Using GitHub a learn.sparkfun.com tutorial

Available online at: http://sfe.io/t11

Contents

- What's a Repo?
- Download ZIP
- Managing Repos
- Pull Requests
- Wiki and Issues

What's a Repo?

Repo is short for **repository**. Think of a repo as a folder of files and all the changes made to the files are recorded. If there's ever a problem with a file you can go back in time to figure out what changes you made. The most common use for repos are for managing large code projects but repo tracking is good for a variety of applications in the hardware world including PCB layouts, firmware, datasheets and documentation.

For example, let us imagine someone has created an Arduino sketch to demonstrate how to read an analog sensor.

```
byte myValue = 0;
myValue = analogRead(A0);
```

There's a couple improvements that could be made to this code (analogRead returns an int not a byte!). If the code was just a file on someone's website you'd have to send them an email and suggest the improvements. This is a bit tedious, and when a project gets longer than a few lines of code, email is not a viable way to collaborate on projects. GitHub allows one person to manage their own projects (also called revision or version control) and it also allows lots of people to work together on large projects (source code management).

What is this git thing?

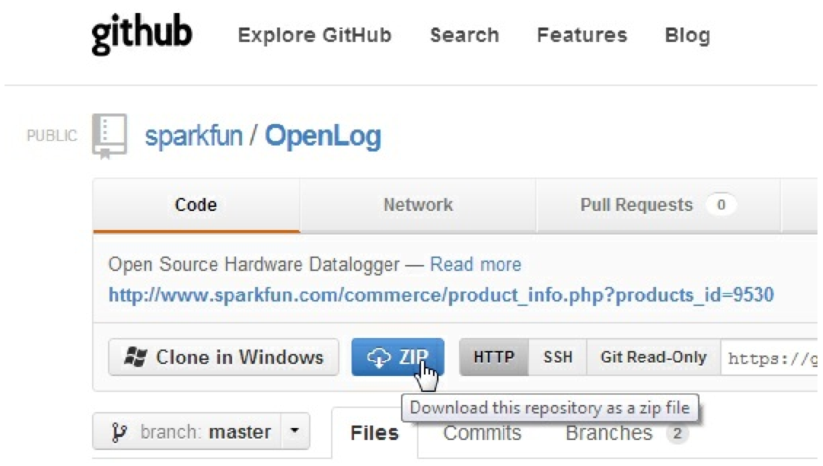
Git is a software management tool designed for extremely large coding projects (such as Linux). Because the majority of work that we do at SparkFun is on smaller projects, we use only a fraction of its capabilities. While Git uses a command line interface, GitHub was created to give Git a slicker looking web interface. Furthermore, GitHub released a Windows GUI (graphical user interface) that makes moving repos around even easier.

We're going to cover a few things in this tutorial:

- **Download ZIP** - How to get something from GitHub
- **Manage** - How to manage your own stuff on GitHub
- **Pull Requests** - How to improve something on GitHub
- **Wiki and Issues** - There is lots more on GitHub including Wikis and Issue Tracking

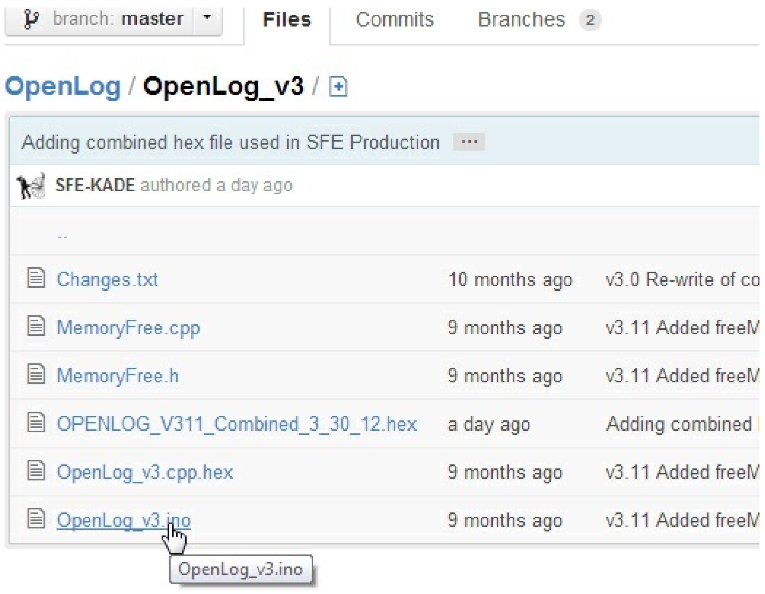
Download ZIP

Just need to get some code from a public GitHub project? Here's how to get something from GitHub:



Open Source Hardware Datalogger — Read more
http://www.sparkfun.com/commerce/product_info.php?products_id=9530

On every project there's an easy to use 'Download as ZIP' button that will give you the entire contents of the project. This is useful if you just need to grab and go (you leech you). However, this is not the correct way if you plan to contribute back.



Adding combined hex file used in SFE Production

SFE-KADE authored a day ago

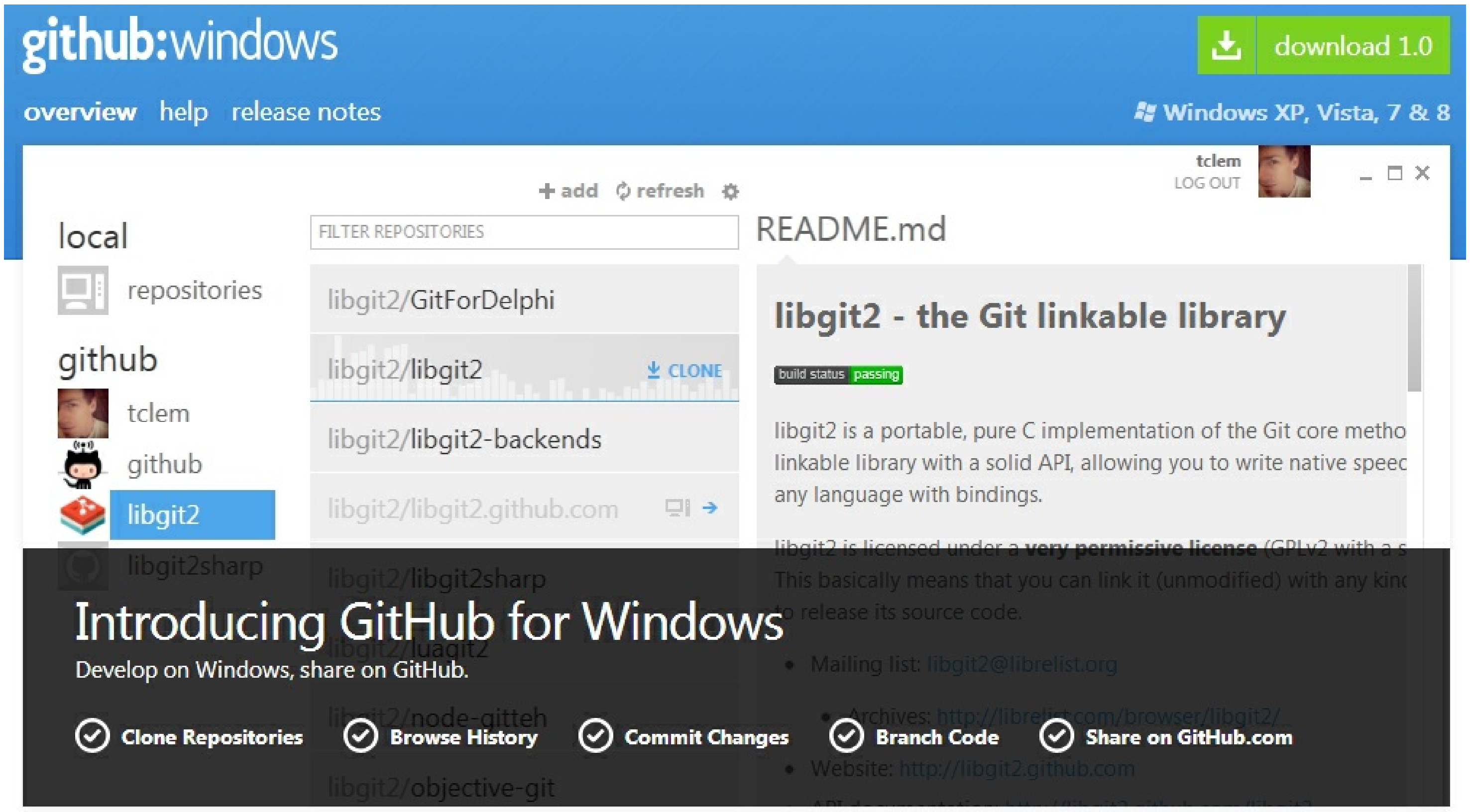
Changes.txt	10 months ago	v3.0 Re-write of co
MemoryFree.cpp	9 months ago	v3.11 Added freeM
MemoryFree.h	9 months ago	v3.11 Added freeM
OPENLOG_V311_Combined_3_30_12.hex	a day ago	Adding combined
OpenLog_v3.cpp.hex	9 months ago	v3.11 Added freeM
OpenLog_v3.ino	9 months ago	v3.11 Added freeM

Right-clicking won't work

Note: If you're navigating around a project and see a file you'd like to grab, right clicking and selecting save-as will not get you the file. You will get an HTML file instead of the raw file you might be expecting. You should either use the ZIP download button or clone the repo to a local folder. Keep reading! We'll show you how.

Managing Repos

To start, you'll need to create an account on GitHub. Don't worry, it's free for regular users.



libgit2 is a portable, pure C implementation of the Git core metho linkable library with a solid API, allowing you to write native speec any language with bindings.

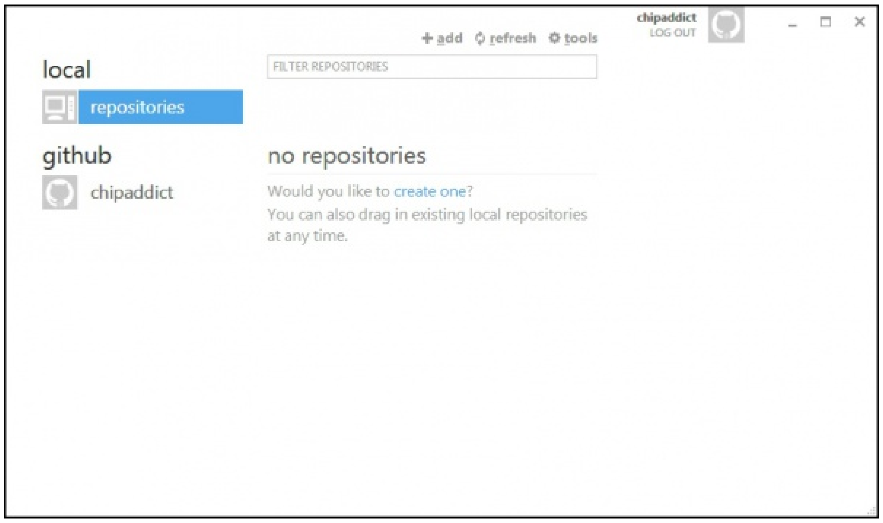
libgit2 is licensed under a **very permissive license** (GPLv2 with a s This basically means that you can link it (unmodified) with any kinc to release its source code.

- Mailing list: libgit2@librelist.org
    - Archives: http://librelist.com/browser/libgit2/
- Website: http://libgit2.github.com

There are plenty of Git clients for Mac and Linux but this tutorial will focus on Windows. If you're a Windows user I highly recommend you try out the GitHub Windows GUI. The following tutorial will focus on this client so please download and install the software.

During installation, it will ask you for things such as your login information, name and email address. All of this information will be associated with the commits you make.

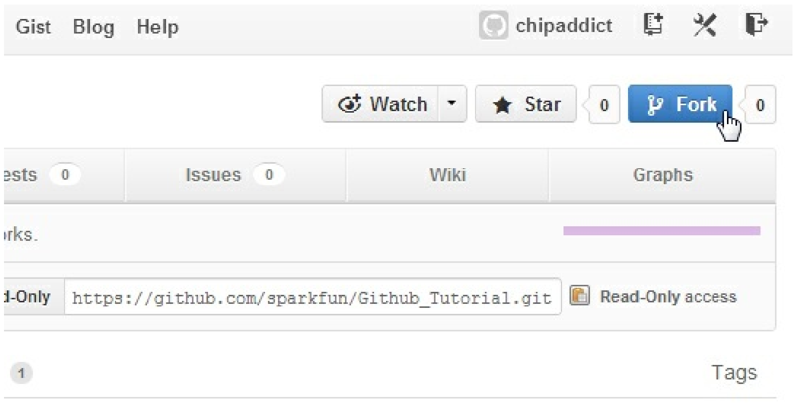
During installation, it may ask you to scan for repos on your computer. I recommend you *don't* do this. The scan can take a long time and because you're reading this tutorial, you probably won't have any repos.



no repositories

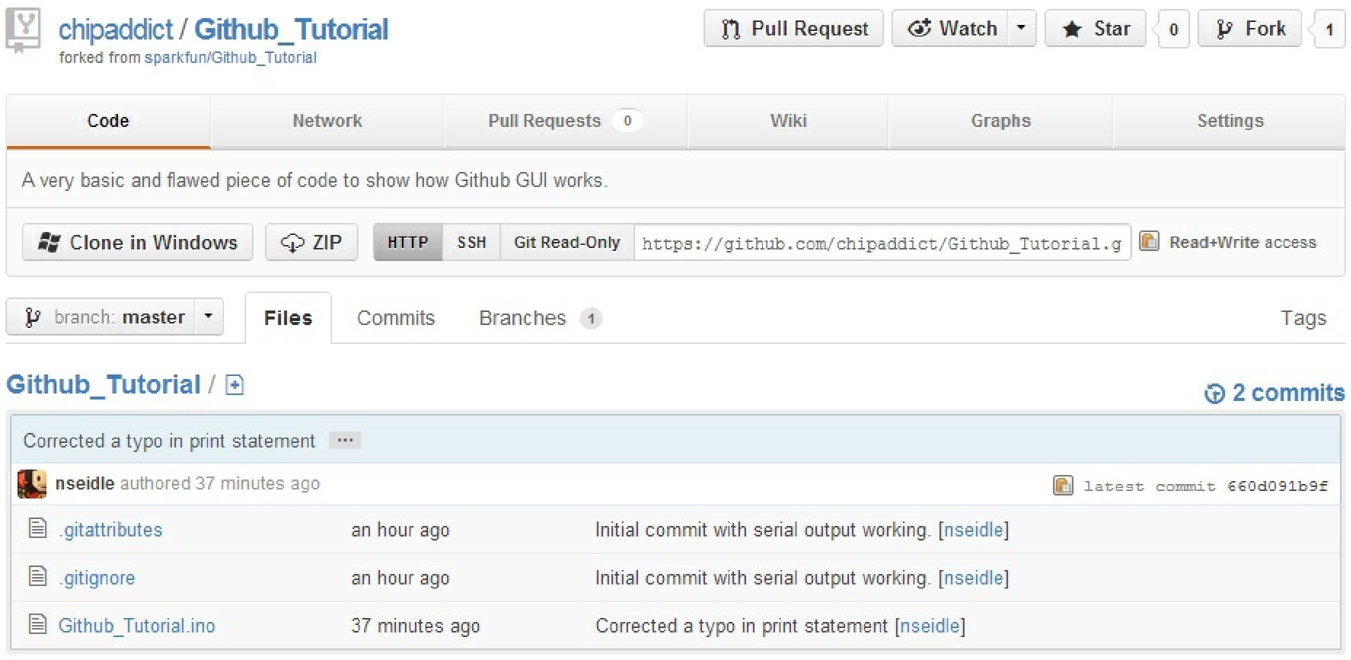
Would you like to create one?
You can also drag in existing local repositories at any time.

Once the GUI is launched you'll probably not have any local repos. Let's go get one from the SparkFun GitHub account. Let's grab the GitHub Tutorial repo.



https://github.com/sparkfun/Github_Tutorial.git

Once you are logged in to GitHub, click on the 'Fork' button.



A very basic and flawed piece of code to show how Github GUI works.

https://github.com/chipaddict/Github_Tutorial.g

Github_Tutorial /

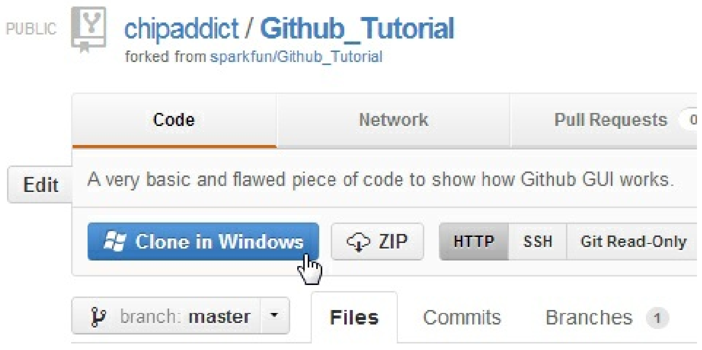
Corrected a typo in print statement

nseidle authored 37 minutes ago — latest commit 660d091b9f

.gitattributes	an hour ago	Initial commit with serial output working. [nseidle]
.gitignore	an hour ago	Initial commit with serial output working. [nseidle]
Github_Tutorial.ino	37 minutes ago	Corrected a typo in print statement [nseidle]

We have now created a ‘fork’ or a copy of this repo and is located within your GitHub account. Note the words in the upper left corner of the window “**chipaddict/Github_Tutorial**” and the words underneath “**forked from sparkfun/Github_Tutorial**”. This shows that you have this project on your account (your account name will be different).

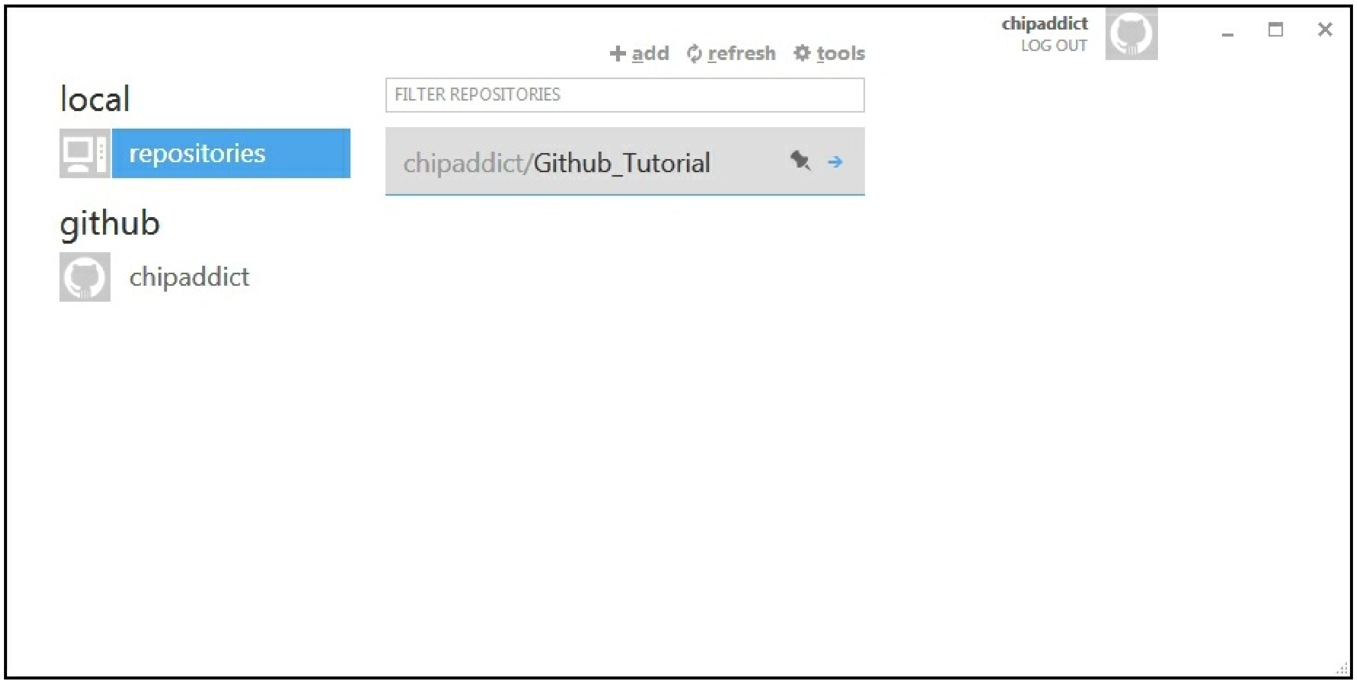
Now you can make lots of changes to this repo without affecting the original project. This is helpful if someone has some example code that is close to what you need but needs lots of modifications for your plans.



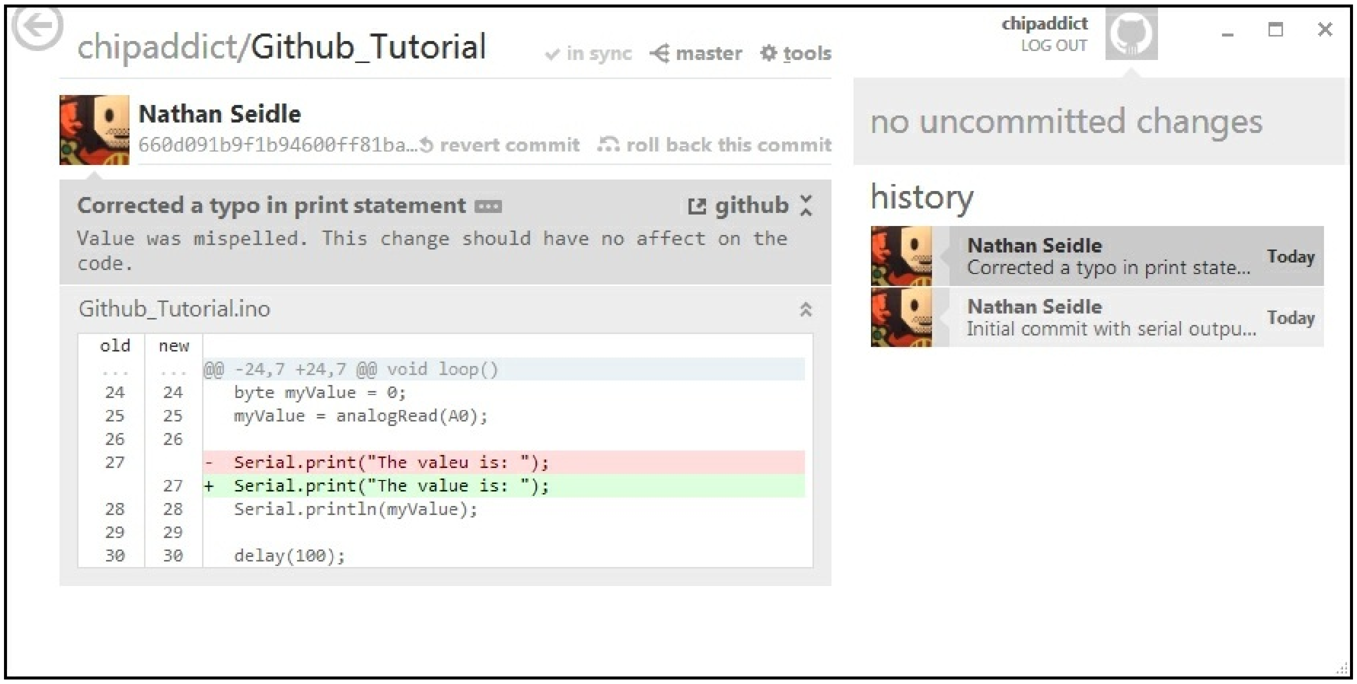
A very basic and flawed piece of code to show how Github GUI works.

Now that you have your own copy of the project online, click on the ‘Clone in Windows’ button. If you are not logged in, it may take you to the Github Windows page.

Windows may ask you for permission to allow the link to launch and use the Github software. This is ok. The GitHub GUI will open and a download will begin.



Oh! Look at that. The GUI has downloaded the repo to a local folder. Double click on the gray ‘sparkfun/Github_Tutorial’ box.

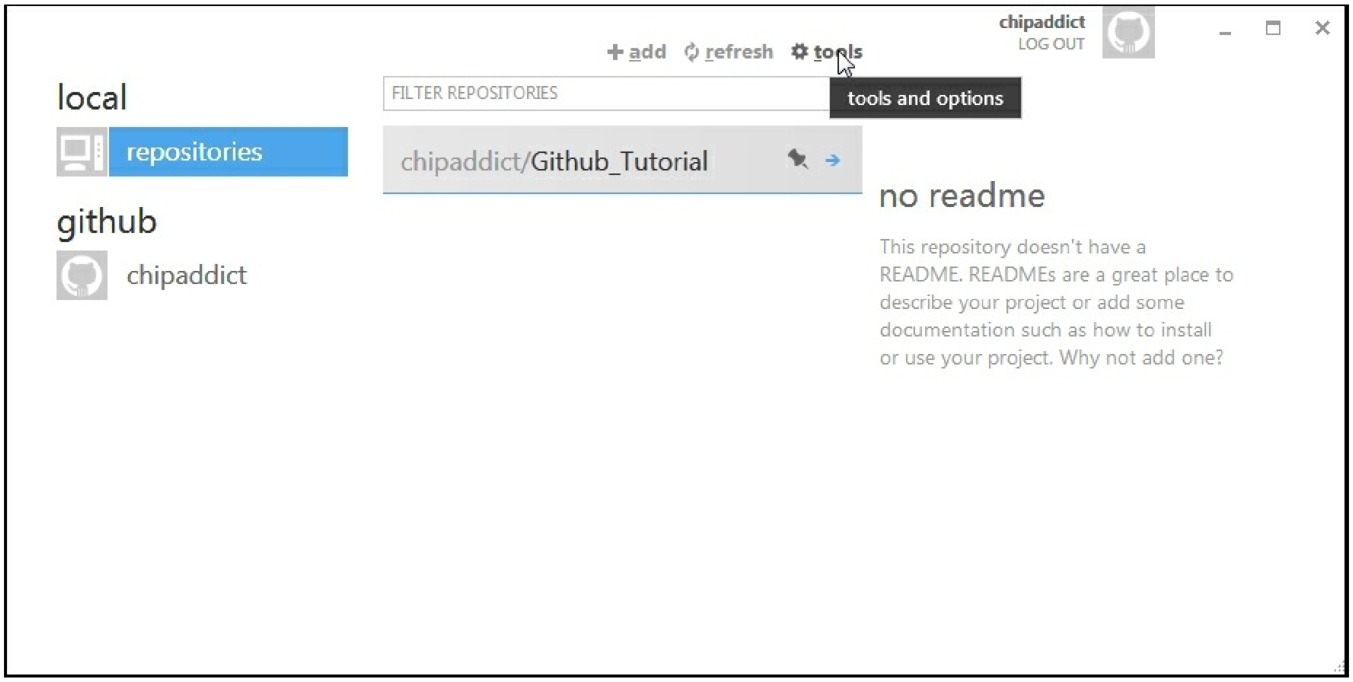


Value was mispelled. This change should have no affect on the code.

```
old  new
...  ...  @@ -24,7 +24,7 @@ void loop()
24   24      byte myValue = 0;
25   25      myValue = analogRead(A0);
26   26
27        -  Serial.print("The valeu is: ");
     27   +  Serial.print("The value is: ");
28   28      Serial.println(myValue);
29   29
30   30      delay(100);
```

Here we can see what has been going on with this repo. We can see that during the last commit, there was a comment ‘Corrected a typo in print statement’ and below that we can see the lines of code that were altered - red was removed, green was added. We can do lots of fun stuff like ‘revert commit’ and ‘roll back this commit’ but for now, let’s see how revision tracking works.

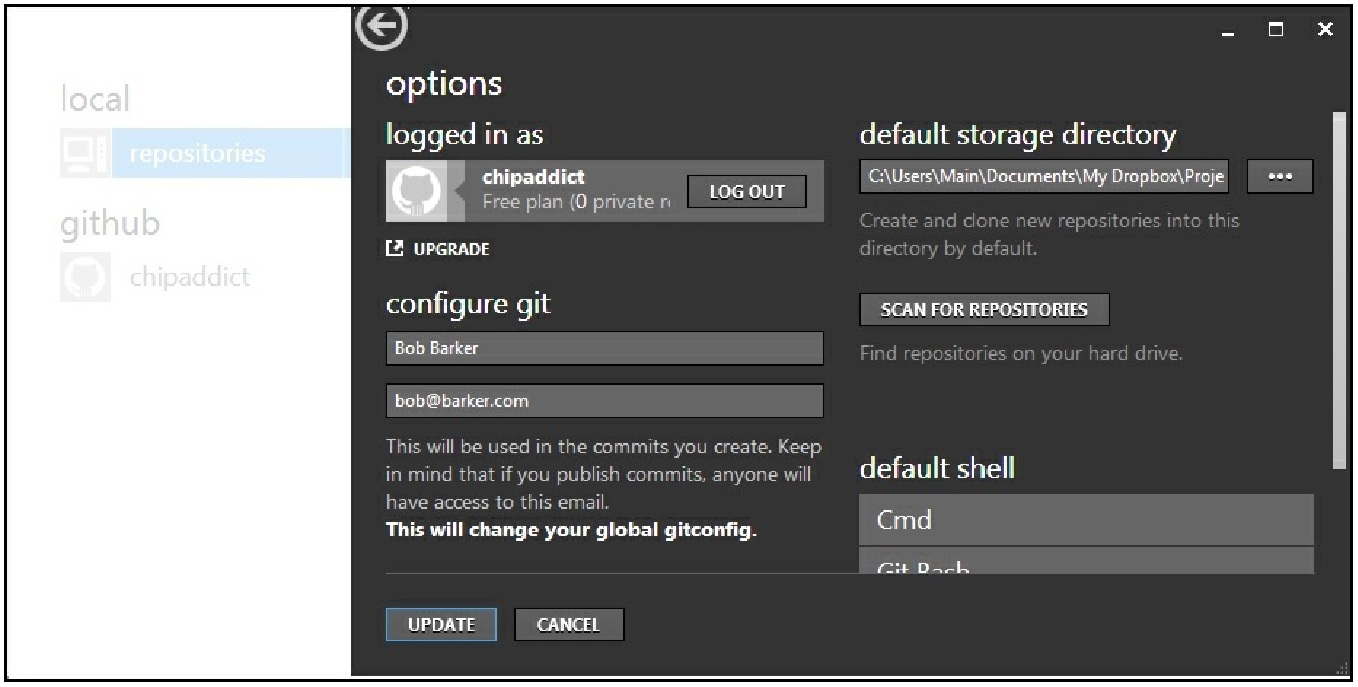
Click on the back arrow in the upper right corner to return to the main GitHub menu.



no readme

This repository doesn't have a README. READMEs are a great place to describe your project or add some documentation such as how to install or use your project. Why not add one?

From the main window, click on ‘tools’ and then ‘options’.



Bob Barker

bob@barker.com

This will be used in the commits you create. Keep in mind that if you publish commits, anyone will have access to this email.
This will change your global gitconfig.

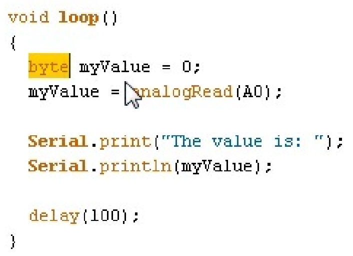
C:\Users\Main\Documents\My Dropbox\Proje

Create and clone new repositories into this directory by default.

Find repositories on your hard drive.

You will be able to find where your repositories are being stored under the ‘default storage directory’. I changed this to store my repositories in a Dropbox folder. I use Dropbox liberally in conjunction with GitHub so that I can work on projects across devices and then push code up to GitHub once the project has reached a level of stability.

You should now know where your repos are stored. Navigate to that directory and open up the *Github_Tutorial.ino* file.

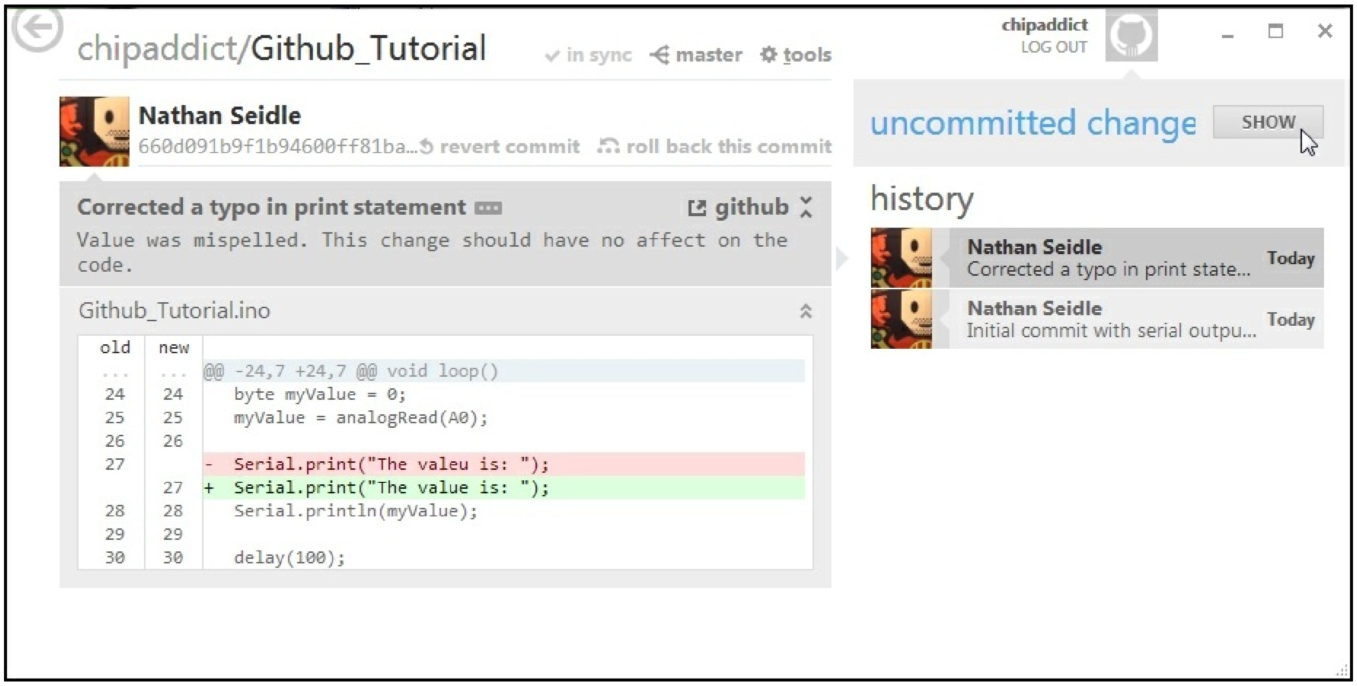


```
void loop()
{
  byte myValue = 0;
  myValue = analogRead(A0);

  Serial.print("The value is: ");
  Serial.println(myValue);

  delay(100);
}
```

From the Arduino IDE or Windows Notepad let's correct the variable declaration from **byte** to an **int**. Save the modification and return to the GitHub GUI:



Nathan Seidle
660d091b9f1b94600ff81ba...

Corrected a typo in print statement

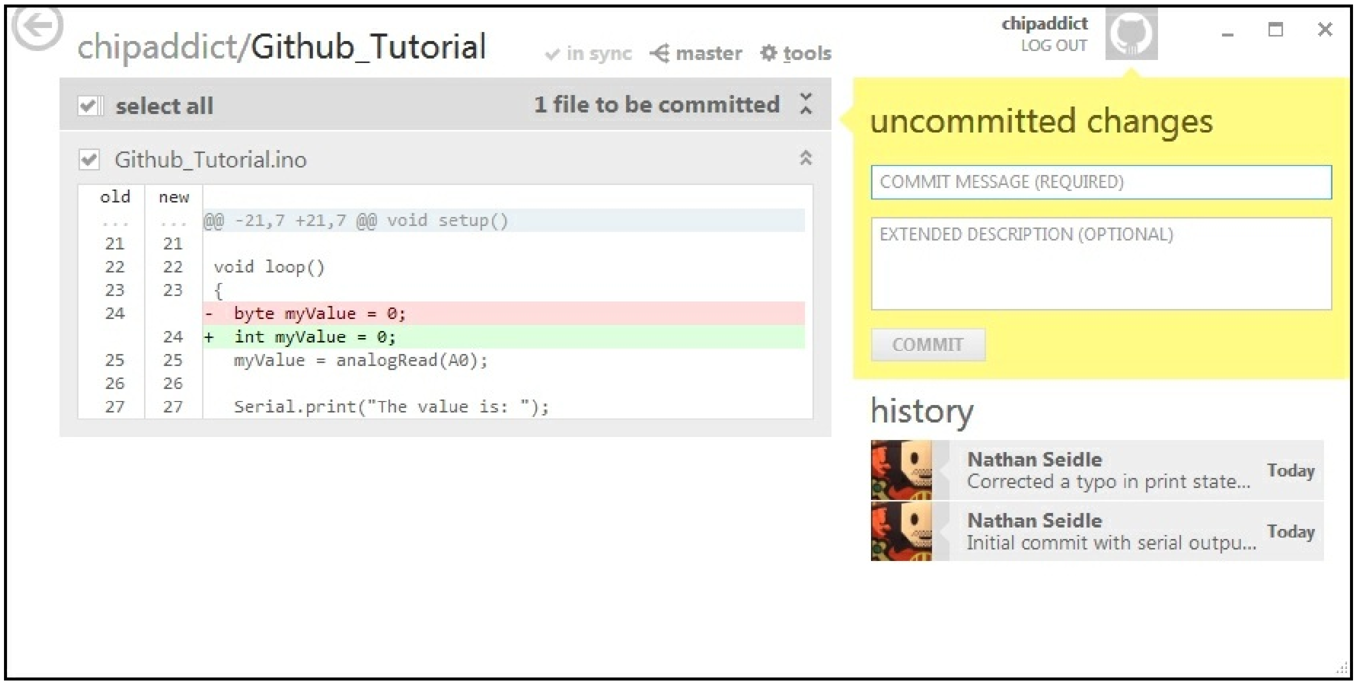
Value was mispelled. This change should have no affect on the code.

Github_Tutorial.ino

```
old  new
...  ...  @@ -24,7 +24,7 @@ void loop()
24   24      byte myValue = 0;
25   25      myValue = analogRead(A0);
26   26
27        -  Serial.print("The valeu is: ");
     27   +  Serial.print("The value is: ");
28   28      Serial.println(myValue);
29   29
30   30      delay(100);
```

Nathan Seidle Corrected a typo in print state...	Today
Nathan Seidle Initial commit with serial outpu...	Today

GitHub has noticed that a file has changed. Click on the 'Show' button.



Github_Tutorial.ino

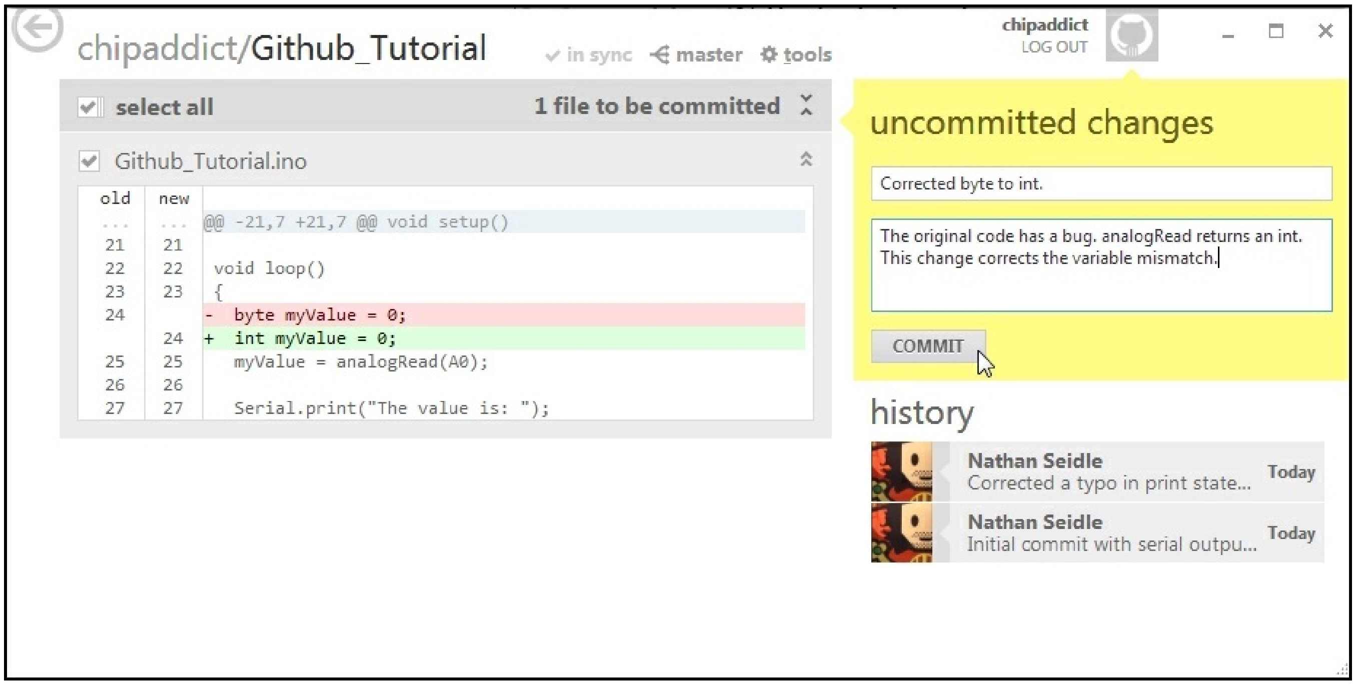
```
old  new
...  ...  @@ -21,7 +21,7 @@ void setup()
21   21
22   22   void loop()
23   23   {
24        -  byte myValue = 0;
     24   +  int myValue = 0;
25   25      myValue = analogRead(A0);
26   26
27   27      Serial.print("The value is: ");
```

Nathan Seidle Corrected a typo in print state...	Today
Nathan Seidle Initial commit with serial outpu...	Today

Wow! The small change we made is now nicely highlighted.

Now let's talk about how repos work. You have a **local working copy**, a **local repo**, and a **global repo**.

Local working copy: You generally write code, layout PCBs, and hack on documentation on your local computer on a local copy. Throughout the day you would use the GitHub window to 'commit' these changes to the a local repo. The changes you've made throughout the day are *not* known to the world, only to your local repo on your local computer.



```
old  new
...  ...  @@ -21,7 +21,7 @@ void setup()
21   21
22   22    void loop()
23   23    {
24        -  byte myValue = 0;
     24   +  int myValue = 0;
25   25      myValue = analogRead(A0);
26   26
27   27      Serial.print("The value is: ");
```

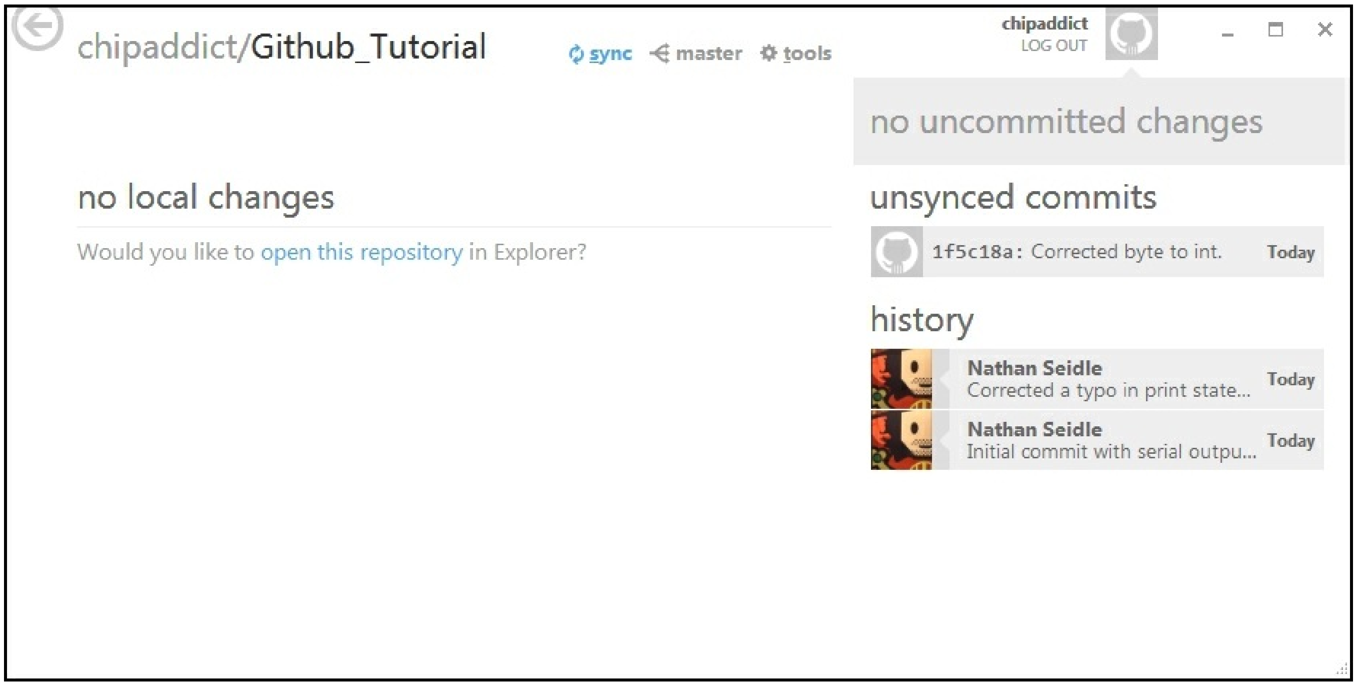
Corrected byte to int.

The original code has a bug. analogRead returns an int. This change corrects the variable mismatch.

Nathan Seidle	Corrected a typo in print state...	Today
Nathan Seidle	Initial commit with serial outpu...	Today

Local repo: Now let's commit the change we've made to our local repo. We must comment about what we changed before we can commit it. Thoroughly describe what you did and then hit commit.

As a general rule, try to commit small changes, frequently. If you wait 4 months between commits it is going to be very difficult for you to remember why you changed 5 lines of a subfunction.



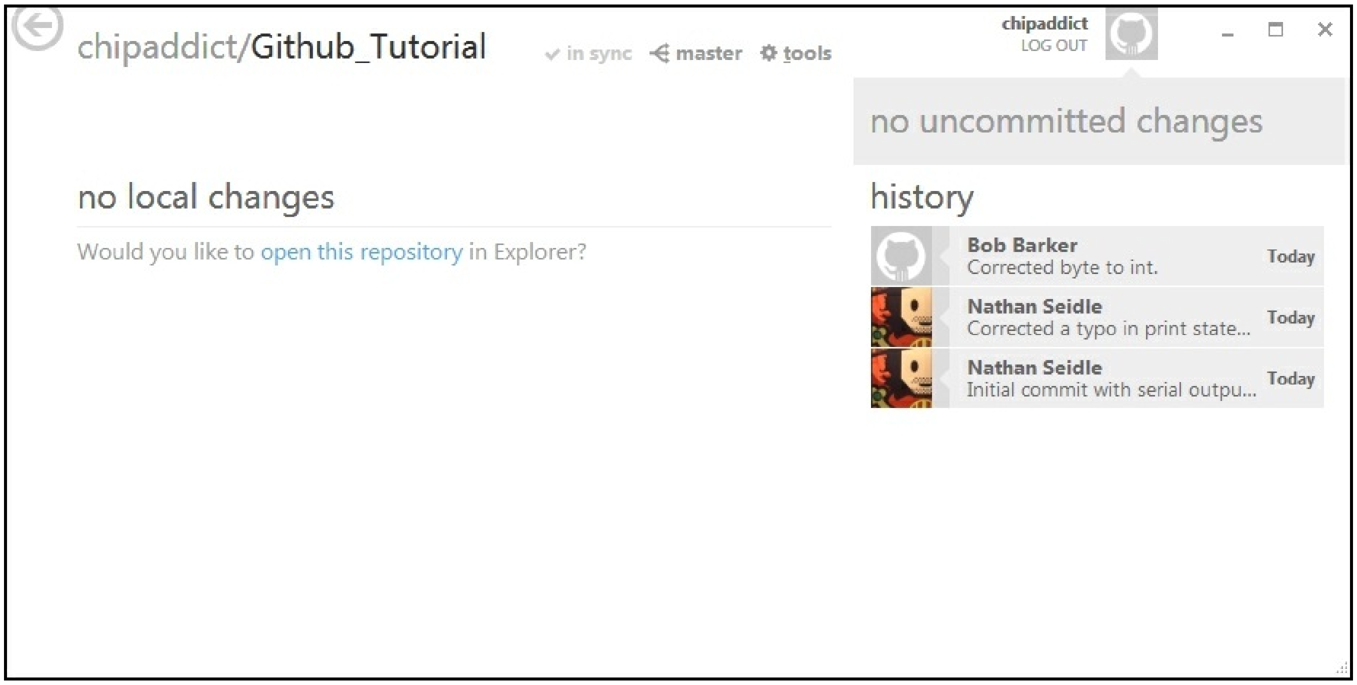
Would you like to open this repository in Explorer?

1f5c18a:	Corrected byte to int.	Today

Nathan Seidle	Corrected a typo in print state...	Today
Nathan Seidle	Initial commit with serial outpu...	Today

We now have *no uncommitted changes* but we **do** have *unsynced commits*. This means we've committed the changes to our local repo but we have not yet pushed (synchronized) the changes to our global repo.

Global repo: Find the 'sync' button on the top bar and hit it. This will push the changes that we made within our local repo up to GitHub and to our account.



chipaddict/Github_Tutorial ✓ in sync master tools chipaddict LOG OUT

no uncommitted changes

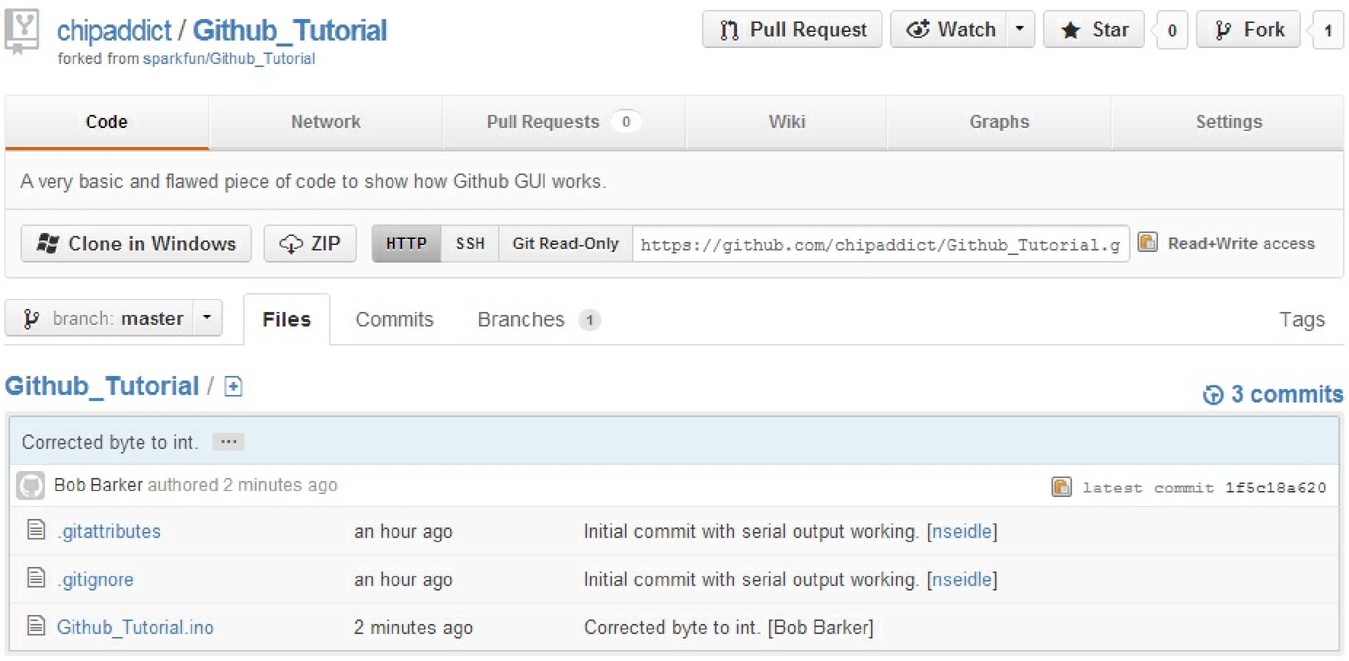
no local changes

Would you like to open this repository in Explorer?

history

Bob Barker	Corrected byte to int.	Today
Nathan Seidle	Corrected a typo in print state...	Today
Nathan Seidle	Initial commit with serial outpu...	Today

Nice job Bob! We have successfully pushed this corrected code up to the global repo on GitHub. Let's look at it online.



chipaddict / Github_Tutorial
forked from sparkfun/Github_Tutorial

Pull Request | Watch | Star 0 | Fork 1

Code | Network | Pull Requests 0 | Wiki | Graphs | Settings

A very basic and flawed piece of code to show how Github GUI works.

Clone in Windows | ZIP | HTTP | SSH | Git Read-Only | https://github.com/chipaddict/Github_Tutorial.g | Read+Write access

branch: master | Files | Commits | Branches 1 | Tags

Github_Tutorial / — 3 commits

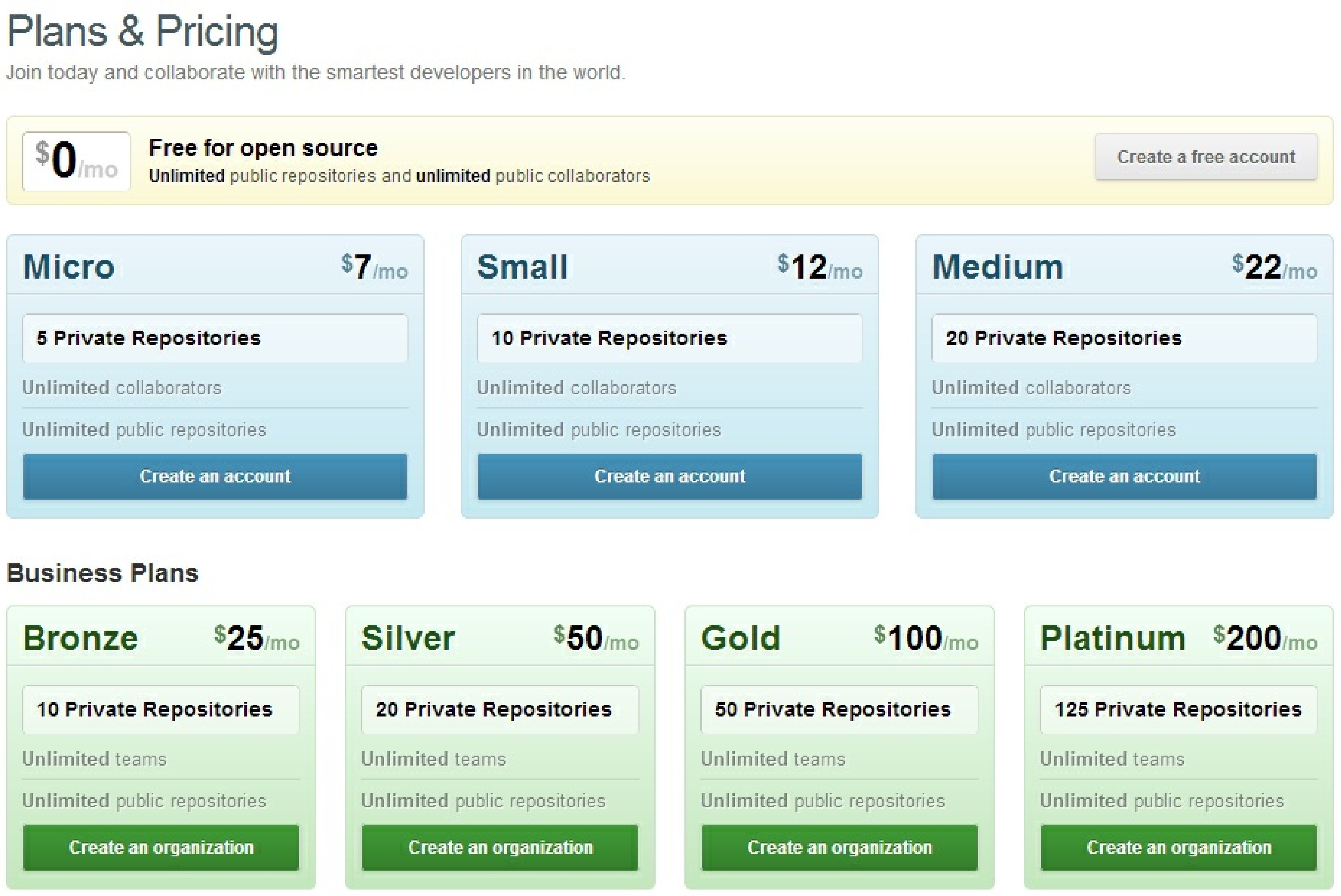
Corrected byte to int. ...

Bob Barker authored 2 minutes ago — latest commit 1f5c18a620

.gitattributes	an hour ago	Initial commit with serial output working. [nseidle]
.gitignore	an hour ago	Initial commit with serial output working. [nseidle]
Github_Tutorial.ino	2 minutes ago	Corrected byte to int. [Bob Barker]

The GitHub web interface is similar to the Windows GUI but adds many advanced options. Use the web for changing properties of the project; use the GUI for routine commits to the local repo and global syncs.

Beyond Free


Plans & Pricing

Join today and collaborate with the smartest developers in the world.

$0/mo **Free for open source** — **Unlimited** public repositories and **unlimited** public collaborators — Create a free account

Micro $7/mo	Small $12/mo	Medium $22/mo
5 Private Repositories	10 Private Repositories	20 Private Repositories
Unlimited collaborators	Unlimited collaborators	Unlimited collaborators
Unlimited public repositories	Unlimited public repositories	Unlimited public repositories
Create an account	Create an account	Create an account

Business Plans

Bronze $25/mo	Silver $50/mo	Gold $100/mo	Platinum $200/mo
10 Private Repositories	20 Private Repositories	50 Private Repositories	125 Private Repositories
Unlimited teams	Unlimited teams	Unlimited teams	Unlimited teams
Unlimited public repositories	Unlimited public repositories	Unlimited public repositories	Unlimited public repositories
Create an organization	Create an organization	Create an organization	Create an organization

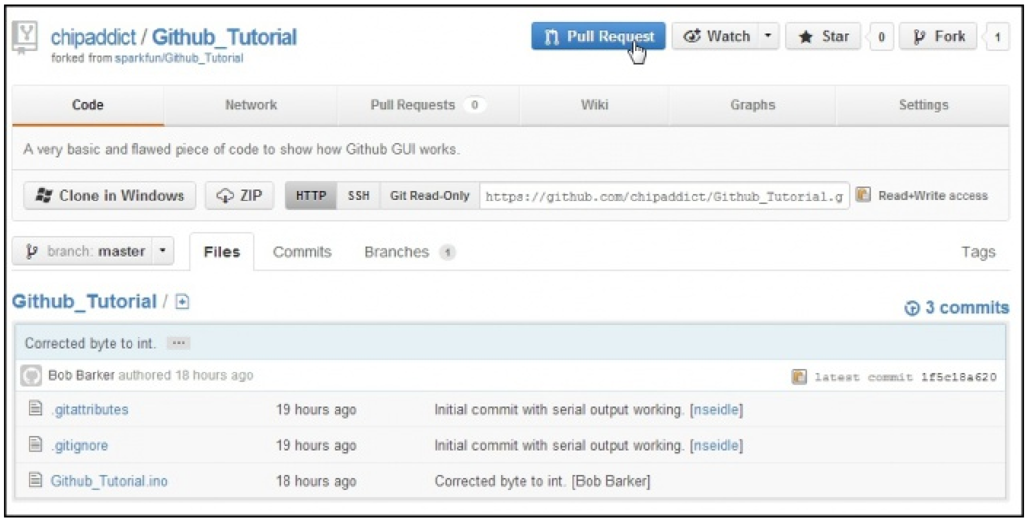
GitHub has a neat revenue model where you can have as many public repositories as you want (yay Open Source Hardware!) but if you want private repos, you have to pay. SparkFun pays at the Gold level ($100 per month) because we love GitHub, use them extensively for our web development and use GitHub for our public hardware projects. We generally create a private repo for a new project and turn it public as we near the release date for the product.

That's it for this section. These steps of forking a repo or creating your repo should allow you to create code projects, PCB layouts (we push Eagle files up to GitHub all the time), documents, images and even binary files.

In this section we found a bug and corrected it, but we have not yet let the original project know about the error. The next section will cover how to send corrections and improvements back to the original project through pull-requests.

Pull Requests

Repositories are great for managing and tracking changes made to code over time. But the real power comes into play when you collaborate with multiple people on a project. When people have multiple improvements, how do we combine them? Pull requests allow contributors to give back to the main project.



A very basic and flawed piece of code to show how Github GUI works.

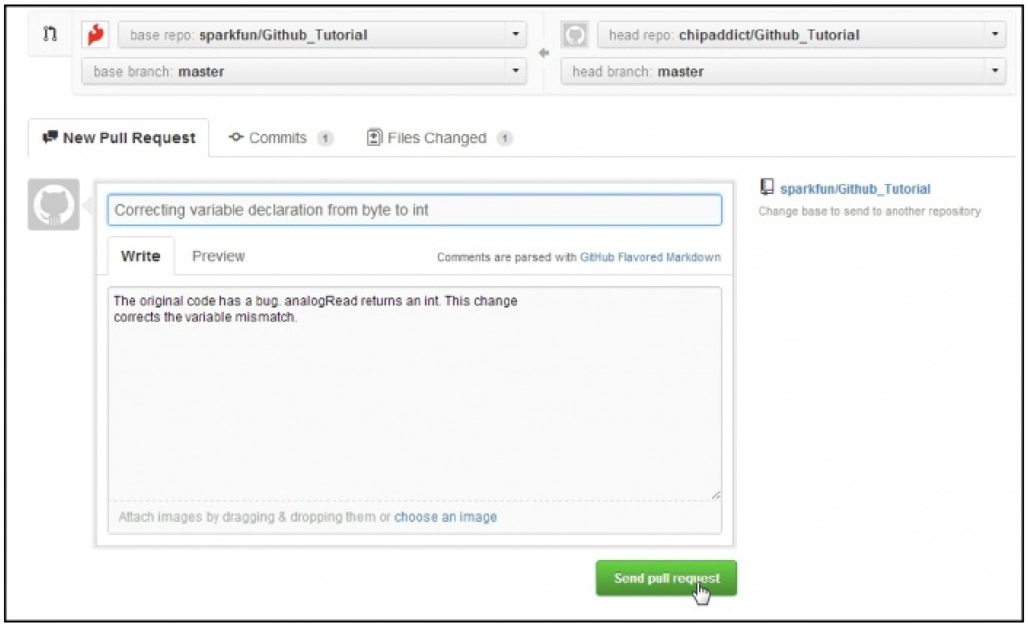
https://github.com/chipaddict/Github_Tutorial.g

Corrected byte to int.

Bob Barker authored 18 hours ago — latest commit 1f5c18a620

.gitattributes	19 hours ago	Initial commit with serial output working. [nseidle]
.gitignore	19 hours ago	Initial commit with serial output working. [nseidle]
Github_Tutorial.ino	18 hours ago	Corrected byte to int. [Bob Barker]

We've made some improvements to our version of the *Github_Tutorial* project. Now let's click on the Pull Request button to let the owner of the project know about the improvements we've made.


New Pull Request
Commits 1
Files Changed 1
Write
Preview
Comments are parsed with GitHub Flavored Markdown
sparkfun/Github_Tutorial
Change base to send to another repository
Attach images by dragging & dropping them or choose an image
Send pull request

base repo: sparkfun/Github_Tutorial
base branch: master
head repo: chipaddict/Github_Tutorial
head branch: master

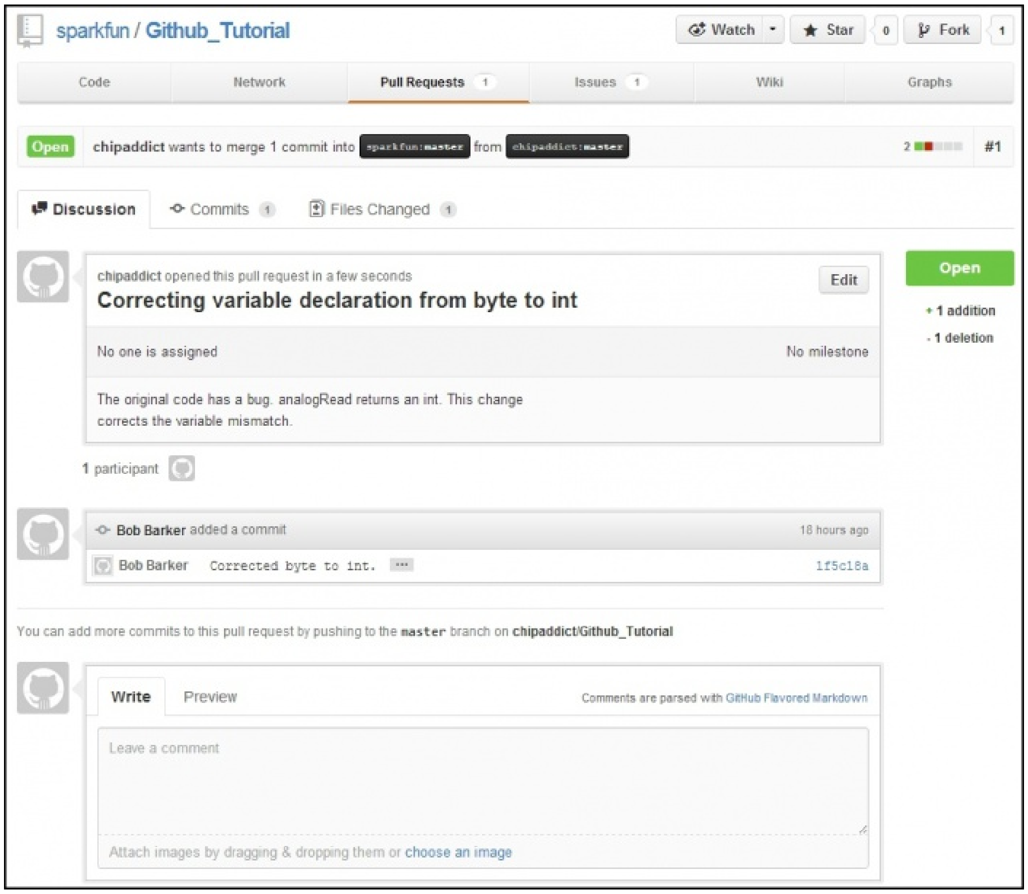
Correcting variable declaration from byte to int

The original code has a bug. analogRead returns an int. This change corrects the variable mismatch.

Here's where we describe the changes that we've made so that the owner of the main project (SparkFun is the owner in this example) knows what to expect.

As with most comments, be as verbose as possible. The changes you've made are obvious to you but to a project owner with thousands of lines of codes and dozens of pull requests it can become confusing.

Once you've written a note about the changes you're proposing click on 'Send Pull Request'.


sparkfun / Github_Tutorial

Open chipaddict wants to merge 1 commit into sparkfun:master from chipaddict:master

chipaddict opened this pull request in a few seconds

Correcting variable declaration from byte to int

No one is assigned — No milestone

The original code has a bug. analogRead returns an int. This change corrects the variable mismatch.

1 participant

Bob Barker added a commit — 18 hours ago

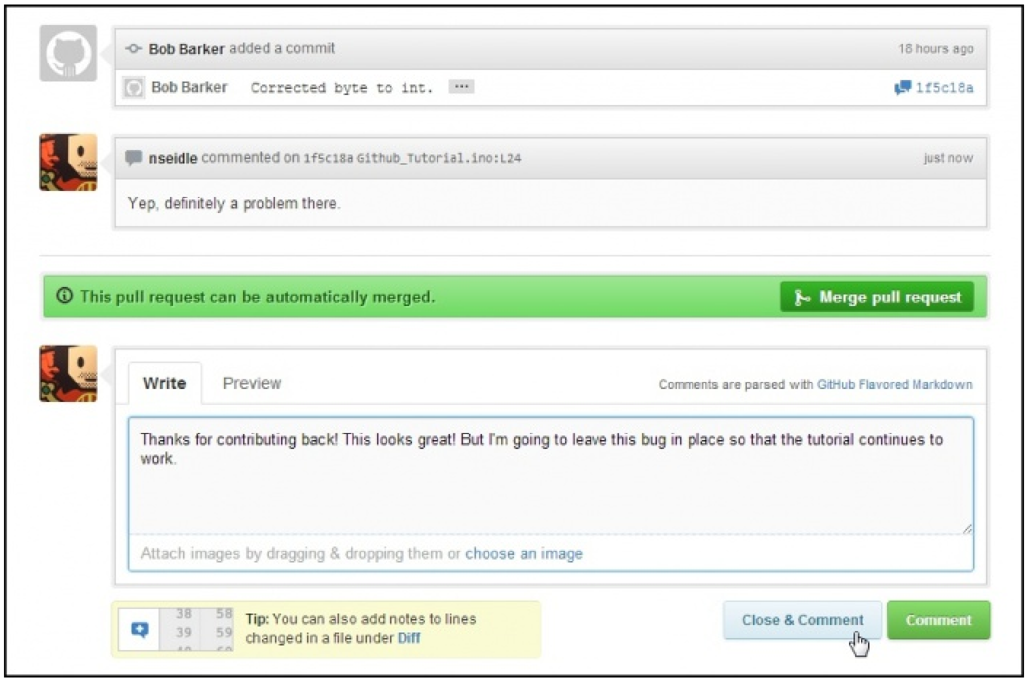
Bob Barker `Corrected byte to int.` 1f5c18a

You can add more commits to this pull request by pushing to the master branch on chipaddict/Github_Tutorial

+1 addition -1 deletion

Once we've sent the pull request, the owner of the main project is notified. Please don't hesitate to send a pull request to SparkFun for this tutorial. We'd love to hear from you!

This is where the owner of the project can review the submitted code (sometimes called a patch). GitHub provides a great discussion system so that the patch can be discussed. You can even comment on individual lines of code.


Bob Barker added a commit — 18 hours ago

Bob Barker `Corrected byte to int.` 1f5c18a

nseidle commented on 1f5c18a Github_Tutorial.ino:L24 — just now

Yep, definitely a problem there.

This pull request can be automatically merged.

Thanks for contributing back! This looks great! But I'm going to leave this bug in place so that the tutorial continues to work.

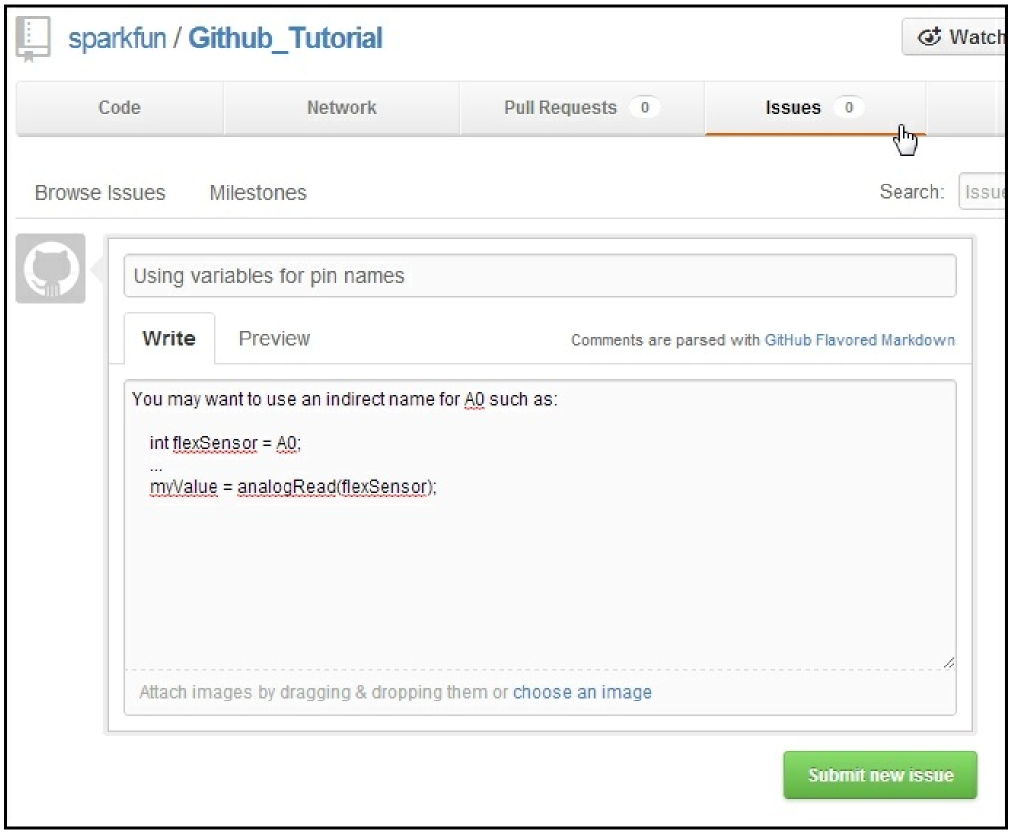
Tip: You can also add notes to lines changed in a file under Diff

Here's what the pull request looks like from the owner's point of view. The owner has the option to merge this pull request or discuss it.

In general, create pull requests that are smaller and more simple in nature. This will make it easier for the project owner to wrap their head around. It's easier to accept pull requests that contain 5 or 10 changes, but a monumental task if you've completely rewritten 400 lines of code.

Wiki and Issues

There are some additional tools built into GitHub as well. **Issue Tracking** allows folks to post problems or issues with a given project. It's kind of like a ticketing system or tech support but with the ability to comment on a specific line of code.

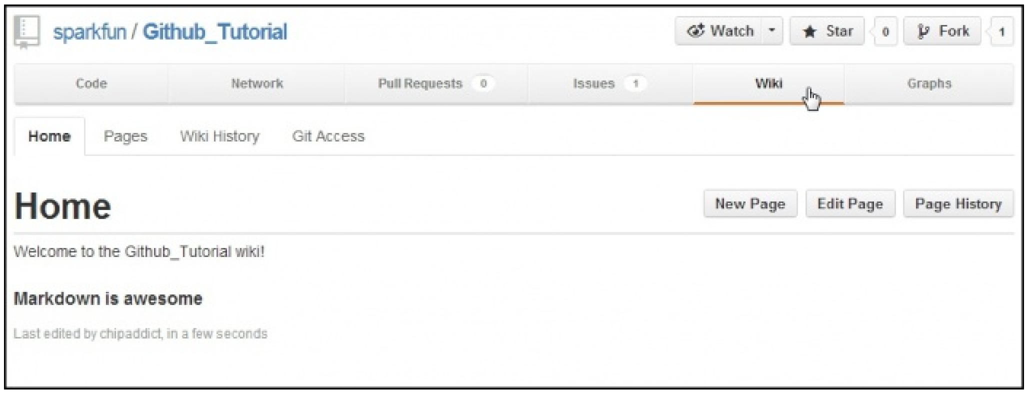

sparkfun / Github_Tutorial
Watch
Code
Network
Pull Requests 0
Issues 0
Browse Issues
Milestones
Search:
Write
Preview
Comments are parsed with GitHub Flavored Markdown
Attach images by dragging & dropping them or choose an image
Submit new issue

Using variables for pin names

You may want to use an indirect name for A0 such as:

```
  int flexSensor = A0;
  ...
  myValue = analogRead(flexSensor);
```

Here's is an example of creating an issue. Nothing too extraordinary but it allows for a good dialog between collaborators. You can see all the open issue on the Github_Tutorial project here.



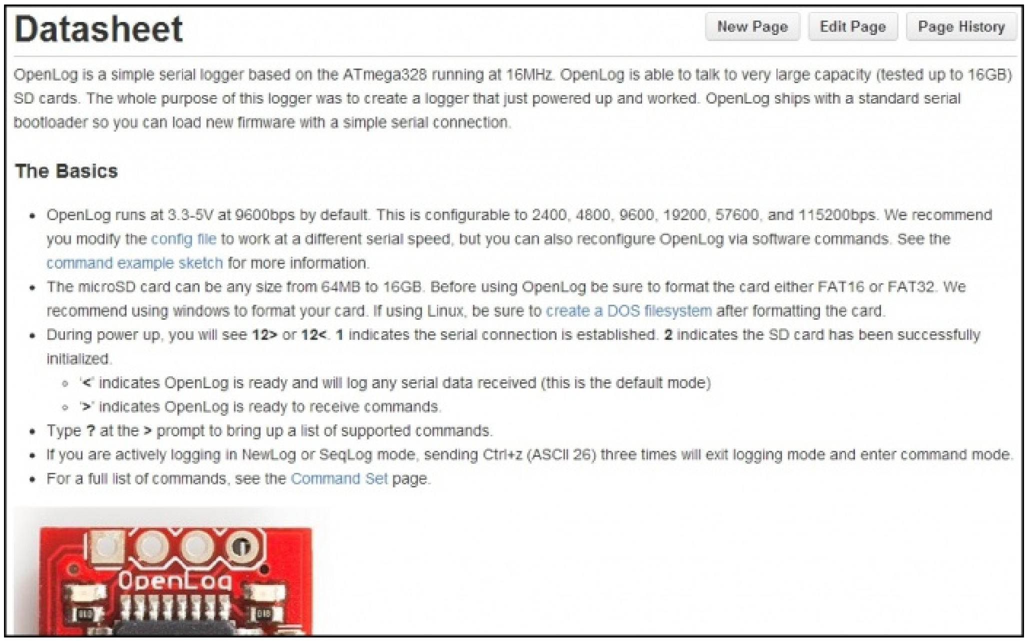
Home

Welcome to the Github_Tutorial wiki!

Markdown is awesome

Last edited by chipaddict, in a few seconds

Every repo also has a **Wiki** built in.



Datasheet

OpenLog is a simple serial logger based on the ATmega328 running at 16MHz. OpenLog is able to talk to very large capacity (tested up to 16GB) SD cards. The whole purpose of this logger was to create a logger that just powered up and worked. OpenLog ships with a standard serial bootloader so you can load new firmware with a simple serial connection.

The Basics

- OpenLog runs at 3.3-5V at 9600bps by default. This is configurable to 2400, 4800, 9600, 19200, 57600, and 115200bps. We recommend you modify the config file to work at a different serial speed, but you can also reconfigure OpenLog via software commands. See the command example sketch for more information.
- The microSD card can be any size from 64MB to 16GB. Before using OpenLog be sure to format the card either FAT16 or FAT32. We recommend using windows to format your card. If using Linux, be sure to create a DOS filesystem after formatting the card.
- During power up, you will see **12>** or **12<**. **1** indicates the serial connection is established. **2** indicates the SD card has been successfully initialized.
  - '**<**' indicates OpenLog is ready and will log any serial data received (this is the default mode)
  - '**>**' indicates OpenLog is ready to receive commands.
- Type **?** at the **>** prompt to bring up a list of supported commands.
- If you are actively logging in NewLog or SeqLog mode, sending Ctrl+z (ASCII 26) three times will exit logging mode and enter command mode.
- For a full list of commands, see the Command Set page.

We've found the the wiki (this is from OpenLog) is a great place to have documentation for a given project or product. As the PCB layout, firmware, and example code changes over time we can update the documentation as well. Collaborators can also help maintain and improve the documentation. GitHub has been pretty powerful over the past few years increasing the ability for SparkFun to collaborate and improve our hardware designs.

Try using Git and GitHub for your next project. There's an undeniable learning curve but it will make it much easier to collaborate with people.

Now that you've got repos under control we recommend you check out these tutorials:

- PCB Basics
- Arduino Basics
- GPS Basics
