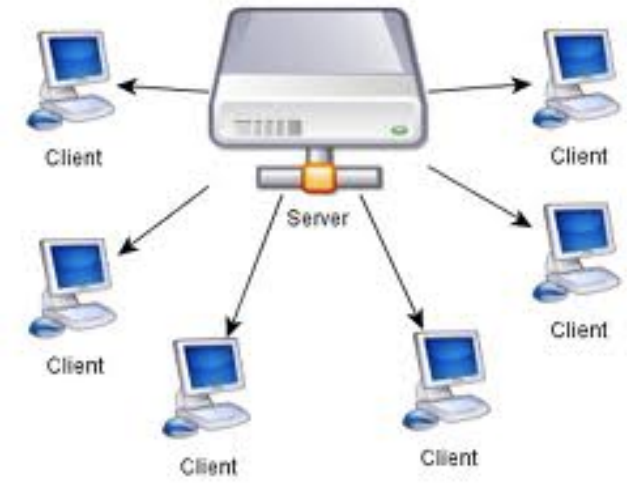
Intro to Servers 1 (Linux Servers)

What is a server?

A server is a machine that can host resources and allow clients to access said resources. Physically, servers are tall racks of machines. They run operating systems and you as a developer can choose what OS you want to work with. You must keep in mind that depending on what you want to do will correlate with what server you will choose. For the most part, linux servers are the norm.

What is a client?

A client is software that (usually) connects to the server to perform actions. The client provide a user interface that allows users to carry out actions. It forwards these requests to the server, which carries out the action and returns a response.



Server Connection - FTP (File Transfer Protocol)

At first, connecting to a server can seem like a pain and depending on your server you might have different protocols to connect. Once you get the hang of it, it will be a snap. Repetition and becoming familiar with how FTP connections work will be more than half the battle.

First off. When connecting to the server you can use the IP address, or alternatively, the web address. Sometimes you may have to use a username.

When logging in you will need:

Host/FTP Address
Username
Password

Host/FTP =
The sites IP address - 00.000.000.000
The sites Web address - www.yousite.com
The sites Web address with FTP instead of WWW - ftp.yoursite.com

Username = Your FTP username that you make in Cpanel/ Admin Panel
Password = Your FTP Password you create in Cpanel/ Admin Panel

Some servers may be set up in a different way. For example they may have a special address for you to use to login correctly. If this is the case you should be able to

find out in the admin panel of your server or by the hosting provider.

Examples of hosting providers:

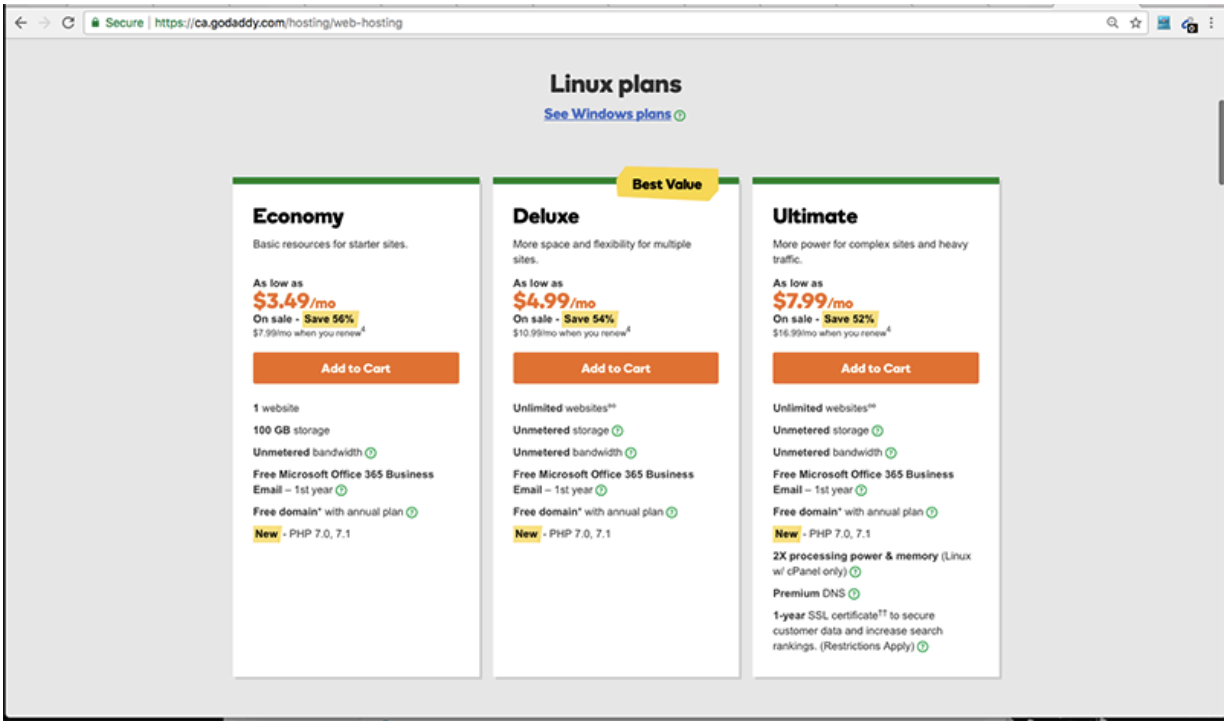
godaddy.com
web.com
hostgator.com
bluehost.com
netfirms.ca

These are places where you can register domains, buy hosting packages for shared, Virtual Dedicated, and fully dedicated servers.

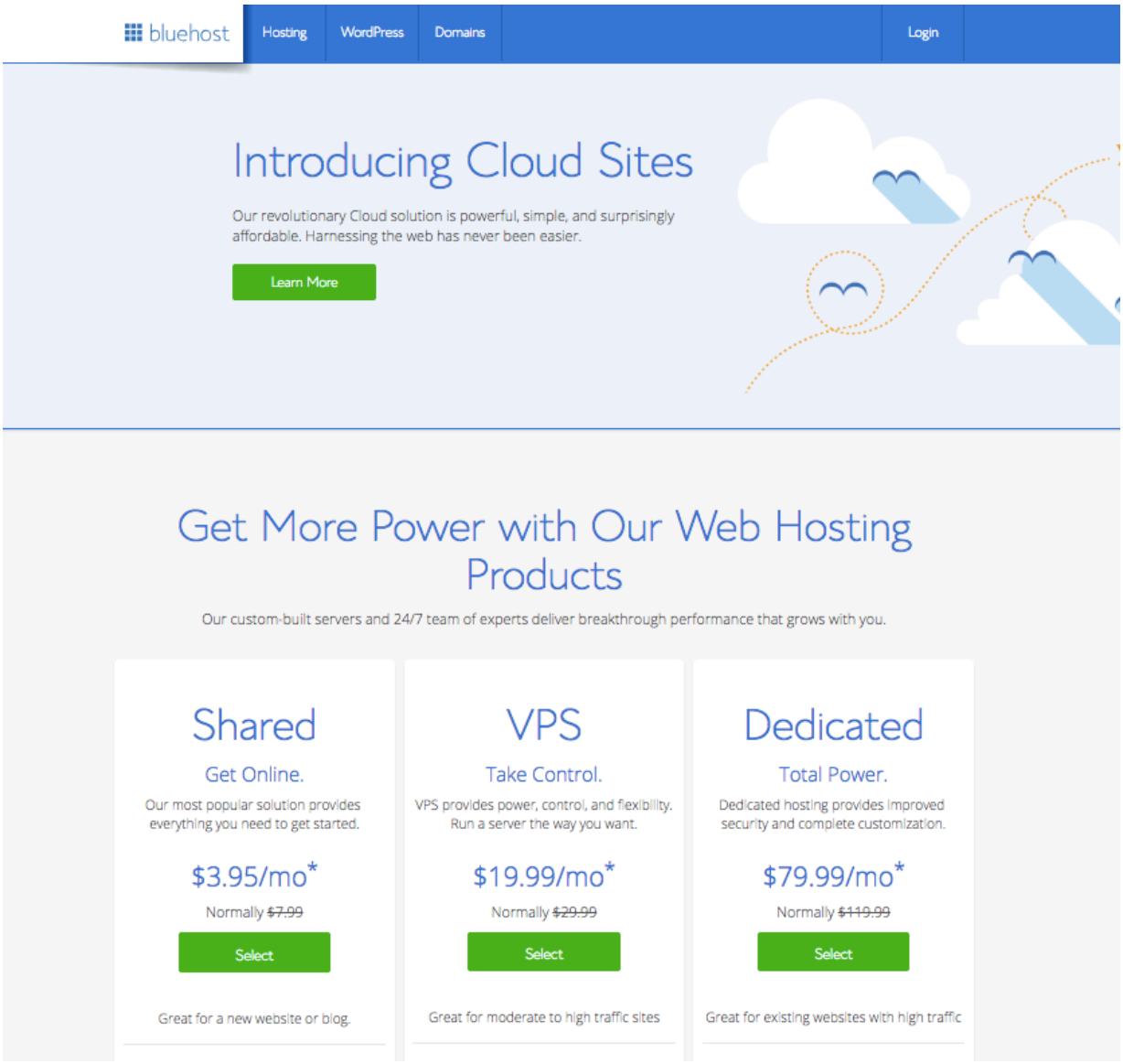
Buying your hosting

Buying your hosting is easy. All you need to do is choose the hosting you want to work with and than pick the hosting plan that is right for you and what you need to do. Hosting companies usually have different plans that will range in price.

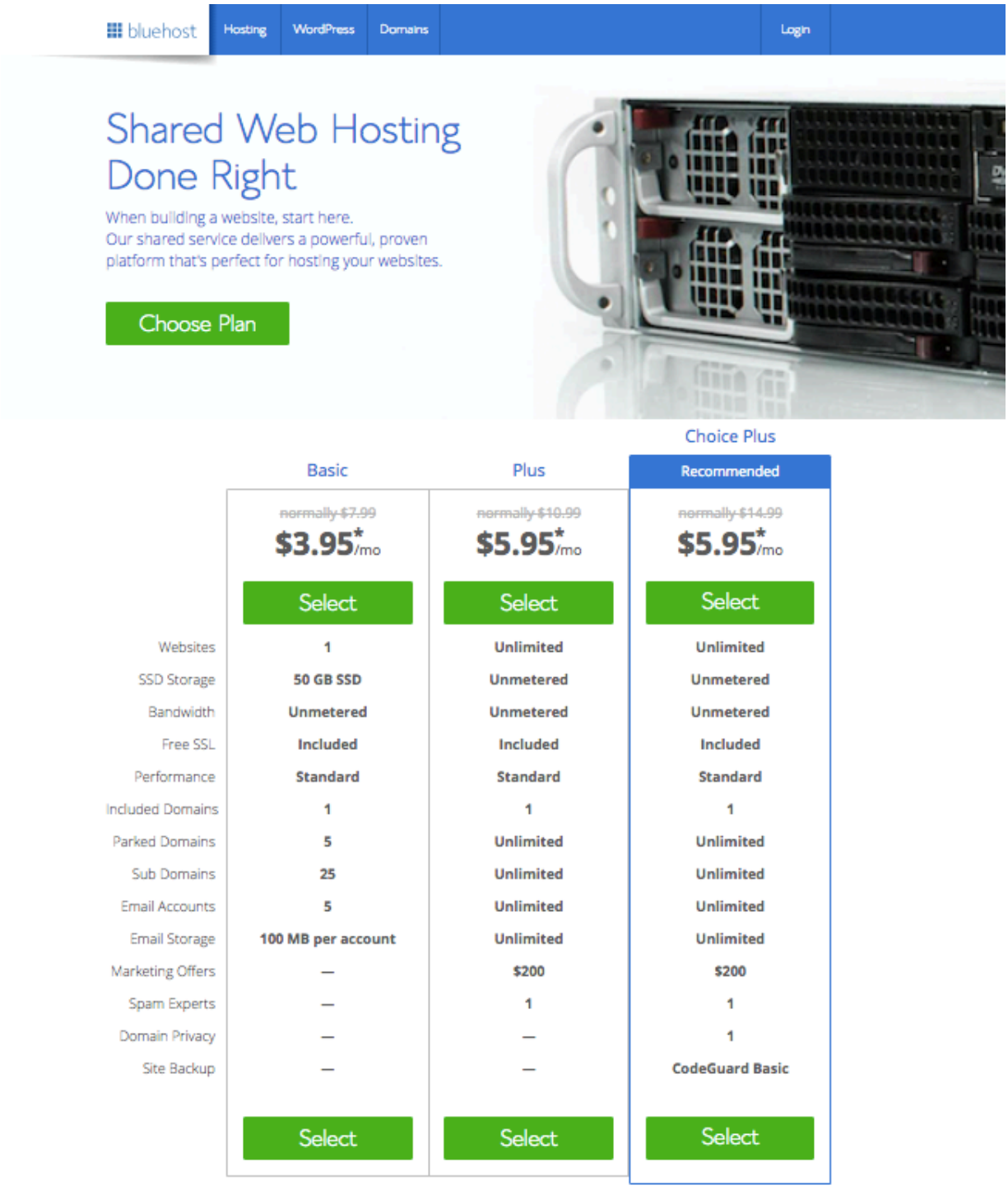
Godaddy shared Hosting plans



Blue Host Shared / VPS / Dedicated Server plans



Blue host shared Hosting plans



From here you will need to know what you need. For example, if you need to host multiple sites you need a hosting plan that allows this. If you need a database you

will also need a plan that allows this (for example if you are going to be hosting a content management system). The amount of storage and bandwidth will also vary with the price. For the sake of this course you will most likely be okay using the cheapest plans with the hosting company of your choosing but I would suggest using the middle one. In the above example the extra dollar isn't much and will allow you more room. I would also suggest to use a hosting package that supports cPanel.

Once you have made a choice, you will need a credit card, paypal or, depending on the hosting company they may allow the use of various payment gateways to be used such as skrill, Union Pay, your checking account, gift cards or even cryptocurrencies.

Hosting payments are usually made on:
A monthly basis
A quarterly basis
A 6 month basis
A yearly basis

Depending on how frequent you pay it may lower the price. For example if you pay every year the monthly price may decrease.

Things you want in a good host

Reliable up time - 99.9% Up time
cPanel

Reliable support - 24/7 365 contactable support
Reasonable Prices
Scalable Plans for future growth
Ability to host more than one domain is a plus
Ability to have/create databases - MySQL (this is done in cPanel)
Linux
Ability to have your own email (IE info@whateverdomainyouregistered.com)

Server Types

Servers are the backbone of every website/ webapplication. There are several types of servers you can work with. This lecture will focus on linux servers which are most common. However, it is good to know that there are options and what those options are, and for future reference out in the field as a developer, designer, webmaster or admin.

Linux/Apache Servers. The most common and what we will be focusing on. Widely used because of the versatility and running PHP with ease. These servers have been the "norm" since 1994/1995 and will be going forward. These servers usually have cPanel as well. Which is also widely used to manage hosting. In cPanel you can create email accounts, manage files on the server and easily accomplish various admin tasks on the server. We will discus this later more in depth.

Windows Servers. These are known for running ASP.net windows servers are not as widely used as linux, but are pretty much just as readily available for purchase. However, the cost of running a windows server may be more expensive and more work. These servers *can* run php, but not as easy and smoothly as a linux server, running php on a windows server will also result in limitations.

Mac Servers. Just don't bother lol. All jokes aside. These servers are out there if you want to check them out through your career. These can run apache but if you are running live sites you may as well just use an actual apache server. They are also expensive to buy from a provider. On the flip side though people use mac servers for home applications and some businesses use it. Chances are you will never need to run or work on a mac server whilst working on an online environment.

Now lets take a look at the types of servers you will come across in your career as a multimedia dev. This is important to know the differences and dabble in using the different kinds of servers down the road as your future employers or business endeavours may call on you to work with or even run one of these.

Shared servers. These are the common server space. A shared server is the cheapest monthly fee and quickest way to deploy a site. But, it has its draw backs. With a

shared server your control will vary. You may have to contact an admin at the hosting provider to allow you to run certain server side scripts. You don't have to worry about this now but later on in your career this is something to keep in mind. You may only be able to run 1 or a specific amount of different websites on the shared server space. So, if you are running a development operation full-time or in your spare time, and have a bunch of clients and you host all of their sites this is probably not going to work for you. You will also have a small amount of web space depending on the package you buy. And lastly, your shared hosting is just as the name states, shared with an array of other people as well as the IP for the server. The shared server is what we will be learning on in this course and the best way to get you acquainted with hosting. Certain shared servers will have cPanel access.

Virtual Dedicated Servers. "VDED" also known as "Virtual Private Servers" or "VPS" are virtual machines so they run on physical machines that have many "virtual machines" on them. In essence its a computer that virtually runs individual "machines" on it. What you the developer gets in the end is your own IP, server and you have total root control over it. The monthly fees are a little more expensive. However, you can host as many sites as you can house for the space you have on your hard drive, you can host your own emails, make your own nameserver, you can power cycle the server when needed and control every and all aspect on the machine. With a machine like this you will be using WHM, cPanel, Plesk or something to

that effect.

Fully Dedicated Servers. A fully dedicated server is the next notch up. Again you have full root access to the software on the machine and the machine itself. The fully dedicated server is your own physical machine with all of the stuff above in the VDED section.

*Note all of these services are paid month to month, some hosting companies will allow you to pay 3, 6 and 12 months at a time as well. Also note that virtual dedicated and fully dedicated servers usually will have an option to have someone at the hosting company manage it for you so that you don't need to learn linux and how to run a server. However, if in your career you choose to be a webmaster, it will be good for you to learn how to.

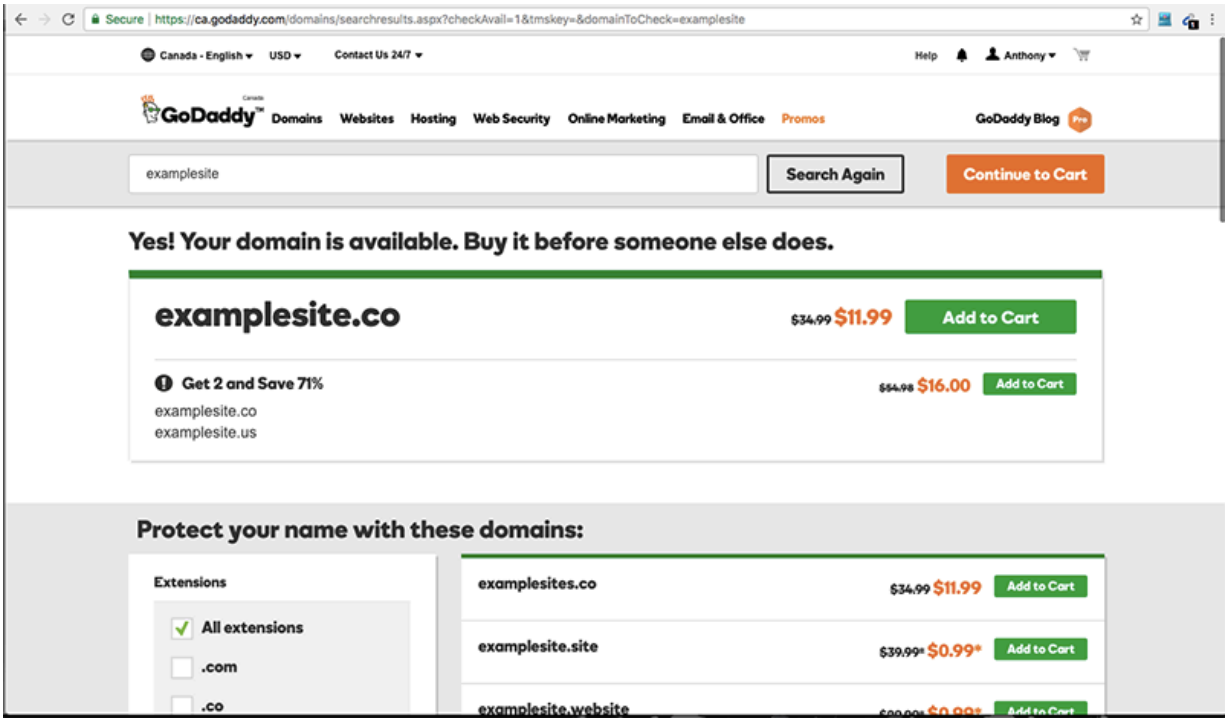
Domains

A domain name is what we use to remember web addresses. Without going into too much detail, we have to register domain names and resolve them to ip addresses or name servers. Domains are registered with a registrar. Many hosting providers will also double as registrars or act as a gateway to a registrar. Having hosting without a domain is possible but for all intents and purposes useless. This is because without a domain name you would be forced to use the IP address of the hosting account to access the resource. It is much easier to work with a domain. Some hosting providers offer deals such as

a free domain with purchase of a hosting account, this will give you the domain for free for the first year, after which you will have a re occurring payment on a yearly basis.

How to register a domain

To register a domain you have to create an account with the registrar (or place you buy your hosting) and go to the domains section. Here you can type in the domain you want and if it is taken already it will notify you and in most cases give you other domains that are similar and available.



Once you have the domain you want you can add it to your cart and complete the purchase.

Choosing a good domain name

Sometimes a client will need help coming up with a good domain name. You should as a webmaster/developer be able to consult with this. As it stands now short domain names are hard to come by as any short domain name is likely to be taken already (1 word, 2 word and many 3 word domains are already taken up some being used and others being parked by people who bought them in order to wait for someone wanting it to buy it from them). You may have to be creative in the process of making new domains. Generally .com .net were the best and to an extent still are in the eyes of some. Most people generally think of a website as a .com but this is changing now that humans are getting used to seeing more diverse TLD's, for example .co is becoming popular. Never over complicate your domain. Stay away from hyphens and numbers if possible. It should be as simple and memorable as it possibly can be. For the sake of this course I recommend something like your first and last name.com

Pointing a Domain

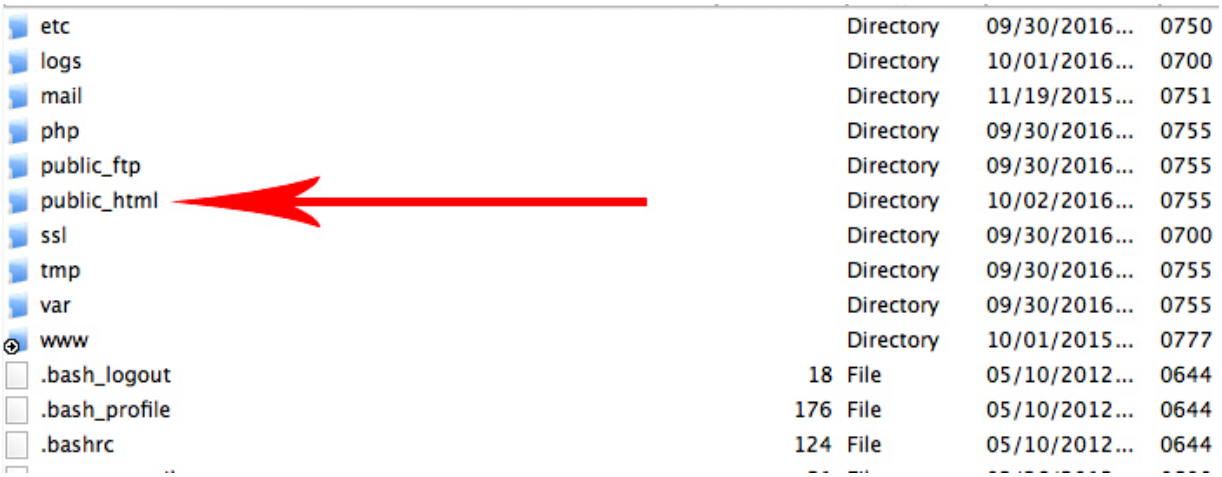
Depending on the set up you can point the domain different ways. In some cases you can just use the name server and in the MX records, enter the NS1 and NS2 addresses and save the record. In other cases you might have to set the A Record and IP (A record hostname being 'www' and the destination IP being the ip address of the server/hosting). We will explore this more in detail in part 2

of this lecture.

Structure

For now you will be learning on a shared server so the rest of this lecture will focus on that. This information should be taken seriously as your multimedia classes and ultimately career will need you to be able to complete tasks on your server to publish them to the web, luckily its not hard at all!

In shared servers you may or may not have access above the public_html folder. You will see this as soon as you login via FTP (File Transfer Protocol). For the purpose of this lecture and application this is the folder we need to concentrate on for now as this is where all the files being published to the web will go. If you are working with a CMS (Content Management System), all the files for the CMS will also go in this folder.



*Note some shared servers use a different name for the public_html folder for but for the most part it will be named “public_html”.
*Note the "WWW" folder or is a mirror of the public_html account. Anything done in one folder will mirror in the other.

Now that we know a little more about our work space on the server we can start uploading things for the world to see. Lets imagine a simple website with 3 pages on it, Home, About Us and Contact us pages.

The home page in a static website is always named "index.html" (index.php if you are running php script) this makes the server and browser know that this is the goto page when someone goes to your URL. In other words, when you go to www.humber.ca the home page is actually www.humber.ca/index.php (because the site is running php. If it was an HTML site the home page/index page would be www.humber.ca/index.html) So this will be the first page we work on and upload.

For the purpose of only uploading a splash page (coming soon page) We could stop here and call it a day. But, in this lecture going forward with our 3 page site we will press on.

Other pages in our website will be built and added following the index.html page and than linked to each other through the main navigation bar. In the end the files on the server (using Filezilla) will look like this.

To keep things organized it is good practice to make a

folder named "images" to add picture files to, they can be used anywhere needed in your sites content from the location in the images folder.

It is also good practice to make a folder called “css”. This folder will contain all the css scripts you will be using (even if there is only one).

At this point we have a working, published site. Properly structured and, when anyone goes to your website URL the content on the index page will be the 1st thing they see.

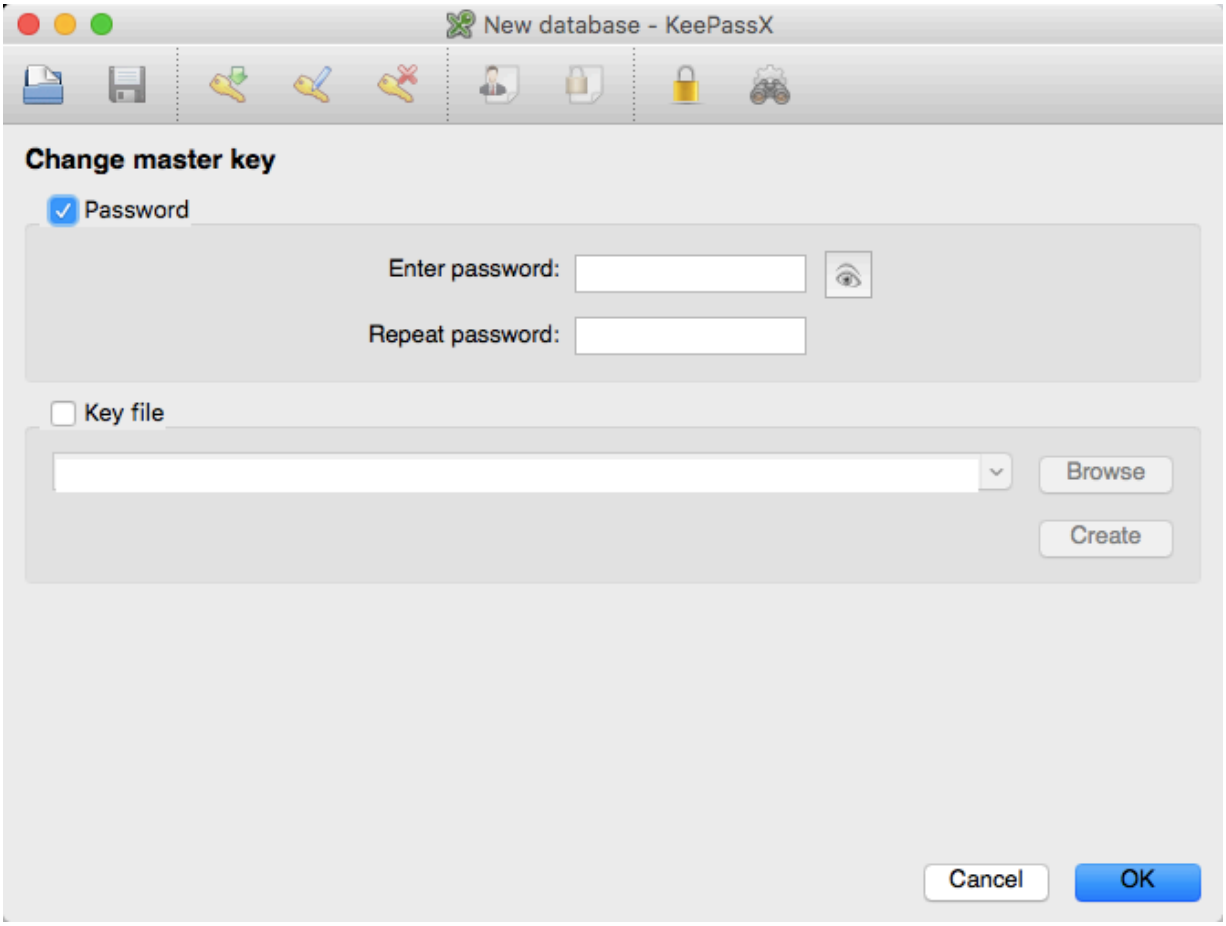
Password storage

Keep your passwords safe and secure, stored all in one location. This way you will not loose them and have them readily available when you need them.

https://www.keepassx.org/

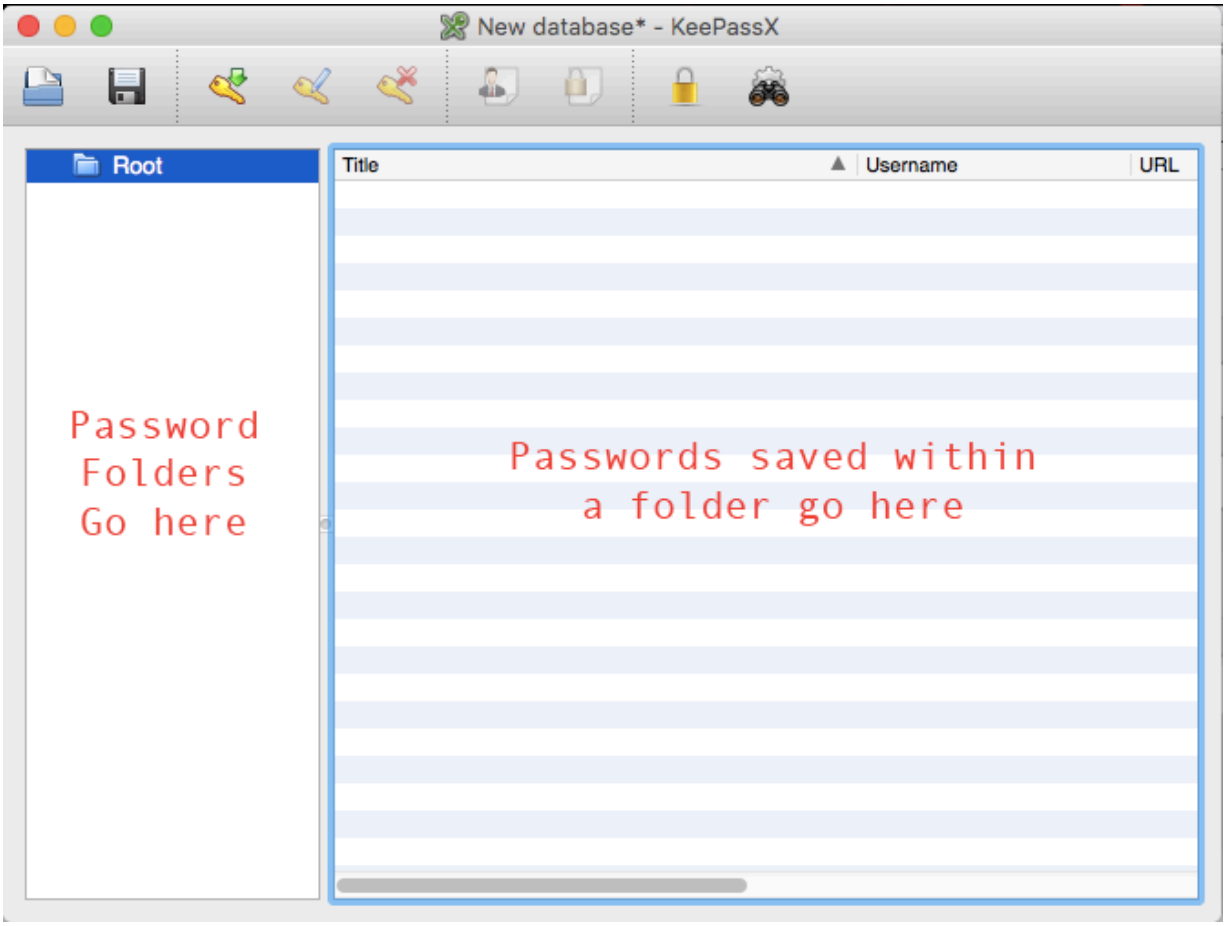
This is one of many available tools to do this.

Once you open the program it will look like this:

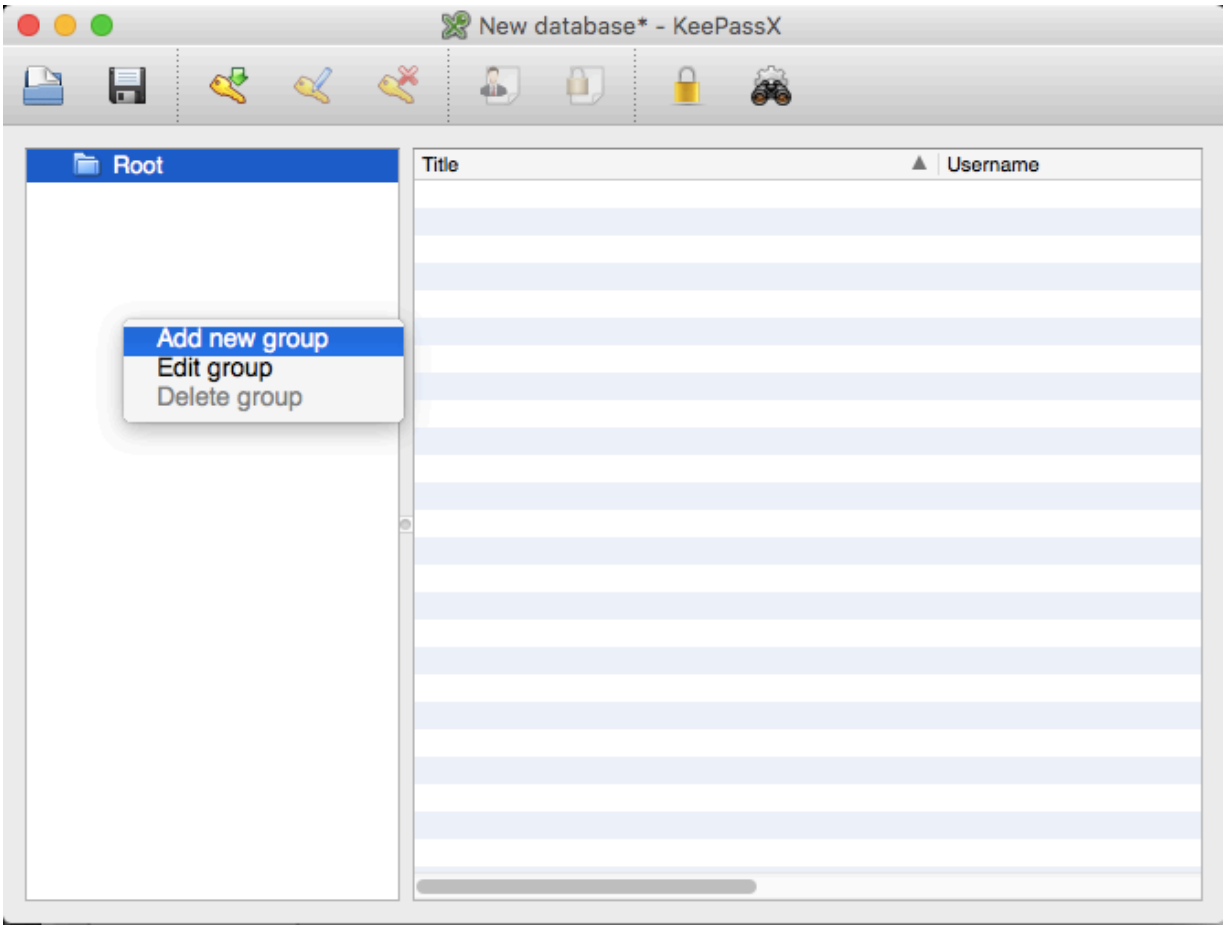


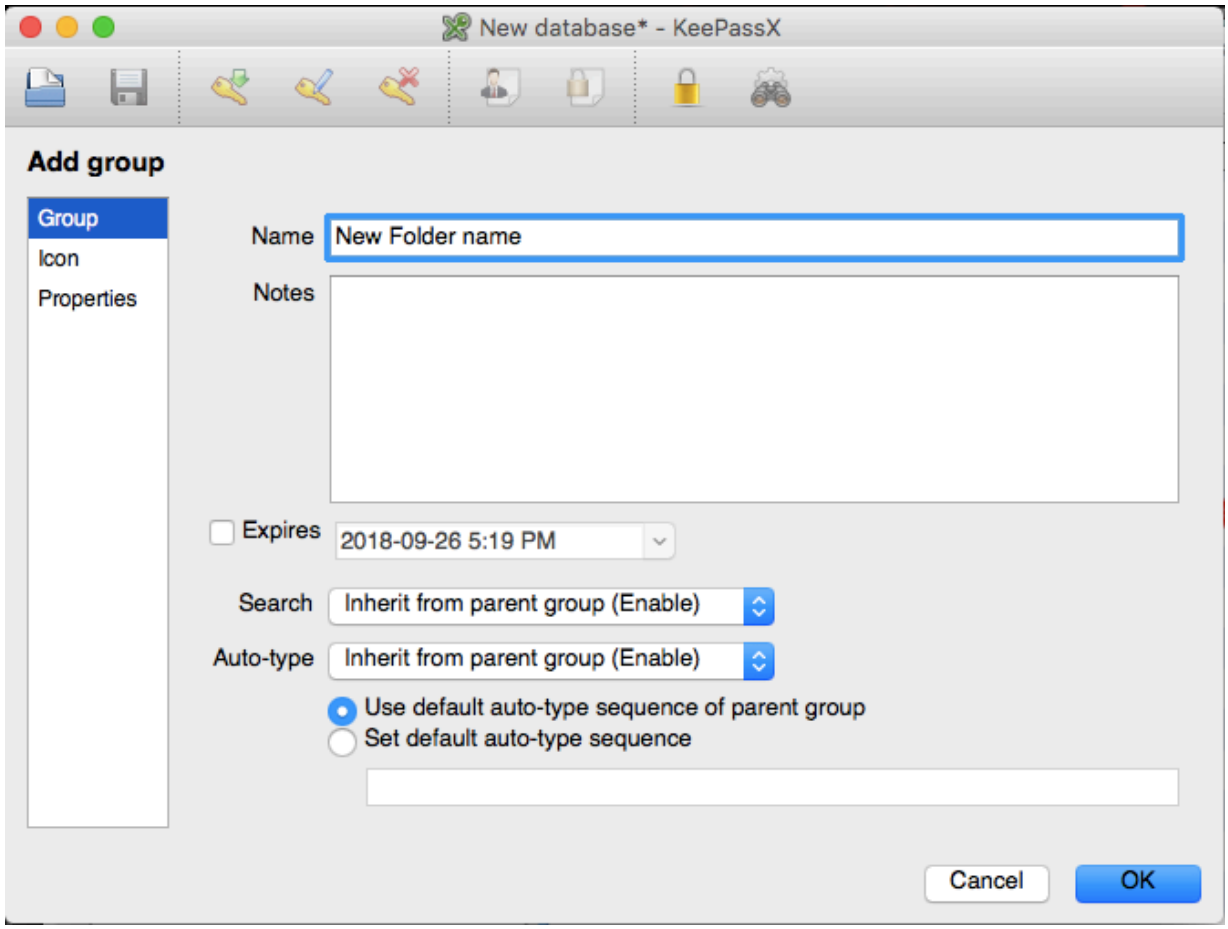
Enter a password and press OK, ignore the "key file" section. Once you do it will open a new database where you can store your passwords.

The GUI will change to look like this:

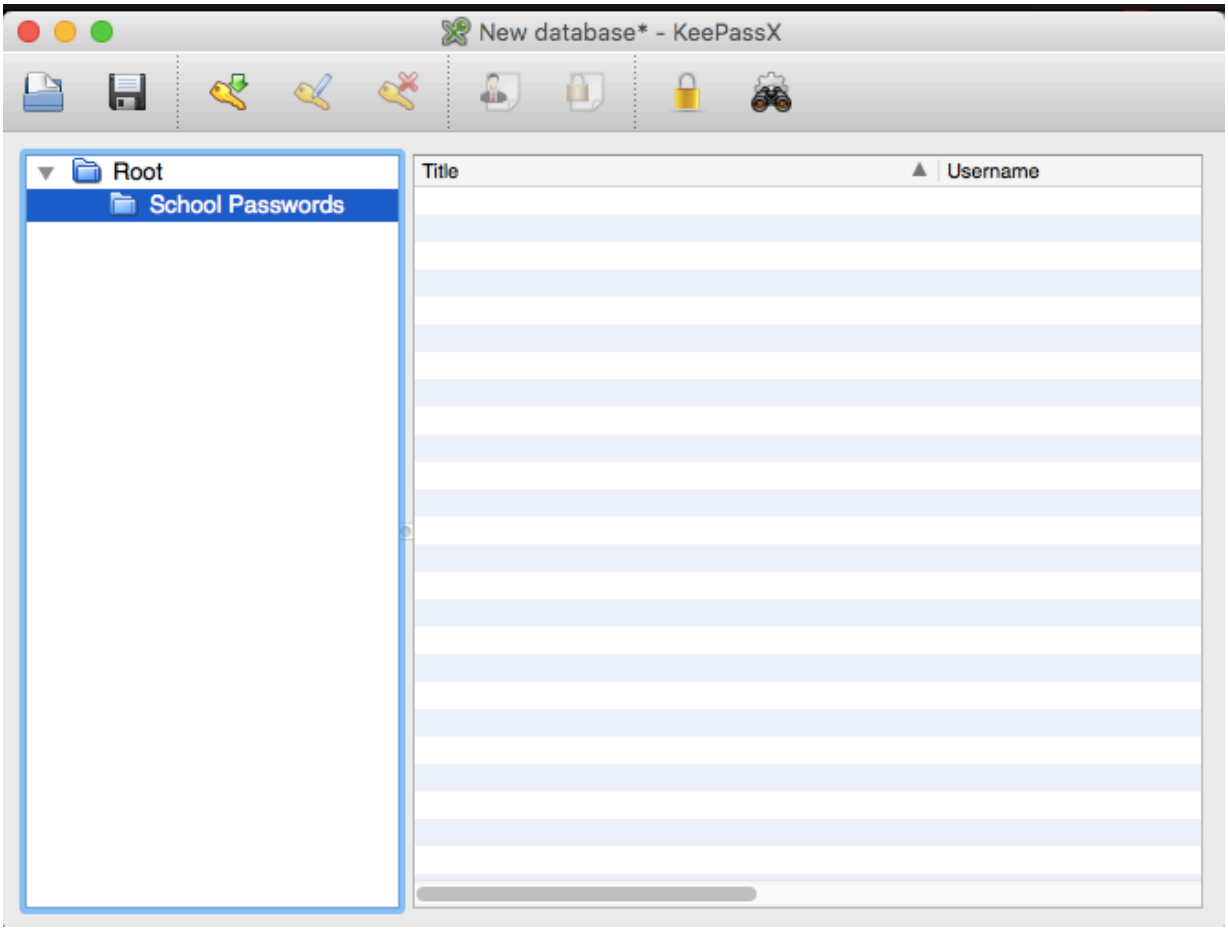


You can now either make a new folder to store passwords, or just start saving them in the root folder. To make a new folder right click on the whitespace and click "add new group". If you want to make a new folder within any other folder, right click on the folder itself and click the same "add new group". Give it a name and press OK

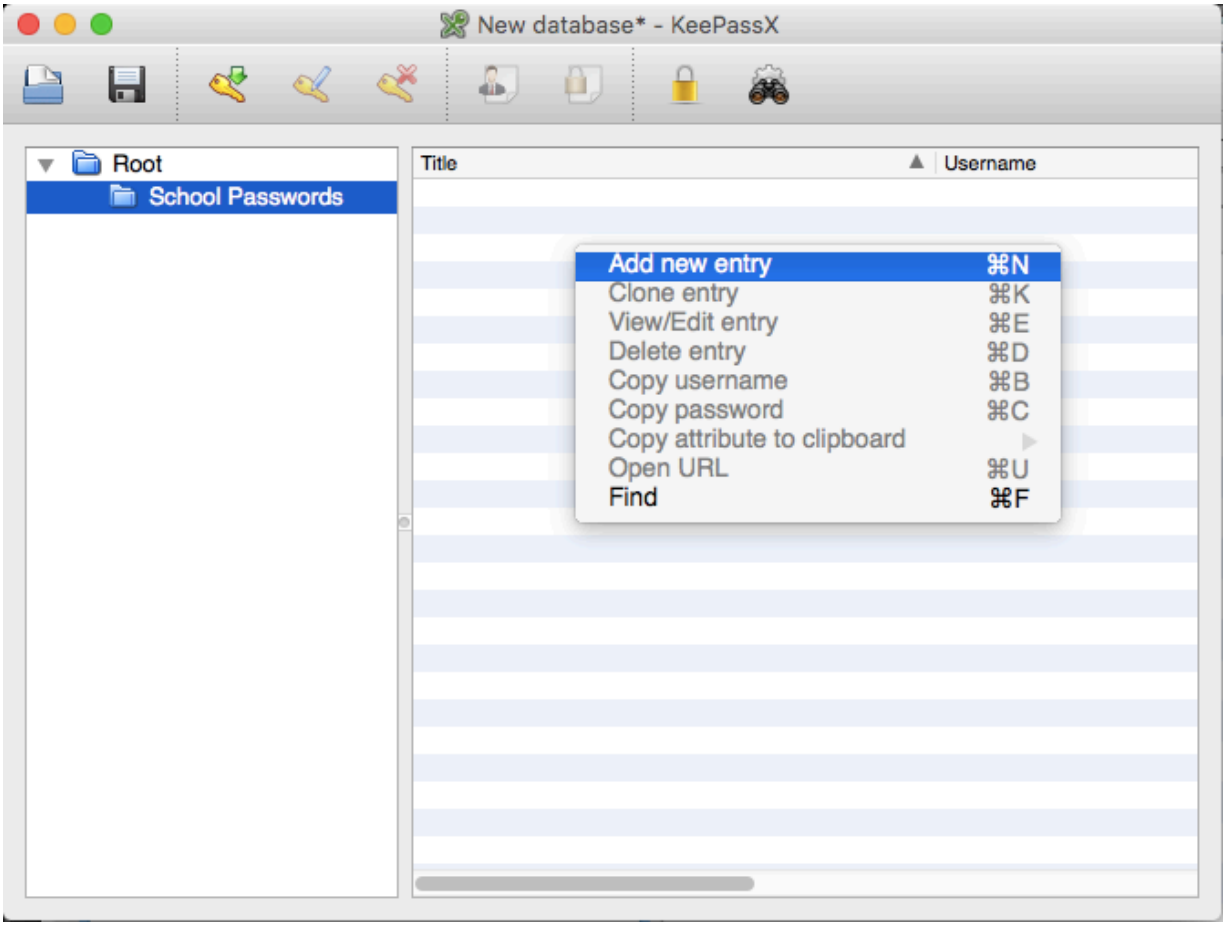

New database* - KeePassX
Root
Title
Username
Add new group
Edit group
Delete group



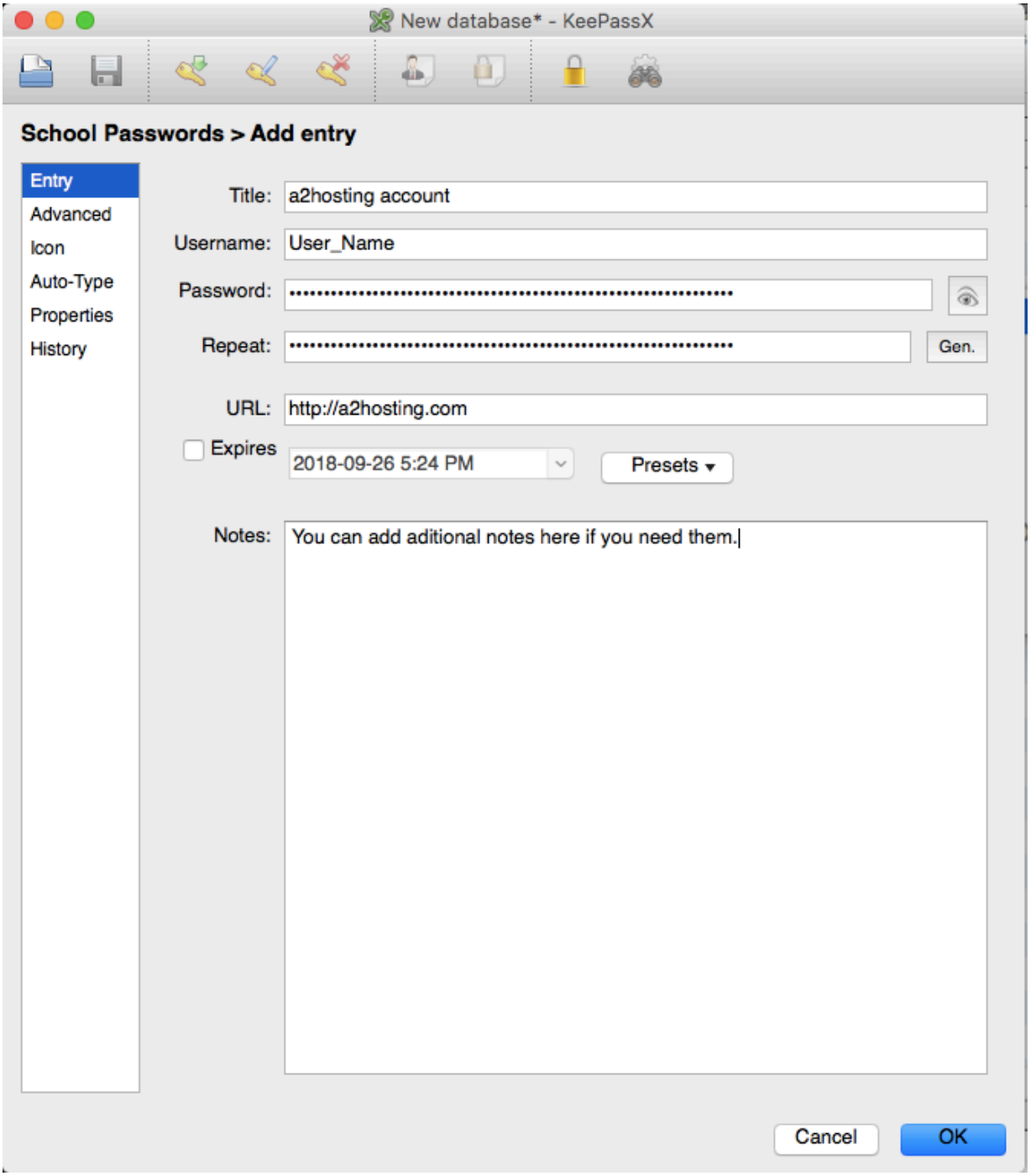
Now we have a new folder to keep things organized.



Now we can add a 1st entry. Right click in the white space on the right side, and click "Add new entry".



The GUI will change to this window:

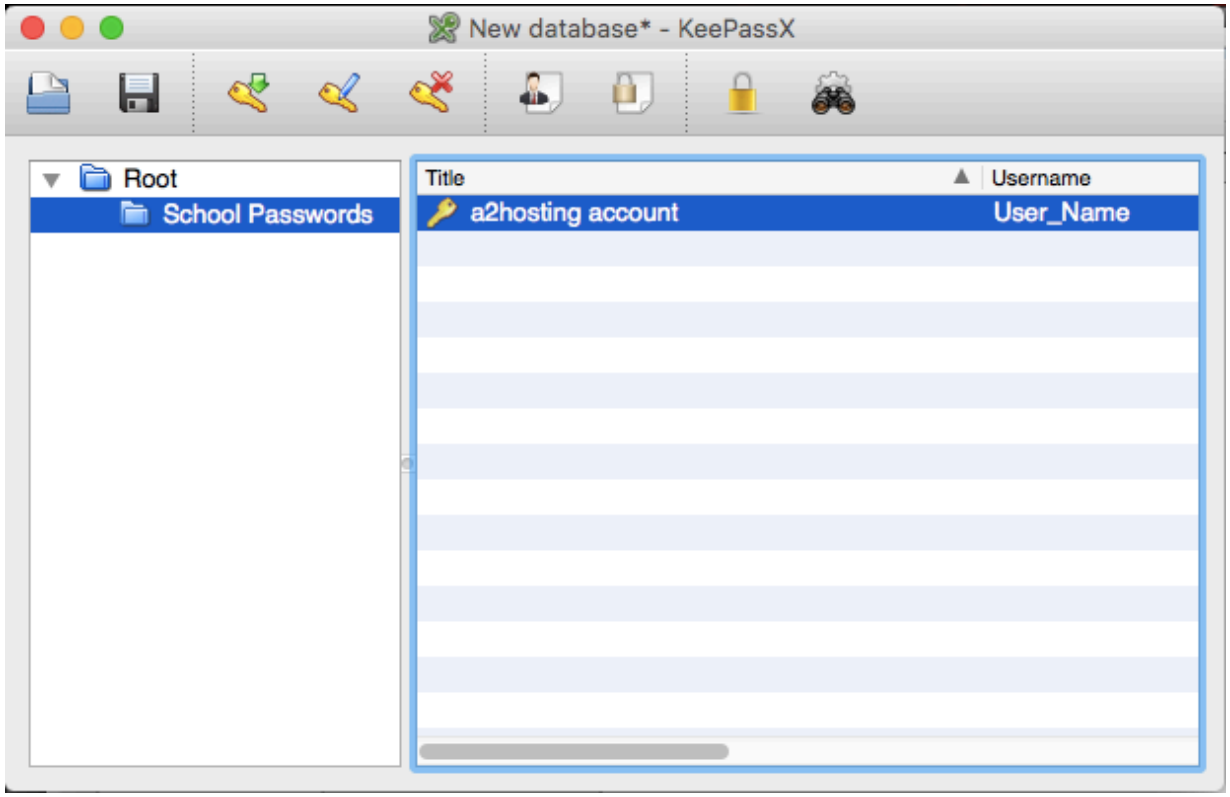


The eye icon will show the password so you can copy and paste it.

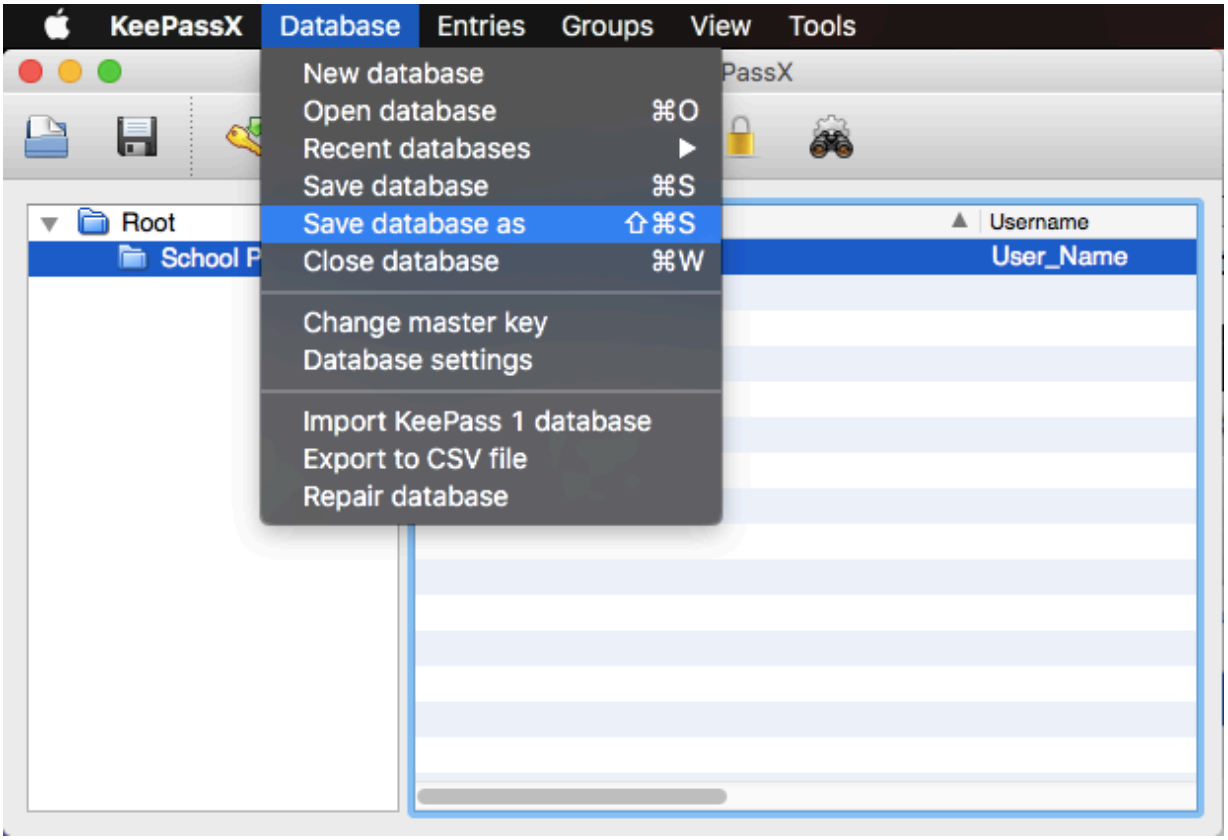
The "Gen" button is a password generator that you can

use to generate strong passwords. You should use it.

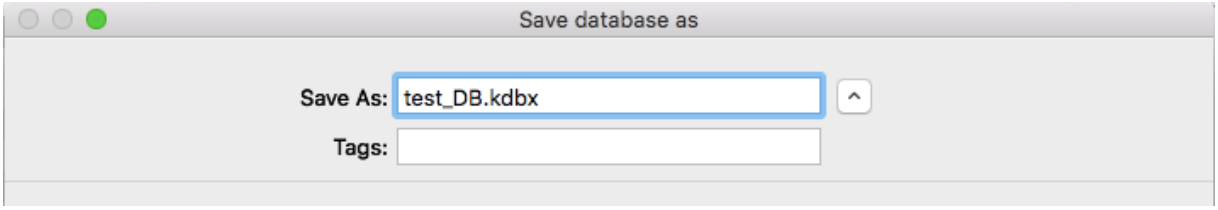
Once done press ok and your password will be saved. You can save as many as you need within any folder.



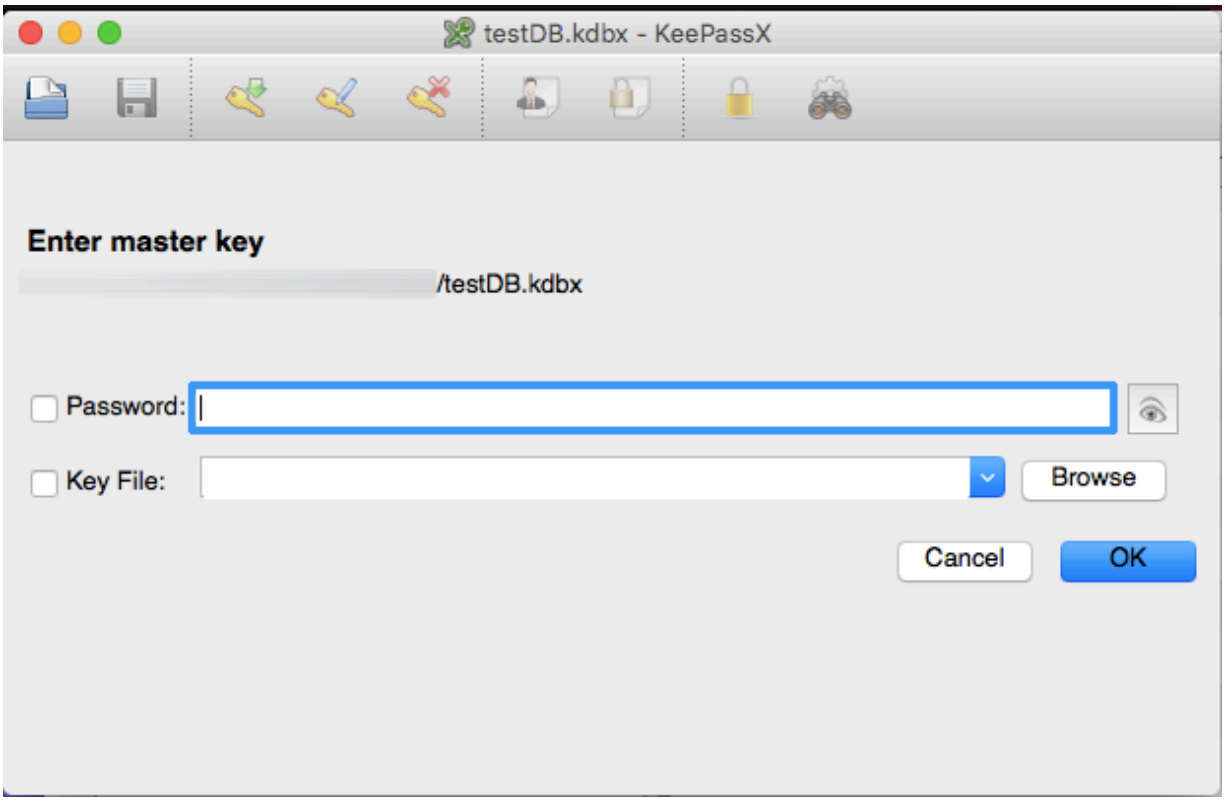
Now you must save the database before closing the program… Go to "Database" and click "save database as"



Save the file in a secure location where you will not loose it. Once saved you may close the program.



The next time you open the program, the database you made should already be loaded (Note the name of the file we named it as in the top bar testDB.kdbx and also under where it says "Enter master key), all you will have to do is enter the password you choose and it will open your password vault. Remember to choose a password you will not forget.


testDB.kdbx - KeePassX
Enter master key
/testDB.kdbx
Password:
Key File:
Browse
Cancel
OK