

Department of Mathematics

About
Resources
Computing FAQs
Making a Website
Two Factor Authentication (2FA)
Teaching Resources
Support
History
Research
People
Undergraduate
Graduate
Internal Resources
Useful Links
News
Events
Login
My Account
Faculty test

Resources — Making a Website

Making a Professional Website: A Brief Guide for Academics

by Zev Chonoles

What is this page about?

If you are an academic – especially a young academic – a professional website is vital for making your research easily available to the world and increasing your visibility. You can also give information for the courses you’re teaching, share notes from seminars, and work with collaborators.

Knowing this, most departments offer their faculty and students the ability to set up a professional website. But if you don’t know how to use that ability, you’re missing an important tool for your academic career.

This page is a guide for members of UChicago’s math department to making a professional website

`http://math.uchicago.edu/~yourname/`

though the general process works equally well for most readers, with some minor changes. At the end, I’ll discuss further steps and alternatives, which also should be useful to most readers.

Follow Us

Pages

A simplistic explanation of how websites work

Every website has a server, which is a computer that stores all of the files composing the website. As its name implies, the server’s job is to serve that website to anyone who requests it. When you click on a link or type in an address, your computer contacts the relevant server with a request, and the server sends back the relevant files. Your browser turns them into the finished product you see or hear.

Websites can use many kinds of files, but HTML is the foundation of all websites. An HTML file is simply text, some of which is information about the text, such as where paragraphs begin and end.

`<p>Hello, friends!</p>`

With this explanation, you can formulate your goal more concretely: to make an HTML file and have the math department’s server provide it to the world.

How to make a website



In order to provide explicit examples and encourage what are (in my opinion) good practices, I've made specific recommendations about what programs to download and how to use them. Of course, my way is not the only way of doing things, and I explore many alternatives at the end of this guide. But please try to follow my recommendations first.

Step 1: Get a program for creating and editing HTML files

It's quite convenient to have a text editor which knows the syntax of HTML, thereby catching mistakes and filling things in for you. I recommend, and personally use, [Bluefish](#).

Step 2: Get a program for connecting to servers

You'll need to remotely connect to the math department server in order to put your HTML file on it. I recommend, and personally use, [Filezilla](#).

(In case you were curious: the server is physically located in [the basement of Hinds](#).)

Step 3: Log into the server

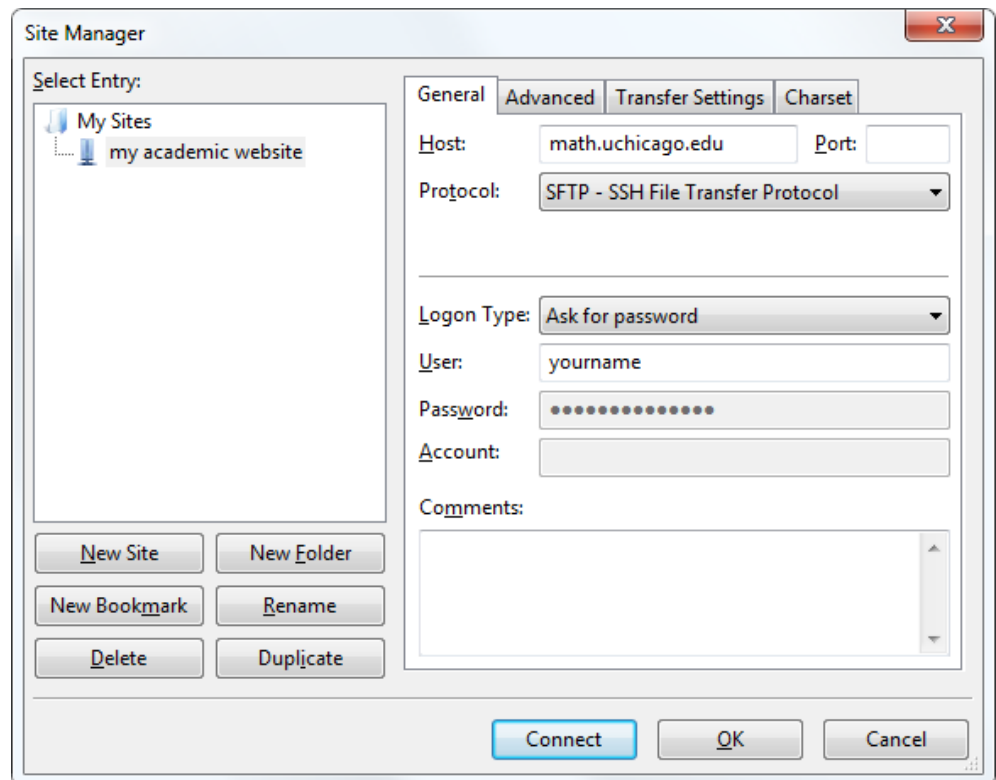
If your math department email address is

`yourname@math.uchicago.edu`

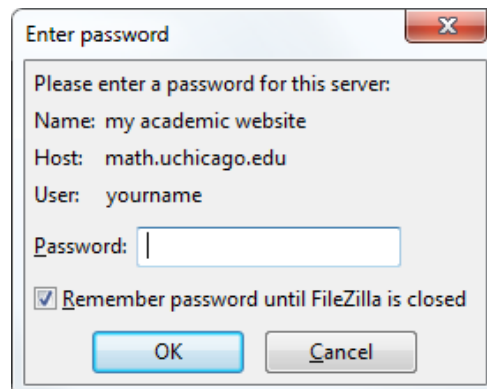
then you have an account named **yourname** on the math department server **math.uchicago.edu**. Your account also has a password, which you chose as a part of setting up your email. If you don't remember it, you'll have to ask [Ed or John](#), the department's sysadmins, to reset it for you.

Start Filezilla. Go to "File", then "Site Manager". Create a new site, and name it whatever you like. The host is **math.uchicago.edu**, and the protocol is SFTP. Leave the port number blank, because Filezilla knows that the default for SFTP is 22. Lastly, change the logon type to "Ask for password", and enter your account name. Don't worry about the dots in the password field.

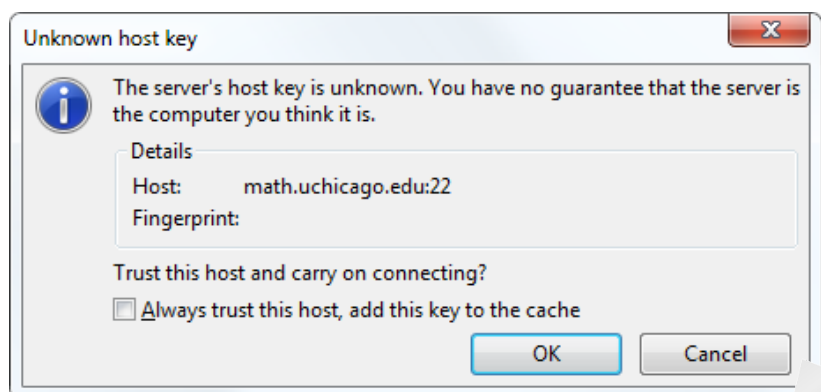




Click “Connect”, and enter your password when prompted. I recommend leaving the “Remember password until Filezilla is closed” box checked.



Now you’ll be warned about an unknown host key. The “Fingerprint” will be a hexadecimal string.



This is your first time interacting with this server, so Filezilla has nothing to check the claimed identity against. To find out whether hackers, the NSA, or your nosy neighbor

performing a **man-in-the-middle attack**, ask Ed or John to tell you the server's key and compare it with the "Fingerprint". Or, just check the box and click OK to log into the server.














Step 4: Prepare the folder that will contain your HTML file

In Filezilla, there are four navigation panes. The left two are "local" (your computer), and the right two are "remote" (the server). The top two show the overall folder structure, and the bottom two show the contents of the current folder. In the bottom two panes, the folder named `..` is a link to the parent of the current folder. We'll interact only with the bottom-right pane.

Any files that are a part of your website will go in an aptly-named folder `public_html` in your account. On the math department server, this folder doesn't exist by default, so we have to create it. Right-click anywhere in the bottom-right pane, and choose "Create directory".

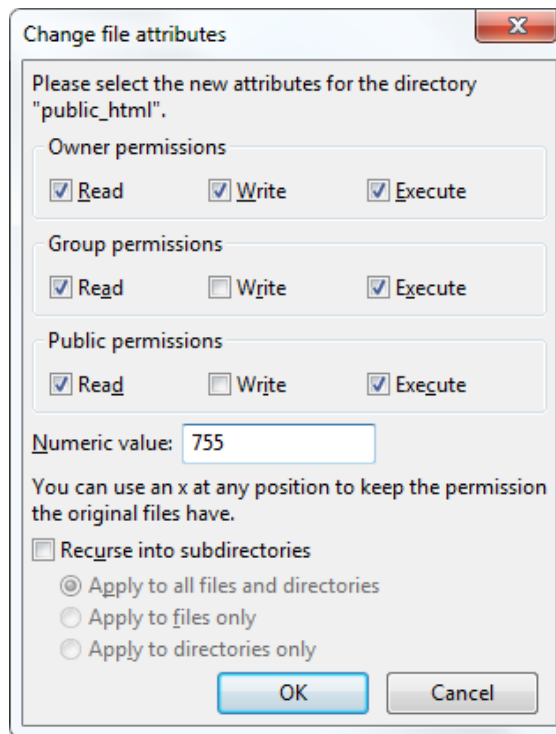


Here's the finished product. Note that you may have different folders than those shown below, and also that periods will come first in alphabetical order.

Filename	Filesize	Filetype
 .thumbnails		File folder
 .wallpapo		File folder
 bin		File folder
 Desktop		File folder
 Documents		File folder
 Downloads		File folder
 mail		File folder
 Music		File folder
 Pictures		File folder
 Public		File folder
 public_html		File folder
 R		File folder
 Templates		File folder
Selected 1 directory.		

Despite its name, the contents of `public_html` are not publicly visible by default. To change this, right-click on the folder and choose "File permissions". Enter `755` and click OK.





If you understand binary, the image above may help you to see the significance of $7 - 5 - 5$.

By doing this, any file in the `public_html` folder, as long as the file itself has the correct permissions, will be available under the address `http://math.uchicago.edu/~yourname/`. For example, to share `awesome_research.pdf` with the world, just put it inside `public_html`, set its permissions to `755`, and tell people to go to

`http://math.uchicago.edu/~yourname/awesome_research.pdf`

The goal of this guide is to share an HTML file with the world, which brings us to our last step.

Step 5: Create an HTML file and put it on the server

Now start Bluefish. Paste the following text into the blank file, and save it as `index.html`.

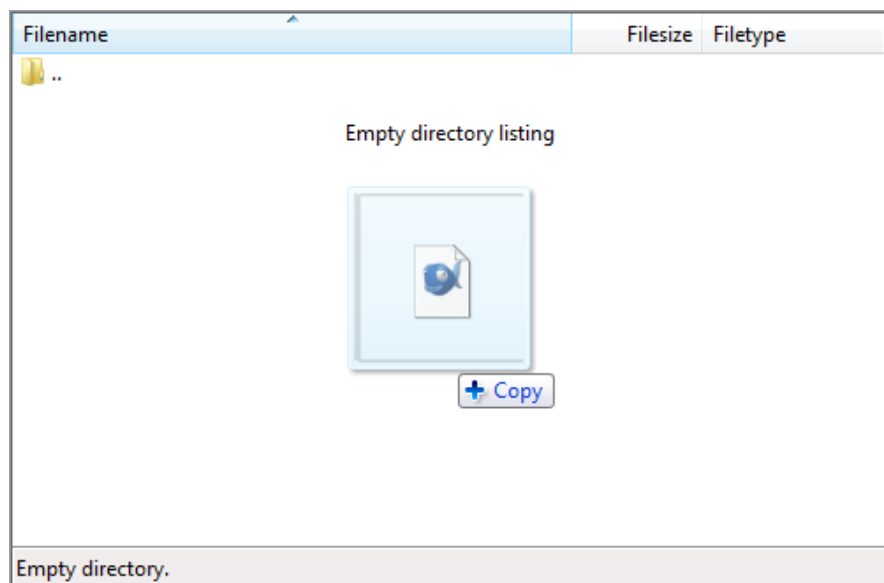
```
<!doctype html>
<html>
<head>
  <title>My first website</title>
</head>
<body>
  <p>Hello, friends!</p>
</body>
</html>
```

Observe that, once Bluefish knows you're making an HTML file, it provides syntax highlighting, visually distinguishing the HTML code from your text.





In Filezilla, enter the `public_html` folder by double-clicking on it. Add the file `index.html` to the folder by dragging-and-dropping from your computer's file manager to the bottom-right pane.



Then set the permissions of `index.html` to 755. By what I've already said, you know that going to

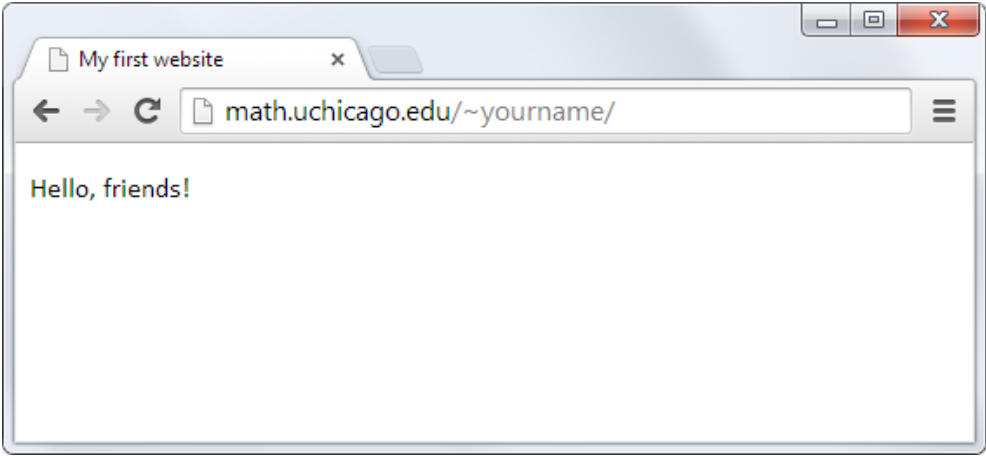
`http://math.uchicago.edu/~yourname/index.html`

in your browser will bring up your website. But servers give special treatment to files named `index`, so you can also go to the sleeker address

`http://math.uchicago.edu/~yourname/`

and it will come up as well.





Congratulations on setting up your website!

Further steps

Adding more to your website

Back at the beginning of this guide, I said that

An HTML file is simply text, some of which is information about the text, such as where paragraphs begin and end.

But there’s a whole lot more to HTML than paragraphs: just for starters, there are links (`<a>`), headings (`<h1>` through `<h6>`), lists (`` and ``), images (``), and the ubiquitous `<div>`.

CSS is fundamental to styling websites. This is a language for telling the browser how to display an HTML file. You control the dimensions, fonts, colors, etc. of the HTML elements you specify.

HTML	CSS rule	What the browser shows
<code><p>Hello, friends!</p></code>	<code>p { color: red; }</code>	Hello, friends!

If you do math or science, you might be interested in including [MathJax](#) on your website, so that you can use a large (but proper) subset of LaTeX and get formatted output like

$$f(a)=12\pi i\oint\gamma f(z)z-adzAx|\downarrow|C-\rightarrow-----w-\rightarrow-----zB|\downarrow|yDf(a)=12\pi i\oint\gamma f(z)z-adzA\rightarrow wBx\downarrow yC-$$

Lastly, a nice bit of flair for your website is the favicon (the little image used in bookmarks and tabs). You can copy the one on your school’s website, or make your own.

Googling any of these terms will lead you to a wealth of information. Additionally, you can examine the HTML and CSS of my website in detail by clicking the links at the bottom of each page.

Learning from other websites

In any modern browser, you can right-click on a page and choose “view source”, and see all of the code your browser was sent to put together that page. I provided the links in the footer of my site for curious people who don’t know this, and because it was fun to implement.



However, take care to avoid **cargo cult programming**. When you see a website that does something you like, make an effort to understand how it works before you copy everything. If you're not sure what a piece of code does, take it out temporarily and see what the effect is; you can always put it back later. Don't be afraid to look things up or ask people.

Using PHP to avoid repeating yourself

As I just mentioned, you can inspect all of the files that the server **sends** to your browser. But for many websites, these files are generated, not static: when you request such a website from its server, the server in that instant creates the files to be sent. This is an extremely powerful tool. Here's one basic use: this page was generated and sent to you at Chicago time

PHP on the server	Sent in its place
--------------------------	--------------------------

<code><?php echo date("g:ia"); ?></code>	<code>1:23pm</code>
--	---------------------

If you are going to repeat certain things many times in your website – for example, the title, or most of the content in the `<head>` element – you can use a PHP **include** to make your life easier, and ensure consistency between the many instances of the repeated code.

PHP on the server	Sent in its place
--------------------------	--------------------------

<code><?php include('file.php'); ?></code>	<code>the contents of file.php</code>
--	---------------------------------------

By its very nature, you cannot normally see PHP code with your browser's "view source". But for the curious, I've provided links to my PHP files in the footer; compare them with the HTML.

Copyright and licensing

Now that you have a website, you can use it to share your valuable work with the world. As a part of this, it is important to understand the basics of copyrights.

At least in the US, you automatically have the copyright to anything you produce – there are no court papers to file, you don't even have to put a little © symbol on it, the copyright is yours. And the most important thing to remember, from [the Wikipedia page on free content](#):

... copyright law in most countries by default grants copyright holders monopolistic control over their creations ...

So if you decide "I don't want to bother with any legal nonsense, I just won't say anything about it", you are actually being **as restrictive as possible** with how other people can use your work!

You might reply "People will do whatever they want with my work regardless of whether it's legal." You may even have been unknowingly doing so yourself. But working from this assumption only perpetuates the current culture of ignoring copyright, and any actual law-abiding people might avoid using your work entirely (see [this discussion](#), for example).

Luckily it's quite easy to change how others are allowed to use your work, by specifying a license. Usually, to apply a license it's enough to simply say "I am using the ____ license". Open licenses, such as the **WTFPL** or those from **Creative Commons**, are an excellent choice for academics. In the footer of my website, you can see that I use the Creative Commons BY license.



Incidentally, this means you are welcome to use any code from my website that you like, as long as you provide attribution. [See here for the details.](#)

Editing your website

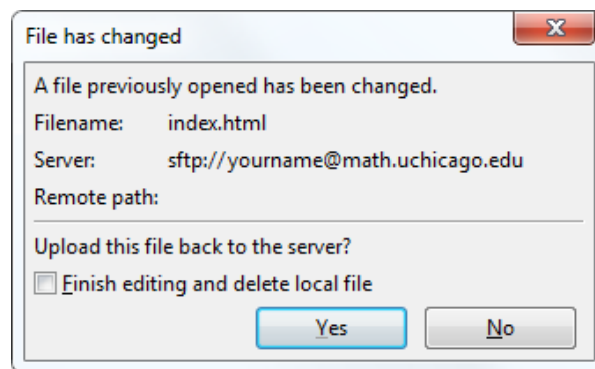
You've just finished setting up your `index.html` file, but now I've gone and told you a bunch of things to start adding to your website. How do you edit this file?

In Filezilla, right-clicking on any file in the bottom-right pane brings up the option "View/Edit", which seems promising, but your computer will instinctively open HTML files with your browser.

 View/Edit dialog

Telling your computer to associate HTML files with Bluefish will make "View/Edit" work the way you expect, and won't interfere with your everyday internet browsing. Use Google to learn how to change file associations on your operating system.

One of the advantages of this approach is that Filezilla will keep track of files as you edit them on your computer, and any time you click "save" in Bluefish, Filezilla will ask whether you want to update the copy on the server with your changes.

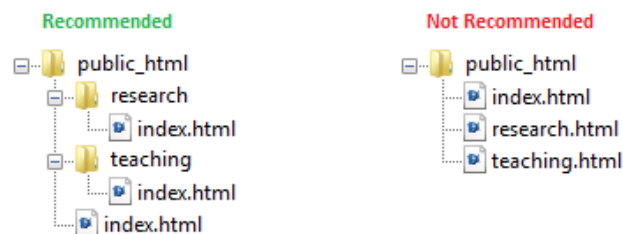


This makes for a very smooth and easy workflow: type a bit in Bluefish, click "save", click "yes" in the above dialog, click "refresh" in the browser to see results. Repeat.

A less elegant approach would be to keep a copy of all of your website's files on your computer, and as you edit them on your computer, dragging-and-dropping to Filezilla when you want to update the version on the server. But instead of the above dialog, you'll have to deal with one complaining that there's already a file on the server with that name.

Using a folder structure for multiple pages

I recommend organizing your website using folders, with one `index.html` per folder, instead of different HTML files all at the top level.



As I mentioned earlier, this lets you use addresses that do not include a file name.



`http://math.uchicago.edu/~yourname/research/`

In my opinion, such addresses look sleeker and more professional without the trailing `.html`. It hides the inner workings and inessential details of your website.

Also, you might eventually want to use a web technology, such as PHP, that requires a file ending other than `.html`. Changing those file endings will break every previous link to your site... but if you use the approach I'm advocating, you can change to `index.php` and everything will be fine.

Lastly, a website will quickly become an unorganized profusion of files in the `public_html` folder unless you provide some structure with folders underneath it, and your HTML files should follow that structure as much as anything else.

If you follow my recommendation, don't forget to also set the folders' permissions to `755`.

Can I use this guide if I'm not in UChicago's math department?

For simplicity, let me reserve the word "school" for an entire educational institution, and extend the word "department" to potentially mean any proper subset of a school, including research groups, divisions, etc. Please keep this in mind as you decide how to interpret my discussion for your situation.

You should be able to use this guide if the websites of other people in your department look like

`http://dept-web-address/~theirname/`

and if you have an email address

`yourname@dept-web-address`

This almost certainly indicates that your department has a server `dept-web-address`, that you have an account `yourname` on that server, and that you can create a website. Just replace every instance of `math.uchicago.edu` in this guide with `dept-web-address` and you're good to go.

The most common complications to this are:

1. You don't have a departmental email address / your department itself doesn't have a website.
2. Your department has a website, but doesn't have any content about its members.
3. Your department has a website and maintains profiles of its members that you can request changes to, but don't have direct control over.
4. Your department has a website and hosts members' professional websites, but you have to send your files to the department sysadmin who will check them for viruses, copyright infringement, etc. before putting them online for you.
5. Your department has a website and hosts members' professional websites, but the website addresses don't match the pattern of departmental email addresses above way.



In cases 1 through 4, you could find out if there's a [school-wide server you can use](#). In case 5, you'll need to find the correct server and the name of your account on it. Ask for help from your department's sysadmin, or from someone in your department who has already made their website.

Alternatives...

to using Bluefish

If you don't want to write any code yourself, but still want to completely design your own site, you can use a [WYSIWYG](#) editor. [BlueGriffon](#) is probably the best free option at the moment. [Dreamweaver](#) is better-known and has more features, but it costs money.

If you don't mind writing code, but want help getting started, use Google to look for free HTML and CSS templates. [Andreas Viklund's templates](#) are popular at the UChicago math department.

A [content management system](#) is sort of a heavy-duty, general template. After setup, you don't connect to the server to use it; creating or editing content takes place in your browser. A CMS is overkill if you only want a simple, single-page website, but they're the right tool for some jobs. [WordPress](#) and [Drupal](#) are popular, or if you want a wiki, [MediaWiki](#) and [Dokuwiki](#).

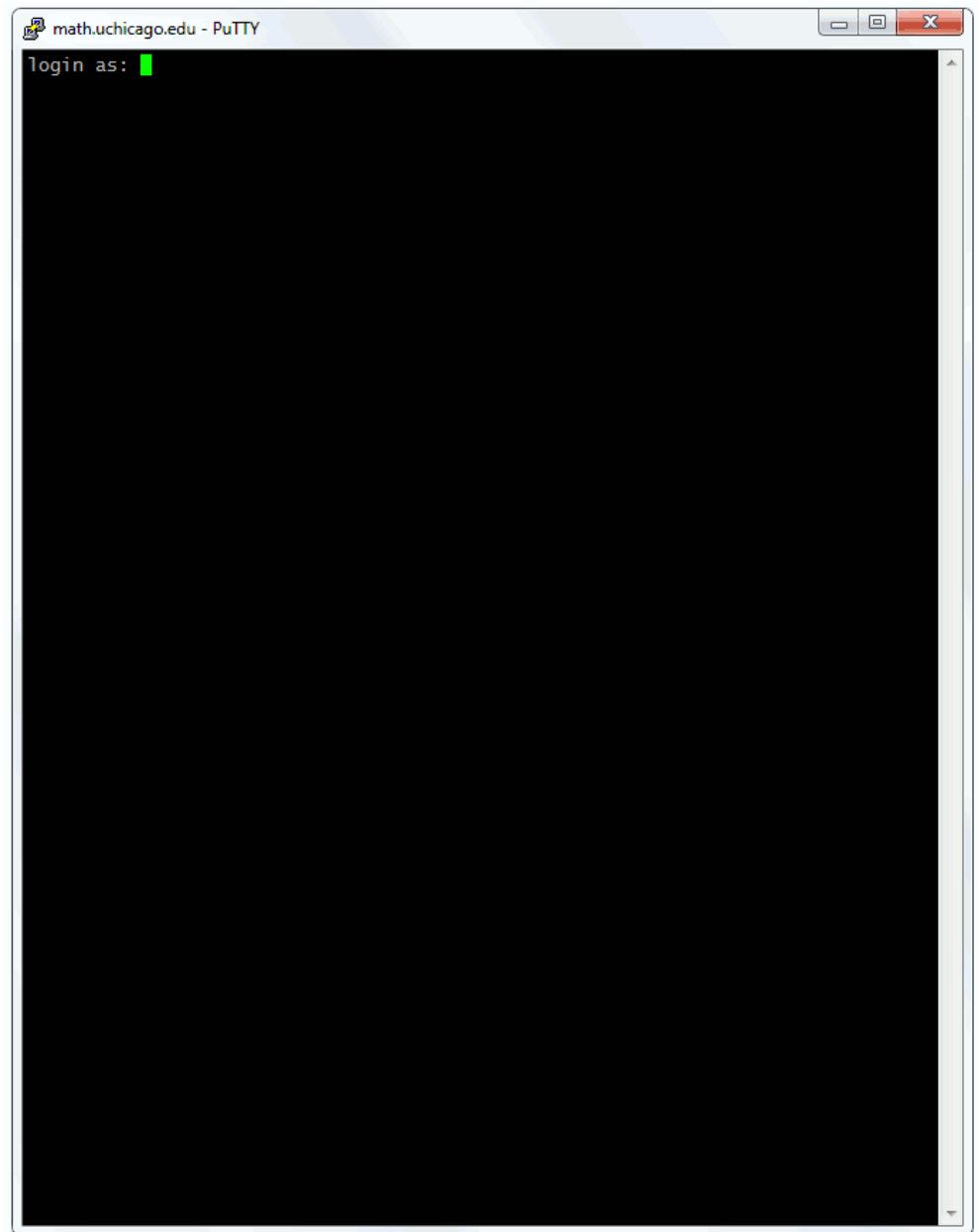
If even Bluefish was too fancy for your taste, you can use your operating system's built-in text editor, which is Notepad for Windows, and TextEdit for Mac.

to using Filezilla

[Cyberduck](#) is another popular option available for Windows, Mac, and Linux. In my opinion, its main advantage over Filezilla is that it has a funnier logo.

If you'd like to overcome your fear of command line interfaces, you can use [PuTTY](#) and [PSFTP](#) on Windows, or the commands `ssh` and `sftp` in the ["Terminal" program](#) on Macs. You'll want to look up the commands `ls`, `cd`, `touch`, and `chmod`, and the bare basics of the text editor `vi`. To see this all in action, click below to watch me use PuTTY to create a file `test.html`.





It may seem like the only benefits of the command line are that writing arcane commands on a black background looks impressive, and gives one a better appreciation of how easy life is with graphical interfaces. But its true power is not evident in this simple demonstration.

Lastly, it's worth mentioning that Filezilla and its alternatives are all just substitutes for the original approach, namely, physically being at a department computer and logging in.

to using a department server

Your school may provide a server where you can host a website. For example, here at the University of Chicago, anyone with a CNet ID can access the server `webspace.uchicago.edu` and create a website `http://home.uchicago.edu/~cnetid/` (see the [full details here](#)).

You can also host your website on a commercial server. You'll get a website that's in one place even when you move from your current school, and has any address you want. Read more about [the pros and cons of doing so](#), as well as [where to host your](#)



to my entire guide

If you absolutely refuse to do more than the minimum amount of work to make a website, then you can use [Google Sites](#) or [WordPress.com](#) (which uses, but is different than, WordPress). These services host the website for you, give you a template, and give you an online WYSIWYG interface for editing it. This is still hosting on a commercial server, so **the same pros and cons** apply.

By default, you'll get an address of the form

`http://sites.google.com/site/example/` `http://example.wordpress.com`

Luckily, it is possible to redirect these ugly commercial addresses to one at your university, or one that you own (instructions for [Google Sites](#), [WordPress.com](#)).

Acknowledgements

Thank you to Vipul Naik, Subhadip Chowdhury, and John Zekos for indulging my many basic questions when I was first making my website, and teaching me many of the things that went into this guide.

Feedback

Please send me an email if you have any questions, comments, or suggestions about the contents of this guide, especially if you encountered a difficulty that I did not cover, or if you are experienced with this material and have a complaint about the way I presented something.

I would also greatly appreciate it if you would take **a brief survey** about the quality of this guide.

If this guide helped you get started with your website, share a link to it on your new website if you want to spread the word.

© 2018 Department of Mathematics
The University of Chicago
5734 S University Ave
Chicago, IL 60637
Phone: (773) 702-7100

Last updated 2017-09-30
Contact the webmaster

