This week we hear from Toshi Takeuchi about how to take advantage of MATLAB's recent improvements to Git integration. Toshi is a Senior Marke Manager at MathWorks.

Quick Introduction to Git with MATLAB

by Toshi Takeuchi

One of the new R2014b features that deserves your attention is Git integration. Git is a source control system (also known as version control or sou code management system) that enables collaborative software development. Why does that matter to you? Programming is an essential skill in ma technical fields even outside computer science, and some universities now offer software carpentry workshops to enhance coding skills for researcl Source control is one of those essential skills in software carpentry.

Until now, you may have tinkered alone with code you needed for your project. However, there are other people who may be working on similar problems and they may be writing similar programs. Source control enables you to work with other people so that you don't have to do it all alone. Collaboration lets you be more productive in other aspects of your project.

Even if you don't care about such collaboration, wouldn't it be cool to share your personal project and see other people using it? They may even fix bugs and improve your code for you!

GitHub is one of the most popular websites that host Git repositories. The best place to share your MATLAB projects is File Exchange because of it popularity with the MATLAB user community. And guess what – File Exchange is integrated with GitHub! Now you see the connection?

Basic terminology

What is a Git repository? A repo (repository) is a directory that holds your source code and any associated files. Local repos are on your local drive the remote repos are on GitHub or other hosts, and you sync the local repos to remote repos as you write your code. You can start with either the lo or remote repos, but in this example I am going to start with a remote repo.

The process looks like this for a single developer:

1. Create or fork a repo on GitHub
2. Clone the repo to your local drive – this is your local repo
3. Add your files to the local repo
4. Sync your local repo to remote repo
5. Repeat this process to keep your source code in sync as you write more code
6. Share your GitHub repo on File Exchange

What is forking?

When you talk about Git, you cannot go without mentioning "forking". In the simplest terms forking means copying someone else's public repo on th remote server, rather than starting a repo from scratch. In practice forking is used as a way to contribute to the existing projects or to start a new pro using an existing project as a starting point. Once you make changes to your forked project, you can send a merge request to the original develope and your changes may be accepted and incorporated into the main project.

Forking enables a flexible distributed style of collaboration and number of forks you have on your project acts as a measure of popularity – similar t count of likes or followers on Facebook or Twitter. The social aspect of forking is an interesting topic on its own, but we need to skip it for this post.

Getting ready

Signing up with GitHub is easy – just click the *sign up* button on the homepage and follow instructions. A free account would do for now. You also n to download and install Git. Even though GitHub has GUI apps for Windows and Mac, you need to set up the command-line tool for use with MATL. You also want to follow these instructions for registering binary files with Git.

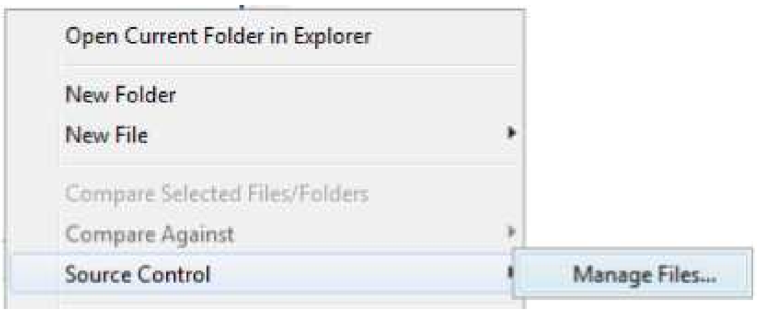
Creating a repo

Creating a repo on GitHub is very easy – just follow these instructions. From this point on I will assume you named your repo *Hello-World* and initia it with a README file. Please note that you can only create a public repo with a free account.

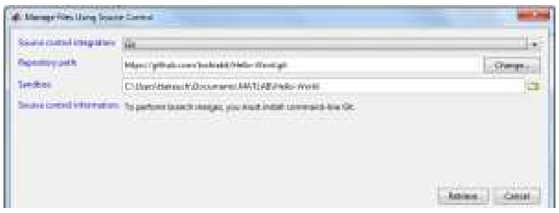
Cloning the repo to your local drive with MATLAB

Until recently, you needed to use the command line tool for this step, but starting with R2014b we can just use MATLAB's Current Folder window. N more Git commands like `git init`, `git status`, `git add`, or `git commit`!

Open your copy of MATLAB and create an empty folder. Right-clicking the empty space in the Current Folder window to bring up a contextual menu and select *Source Control* > *Manage Files*.



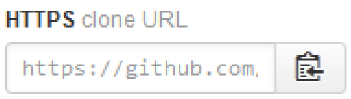
This will open a new dialog box: *Manage Files Using Source Control*.



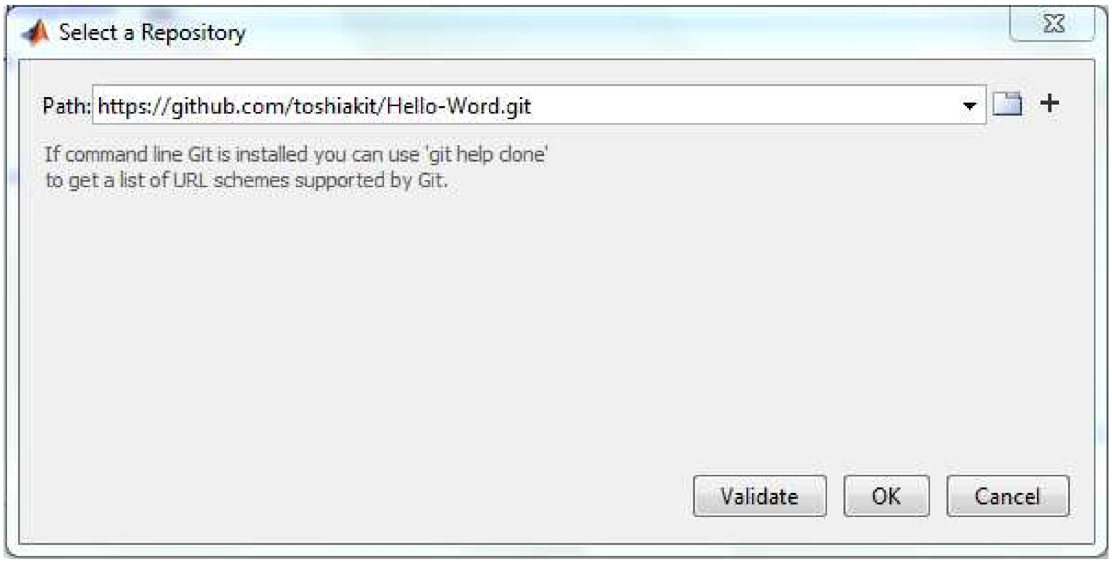
Source control integration: Git
Repository path: https://github.com/toshiakit/Hello-Word.git
Sandbox: C:\Users\[illegible]\Documents\MATLAB\Hello-World
Source control information: To perform branch merges, you must install command-line Git.

1. For *Select control integration*, choose *Git*
2. For *Repository path*, click *Change*

You now see a new dialog box: *Select a Repository*. Copy and paste the URL of the remote repo you just created. You can find the URL in the right sidebar of your new repo on GitHub.



You choose either SSH or HTTPS depending on how you setup your authentication on GitHub.



Path: https://github.com/toshiakit/Hello-Word.git

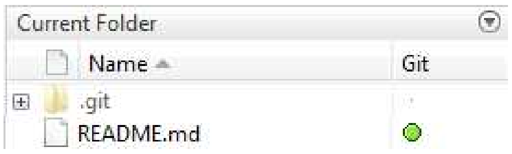
If command line Git is installed you can use 'git help clone'
to get a list of URL schemes supported by Git.

Click *Validate*. You may be asked for your login password for authentication. You can close the dialog box when your path is validated.

Back in *Manage Files* dialog box, the *sandbox* should be already set to your current folder. All you need to do now is hit *Retrieve*.

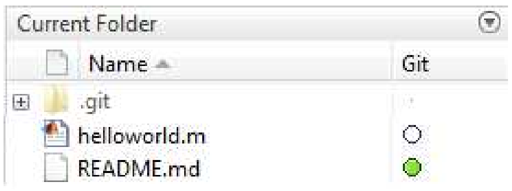
You have now successfully cloned the remote repo to your local drive. Check your Current Folder window. You should see just one file – README.

but with a green circle next to it. This is just a text file but you can apply wiki-like syntax called markdown to make it appear like a regular web page when viewed on GitHub. README serves as the front page of your repo on GitHub.



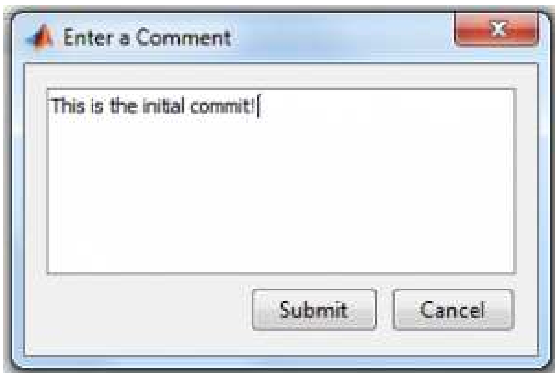
Adding files to your local repo

Let's add a new MATLAB script file *helloworld.m*. It will appear with a blank circle – it means it is not added to Git source control yet. To add it to Git right-click on the file and select *Add to Git*. The empty circle changes to "+" symbol. When you modify a file already under source control, the symb becomes a blue square.



Taking a snapshot with commit

You can continue editing files as you like, but at some point, you want to take a snapshot of the edits you made. That's when you do a commit. You select any empty space in the Current Folder window to bring up the contextual menu and select *Commit to Git Repository*. This will bring up a dial box where you can add your comment about the changes you made since the last commit. Comments will be helpful to keep to track of your chang and revert back to earlier commits if necessary.



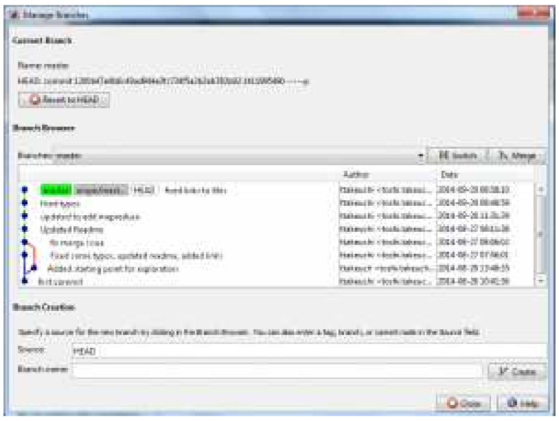
This is the inital commit!

Synching your local repo to remote repo

When you commit, the snapshot is saved in the local repo, but it is also a good idea to mirror the changes to the remote repo as well. To do so, brin the contextual menu by right-clicking an empty space in the Current Folder window and select *Push*. That will push your changes to the remote rep You may need to enter your password.

Branching and Merging

The real power of source control comes from the ability to create multiple branches from your project. By default, you have a branch called "master your repo. You can create a new branch from the master branch, makes changes, and then merge those changes back to the master. This mechan is used for working on new experimental features without affecting the working code on the master. You can branch and merge in MATLAB but the details are beyond the scope of this post.



Closing

If you have been curious about Git but put off by its complicated command line interface, Git integration in R2014b makes Git much more pleasant
approachable. I hope this quick introduction motivates you to take advantage of this new feature. When you do, please don't forget to post your pro
to the File Exchange. To learn more about Git, it is actually helpful to start with reviewing the underlying concept about how Git works.