How to build your personal academic website using free software

Alexa Ruel & Jillian Caplan

March 4, 2021

In this workshop, we will cover the basic steps of setting up your own academic website using Rmarkdown, blogdown and Hugo. Your finished website should contain your biography, a photo, your CV, contact information and recent publications.

Set up

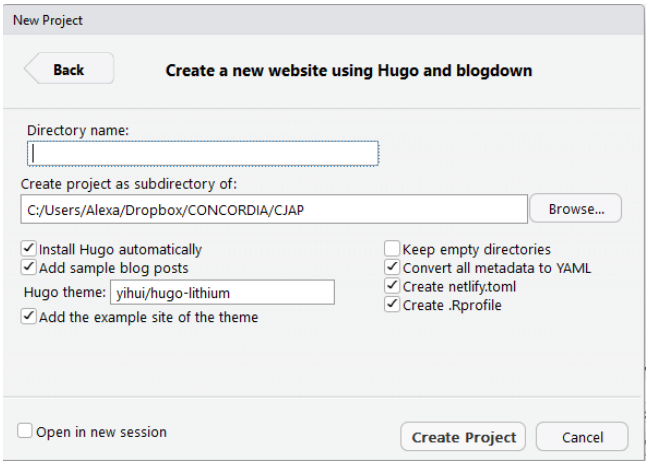
1. Create a Github account if you don't have one already
    a. Go to https://github.com and create an account
    b. Your website will be hosted through this account, therefore give it a meaningful name such as your first and last name.
2. Download R and R studio if you don't have it downloaded already
    a. Got to https://www.r-project.org/ to download R (this is the coding program)
    b. Go to https://rstudio.com/ to download RStudio (this is the coding interface)
3. Install the 'packages' to create your website: Rmarkdown, blogdown and hugo
    a. In the console, type:

```
install.packages("rmarkdown")
install.packages("remotes")
      remotes::install_github('rstudio/blogdown')
blogdown::install_hugo()
```

Choosing and Downloading your Template

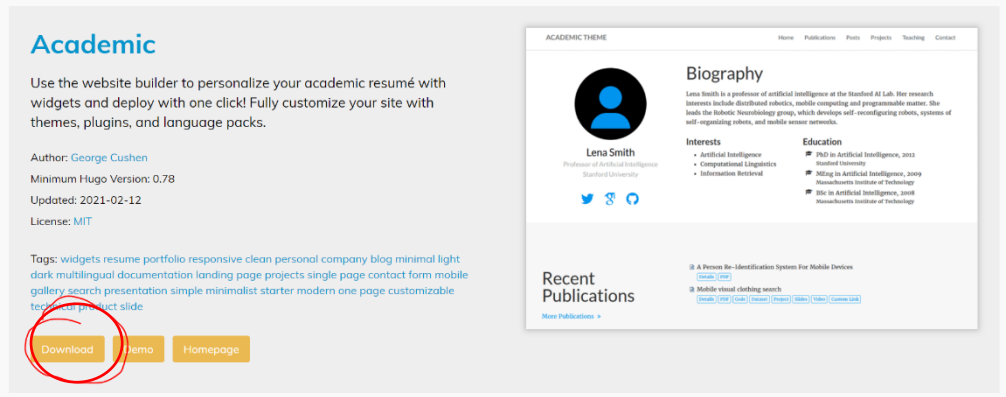
Although you will be using the academic template in this workshop (https://themes.gohugo.io/academia-hugo/) to learn the basics of website creation using R and blogdown, Hugo offers hundreds of free website templates that can be used in the same way. These websites can be found at: https://themes.gohugo.io/

1. Once you have chosen your Hugo template, import the theme into R
    a. File > New Project
    Click on New Directory > Website using blogdown
        i. **Directory name** will become the name of the folder that contains all the content for your website, name it accordingly. (E.g., My Website).
        ii. **Create project as subdirectory of** this is the location where the new directory will be created. To change this path, click on "Browse".

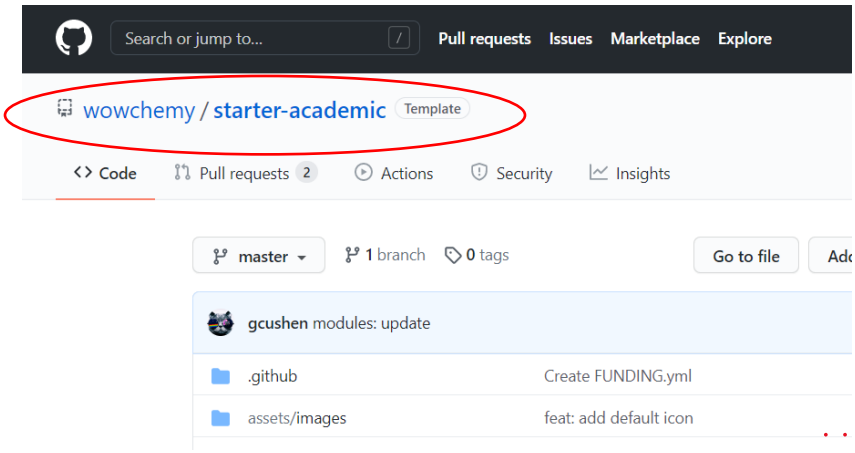


iii. **Hugo theme** this is where the name of the template you have chosen is to be entered. This is retrieved by going to the GitHub page where the template was created:

1. Click on Download from the template's page



2. Rewrite the entire github repository name next to to **Hugo Template**. (E.g., wowchemy/starter-academic). Make sure you don't add extra spaces or it will not work.



3. Then click **Create project**.

2. You now should have all the files for your website in the folder you created called "My Website".
   a. You can view these folders right in Rstudio on the right side of the Rstudio window. This is where all the files you will be editing can be found.

Start Editing

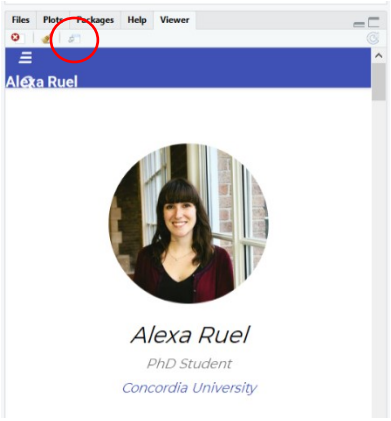
Your finished website should contain your biography, a photo, your CV, contact information and recent publications. However much more than that can be edited here. Feel free to make more edits after you've covered these basics.

Before you start, lets go over how to save edits and view them (you'll be doing this a lot!)

- Each time you make a modification to one of your files, all you need to do is hit the save button at the top of the Rstudio window.
- To see what these changes look like in your website, enter the following command in the console:

```
blogdown::serve_site()
```

- This will serve (or generate) your website locally (visible on your computer only).
    - You will first be able to view the website in the bottom right corner of your Rstudio page under the **Viewer** tab.
    - To open this in your web browser, click on the 'show in new window'.
    - Now you can keep this web browser page open and simply refresh the page when you save new changes!
    - Note: avoid using **blogdown::serve_site()** twice in a row, as it can cause problems. Instead, use **blogdown::stop_server()** to stop it, then serve the website again using **blogdown::serve_site()**.



1. Remove the sections of the website you don't want included in your final website:
    a. **Config folder > _default > menus**
        1. This page codes for the links to the different sections/pages of your website.
        2. To remove a section/page, delete the lines that code for it or turn these lines into comment (by adding a # in front of each line).
            a. **Note**: The CV page is not a web page but the pdf to your CV. To make the CV tab link to your CV pdf
                i. Save your **cv.pdf** to **static > media** in your file explorer, then change the url to "media/yourcvname.pdf"
        3. **Note**: Change #featured under publications to #publications
    b. Config folder > _default
        1. Privacy.md and Terms.md documents exist but are blank in the template email. You can either create your own content in these or delete them if you don't want them showing up at the bottom of your website.

2. Code the basics
    a. In your main folder, you'll find the **config file**. This file codes the basics of your website.
        1. Change Title to your name.
            1. E.g. title = "Alexa Ruel"
    b. In the main folder, you'll find the **config folder**. This has more files that code the basics of your website.
        1. Config folder > _default > config
            1. Title: *put your name here*
                a. This will change the word "Academic" at the top of the website to your name
        2. Config folder > _default > params
            1. Change the color scheme of your website by using one of the themes here: https://wowchemy.com/docs/customization/#color-themes
                a. (here are the details about changing the theme color and font colors of your website beyond the themes: https://rstudio-pubs-

static.s3.amazonaws.com/495947_8a997f234d804fd9b88887812450db60.html)

2. Change your website's font size
3. Enter your **contact details** (line 84)
    a. This information is for the contact section only. We will change the icons under your picture elsewhere.
4. Add your social media handles (line 123)

3. Content
    a. In your main folder, you'll find the **content folder**. This is where you will make most of the edits for your website.

Content > Authors > Admin > index.md

1. This file contains all the information for the **about** section of your website.
    a. **bio**: add a very short description here.
        i. E.g. PhD Student in Experimental Psychology at Concordia university.
    b. **Education:**
    courses: enter your education here
        i. E.g.
        - course: PhD in Experimental Psychology
        institution: Concordia University
        year: 2018-present
    c. **Email**
    d. **Interests** e.g.
    - Decision-Making Strategies
    e. **Organization** e.g.
    name: Concordia University
    url: (link goes here, starting with https://)
    f. **Role** e.g. PhD Student
    g. **Social**: social media links go here.
        i. If there is one you don't want to use or don't have, you can delete the icon, icon_pack and link.
    h. After the dashed line (---) you will enter the text that will appear next to your picture. You can just type the text you want to appear here.
        i. If you want to add a link [text to be linked goes here](link goes here)
        ii. Add link to your CV
            1. Drag and drop your own cv as a pdf to **static > media**
            2. Then change the name of the pdf to the name you called your PDF: {{< icon name="download" pack="fas" >}} Download my CV {{< staticref "media/ARUEL.pdf" "newtab" >}}here{{< /staticref >}}.
    i. **To change the photo to one of yourself**, you will need to replace the avatar.jpg file with the picture you want to use. This is in the authors > admin folders. Keep the name and the file type the same.
        i. You will need to do this in the folder where your website is saved on your computer.

Content > Home

2. Here you will find all the files (e.g. about.md) that guide the academic's page widgets. (e.g., all the sections of the website). If you want to deactivate some of the sections, you need to do so here. *Here is our recommendation as to which widgets to activate for a personal academic website by making active: true (all others you can deactivate by making active: false):*
    a. About
    b. Contact
    c. Publications
        i. Delete the call out notes after the ---
        ii. Here is where you will change how many publications will be displayed on your home page. (line 19)
    d. Skills (*optional, this can be deactivated too*)
        i. Some pages do not have the Activate this page section, you will need to copy the section from another page and make it false here.
        ii. *Exception*: **hero.md**, you will need to add 'active: false' under headless. Do the same for **gallery > index.md**
3. The **about.md, contact.m and publicaitons.md** pages do not contain actual content.
    a. As seen above, the content for the **about section** is edited in **Content > Authors > Admin > index.md.**
    b. The **contact section** is edited in **Config folder > _default > params.md**
    c. The content for the **publications section** is created and edited in **Content > Publication**, which is described below.
    d. The skills section is edited in directly in the **Content > Home > skills.md**

E.g.

```
[[feature]]
icon = "laptop-code"          ← this is the icon for this skill
icon_pack = "fas"             ← package the icon comes from
name = "Programming"          ← name your skill
description = "R, Python, Excel & MATLAB"      ← describe the skill
```

For more icons see:
https://fontawesome.com/icons?d=gallery&p=2
https://jpswalsh.github.io/academicons/

Content > Publication

4. In the index.md file, you can change the number of publications you want to see in the home page of your website.
5. Within the publications folder, you will save each publication as a folder and within it, an index.md file with:
    a. The abstract
    b. The authors (your own name is entered as *admin*)
    c. Publication in (journal name goes here)
    d. Published Date
    e. The title

f. URL PDF: you can link to the article here either as a website link or pdf)
g. URL dataset: you can also link to the data if its publicly available

Build your Website

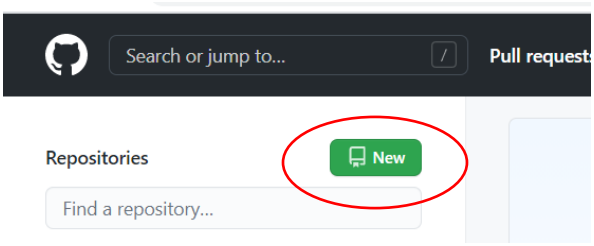
Once you're done editing your website, and it looks good when viewed locally, you will need to build the website. This compiles all your files and prepares the whole thing to be put online. This is done by entering the following in the R console:

```
blogdown::build_site()
```

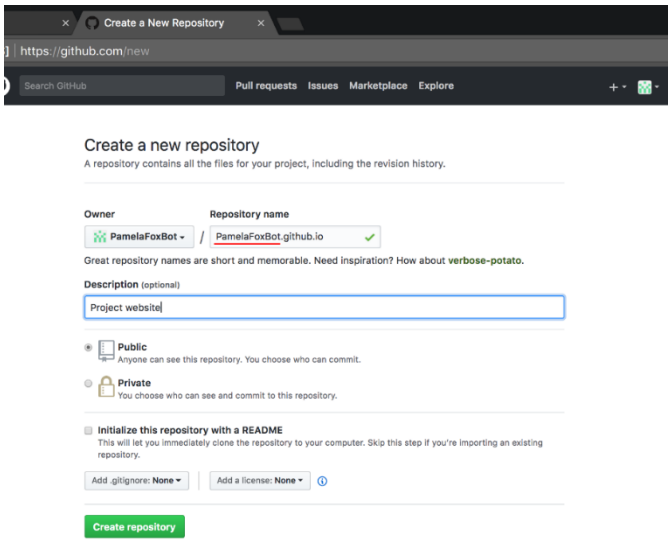
Push your files to a GitHub repository

Now we can add our files to our GitHub page.

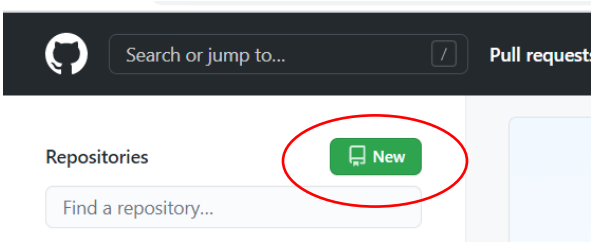
1. **Create a GitHub repository for your website**. There are two ways to do this.
   a. Host the website directly on GitHub. **Note**: you can only do this once on your GitHub account.
      i. Create a new repository



      ii. **Make sure to name your project exactly "YOUR_USERNAME.github.io".** That tells GitHub that you're making a special webpage project, so that it knows to upload your files to that user-facing URL whenever you change them. The description is optional. The repository can either be public or private but either way, only you will be able to make edits to the website. Finally, DO NOT initialize a README file. Click Create repository.



b. Place the files on GitHub, serve the website on Netlify. In this method, we create a repository that will not serve as a website on its own; it will not end with 'io'.
    i. Create a new repository



    ii. Using this method, you can call the repository whatever you would like, but still do not initialize a README file.

2. **Getting your files into the repository** you created is the same regardless of which method you followed above. However, this is more complex than just drag and dropping them to your GitHub repository and depends on if you are using a MAC or PC computer.
    a. On PC
        i. Download git bash here: https://git-scm.com/downloads
        ii. Open git bash from your website's **public folder** (right click somewhere within the folder and click "Git Bash here". (this will make sure the path is already set to the public folder)
        iii. Now you can enter a series of commands in GitBash to push changes to your GitHub page. **Note**: they should be entered one at a time and have to be exactly as written below (including spaces).

```
git init
```

The command above created a git folder in the public folder which will allow you to push all the website content to your GitHub repository. Therefore, only needs to be done once.

```
git remote add origin https://github.com/alexaruel/alexaruel.github.io
```

Here the https:// link is the link to your repository.
This tells your computer where its going to push your files to. This also will only need to be done once.

```
git add .
```

Note: the space between add and . is necessary here.

This compiles which files are new and therefore which files need to be pushed to the GitHub repository. If this is your first time pushing this should be all your website files. This is the first step for any additional times you are pushing updates to your GitHub page.

```
git commit -m "initial commit"
```

This adds the comment you add between the quotation marks to your GitHub repository so you can see which edits were made when and what was done, so try to make the comment as descriptive as possible without being long. E.g., "CV updated".

```
git push --set-upstream origin master
```

This does the actual pushing, moving files from your website's public folder to your GitHub repository. This will open a pop-up window to have you enter your GitHub credentials. This ensures that if someone else did all the other steps, only you know the username and password to your GitHub account and therefore, only you can make these changes.

b. <u>On MAC</u>

i. Download Xcode here: https://git-scm.com/downloads This is a third party program that will allow you to do the same steps as on a PC.

ii. Open your computer's **terminal** and change the current path to where you saved the public folder of your website.

1. First check your current path using pwd (which means print working directory) then change this using cd.
    a. Note that cd will add to the current location. For instance, if pwd is Users/YourName and your public folder is in Users/YourName/Desktop/website you need to do: cd Desktop/website to change the path to this public folder.

iii. Now you can enter a series of commands to push changes to your GitHub page. (see PC section: these are the same commands)

3. **If you used a github.io repository**, then you're done! You should now be able to view your website at yourusername.github.io (see alexaruel.github.io). If the website does not work yet, give it some time! Sometimes, the first push can take up to 24h.
    a. If after 24 hours, if you still cannot view your website at yourusername.github.io or the formatting it off, try the next step, using Netlify.
4. **If you did not use a github.io repository**, you will now need Netlify to deploy (to be able to view your website).
    a. Create a Netlify account at https://www.netlify.com/
    b. Create a **New Site from Git,** then click on **GitHub**

i. This step is easiest if you are already logged into your GitHub account. You may need to validate your log in (security measure)

c. Select the repository you want to sync with Netlify (to deploy). Netlify will generate a random and very specific custom URL for your website. You should now be able to view your completed website at this location! You're done!

Buying a Custom Domain Name & Applying to your new Website

If you've followed the steps above, you likely have a website ending with github.io OR some custom domain name generated by Netlify. You can of course keep your website's domain name as is, or pay for a custom domain name and have the domain be whatever you want it to be. Here is an example of how you can do that.

1. Purchase your domain name such as https://ca.godaddy.com/
2. Link it to your github.io or Netlify website by following the steps outline below
    a. Linking to GitHub: https://medium.com/@JinnaBalu/godaddy-domain-with-github-pages-62aed906d4ef
    b. Linking to Netlify: https://levelup.gitconnected.com/netlify-custom-domains-8b4cc5fddb5d
    c. There are many other sources online that can help you link your custom domain to your website if needed.

Please contact us if you have any questions
www.concordiapsychjournals.ca
CJAP: journalofaccessiblepsych@gmail.com
CJPN: concordiajpn@gmail.com

Follow us on Social Media to stay up to date!


CJAP: @AccessiblePsych
CJPN: @ConcordiaJPN


CJAP: @AccessiblePsyc
CJPN: @ConcordiaJPN