

Utilizando o GitHub para projetos Colaborativos

GitHub



INDICE

1	SOBRE O GIT E O GITHUB	4
1.1	Conceitos importantes do Git/GitHub	5
2	UTILIZANDO O GITHUB WEB	6
2.1	Criando uma conta no GitHub Web	8
2.2	Acessando o GitHub Web	8
2.3	Criando um repositório no GitHub Web	9
2.4	Copiando um repositório no GitHub Web.....	9
2.5	Efetuatingo commit no GitHub Web.....	10
3	UTILIZANDO O GITHUB FOR WINDOWS.....	12
3.1	Instalando o GitHub for Windows	12
3.2	Criando um novo repositório	13
3.3	Efetuatingo commits no repositório	15
3.4	Efetuatingo novos commits	17
3.5	Revertendo commits.....	18
3.6	Enviando o repositório local para GitHub Web	18
3.7	Sincronizando alterações locais com o GitHub Web	19
3.8	Clonando um repositório do GitHub Web.....	20
3.9	Trabalhando com branches	24
3.10	Efetuatingo o merge dos commits.....	26
4	EXEMPLOS DE PROJETOS DE CIÊNCIAS SOCIAIS NO GITHUB	27
4.1	Dataverse	27
4.2	OpenScholar	28
5	REFERÊ NCIAS	28

FOLHA DE INFORMAÇÕES

DOCUMENTO	
Arquivo:	Apostila_GitHub
Autor:	Carla Oliveira Santos
Título:	Utilizando o GitHub para projetos Colaborativos.
Objetivo:	Apostila elaborada para aula de laboratório, da disciplina "Temas Contemporâneos" (BH1305) da UFABC, ministrada pelo Prof. Sérgio Amadeu da Silveira.

CONTROLE DE VERSÕES

REVISÕES		
Data da Revisão	Versão	Revisão Efetuada
22/02/2016	1.0	Elaboração do documento.

1 SOBRE O GIT E O GITHUB

Souza e Leitão (2012) definem o *Git* como um sistema de controle de versão distribuído. Este sistema foi desenvolvido por Linus Torvalds, em 2005, para o desenvolvimento do kernel do Linux. Por ser um sistema distribuído, todos os desenvolvedores possuem uma cópia local completa do repositório (também chamado de árvore), não necessitando de acesso à Internet para trabalhar com o código.

Atualmente, além do Kernel do Linux, o Git também é utilizado em diversos outros projetos de código aberto e fechado. Além disso, também é bastante utilizado em empresas em todo o mundo, inclusive no Brasil. O Google e o Facebook, por exemplo, utilizam o Git para organizar seus projetos de código aberto.

Em 2008, Tom Preston-Werner, Chris Wanstrath e PJ Hyett criaram o GitHub, uma aplicação Web que possibilita a hospedagem de repositórios Git, além de servir como uma rede social para programadores. Diversos projetos importantes de código aberto são hospedados no GitHub, por exemplo: jQuery, Spring, JUnit entre outros. Atualmente o GitHub possui mais de 3 milhões de usuários e mais de 12 milhões de repositórios.

Além do GitHub os principais serviços que suportam o Git são: Bitbucket, Google Code e Project Locker.

É importante explicar que o Git e o GitHub não são a mesma coisa. O Git é o sistema de controle de versões, com o qual interagimos na linha de comando. Por outro lado, o GitHub é uma rede social para programadores que disponibiliza repositórios Git acessíveis remotamente. O GitHub é muito utilizado em projetos open source que possuem vários colaboradores espalhados pelo mundo inteiro.

Para facilitar a vida de quem não gosta de utilizar linha de comando a equipe do GitHub desenvolveu uma aplicação visual (software) para o Git no Windows chamada de GitHub for Windows.

De acordo com o Laboratório de Cultura Digital (2013) parte-se do princípio de que, embora a tecnologia esteja amparada pelo suporte eletrônico, as trocas realizadas por meio dela são substancialmente humanas.

O processo de desenvolvimento e o trabalho da equipe só têm sentido se a tecnologia vier a cumprir sua função social. O Coordenador de Cultura Digital do Laboratório de Cultura Digital, João Paulo Mehl, ressalta que a concepção de softwares livres não deve ser tarefa exclusiva dos especialistas e programadores, mas também de outros atores da sociedade. ***“Queremos envolver os quilombos, grupos de teatro, os povos indígenas. Queremos que todos deem a sua contribuição enquanto usuários ou desenvolvedores, mostrando o que cada organização precisa”.***

De acordo com o Laboratório de Cultura Digital (2013), a Rede Livre e suas tecnologias foram elaboradas em software livre e com código aberto. O objetivo é que todos que queiram ver seu funcionamento e participar de seu desenvolvimento possam ter acesso

ao código. ***“O conhecimento não deve ser bloqueado. Queremos que as pessoas tenham acesso a essa informação. Em um código aberto as pessoas podem ver que não há nada ilícito”***, afirma. O código aberto é o futuro da computação, pois proporciona um ganho de qualidade e de capacidade na geração do software. ***“No software aberto as pessoas podem ajudar, contribuir, opinar”***.

Segundo o Laboratório de Cultura Digital (2013), para desenvolver a Rede Livre foi escolhido o GitHub, um repositório de códigos. Este repositório é uma ferramenta global que permite a contribuição entre diversos desenvolvedores para a geração e construção colaborativa de um novo código.

A plataforma funciona através do versionamento dos códigos, mantendo as versões anteriores às mudanças feitas pelos programadores, deixando essas mudanças documentadas em linhas de tempo, comentários e arquivos. Assim, evitam-se perdas e a comunicação entre os diversos colaboradores torna-se mais simples.

O GitHub opera também dentro da lógica do software livre. O que é produzido na plataforma está disponível para acesso, leitura e checagem. As mudanças, no entanto, só podem ser feitas por quem está autenticado como um dos desenvolvedores.

A plataforma ainda permite acesso a uma linha do tempo de todo o desenvolvimento do código, ferramentas úteis para os envolvidos e até uma enciclopédia sobre o código, explicando do que se trata e todo o projeto que está sendo desenvolvido.

Como ponto positivo é que há controle e facilidade de apreensão do modo de desenvolvimento. Para tanto, um grupo fica responsável por organizar o processo: aprovar envios e mudanças de código, gerenciar o uso das diversas ferramentas disponíveis e gerir grupos de desenvolvimento e colaboração, tudo através de um processo transparente e aberto. ***“Ferramentas de uso popular, de governos, voltadas à população em geral, deveriam ser feitas na lógica do software livre, pois não haveria donos para esse conhecimento, ele seria de todos”***, ressaltam.

Para mais informações sobre a Rede Livre acesso o site: <http://redelivre.org.br/>

1.1 Conceitos importantes do Git/GitHub

- **Repository:** Repositório é o local onde fica todos os arquivos do projeto, inclusive o histórico de versões.
- **Commit:** Funcionalidade que permite efetuar o controle das alterações realizadas. É como se fosse um “checkpoint” do projeto. Sempre que for necessário é possível retroceder um commit.
- **Revert:** Reverte (retrocede) um commit efetuado.
- **Branch:** É uma ramificação do seu projeto. Cada branch representa uma versão do seu projeto. Podemos seguir uma linha de desenvolvimento a partir de cada branch.

- **Fork:** Consiste em realizar a cópia de um repositório que pertence a outra pessoa, adicionando esse repositório ao nosso repositório. Ou seja, nos tornamos os “proprietários” do repositório o qual estamos realizando o fork, porém o original se mantém intacto.
- **Clone:** Obtém uma cópia de um repositório remoto.
- **Push:** Envia os commits dos arquivos locais para um repositório remoto.
- **Sync:** Sincroniza as alterações locais efetuadas com o repositório remoto.
- **Pull request:** Consiste em uma solicitação de integração das nossas modificações com o repositório remoto. Basta informar na caixa de comentários um detalhamento do que foi criado/modificado. O responsável pelo repositório agora poderá simplesmente aceitar sua sugestão, dialogar com você tirando dúvidas ou solicitando alterações ou até mesmo negar sua proposta. Neste cenário toda a comunidade do GitHub poderá comentar o que você fez.
- **Merge:** É a capacidade de incorporar alterações do Git, onde acontece uma junção dos branches.

A Figura 1 mostra o diagrama das interações das funcionalidades do Git/GitHub.

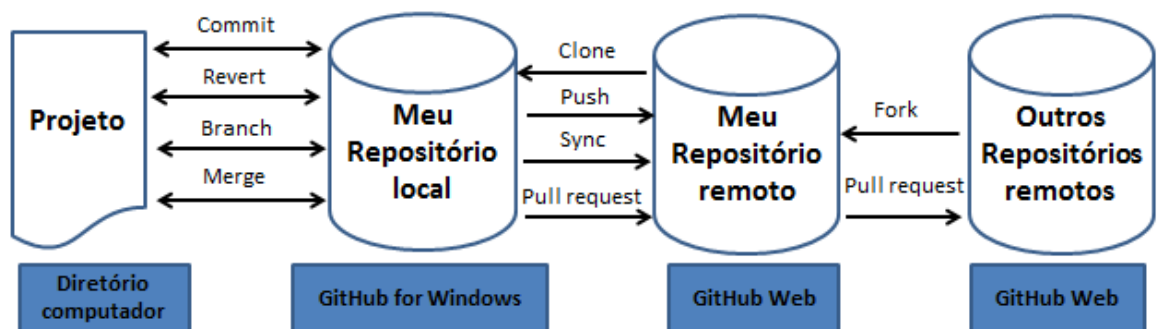


Figura 1 – Diagrama das interações das funcionalidades do Git/GitHub

2 UTILIZANDO O GITHUB WEB

De acordo com Freire (2014) um dos maiores problemas das ciências sociais é a questão da replicabilidade. Reproduzir o trabalho de um pesquisador e conferir seus resultados é um dos pilares fundamentais do processo científico, e embora esta seja prática corrente em diversas áreas do conhecimento, seu uso ainda é limitado nas humanidades.

Para Freire (2014) é certo que, em muitos casos, as análises sociais não se prestam facilmente à replicabilidade: pesquisas de campo e descrições de fatos históricos, por exemplo, são por definição baseados em eventos únicos. Mas e quanto aos trabalhos quantitativos? Se a coleta dos dados quantitativos também não é facilmente replicável (sobretudo por custos financeiros), os processos utilizados na análise de dados podem ser analisados e refeitos sem grandes problemas. E quais as vantagens disso?

Neste sentido Freire (2014) destaca três pontos importantes: Em primeiro lugar, ao colocar seus dados abertos para o escrutínio dos pares, o cientista atesta de boa fé que não tem nada a esconder, e que acredita que seus resultados são robustos. Isso dá credibilidade a sua pesquisa, e em termos práticos pode significar o aumento do número de citações do trabalho em questão. Em segundo lugar, é claramente desejável que a ciência seja “autocorretiva”, ou seja, que o conhecimento científico se acumule a partir da revisão, crítica e aprimoramento das pesquisas anteriores.

Nesse sentido, abrir os dados e mostrar os procedimentos de um trabalho colabora para que outros possam avaliar e corrigir eventuais erros e formular melhores teorias no futuro.

Por fim, há também um importante caráter didático na replicação, cujo valor nem sempre é apreciado. Ao entrar em contato com bancos de dados e ver as ferramentas utilizadas por um autor, um aluno pode entender como procedimentos estatísticos são utilizados na prática, como estimar modelos em uma linguagem de computador e, também, entender que as análises são fruto de muita tentativa e erro.

Os gráficos e tabelas que aparecem nas prestigiadas revistas da área não surgiram como mágica, mas são o resultado de diversas modificações até se encontrar a forma adequada para entender o que os dados querem dizer.

No post publicado no blog, “Sociais e Métodos”, Freire (2014) sugere que os cientistas sociais usem o Git, pois ele permite que os usuários rastreiem qualquer mudança feita nos dados — sejam eles scripts, bancos, etc — e que eles possam voltar a uma versão prévia desses objetos a qualquer momento. Assim, todas as vezes que algo é salvo no Git, o sistema mostra quais foram as alterações feitas por cada um dos usuários, e caso alguma coisa dê errado, em segundos você pode voltar ao objeto antigo e corrigir os erros. Tudo em um ambiente organizado, sem necessidade de alterar os nomes e os tipos dos arquivos.

Além disso, o Git também é capaz de criar vários “ramos” (branches) de um mesmo projeto, assim várias pessoas podem trabalhar no mesmo código sem modificar o objeto original. Na opinião de Freire (2014), não há melhor ferramenta para trabalhos colaborativos, mantendo a transparência de todo o processo.

O site mais popular para armazenar repositórios feitos com o Git é o GitHub. No caso do GitHub, existem centenas de projetos de cientistas sociais hospedados lá (por exemplo, o famoso Instituto de Ciências Sociais Quantitativas de Harvard guarda o código de todos os seus programas em um repositório), e você pode segui-los como no Twitter ou Facebook. Assim você também pode interagir e, se quiser, fazer parte dos projetos mais recentes da área.

Freire (2014) montou um pequeno tutorial para começar a usar o Git. Segundo ele, a primeira coisa a ser feita é criar sua conta no GitHub. Caso você só crie repositórios

públicos, a conta é totalmente gratuita. Se você quiser guardar códigos privados, os planos começam com 7 dólares por mês, mas há um plano gratuito para estudantes por dois anos.

É possível utilizar as funções mais básicas do GitHub sem instalar nenhum programa. Criar um repositório (uma pasta com seus arquivos), colocar seus scripts online (copiando e colando manualmente), e criar uma cópia de um repositório de outra pessoa (fork, no linguajar do Git), tudo pode ser feito direto no site.

2.1 Criando uma conta no GitHub Web

Para criar sua conta no GitHub Web acesse o site: <https://github.com/>

Em seguida clique no botão **Sign up**. Na tela da Figura 2 preencha os campos abaixo:

- **Username:** Informe o nome do seu usuário.
- **E-mail Address:** Informe seu endereço de e-mail.
- **Password:** Informe sua senha.

Após informar corretamente os campos clique no botão **Create an account**. Feito isso sua conta será criada no GitHub Web.

Join GitHub
The best way to design, build, and ship software.

Step 1: Set up a personal account | Step 2: Choose your plan | Step 3: Go to your dashboard

Create your personal account

Username
[Input field]
This will be your username — you can enter your organization's username next.

Email Address
[Input field]
You will occasionally receive account related emails. We promise not to share your email with anyone.

Password
[Input field]
Use at least one lowercase letter, one numeral, and seven characters.

You'll love GitHub

- Unlimited collaborators
- Unlimited public repositories
- Great communication
- Friction-less development
- Open source community

Figura 2 – Criando uma conta no GitHub Web

2.2 Acessando o GitHub Web

Para usar o GitHub Web acesse o site: <https://github.com>

Em seguida clique no botão **Sign in**. Informe seu usuário e senha e clique no botão **Sign in**.

2.3 Criando um repositório no GitHub Web

Após criar sua conta no GitHub e acessá-la, o próximo passo é criar um repositório. Para criar um repositório é só clicar em no + no canto superior direito (Figura 3), e escolher a opção **New repository**.

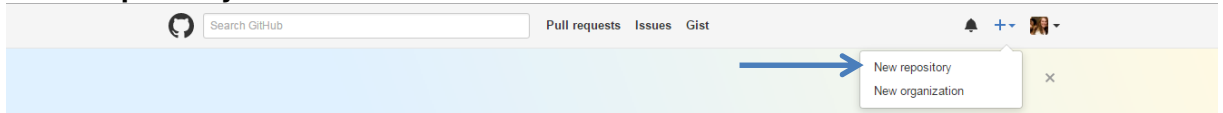


Figura 3 – Novo repositório

Ao clicar na opção **New repository** será exibida a tela da Figura 4 abaixo. No campo **Repository name** informe o nome que desejar para o repositório (geralmente separados por hífen, como por exemplo, “meu-primeiro-repositorio”). No campo opcional **Description** coloque uma descrição se assim desejar e para finalizar clique em **Create repository**.

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

Repository name

 carlaolivei /

Great repository names are short and memorable. Need inspiration? How about [studious-waffle](#).

Description (optional)

☒ **Public**
Anyone can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾

Add a license: **None** ▾ ⓘ

Create repository

Figura 4 – Criando um repositório

2.4 Copiando um repositório no GitHub Web

Para copiar um repositório de outra pessoa para a sua conta também é bem fácil. Para isso pesquise o projeto que deseja clonar. Quando localizar o projeto desejado clique no link desse projeto como mostra a Figura 5.

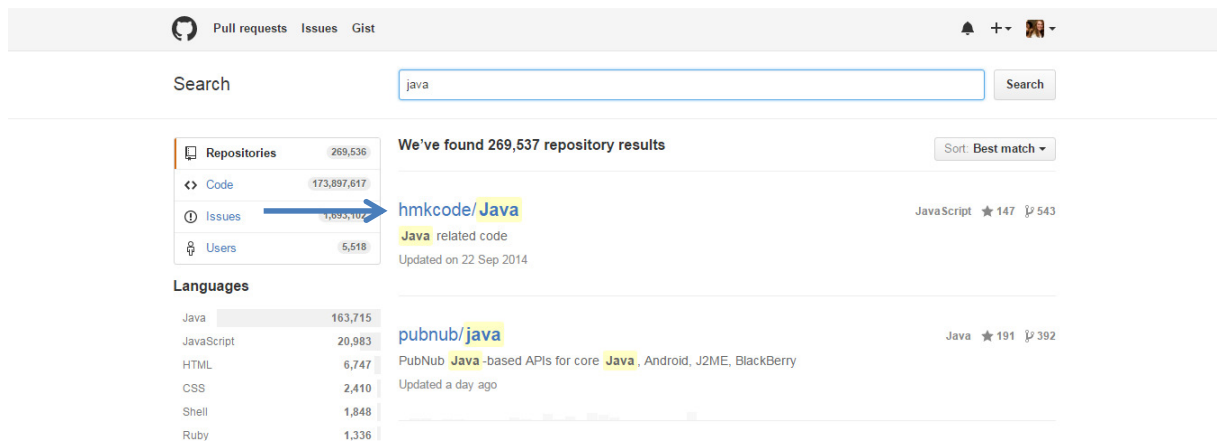


Figura 5 – Pesquisando um projeto

Ao clicar no link será exibida uma tela similar à tela da Figura 6. Em seguida clique em **Fork** no canto superior direito. Após isso o repositório desejado estará na sua lista de repositórios.

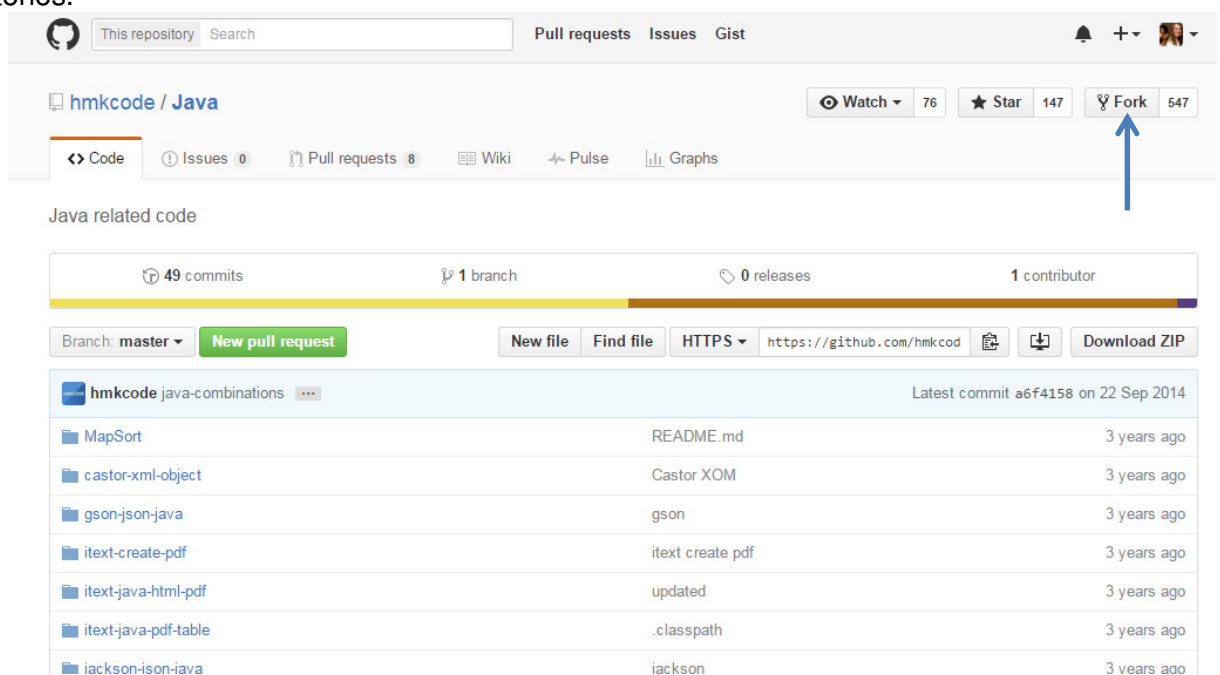


Figura 6 – Clonando (copiando) o repositório de outra pessoa

2.5 Efetuando commit no GitHub Web

Para adicionar ou modificar arquivos, você precisa fazer um *commit*. Vamos ver um exemplo. Escolha o projeto e em seguida clique no arquivo que deseja alterar conforme mostra a Figura 7.

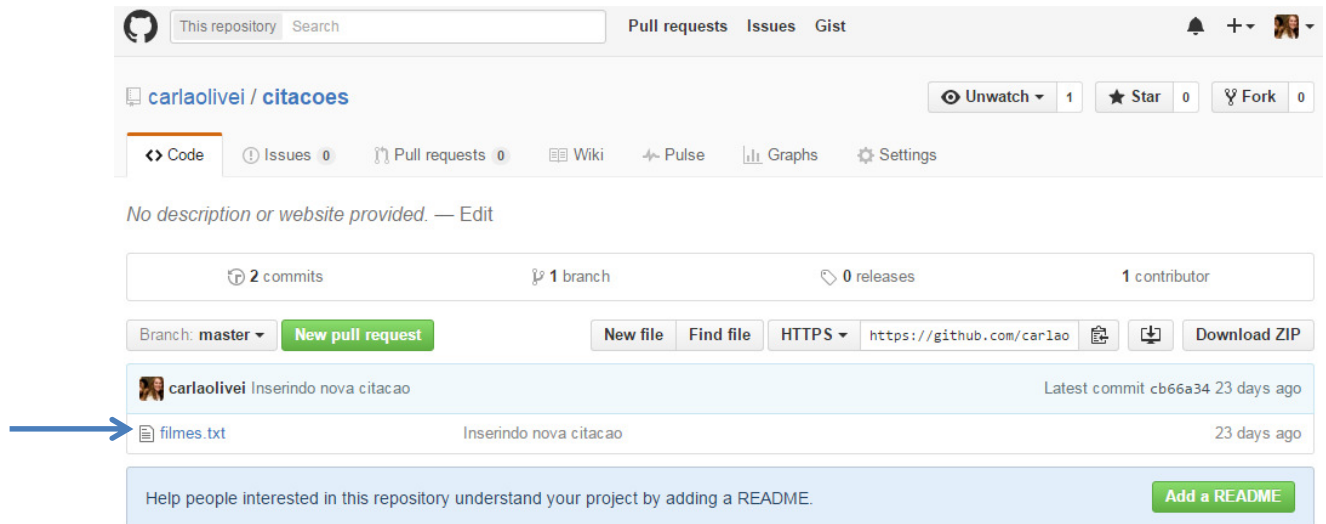


Figura 7 – Escolha do arquivo que será alterado

Quando abrir a tela Figura 8 clique no ícone em forma de lápis, disponível à direita para habilitar a edição do arquivo.

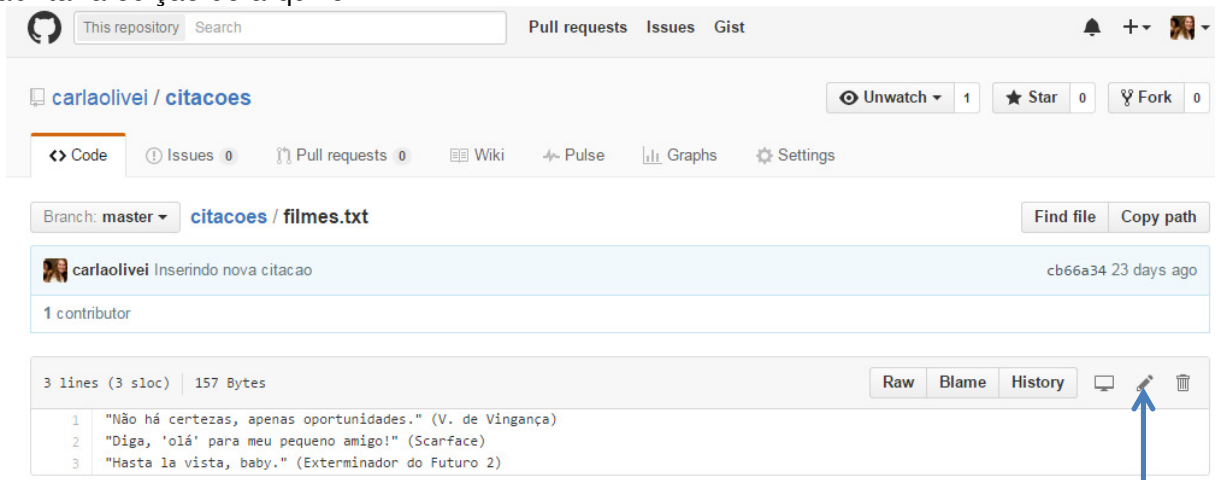


Figura 8 – Modo de edição do arquivo que será alterado

Edite o arquivo conforme desejar e na parte inferior da tela (Figura 9) efetue o commit. Para isso informe o texto do commit, uma descrição opcional e em seguida clique no botão **Commit changes** conforme Figura 9. Após isso as alterações efetuadas no seu arquivo serão executadas e registradas.

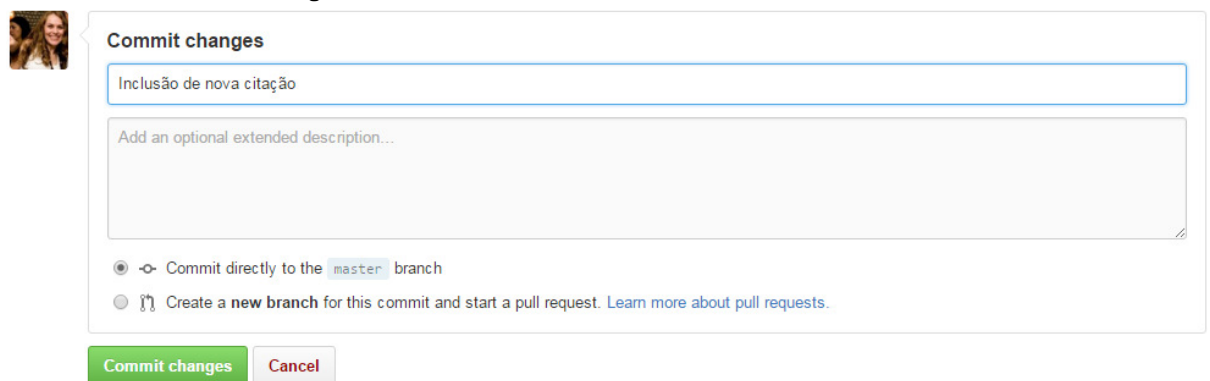


Figura 9 – Efetuado commit das alterações efetuadas no arquivo

Com isso você já pode usar o GitHub Web. No entanto, para usar o todo o potencial to Git é necessário instalar o programa no seu computador.

O Git tem versões para Windows, MacOS e Linux, então não há problema quanto à compatibilidade.

Para facilitar a vida do usuário a equipe do GitHub desenvolveu uma aplicação do GitHub para o Windows (GitHub for Windows). Para mais detalhes vide o item 3.

3 UTILIZANDO O GITHUB FOR WINDOWS

3.1 Instalando o GitHub for Windows

Para instalar o GitHub for Windows, precisamos baixar a aplicação. Para isso acesse o site: <http://windows.github.com> conforme mostra a Figura 10.

Em seguida clique no botão “**Download GitHub Desktop**”.

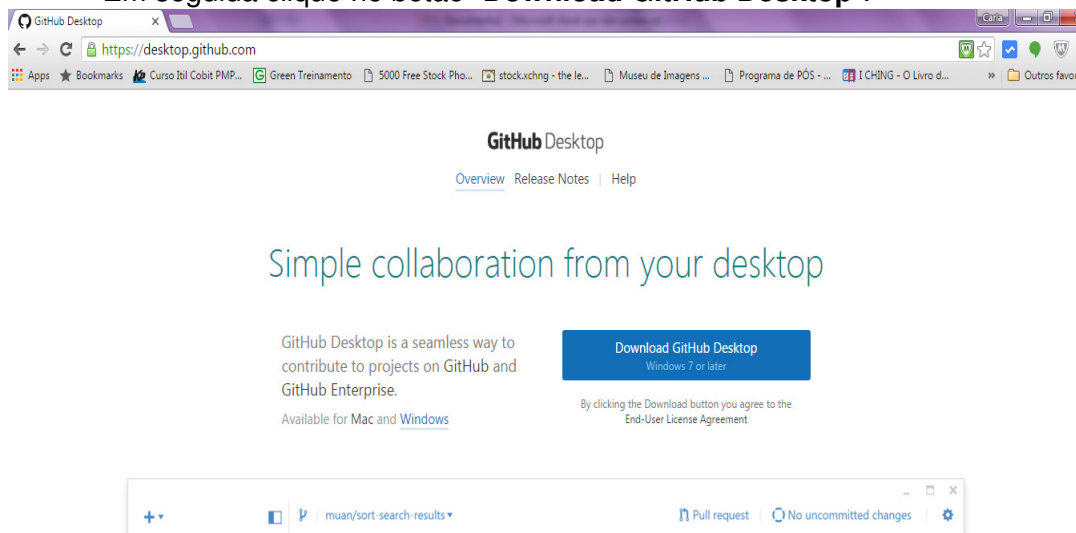


Figura 10 – Tela de Download do GitHub for Windows

NOTA:

⇒ **VERSÕES DO WINDOWS SUPORTADAS:** O GitHub for Windows funciona apenas no Windows Vista, Windows 7 e Windows 8.

Após a conclusão do download do arquivo “**GitHubSetup.exe**”, devemos executá-lo clicando duas vezes sobre ele.

Ao clicar duas vezes no arquivo “**GitHubSetup.exe**” será exibida a tela de instalação.

Após a conclusão da instalação será exibida uma tela onde deve ser informado o usuário e senha cadastrados no **GitHub Web**. Após informar o **usuário** e **senha** clicar no botão **Log in**.

Na próxima tela informar o **Nome** e o **E-mail** do usuário Git. Em seguida clicar no botão **continue**.

Será exibida uma tela onde serão listados os repositórios Git encontrados no computador. Caso não seja encontrado nenhum repositório, será exibida uma mensagem informando que não foram encontrados repositórios. Neste caso, clicar no botão **Skip**, para adicionar os repositórios posteriormente.

Após clicar no botão **Skip**, será direcionado para a tela principal do GitHub for Windows.

Até aqui a aplicação foi instalada. Se verificar na área de trabalho do Windows foram adicionados dois novos atalhos, sendo um chamado **GitHub**, que serve para executar o GitHub for Windows, e o outro chamado **Git Shell**, para executar o Git via prompt de comando.

IMPORTANTE:

- ⇒ O GitHub for Windows criará um novo diretório (pasta) chamado **GitHub**, localizado na pasta **Documentos** do usuário (C:\Users\nome\Documents\GitHub). Esse é o diretório padrão onde os novos repositórios serão criados.

3.2 Criando um novo repositório

Agora que o GitHub já foi instalado e configurado, podemos criar um novo repositório. Para isto, basta clicar no botão **+** localizado na tela principal da aplicação conforme mostra a Figura 11.

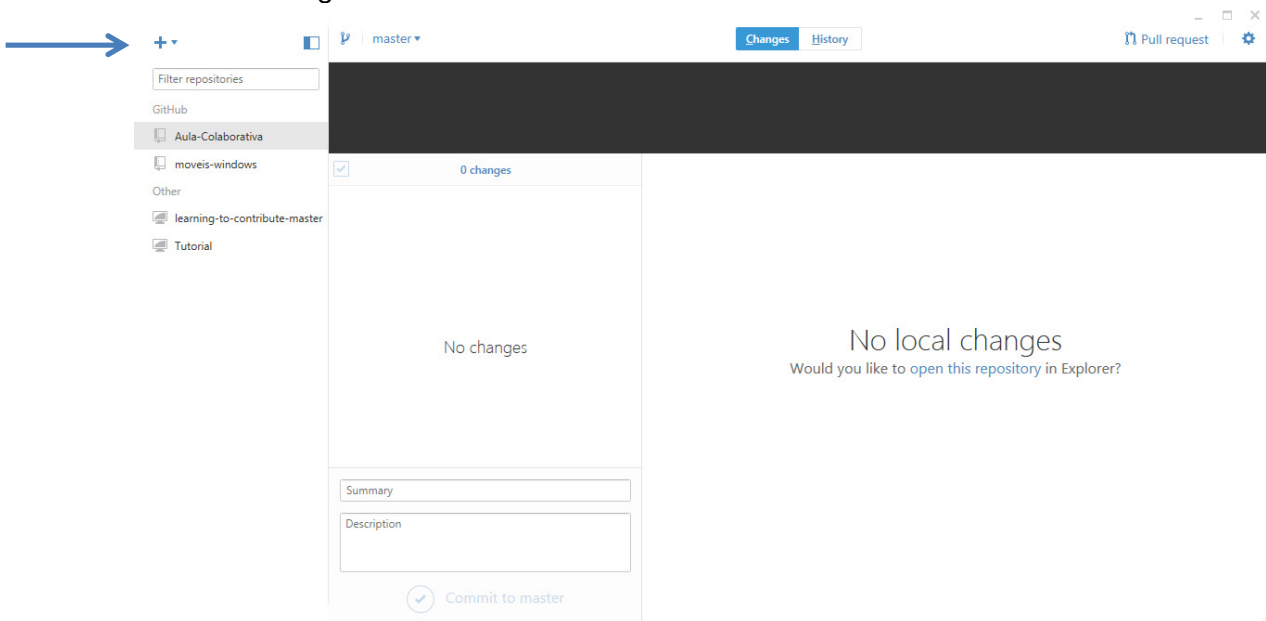


Figura 11 – Novo repositório no GitHub for Windows

Ao clicar no botão **+** será exibida a tela da Figura 12.

No campo **Nome** preencher com o nome do repositório, no nosso caso preencher com **Laboratorio**. No campo **Local path** escolher o diretório onde o repositório deverá ser criado. Neste caso, a sugestão é deixar preenchido com o diretório padrão.

Observe que há também o campo chamado **Git ignore**, onde é possível escolher a linguagem de programação utilizada no projeto, dentre as opções disponíveis, e com isto o arquivo **.gitignore** será criado automaticamente. No nosso caso selecione a opção **None**.

Após preencher os campos, clicar no botão **Create repository**. Aguardar a criação do repositório e então o mesmo será listado na tela.

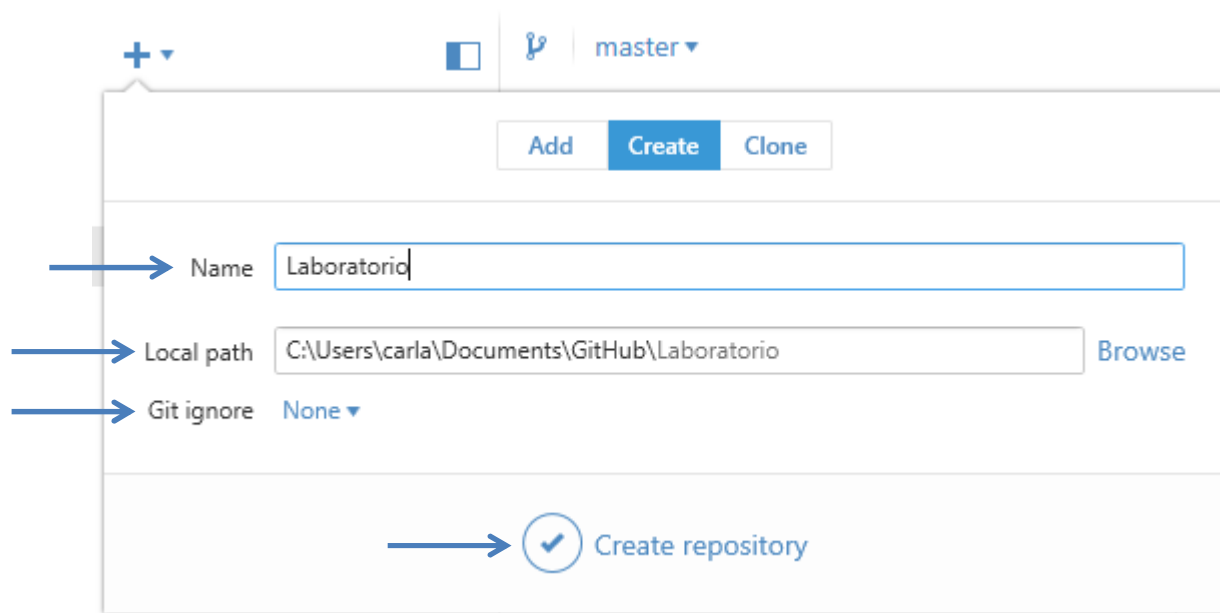


Figura 12 – Criando um novo repositório

Após o repositório ser criado repare que a tela está dividida em três colunas conforme mostra a Figura 13. Na primeira são listados os repositórios. Na segunda são exibido os commits do repositório juntamente com um formulário para efetuar um novo commit. Na última coluna são exibidos os arquivos do repositório.

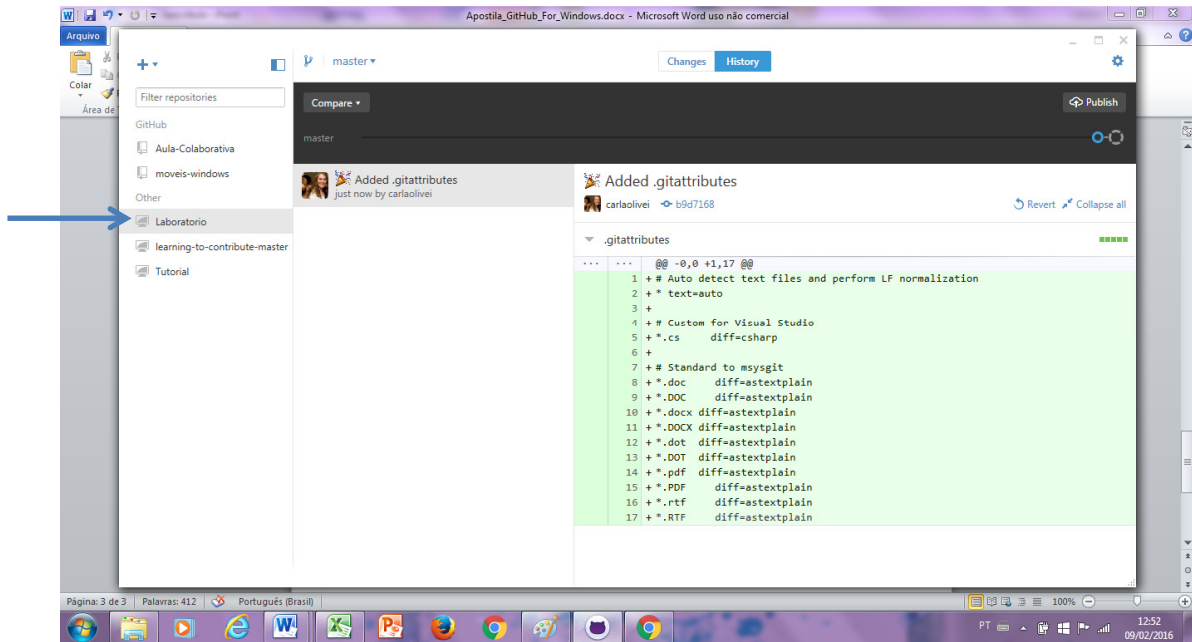


Figura 13 – Tela do novo repositório criado

3.3 Efetuando commits no repositório

Agora vamos criar um novo arquivo no repositório **Laboratorio** criado. Em seguida efetuar um commit.

Para isso acesse o diretório através do seguinte caminho:
C:\Users\nome\Documents\GitHub\Laboratorio.

Crie um novo arquivo chamado **lab1.txt** com o seguinte conteúdo:

GitHub for Windows

Seja bem vindo!

Primeira Laboratorio.

Agora ao voltar para o GitHub for Windows veremos o novo arquivo sendo listado. Para isso clique no botão **Changes**, como mostra a Figura 14 abaixo.

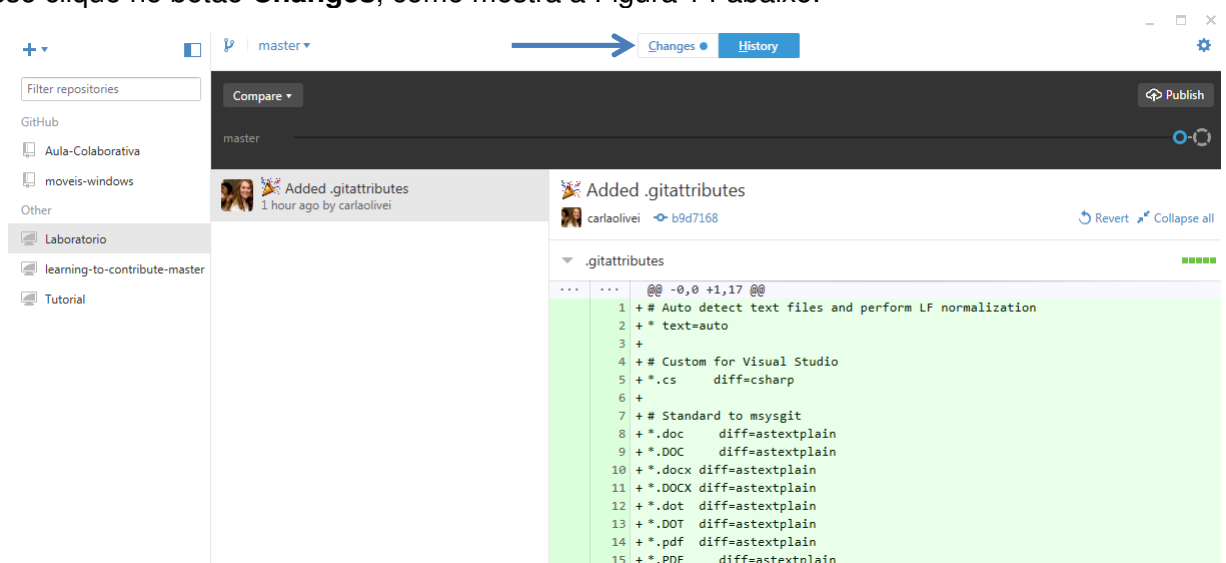


Figura 14 – Exibição do novo arquivo criado

Ao clicar no botão **Changes**, o novo arquivo criado será listado conforme mostra a Figura 15.

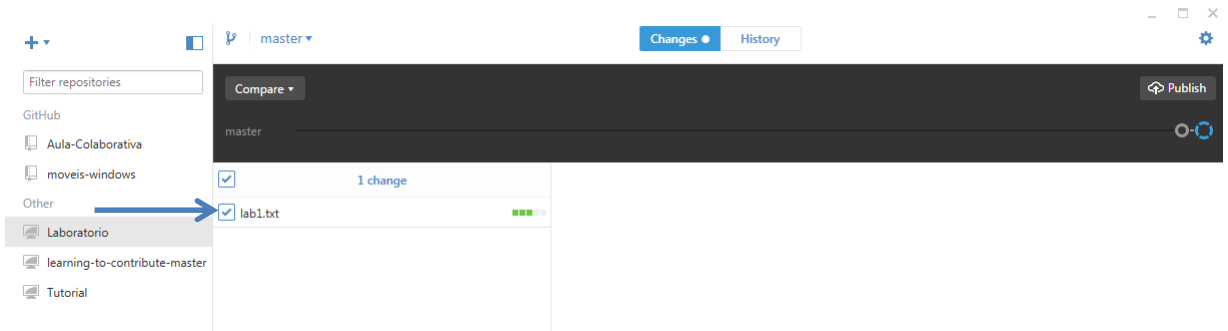


Figura 15 – Novo arquivo criado

Para efetuar o commit do arquivo, basta digitar a mensagem do commit no campo **Summary**, e opcionalmente, preencher uma descrição mais detalhada no campo **Description**. Após isso efetuar o commit clicando no botão **Commit to master** conforme mostra a Figura 16.

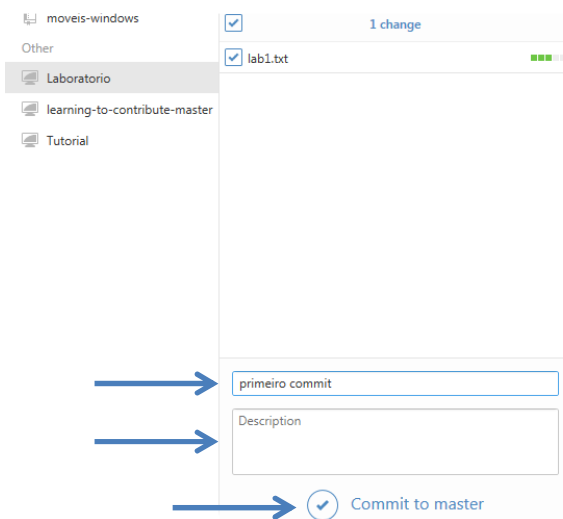


Figura 16 – Efetuando commit de um arquivo

Para visualizar o commit efetuado clique no botão **History** conforme mostra a Figura 17.

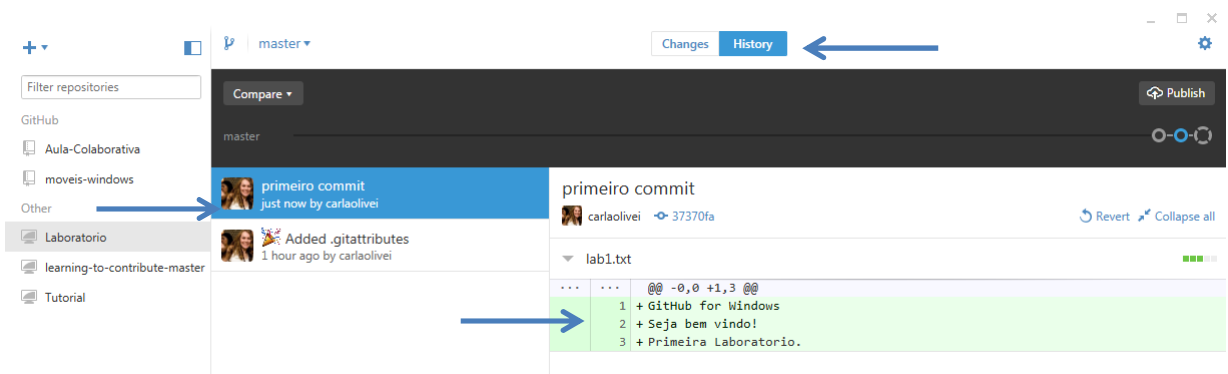


Figura 17 – Visualizando o commit de um arquivo

3.4 Efetuando novos commits

Vamos efetuar um novo commit no repositório. Para isso, altere o arquivo **lab1.txt** incluindo o trecho abaixo:

GitHub for Windows

Seja bem vindo!

Primeiro Laboratorio.

Alterando o arquivo para o segundo commit.

Agora ao voltar para o GitHub for Windows e visualizar a alteração efetuada, clique no botão **Changes**. Ao clicar no botão **Changes** a alteração será exibida. Após isto registre a alteração com um novo commit conforme mostra a Figura 18.

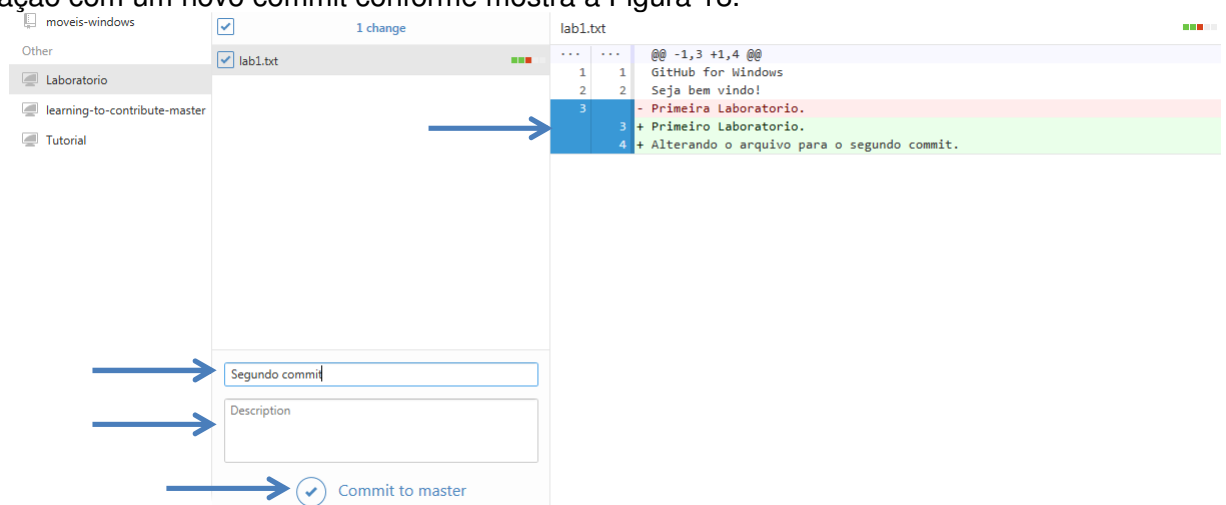


Figura 18 – Registrando a nova alteração no arquivo com o commit

Repare na Figura 19 que o conteúdo já existente que foi alterado é exibido com o sinal de -. O novo conteúdo inserido é mostrado com sinal de +.

Após efetuar o commit repare também que o mesmo é exibido na listagem de commits, onde é exibida a mensagem, o autor e a data em que cada commit foi executado (Figura 19).

Ao clicar em algum dos commits serão exibidas as alterações realizadas naquele commit em específico.

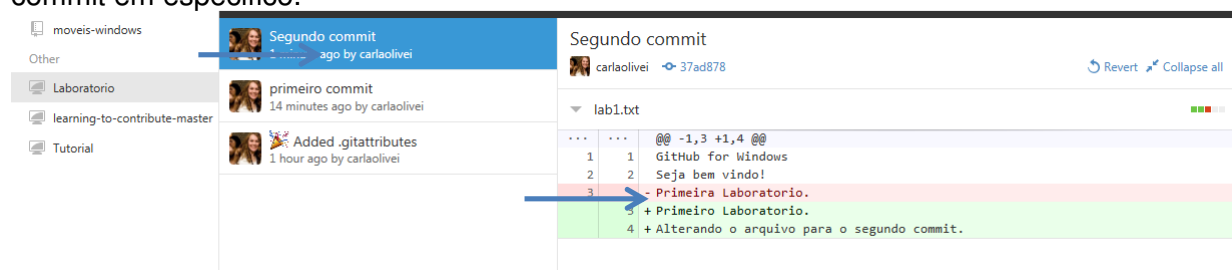


Figura 19 – Detalhes do commit

3.5 Revertendo commits

É possível reverter os commits facilmente, bastando para isso clicar no commit desejado, e em seguida clicar no botão **revert** conforme mostra a Figura 20.

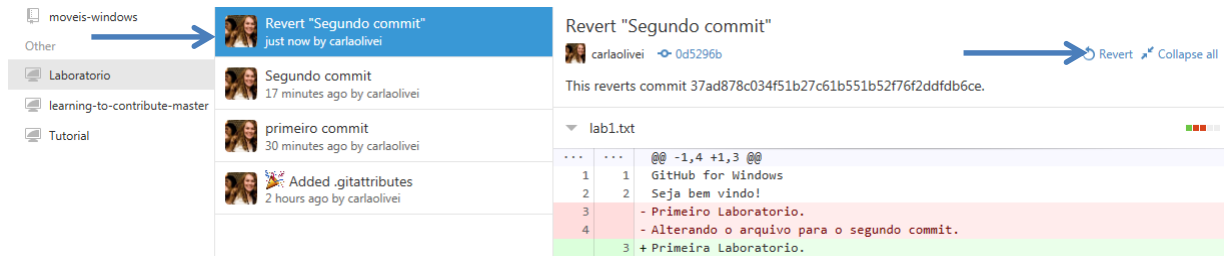


Figura 20 – Revertendo um commit

Após clicar no botão **revert** um novo commit será efetuado automaticamente, desfazendo as alterações do commit selecionado.

Para confirmar é só abrir o arquivo **lab1.txt**. O conteúdo voltará a ser o que era antes da alteração efetuada.

GitHub for Windows
Seja bem vindo!
Primeira Laboratorio.

3.6 Enviando o repositório local para GitHub Web

O GitHub for Windows possui integração com o GitHub Web. Permite com isto envio do nosso repositório local para o GitHub Web.

Para enviar o repositório local para o GitHub Web basta selecionar o repositório desejado e clicar no botão **Publish** conforme mostra a Figura 21

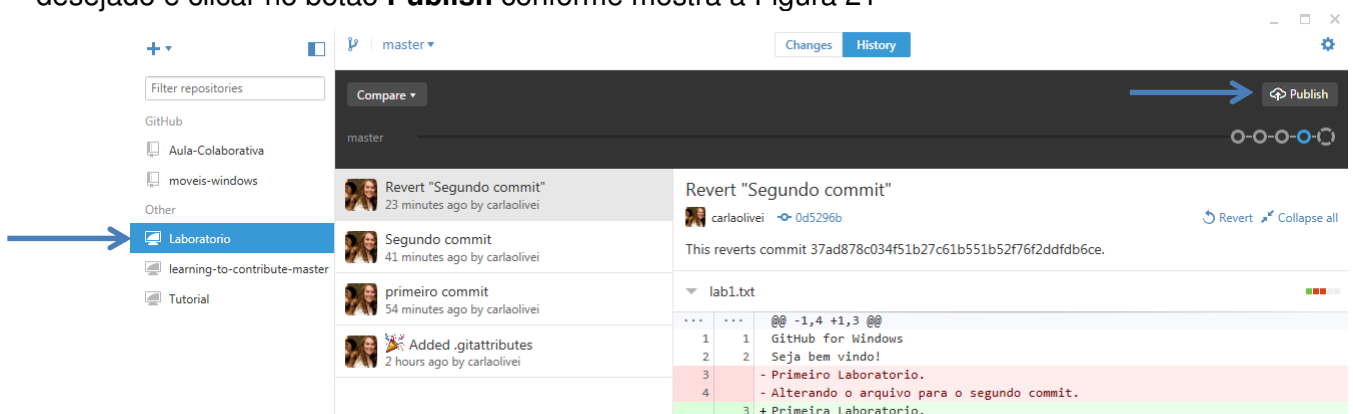


Figura 21 – Enviando o repositório para o GitHub Web

Ao clicar no botão **Publish** será exibida uma tela (Figura 22) onde podemos preencher uma descrição detalhada do repositório, e escolher se o repositório deverá ser privado. Caso o usuário tenha cadastro em algum dos planos pagos do GitHub.

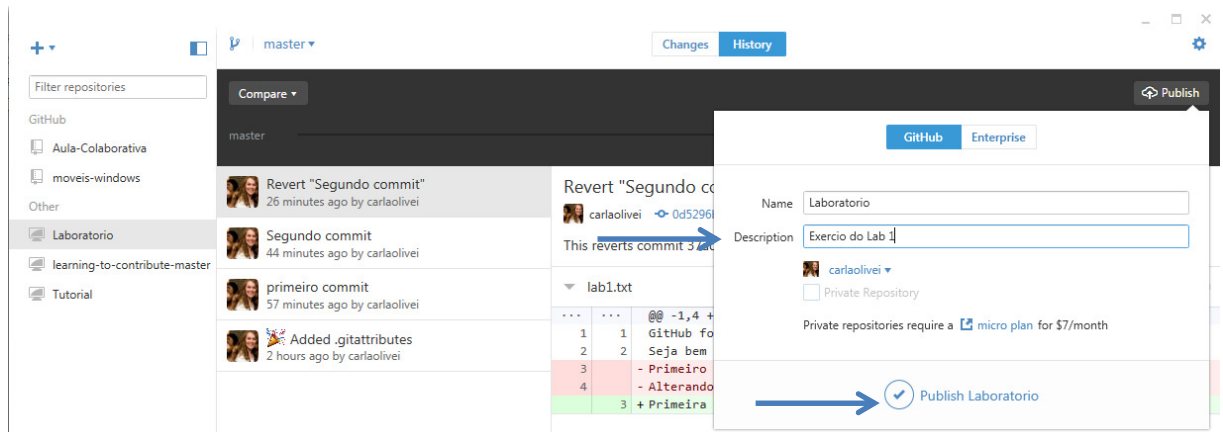


Figura 22 – Tela para enviar o repositório para o GitHub Web

Após preencher as informações basta clicar no botão **Publish Laboratorio** e este repositório será enviado para o GitHub Web.

Se você acessar a página do seu usuário no GitHub Web, verá que o repositório foi enviado corretamente conforme mostra a Figura 23.

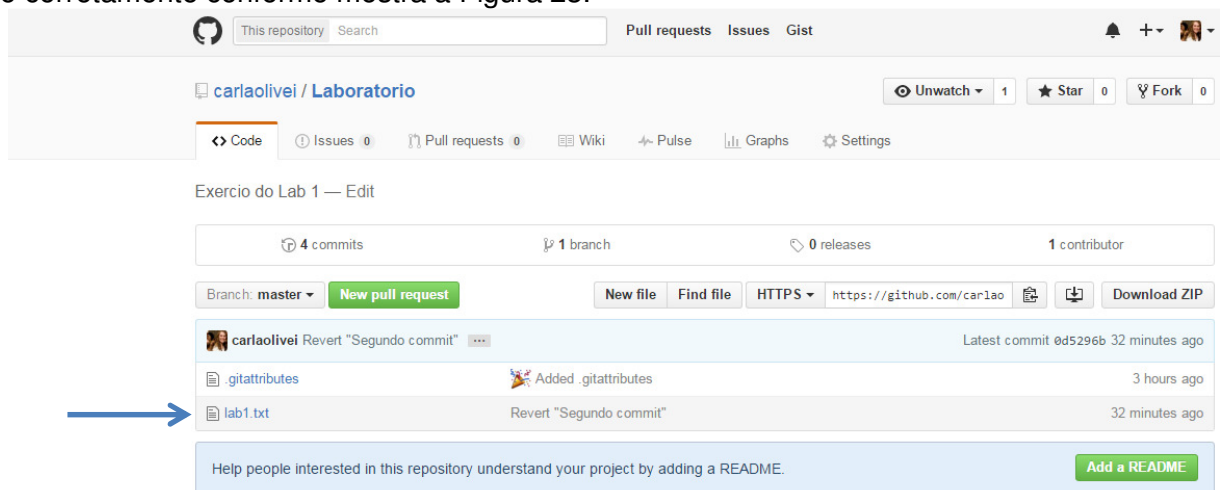


Figura 23 – Repositório enviado com sucesso para o GitHub Web

3.7 Sincronizando alterações locais com o GitHub Web

Após enviar um repositório para o servidor remoto sempre que precisar sincronizar as alterações locais com o repositório remoto no GitHub Web, basta clicar no botão **Sync** conforme mostra a Figura 24.

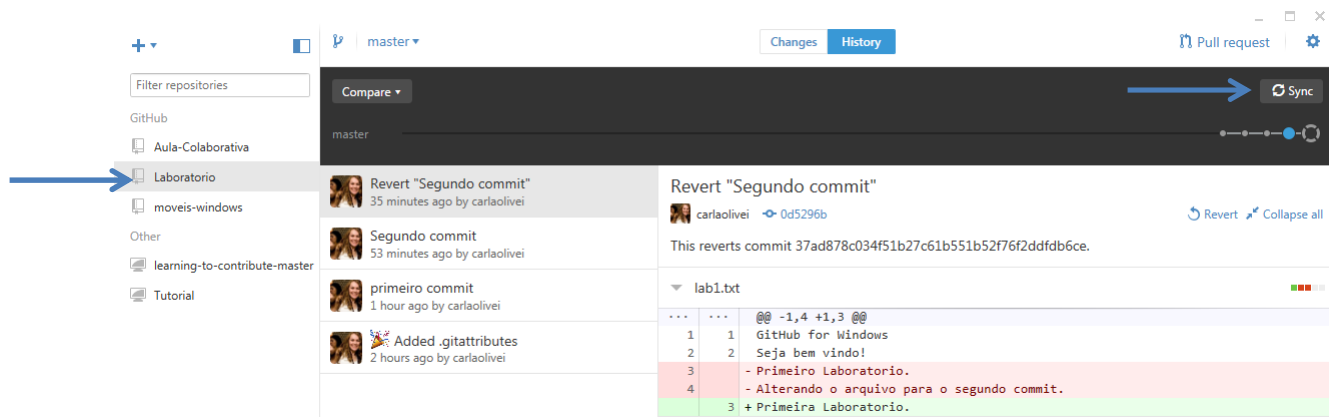


Figura 24 – Sincronização das alterações locais com o servidor remoto

3.8 Clonando um repositório do GitHub Web

Para clonar um repositório disponível no GitHub Web primeiro é necessário acessar o GitHub Web e pesquisar o repositório desejado. Neste exemplo vamos pesquisar por “Manual do GitHub” conforme mostra a Figura 25.

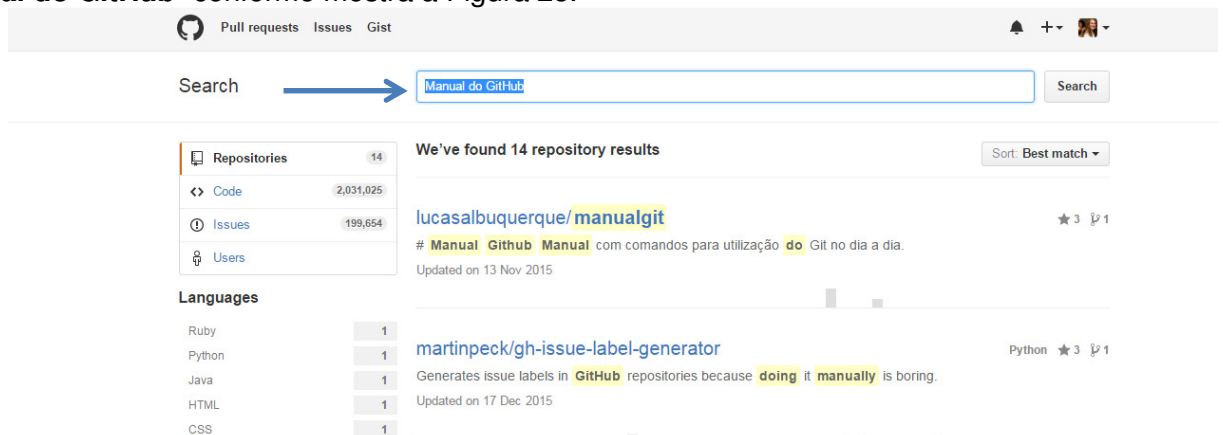


Figura 25 – Pesquisando um projeto no GitHub Web

Em seguida clique no repositório desejado. No nosso caso clique em **lucasalbuquerque/manualgit**. Ao clicar neste repositório será exibida a tela da Figura 26.

O próximo passo agora é clicar no botão **Fork**, no canto superior direito, para criar uma cópia desse repositório.



Figura 26 – Clonando um repositório no GitHub Web

Ao clicar no botão **Fork** será exibida a tela da Figura 27.

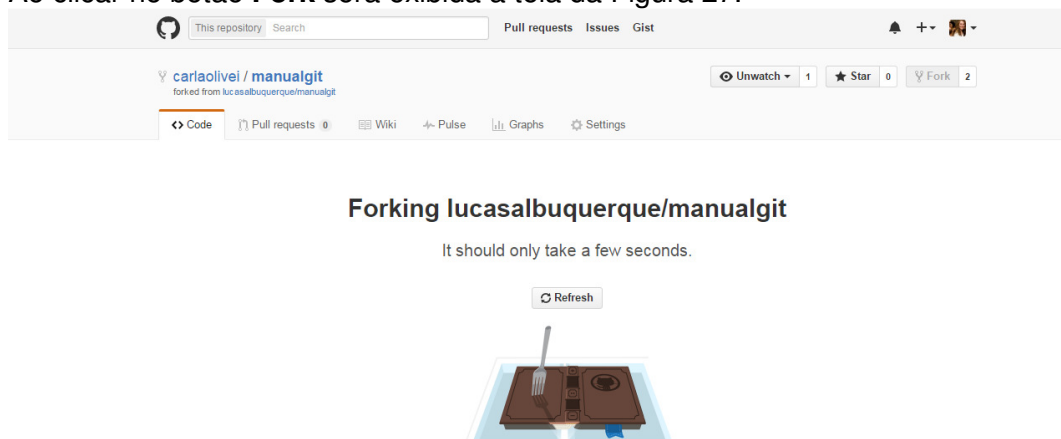


Figura 27 – Andamento do processo de clonagem de um repositório no GitHub Web

Para verificar se o repositório foi copiado corretamente acesse a página do seu usuário no GitHub Web. Filtre pela opção **Fork** e verá o repositório **manualgit** conforme mostra a Figura 28.

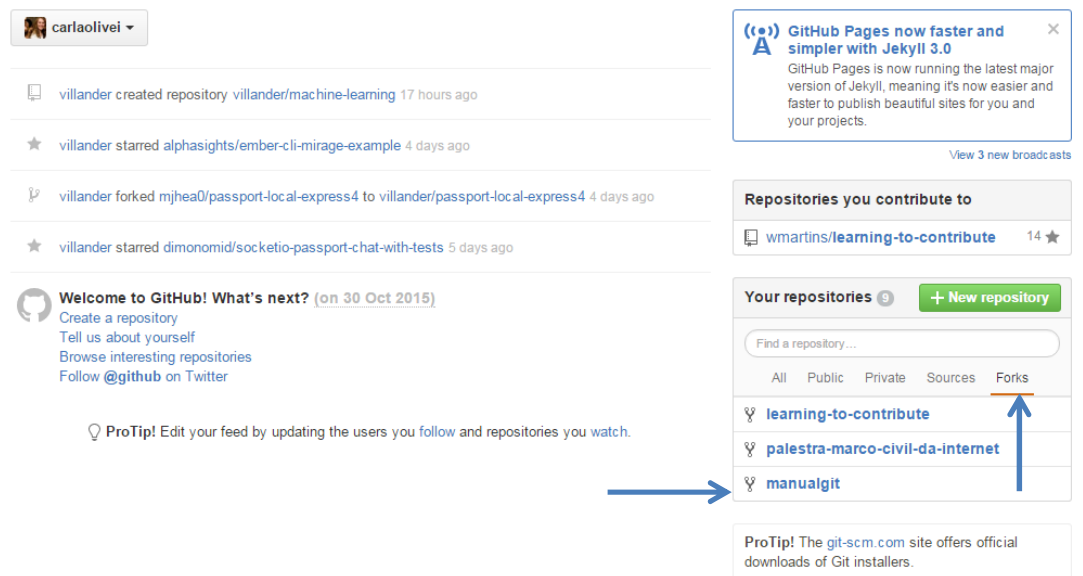


Figura 28 – Confirmando que a clonagem de um repositório no GitHub Web foi efetuada com sucesso

No GitHub for Windows clique no botão **+** e escolha opção **Clone**. Em seguida selecione o repositório **manualgit** e clique no botão **clone manualgit**.

Ao clicar no botão **clone manualgit** será exibida a tela abaixo para seleção do diretório. Mantenha o diretório padrão que GitHub e clique em **OK** conforme a Figura 29.

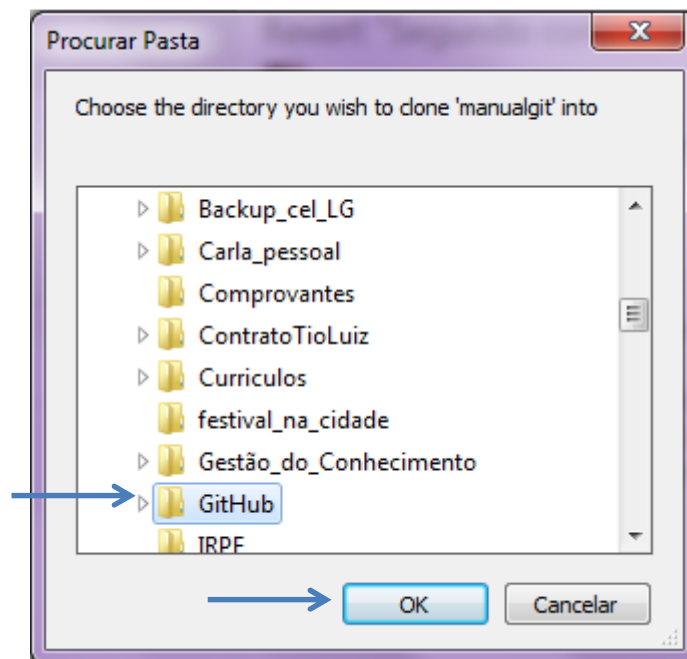


Figura 29 – Diretório padrão do GitHub for Windows

Após clicar em **OK** será exibida a tela da Figura 30.

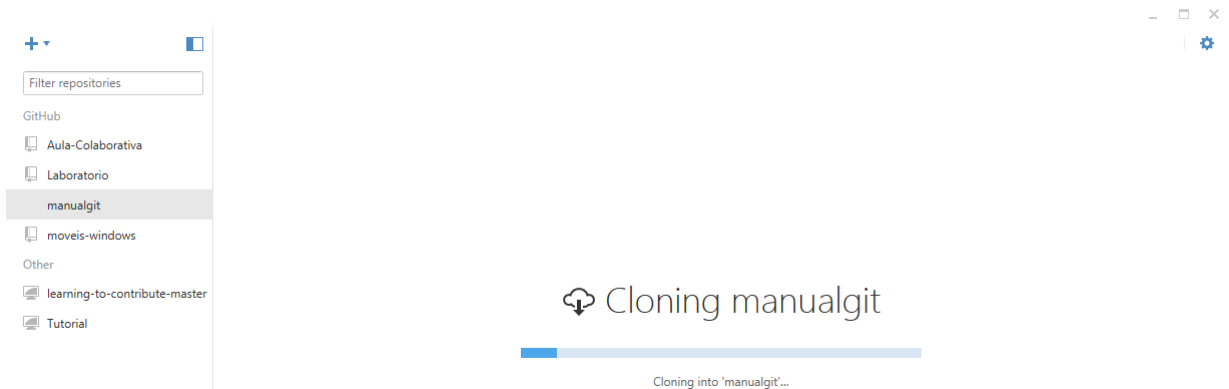


Figura 30 – Clonando um repositório do servidor remoto

Quando o processo de clonagem terminar será exibida a tela da Figura 31 com o novo repositório clonado.

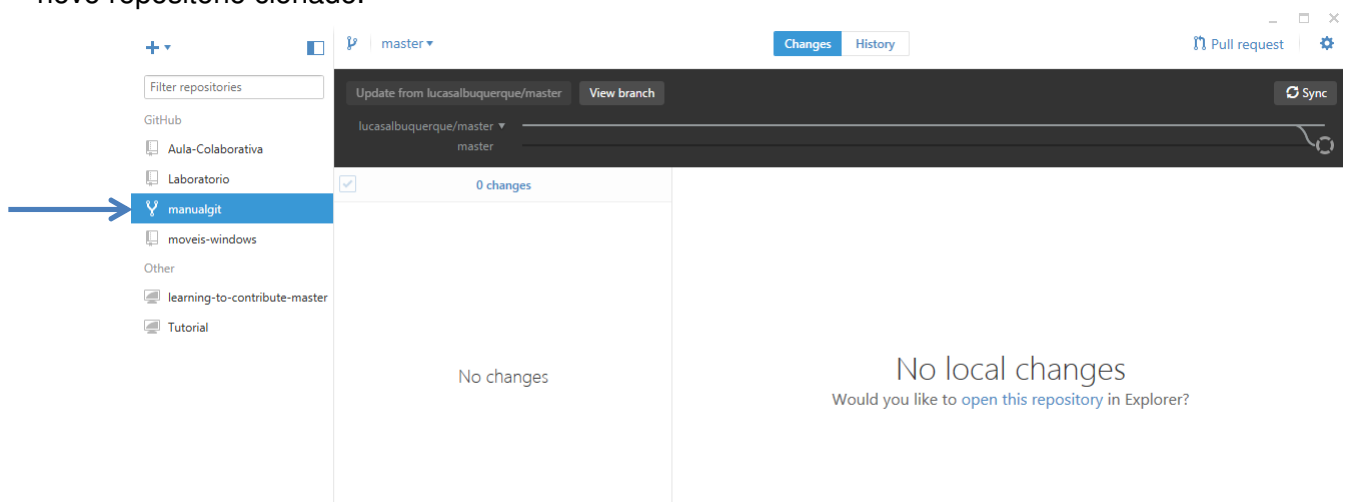


Figura 31 – Novo repositório clonado

Para confirmar que a clonagem foi realmente executada acesse o diretório C:\Users\nome\Documents\GitHub e verifique que foi criada a pasta **manualgit** conforme mostra a Figura 32.

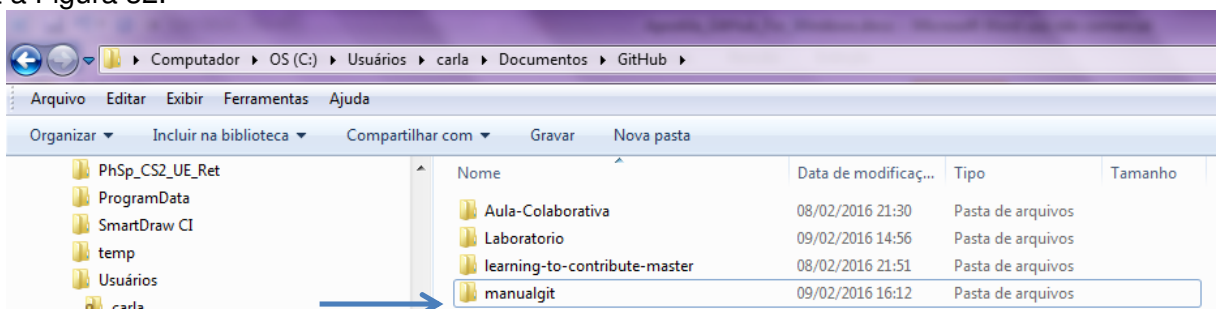


Figura 32 – Confirmando se a clonagem foi efetuada com sucesso

3.9 Trabalhando com branches

Também é possível trabalhar com branches no GitHub for Windows. Na tela principal existe um botão onde podemos gerenciar as branches do repositório conforme mostra a Figura 33.

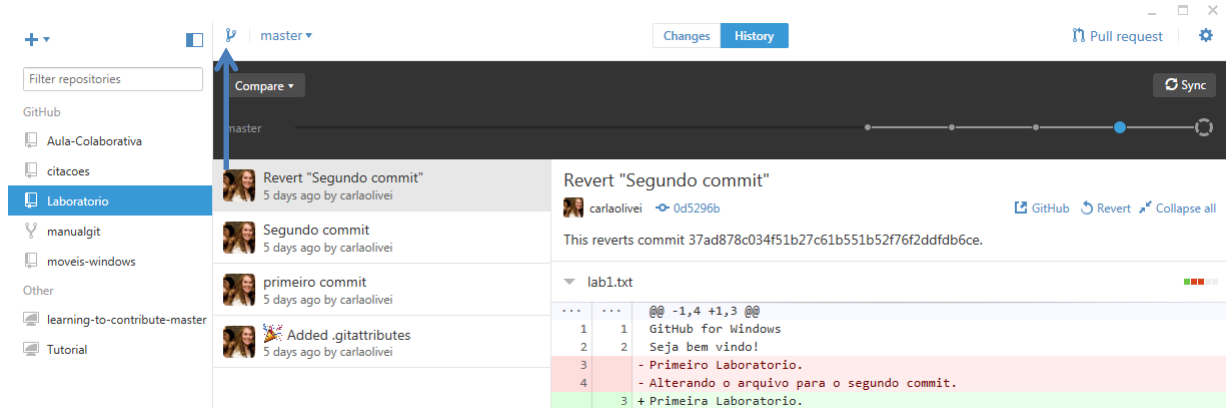


Figura 33 – Gerenciando branches

Vamos criar uma nova branch chamada **teste**, clicando no botão mencionado anteriormente, e em seguida digitar **teste** no campo de texto da tela. Em seguida confirmar clicando em no botão **Create new branch** conforme mostra a Figura 34.

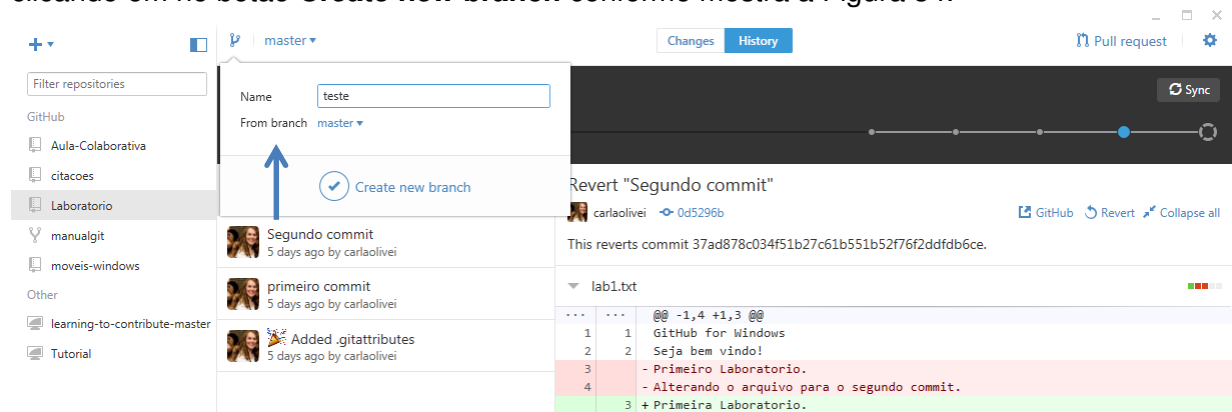


Figura 34 – Criando uma nova branch

Após a nova branch ser criada ela será automaticamente selecionada. Se clicar novamente no botão para gerenciar as branches, verá que agora existirão duas branches no repositório conforme mostra a Figura 35.

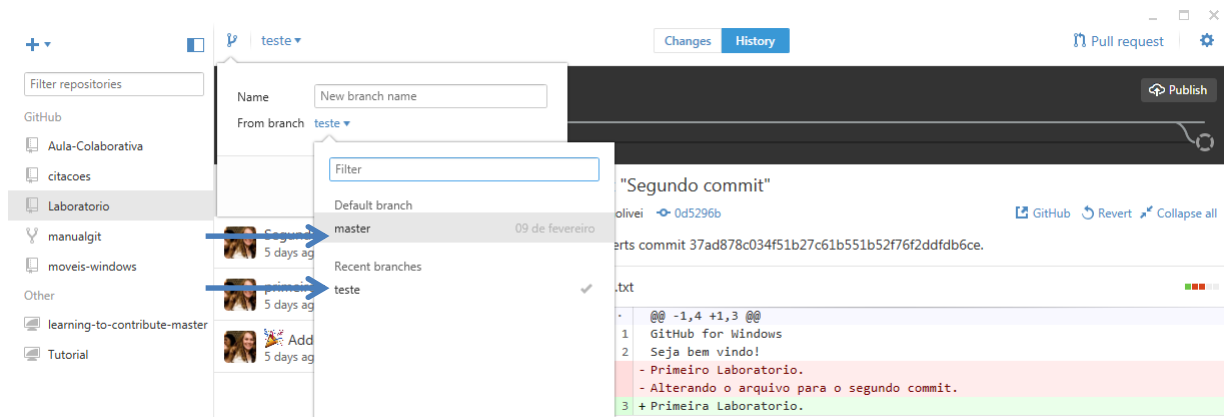


Figura 35 – Branches existentes

Vamos alterar novamente o arquivo **lab1.txt** e em seguida efetuar um novo commit, porém desta vez o commit deverá ser realizado na branch **teste**.

Altere o arquivo **lab1.txt** adicionando o seguinte conteúdo:

GitHub for Windows

Seja bem vindo!

Primeira Laboratorio.

Primeiro teste com branch.

Após efetuar a alteração no arquivo registre a mesma com um novo commit. Certifique-se antes que a branch **teste** está seleccionada como mostra a Figura 36.

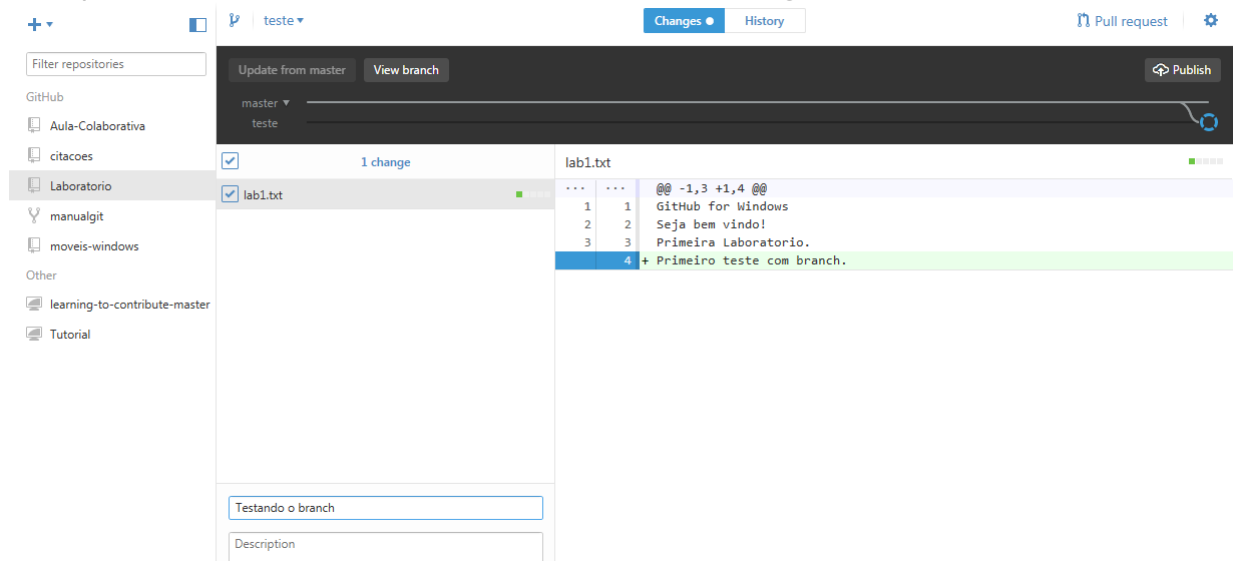


Figura 36 – Efetuando commit na nova branch criada

Observe que após efetuar o commit o mesmo é exibido na listagem de commits da branch **teste** (Figura 37).

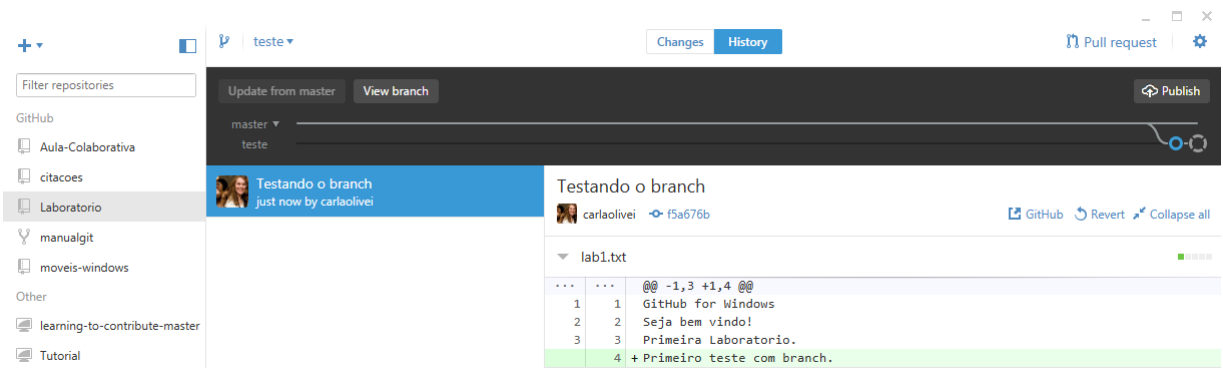


Figura 37 – Commit efetuado na nova branch criada

Porém, se você mudar para a branch **master** (Figura 38) o commit não será exibido, uma vez que ele foi efetuado apenas na branch **teste**.

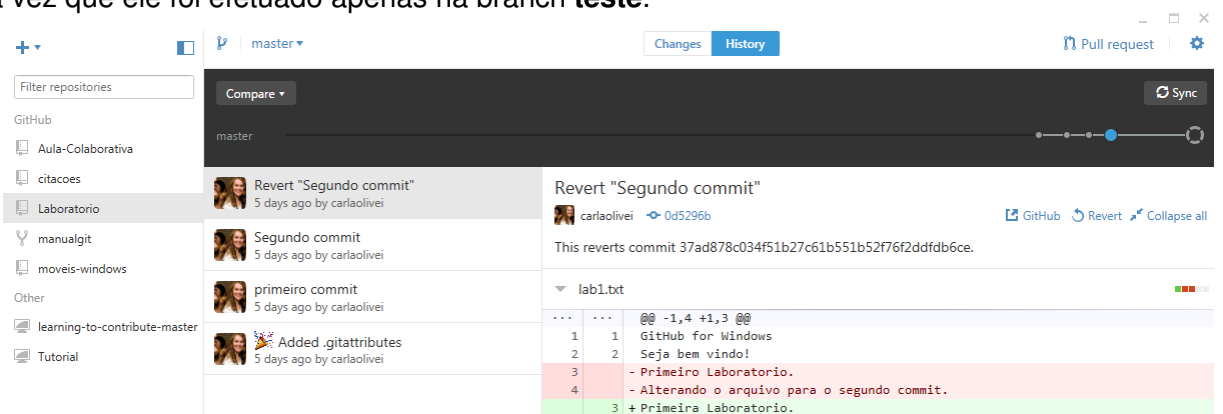



Figura 38 – Confirmando que o commit foi efetuado na nova branch criada

3.10 Efetuando o merge dos commits

No GitHub for Windows também é possível efetuar merges, para mesclar as alterações de duas branches diferentes. Para isso:

1. Próximo ao ícone , clique e selecione a branch que deseja efetuar o merge.

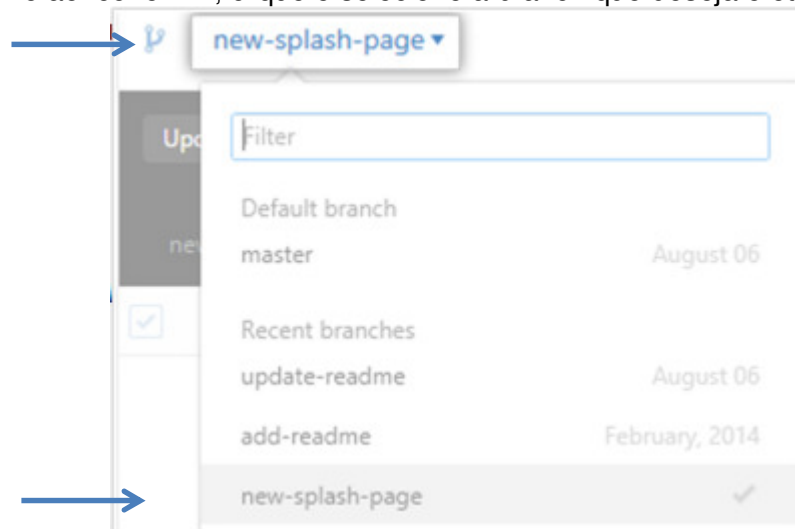


Figura 39 – Seleção da branch

2. No gráfico de comparação, clique em NOME-DA-BRANCH.

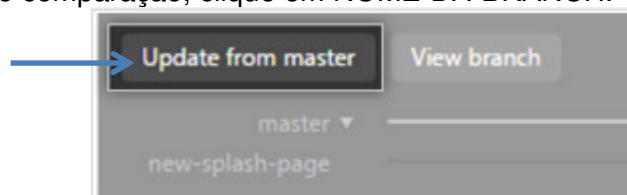


Figura 40 – Efetuando o merge





3. Clique em  Sync para atualizar o repositório remoto.



Figura 41 – Efetuando o merge

Segue abaixo tabela com a descrição dos ícones que aparecem no merge.

ICONE	DESCRIÇÃO
	Um commit que foi movido para o repositório remoto (normalmente chamado de origem/master).
	Indica quando um merge foi efetuado.
	Indica um commit que ainda não foi movido para o repositório remoto.

O manual oficial do GitHub for Windows para quem deseja contribuir com projetos colaborativos está disponível no link abaixo:

<https://help.github.com/desktop/guides/contributing/>

4 EXEMPLOS DE PROJETOS DE CIÊNCIAS SOCIAIS NO GITHUB

Seguem abaixo dois exemplos de projetos open source (software livre) do Institute for Quantitative Social Science (IQSS), da Universidade de Harvard, disponibilizados no GitHub.

4.1 Dataverse

A plataforma Dataverse é uma arquitetura de software livre para a publicação, citação, análise e preservação de dados de projetos de pesquisa. Permite um gerenciamento de dados seguro. O Dataverse dá suporte à criação de termos de uso e restrições para limitar o uso e acesso aos dados, além de cópias de segurança. Por facilitar a disseminação de dados e um compartilhamento efetivo, as equipes de pesquisa podem trabalhar unidas em um espaço para acompanhamento dos processos e das mudanças nos projetos no decorrer do tempo.

O projeto da Rede Dataverse é desenvolvido e mantido pelo Institute for Quantitative Social Science (IQSS), da Universidade de Harvard.

Este projeto está disponível no GitHub no link abaixo:

<https://github.com/IQSS/dataverse>

Para mais detalhes sobre o projeto Dataverse acesse o link abaixo:

<http://dataverse.org/about>

O link abaixo mostra exemplos de trabalhos de pesquisa na área de ciências sociais disponibilizados no Dataverse:

https://dataverse.harvard.edu/?q=&fq0=subject_ss%3A%22Social+Sciences%22&types=dataverses%3Adatasets&sort=dateSort&order=desc

4.2 Openscholar

O Openscholar é um projeto do Institute for Quantitative Social Science (IQSS), da Universidade de Harvard. É uma ferramenta onde é possível criar sites acadêmicos em questão de minutos devido a sua interface lógica e intuitiva, não requerendo nenhum conhecimento de programação. Cada site vem com uma suite de poderosas ferramentas que possibilitam ao professor criar e distribuir informações de maneira fácil e eficiente.

Este projeto está disponível no GitHub no link abaixo:

<https://github.com/openscholar/openscholar>

Para mais detalhes sobre o projeto Openscholar acesse o link abaixo:

<http://theopenscholar.org/pages/about-openscholar>

5 REFERÊNCIAS

FREIRE, Danilo. Git e GitHub: vantagens para sua pesquisa, 2014.

Disponível em: <https://socioisemetodos.wordpress.com/2014/08/14/git-e-github-vantagens-para-sua-pesquisa/>

Laboratório de Cultura digital. A Rede Livre e o desafio do desenvolvimento tecnológico colaborativo, 2013.

Disponível em: <http://labculturadigital.org/2013/11/24/a-rede-livre-e-o-desafio-do-desenvolvimento-tecnologico-colaborativo/>

SOUZA, Kleber Sacilotto e LEITÃO, Breno Henrique Leitão. Como Contribuir para o kernel Linux. IBM, 2012.

Disponível em: http://www.ibm.com/developerworks/br/local/linux/kernel_contribution/