ao estado anterior sem descartar alterações.

Comando: git checkout -- arquivo

- Repositório
- Índice
- Área de Trabalho

Comando: git checkout -- arquivo

- Repositório
- Índice
- Área de Trabalho

Instalação do Git

- O Git não foi desenvolvido para Windows, mas roda nele perfeitamente.

Instalação do Git

- O Git não foi desenvolvido para Windows, mas roda nele perfeitamente.
- Em distribuições Linux basta instalá-lo com o gerenciador de pacotes, procurando por "git" ou "git-core".

Instalação do Git

- O Git não foi desenvolvido para Windows, mas roda nele perfeitamente.
- Em distribuições Linux basta instalá-lo com o gerenciador de pacotes, procurando por "git" ou "git-core".
- Antigamente, "git" respondia pelo "GNU Interactive Tools".

Instalação do Git

- O Git não foi desenvolvido para Windows, mas roda nele perfeitamente.
- Em distribuições Linux basta instalá-lo com o gerenciador de pacotes, procurando por "git" ou "git-core".
- Antigamente, "git" respondia pelo "GNU Interactive Tools".
- No Ubuntu 10.10 ou mais recente:

```
apt-get install git
```

Instalação do Git

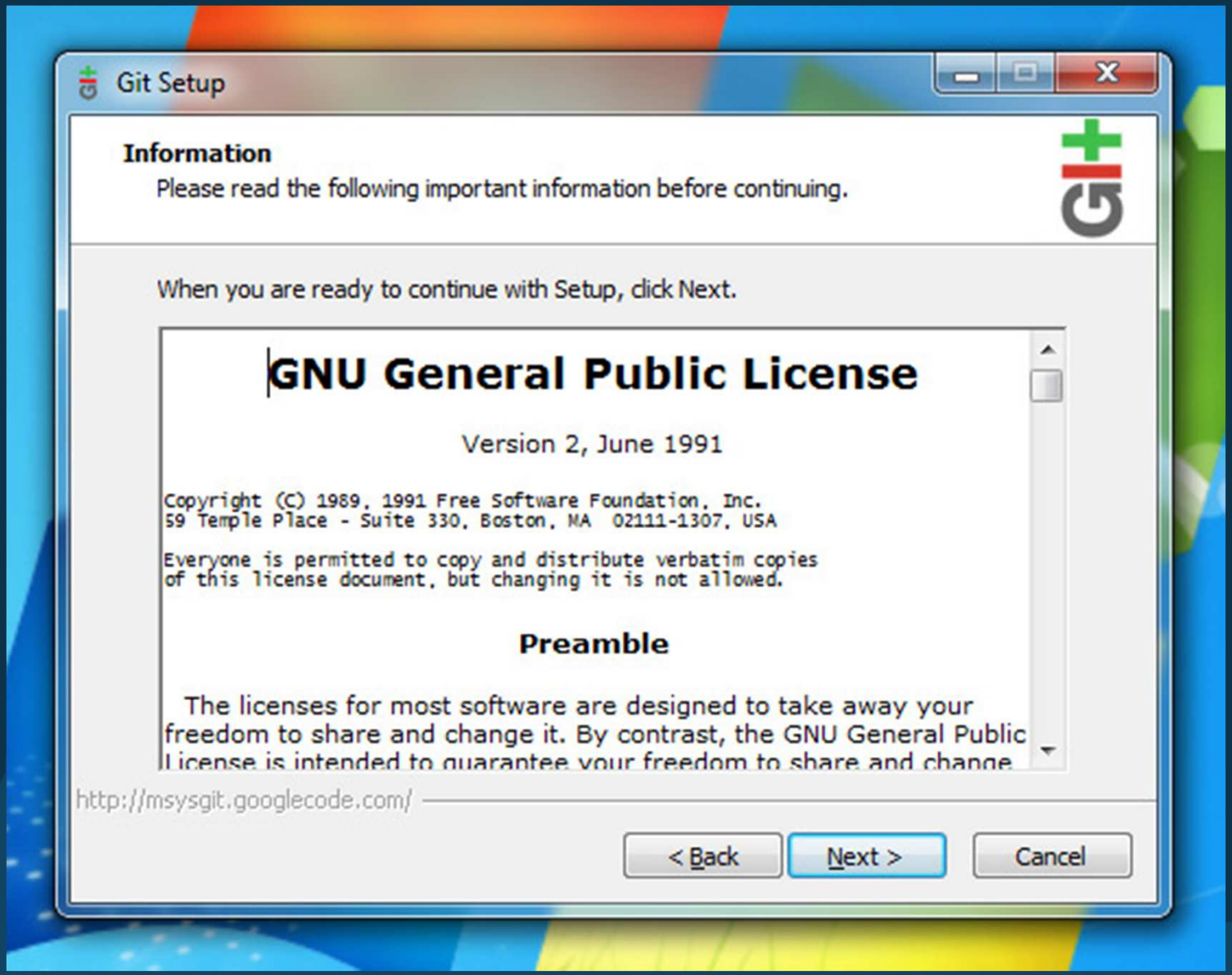
- O Git não foi desenvolvido para Windows, mas roda nele perfeitamente.
- Em distribuições Linux basta instalá-lo com o gerenciador de pacotes, procurando por "git" ou "git-core".
- Antigamente, "git" respondia pelo "GNU Interactive Tools".
- No Ubuntu 10.10 ou mais recente:

```
apt-get install git
```

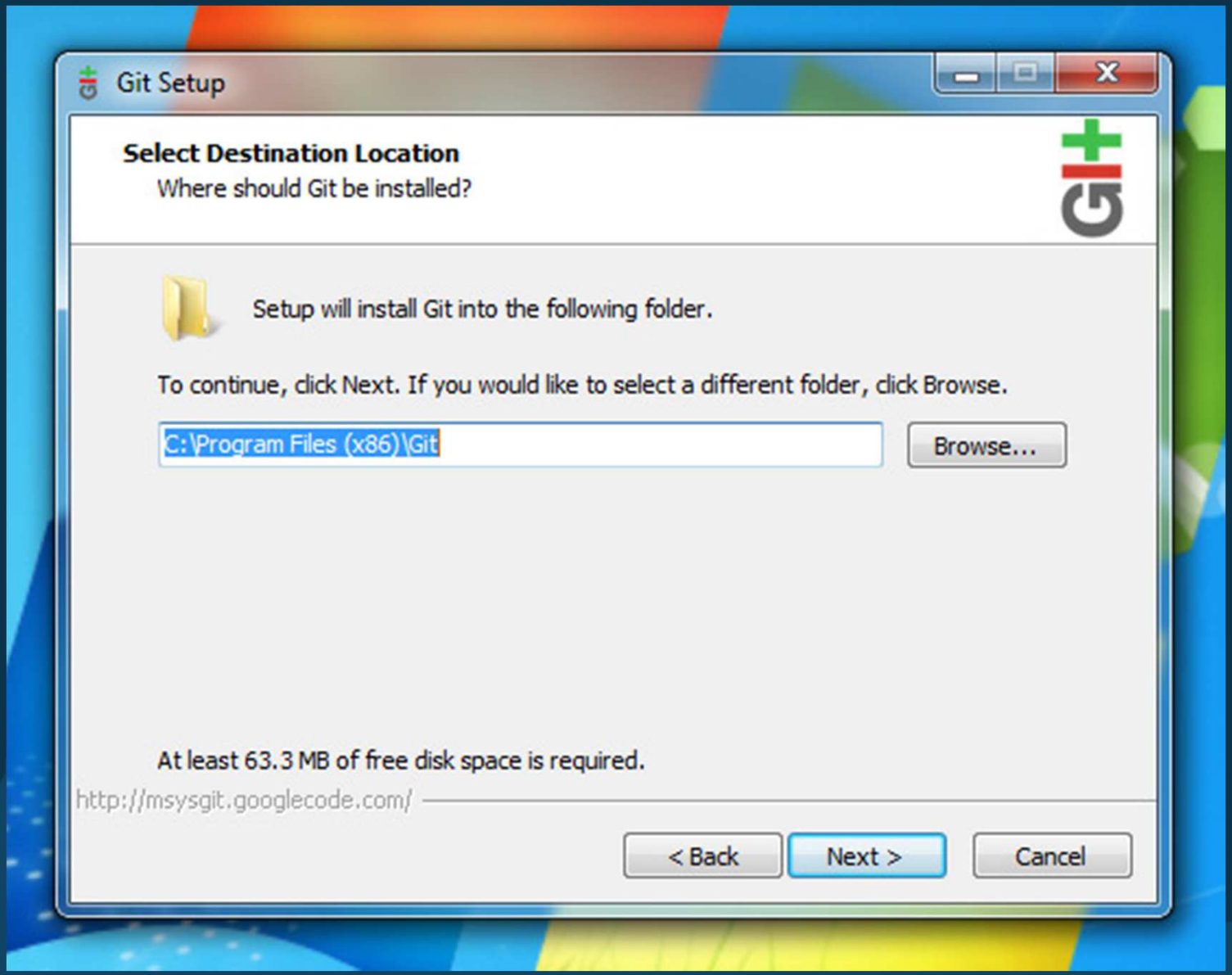
- Aqui utilizaremos como base o tutorial de instalação do Git para Windows feito pelo GitHub.

Instalação do Git

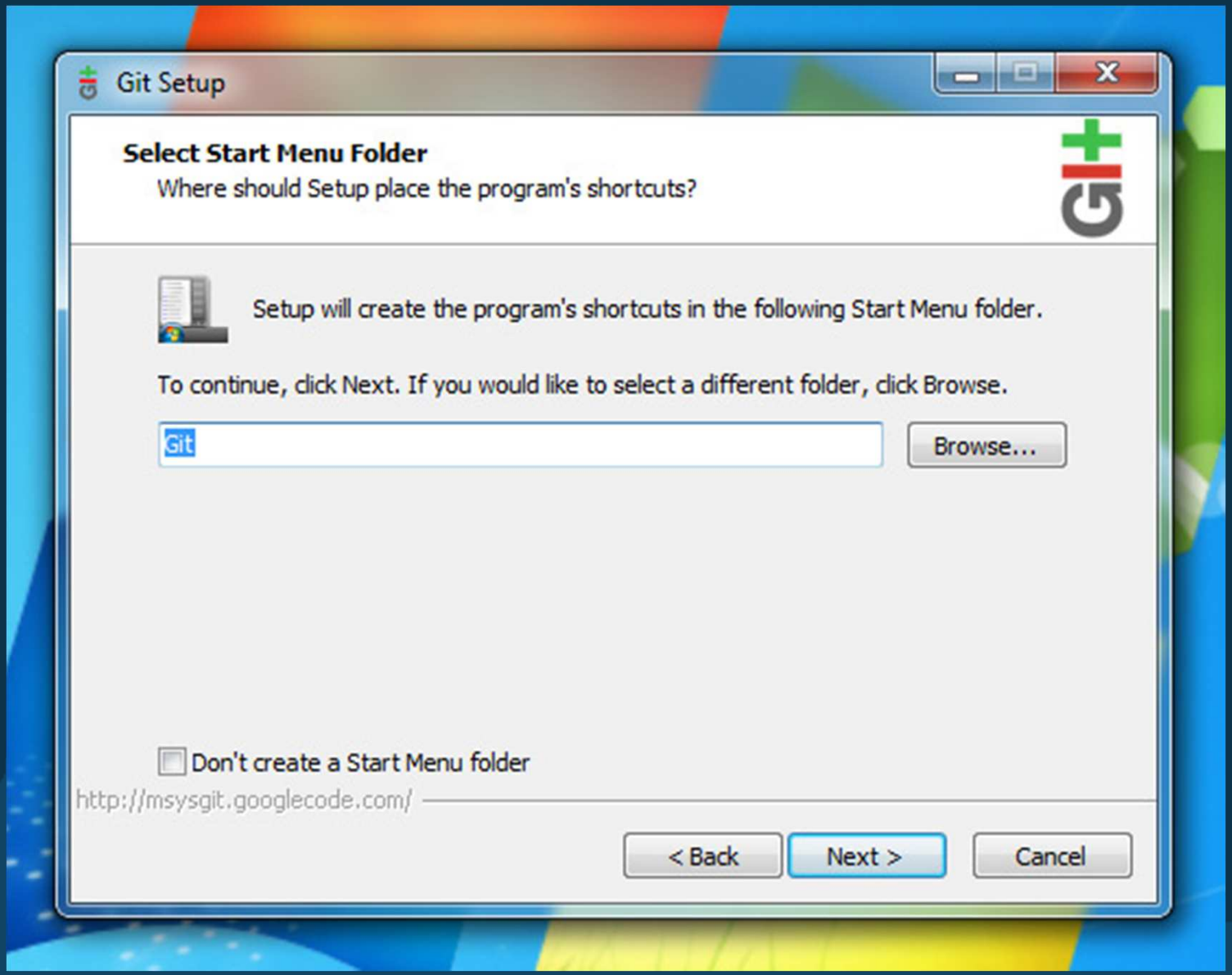
Instalação do Git



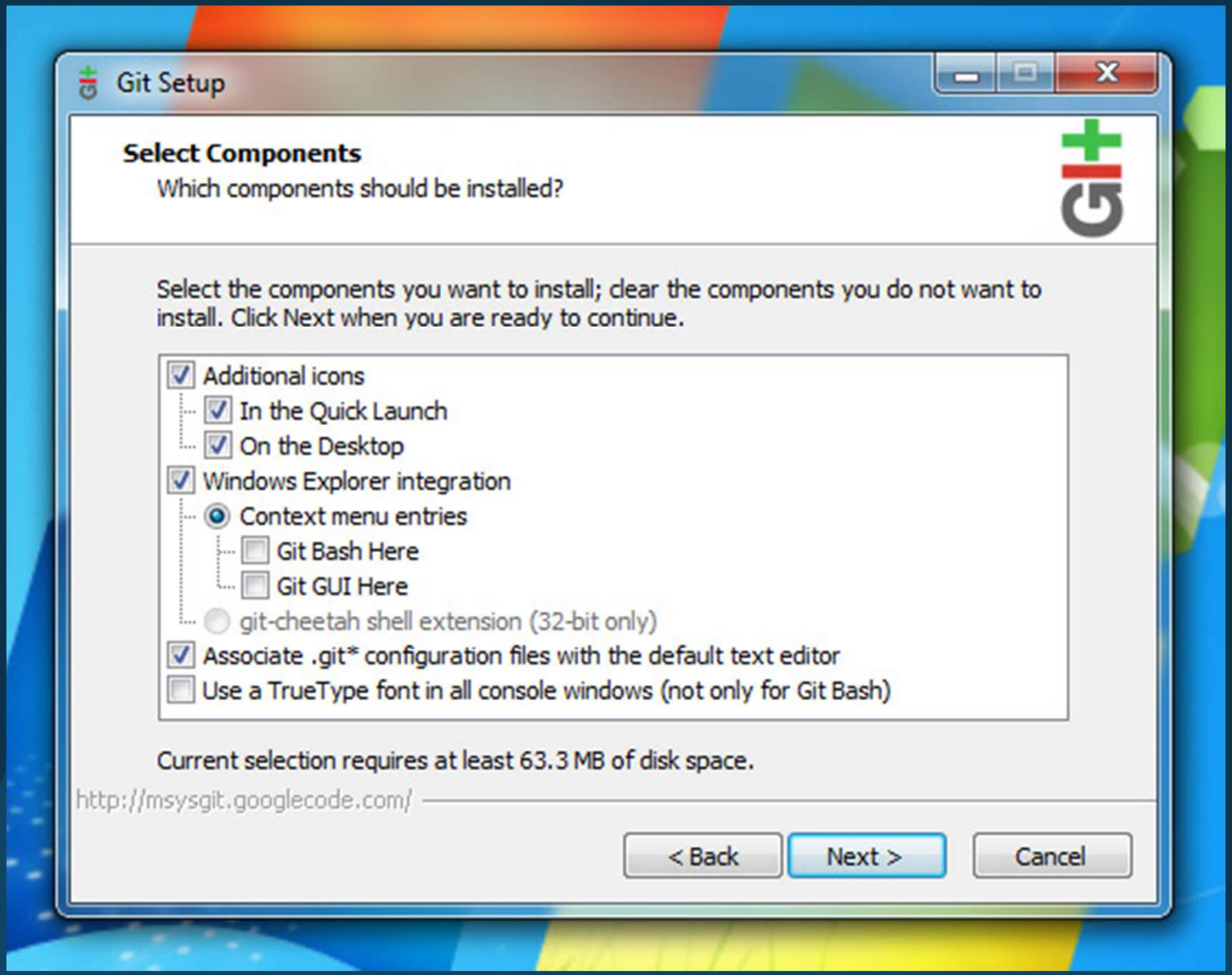
Instalação do Git



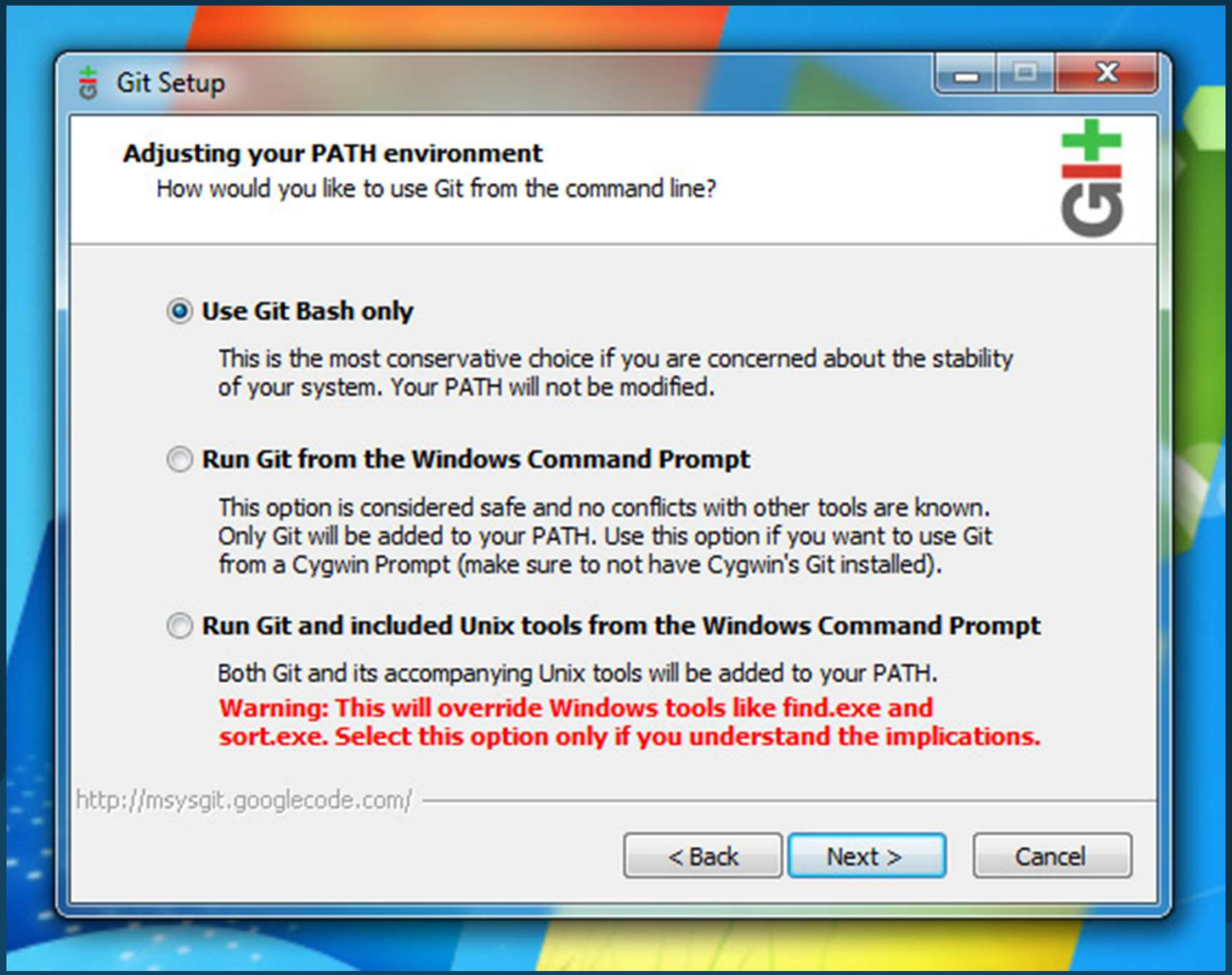
Instalação do Git



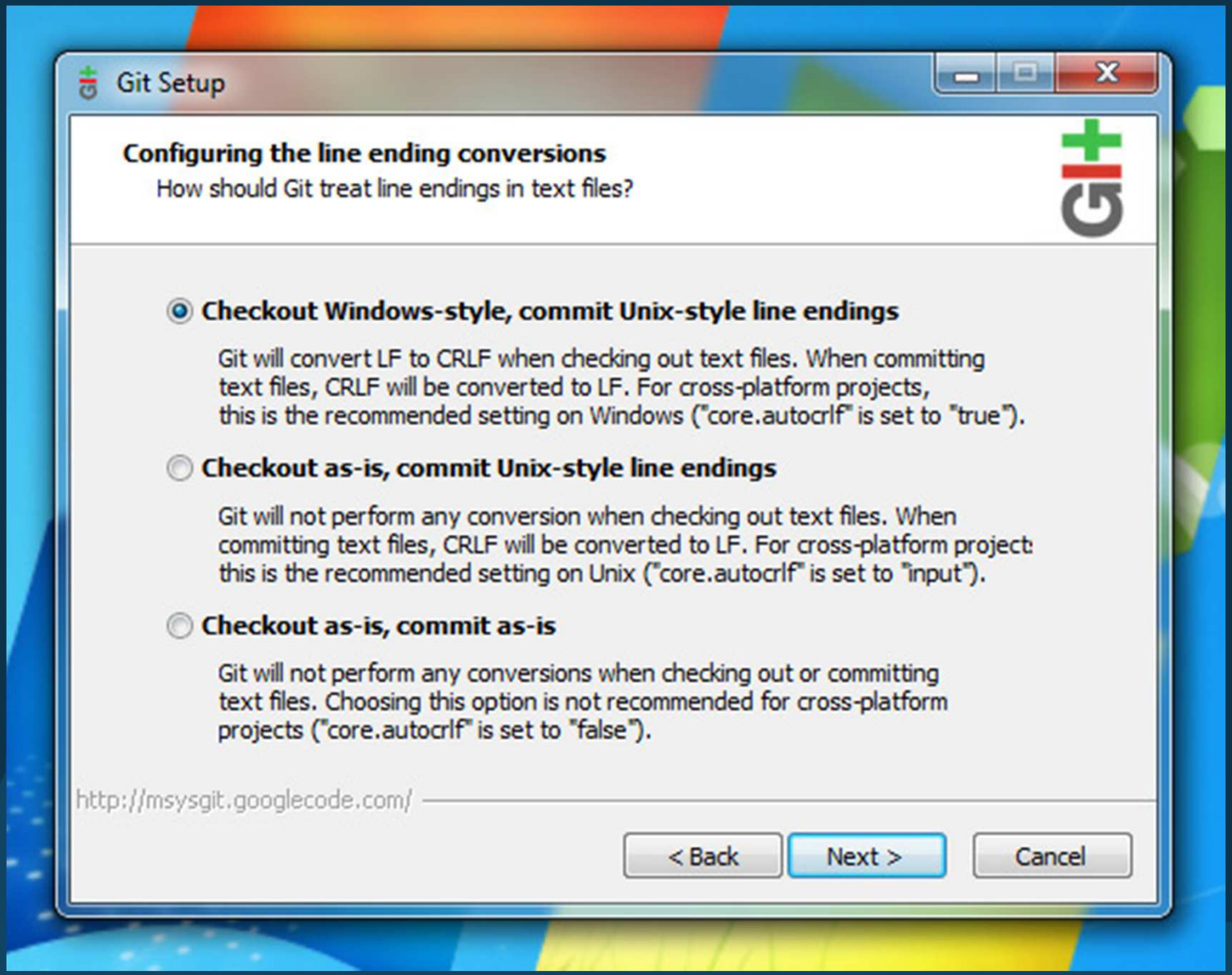
Instalação do Git



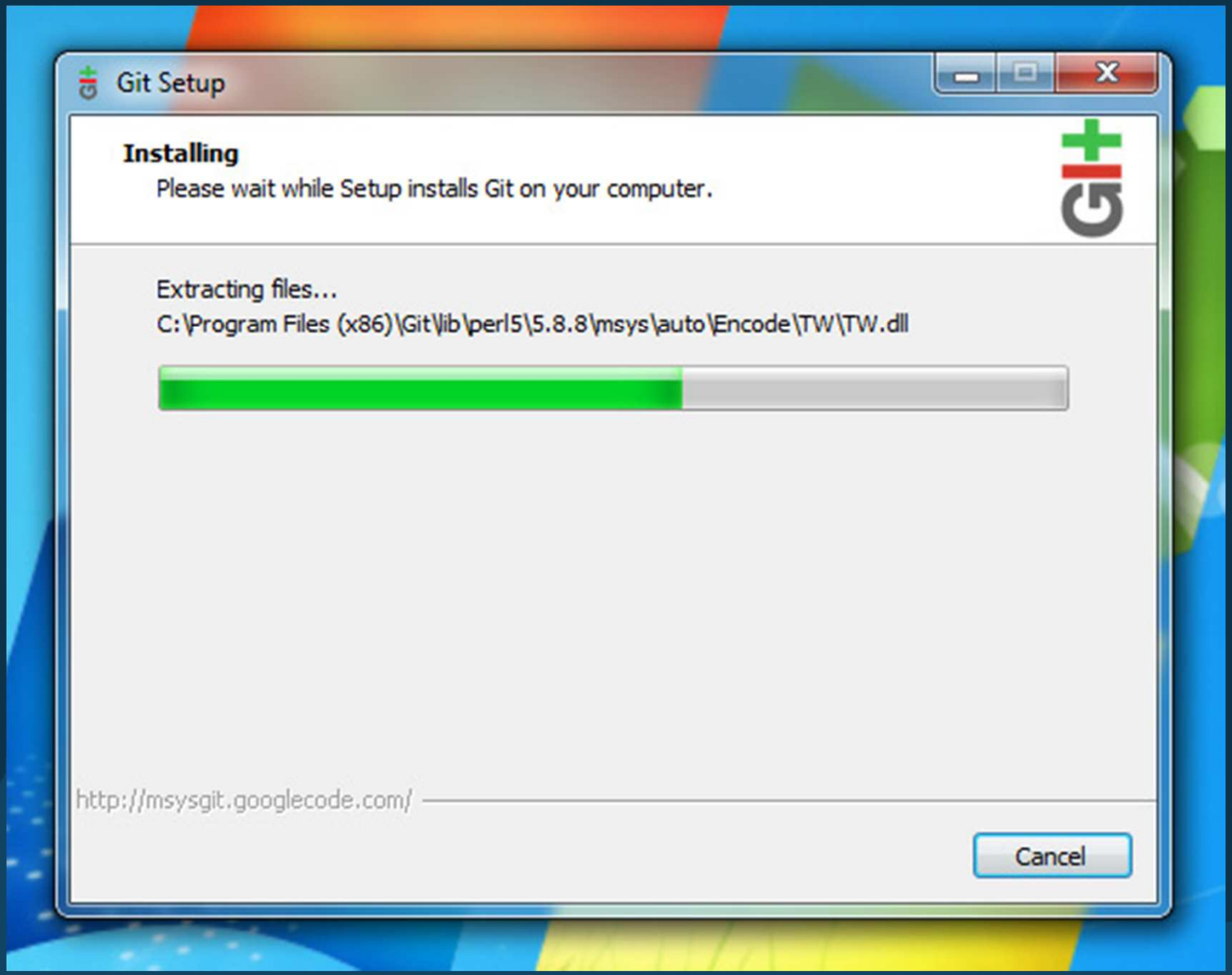
Instalação do Git



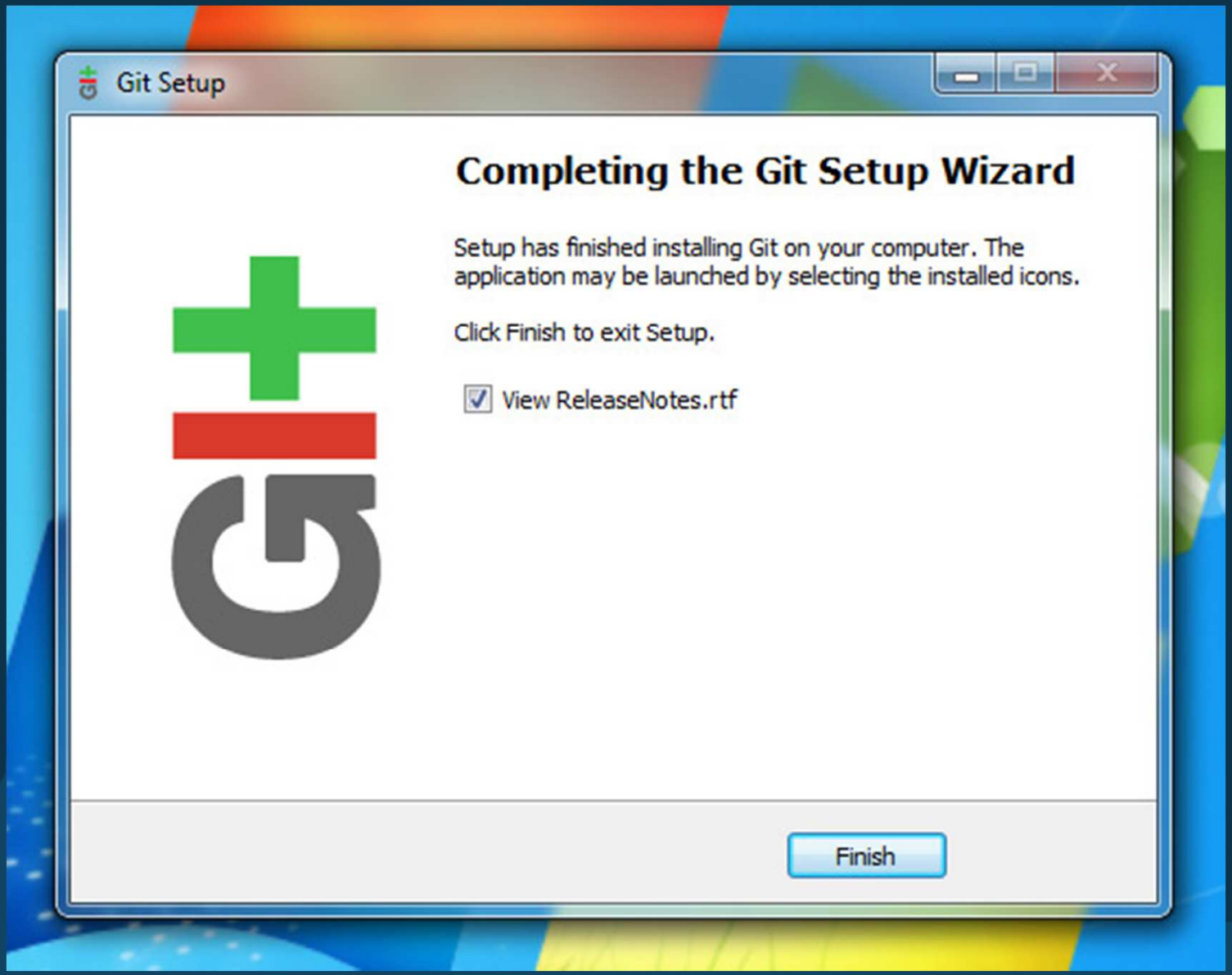
Instalação do Git



Instalação do Git



Instalação do Git



Instalação do Git

- Após a instalação, o passo final antes de começar a fazer qualquer coisa é definir seu nome e e-mail.

```
git config --global user.name "Joao Zinho"
git config --global user.email joao@zinho.com
```

O início:

```
git init
```

Seu melhor amigo:

git status

Adicionando ao índice:

```
git add <arquivo>
```

Removendo arquivo do índice*:

```
git rm --cached <arquivo>
```

Gravando no repositório:

```
git commit -m "Mensagem"
```

Consultando o histórico:

```
git log
```

Revertendo alterações*:

```
git checkout -- <arquivo>
```

ou

```
git checkout <sha1> <arquivo>
```

Retornando ao estado anterior:

```
git reset HEAD <arquivo>
```

Visualizando diferenças:

git diff <arquivo>

ou

git diff <sha1> <arquivo>

Ignorando arquivos:

O utilíssimo ".gitignore"

Listando branches:

```
git branch
```

Criando branches:

```
git branch novo_nome
ou
git checkout -b novo_nome
ou
git checkout -b novo_nome <sha1>
```

Navegando entre branches:

git checkout nome

Apagando branches:

```
git branch -d nome
ou
git branch -D nome
```

Mesclando branches:

```
git merge outro_branch
ou
git merge outro_branch --squash
```

Tornando o histórico linear*:

```
git rebase outro_branch
```

Repositórios remotos

- O Git é um sistema de versionamento distribuído. Desta forma, podem haver várias cópias completas do mesmo repositório em computadores diferentes.

Repositórios remotos

- O Git é um sistema de versionamento distribuído. Desta forma, podem haver várias cópias completas do mesmo repositório em computadores diferentes.
- O SSH é o protocolo mais utilizado para enviar e receber dados do repositório.

Repositórios remotos

- O Git é um sistema de versionamento distribuído. Desta forma, podem haver várias cópias completas do mesmo repositório em computadores diferentes.
- O SSH é o protocolo mais utilizado para enviar e receber dados do repositório.
- Para isto precisamos criar um par de chaves criptográficas, sendo uma pública e outra privada.

Repositórios remotos

- O Git é um sistema de versionamento distribuído. Desta forma, podem haver várias cópias completas do mesmo repositório em computadores diferentes.
- O SSH é o protocolo mais utilizado para enviar e receber dados do repositório.
- Para isto precisamos criar um par de chaves criptográficas, sendo uma pública e outra privada.
- Você não precisa de um servidor. Seu repositório remoto pode ser simplesmente um sistema de arquivos qualquer.

Criando chaves criptográficas

```
ssh-keygen -t rsa -C joao@zinho.com
```

Adicionando o repositório remoto:

```
git remote add origin git@github.com:joaozin/repositorio.git
```

Enviando:

```
git push origin master
```

Atualizando sua cópia local:

```
git fetch origin
ou
git pull origin master
```

Apagando repositório remoto:

```
git remote rm origin
```

Referências

AkitaOnRails.com: Começando com o Git - http://akitaonrails.com/2010/08/17/screencast-comecando-com-git
CNN: At 20, Linux is invisible, ubiquitous - http://edition.cnn.com/2011/TECH/gaming.gadgets/08/25/linux.20/index.html
Myhro Blog: Git para principiantes - http://blog.myhro.info/2011/08/git-para-principiantes/
Pro Git - http://progit.org/book/
Tech Talk: Linus Torvalds on git - http://youtube.com/watch?v=4XpnKHJAok8
Why Git is Better than X - http://whygitisbetterthanx.com/