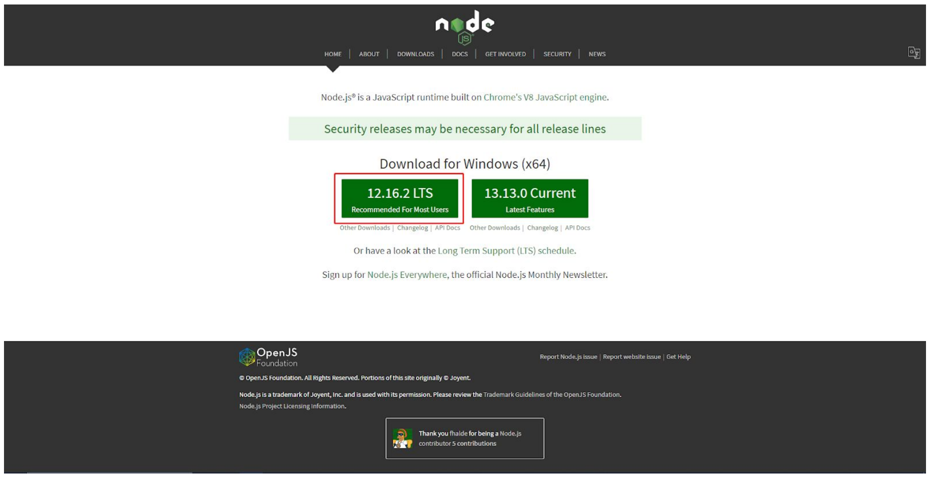
Junior Achievement of Armenia Website Manual

Produced by James Kajon, Zhongchuan Xu, Rozi Aloyan, and Bailey Berg

Getting Started

Download the **LTS** version of Node.js from nodejs.org/en/ and install it.


HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | NEWS
12.16.2 LTS
Recommended For Most Users
13.13.0 Current
Latest Features
Other Downloads | Changelog | API Docs
Other Downloads | Changelog | API Docs
OpenJS Foundation
Report Node.js issue | Report website issue | Get Help

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine.

Security releases may be necessary for all release lines

Download for Windows (x64)

Or have a look at the Long Term Support (LTS) schedule.

Sign up for Node.js Everywhere, the official Node.js Monthly Newsletter.

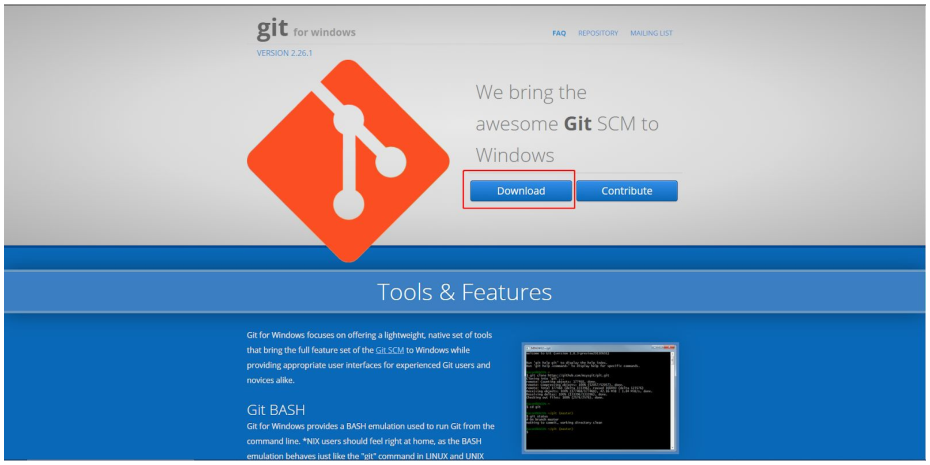
© OpenJS Foundation. All Rights Reserved. Portions of this site originally © Joyent.

Node.js is a trademark of Joyent, Inc. and is used with its permission. Please review the Trademark Guidelines of the OpenJS Foundation.
Node.js Project Licensing Information.

Thank you fhalde for being a Node.js contributor 5 contributions

Open node-vxx.xx.x-x64.msi on windows or node-vxx.xx.x.pkg on mac and install. Run the default install with clicking ‘Next’ all the way, then ‘Install’, and ‘Finish’ at the end. For windows, accept Node.js making changes to your device.

If you want to use git for version control but don’t have it installed, you install it for windows from gitforwindows.org with the following instructions. On mac you can follow the instructions at atlassian.com/git/tutorials/install-git. You, however, do not need to use git.



We bring the awesome **Git** SCM to Windows

Git for Windows focuses on offering a lightweight, native set of tools that bring the full feature set of the Git SCM to Windows while providing appropriate user interfaces for experienced Git users and novices alike.

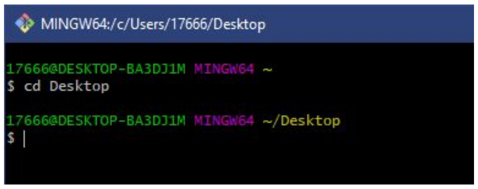
Git BASH

Git for Windows provides a BASH emulation used to run Git from the command line. *NIX users should feel right at home, as the BASH emulation behaves just like the "git" command in LINUX and UNIX

Open Git-x.xx.x-64-bit.exe install package. Run the default install by clicking ‘Next’

We have made a github account for you which is where the code is located. You can sign in at https://github.com/jaarmenia/jaa-website with the username and password we have sent you. From this same url you can use the git bash to clone the github repository.

First open git bash. Clone the repository to Desktop (or elsewhere) by typing ‘cd Desktop’ and press Enter key on keyboard.

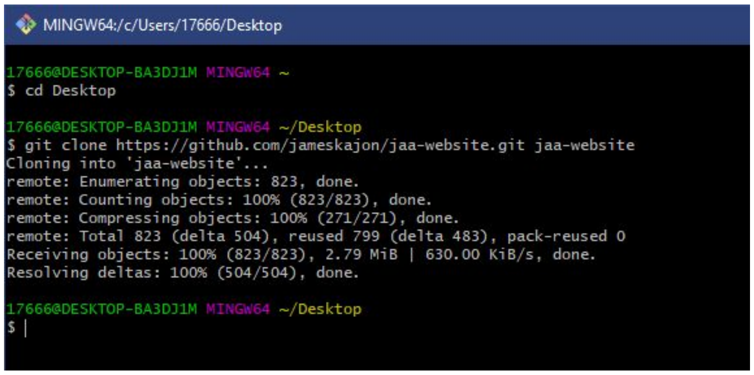


```
MINGW64:/c/Users/17666/Desktop

17666@DESKTOP-BA3DJ1M MINGW64 ~
$ cd Desktop

17666@DESKTOP-BA3DJ1M MINGW64 ~/Desktop
$ |
```

Type in ‘git clone https://github.com/jaarmenia/jaa-website’ and press Enter key to clone the repository to a folder called ‘jaa-website’, and wait until the process to finish. If a window asks you to sign in use the github credentials.



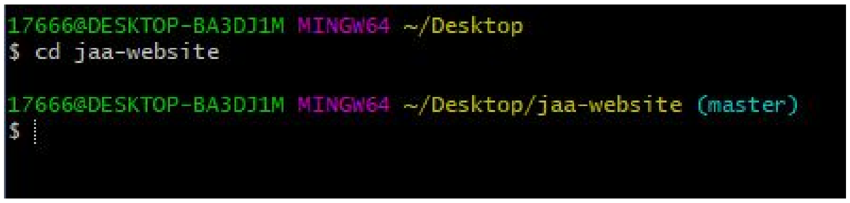
```
MINGW64:/c/Users/17666/Desktop

17666@DESKTOP-BA3DJ1M MINGW64 ~
$ cd Desktop

17666@DESKTOP-BA3DJ1M MINGW64 ~/Desktop
$ git clone https://github.com/jameskajon/jaa-website.git jaa-website
Cloning into 'jaa-website'...
remote: Enumerating objects: 823, done.
remote: Counting objects: 100% (823/823), done.
remote: Compressing objects: 100% (271/271), done.
remote: Total 823 (delta 504), reused 799 (delta 483), pack-reused 0
Receiving objects: 100% (823/823), 2.79 MiB | 630.00 KiB/s, done.
Resolving deltas: 100% (504/504), done.

17666@DESKTOP-BA3DJ1M MINGW64 ~/Desktop
$ |
```

Go to folder ‘jaa-website’ folder by typing ‘cd jaa-website’ and press the Enter key.

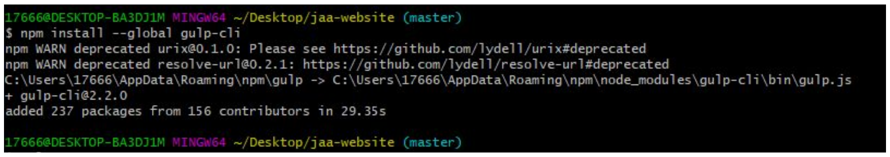


```
17666@DESKTOP-BA3DJ1M MINGW64 ~/Desktop
$ cd jaa-website

17666@DESKTOP-BA3DJ1M MINGW64 ~/Desktop/jaa-website (master)
$ |
```

More detailed instructions can be found in the additional sources section under Git / GitHub.

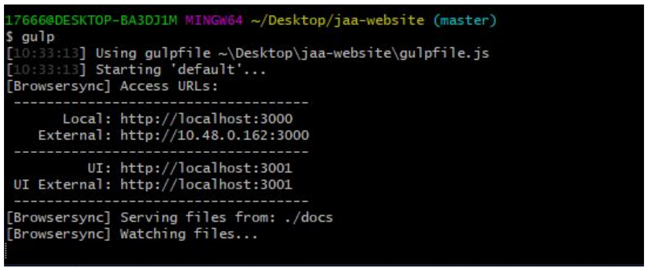
From the command line, while in the ‘jaa-website’ folder, run the command `npm install` and then `npm install --global gulp-cli`. This will install all dependencies (all the other programs needed).



```
17666@DESKTOP-BA3DJ1M MINGW64 ~/Desktop/jaa-website (master)
$ npm install --global gulp-cli
npm WARN deprecated urix@0.1.0: Please see https://github.com/lydell/urix#deprecated
npm WARN deprecated resolve-url@0.2.1: https://github.com/lydell/resolve-url#deprecated
C:\Users\17666\AppData\Roaming\npm\gulp -> C:\Users\17666\AppData\Roaming\npm\node_modules\gulp-cli\bin\gulp.js
+ gulp-cli@2.2.0
added 237 packages from 156 contributors in 29.35s

17666@DESKTOP-BA3DJ1M MINGW64 ~/Desktop/jaa-website (master)
```

Now that everything is installed. You can run a developer server with the command ‘gulp’. This will compile the code, watch for changes to recompile, and open a window in your web browser with the current version of the website. This is not something available on the internet, only your computer and LAN. You can learn about the gulp commands at Gulp Commands.



```
17666@DESKTOP-BA3DJ1M MINGW64 ~/Desktop/jaa-website (master)
$ gulp
[10:33:13] Using gulpfile ~\Desktop\jaa-website\gulpfile.js
[10:33:13] Starting 'default'...
[Browsersync] Access URLs:
 ------------------------------------
       Local: http://localhost:3000
    External: http://10.48.0.162:3000
 ------------------------------------
          UI: http://localhost:3001
 UI External: http://localhost:3001
 ------------------------------------
[Browsersync] Serving files from: ./docs
[Browsersync] Watching files...
```

If you want to stop gulp, you can simply use Ctrl+C to stop the program.

Now download a text editor of your choice, the two we used while building the website were WebStorm and Atom. Sadly, Webstorm doesn't not have a free community edition but you can download a 30 day trial from jetbrains.com/webstorm. We recommend the free editor Atom found here atom.io.

After installing either software, go ahead open the ‘jaa-website’ folder.

Top File Level Descriptions

Now that you have the repository and installed Node.js, the top level of the ‘jaa-website’ folder should contain the following described folders and documents.

In this manual, when we say compile, we are talking about the process of moving the source code in the ‘src’ folder to the ‘docs’ folder and the changes made during that process. Compiling will need to take place after any change of the code is able for the changes to take place in ‘docs’.

docs

This folder contains the compiled version of the website which is used to host a developer server. This folder or its contents should not be edited as all of it is automatically generated from compiling the website. Any changes here may be overwritten by the compiling.

gulp

This folder contains additional javascript to compile the website.

node_modules

This folder was generated by Node.js and contains all installed node packages. This folder or its contents should not be edited. Adding more packages should be done though running the ‘npm install <package_name> --save-dev’ command. Read more about this command here: docs.npmjs.com/downloading-and-installing-packages-locally.

src

This folder contains the source code of the website. This is where all the edits to the content of the website will take place.

.gitignore

This is a file for git to ignore certain files and directories that should not be added to git. One example is the ‘.DS_Store’ file. This is a file used by macOS to manage the file directory. It, however, is not relevant to the website, so it is ignored by git as defined in this file. You can learn more about a gitignore here: git-scm.com/docs/gitignore. The files we decided to have git ignore are: any ‘.DS_Store’ file, the ‘.idea’ used by some IDEs to store user settings, the ‘node_modules’ folder and all of its contents, the ‘deploy’ folder and all of its contents.

gulpfile.js

This file contains the code to compile the website. It also uses the code from the ‘gulp’ folder to achieve this.

package.json

This file contains information about all the node packages used by this project. This file is how the ‘npm install’ command knows what to install. More package information will be added automatically when you use the ‘npm install <package_name> --save-dev’ command. Use the save dev option instead of save as all node dependencies for this project are for development, not production.

package-lock.json

This file is another automatically generated file by node. It tracks the packages currently installed in the project. It should not be manually edited or commited to git.

The following are also top level folders or files but will not yet be there after the getting started instructions are completed.

deploys

This folder contains all deploys of the website with time stamps. A deploy is simply a zip of the docs folder after it has been compiled. They can be generated by the ‘gulp deploy command’.

Source Code Structure and What Compiling It Does

The ‘src’ folder contains two folders ‘public’ and ‘views’. The public folder contains the documents that the browser uses besides the html. And the ‘views’ folder contains almost all the code to compile the html (‘events.js’ and ‘news.js’ in ‘src/public/js’ are also used).

Inside of ‘public’, the ‘js’ folder holds four files: ‘date-layout-data-management.js’, ‘events.js’, ‘news.js’, and ‘main.js’. These three files all work to build the ‘News’ and ‘Events’ sections on the ‘News and Events’ page and the ‘Latest News’ section on the home page. Both ‘events.js’ and ‘news.js’ contain the data used to load in the news and events. The ‘date-layout-data-management.js’ works with the data from those two files to handle showing more news or events on a button press and also showing the full news or event if the news or event is not a link to another site. The way these three files work together is described more in the Adding News or Events section.

The compiling of these files includes minification and transferring them to the same place in the ‘docs’ folder (‘docs/public/js’). The minification takes the js file and makes it as small as possible. This happens in the ‘compileJs’ function in ‘gulpfile.js’ using the ‘gulp-minify’ node module. If you would like to remove minification for whatever reason this can be done by deleting the following lines in that function. Documentation for ‘gulp-minify’ can be found here npmjs.com/package/gulp-minify.

```
.pipe(minifyJS({
   ext:{
       min:'.js',
   },
   noSource: true,
}))
```

Also inside of ‘public’, the ‘css’ folder holds five files: ‘nav.css’, ‘footer.css’, ‘modal.css’, ’layouts.css’, and ‘main.css’. ‘nav.css’ and ‘footer.css’ contain all the css for the navigation bar and the footer respectively. The styles for the modal used to show inpage news stories or event descriptions are in ‘modal.css’. The rest of the styles for the page are split between ’layouts.css’ and ‘main.css’. The difference in the way things are organized between these two files is that ’layouts.css’ contains style for things which we consider a layout, while ‘main.css’ gets the rest which tend to only focus on one element. Our definition of a layout for this was styles that affect a list of elements in a repeating pattern, meaning it doesn’t just give styles to one element on its own but describes styles for multiple elements together. For example, what in code is called the image content layout (selector: ‘.img-content-layout’), is responsible for the sections of the ‘About Us’, ‘Programs’, and ‘How Can I Help’ where the image is displayed inline with the text while alternating the order of the image and text. ‘main.css’ is separated by general styles on multiple pages at the top and page specific styles towards the bottom starting with the home page. The other layouts include the grid item layout which is used on the partner page for the section with the partner logs, the date item layout (selector: .date-item-layout), which is used on the 'News and Events' page to show a date, image and blurb for each new or event item, latest news layout (selector: .latest-news-layout) is used to format the 'Latest News' section on the home page. The simple simple form layout (selector: .simple-form) is used to style the mailing list form on the partners page, and the testimonials layout (selector: .testimonials-layout) styles the testimonials used on the home page and 'programs' page.

The compiling of these files is similar to that of the javascript files, it makes a minified version in the same place in the 'docs' folder ('docs/public/css'). It, however, also used autoprefixer to automatically add support for older web browsers and 'gulp-concat' to merge all of the separate files into one named 'style.min.css'. This is done to increase the loading time of the website by having fewing requests per page load. The compile process is written in the 'compileCss' function in 'gulpfile.js'.

The css minification is done by 'gulp-clean-css' with documentation at npmjs.com/package/gulp-clean-css. 'autoprefixer' documentation can be found at github.com/postcss/autoprefixer#readme. 'autoprefixer' is a plugin for 'postcss', documented at github.com/postcss/postcss#readme

Inside of 'public/assets', there are four folders: 'audits', 'icons', 'newsletters', and 'images'. 'audits' contains the 2017 financial report linked to on the donation page. 'icons' contains the images and files which browsers use to show the JAA logo and colors on various things including the icon in the top-left corner of the tab and a smartphone homescreen icon. This is described more in Updating the Favicon. 'newsletters' contains all newsletters which are linked to on the 'News and Events' page. In 'assets', 'image' holds the majority of the contents. As the name suggests, this folder holds all the images.

'public/assets/image' has 11 folders in it. Many of the folders contain the images that the page with that same name uses, these are: 'about-us', 'alumni', 'donate', and 'programs'. The 'ja-icons' folder contains the icons from JA Worldwide which we have used. The icons can be found from the JA Worldwide media center (jaworldwide.org/mediacenter) at dropbox.com/sh/twywl4xsnlag2jp/AABu0jDzKvUUtECCIPVMH_9Qa?dl=0. The 'social-icons' folder contains the social media images which are used in the footer. The 'news' and 'newsletters' folders contain the images used in the 'News' and 'Newsletters' sections of the 'News and Events' page. 'partner-logos' contains the images shown on the partnership page of all the partners also including the JHM logo is also shown on the home page. Explain the purpose of the two other folders, 'jaa-logo-resize' and'name-resize', is a bit more complicated. The folders don't provide structure to organization of the images but instead mark the folders they contain for specific types of resizing. More about resizing can be found at How to Resize Images. In addition to resizing, the compiling process will also minify them using 'gulp-imagemin' (npmjs.com/package/gulp-imagemin) by removing any unnecessary metadata before finally moving them to the 'docs' folder.

Inside of 'views', the 'en' and 'hy' hold the english version and armenian version of the pages. These are .hbs files which uses handlebars templating (handlebarsjs.com/guide/) to define html. These files get compiled with 'gulp-compile-handlebars' (npmjs.com/package/gulp-compile-handlebars). One of the very useful things templating allows is to define some html in one place and then use it on all or in some of your pages. This describes handlebars's layouts and includes. The site's layout is at 'views/layouts/base-layout.hbs' This file defines the head and body tags used on each page. It adds the includes of the header and footer and defines a block for where the rest of the page should go called 'content'. There is also another block defined for js code named 'js' which will place all js at the very end of the body. This is an ideal location as the page has already loaded here so using the js 'onload' function is no longer required.

```
{{> includes/nav-bar }}
```

The above handlebars syntax in 'base-layout.hbs' is the inclusion of the nav bar. This means this line will be replaced with the contents of the 'partials/includes/nav-bar.hbs' file.

```
{{#> content}}
{{!-- Content of the page --}}
{{/content}}
```

The above handlebars code in 'base-layout.hbs' defines the content block in the code that later gets filled in by a block definition in each page.

```
{{#> base-layout
       title="Welcome" }}
   {{#*inline "content"}}
       <h1>Welcome</h1>
   {{/inline}}
{{/base-layout}}
```

The above handlebars code is an example welcome page which uses the base layout and defines the content block. The parameter to the base layout of 'title' will be used in the text of the tab. In this example the tab title will become 'JAA - Welcome'. The page will have a footer and navbar since that is included in the base layout. But the space between those will be blank besides the heading saying welcome.

There is one additional file in 'views', 'redirect-to-index.html'. Since the pages of the website are in the 'en' folder, this document will make sure that if a user goes to '/index.html', they will then be redirected by this file to the english index page.

Gulp Commands

Gulp is a Node.js module used to process website files. Gulp plugins (also Node.js modules) can be used for things such as restructuring files, resizing images, and many others. In this project we use gulp for the image resizing, handlebars compiling, and minification mentioned in the previous section. This is all done though the script written in ‘gulpfile.js’. This script exposes three functions that can be run as commands: ‘watchBrowser’, ‘compile’, and 'exportProject'. The commands can be run in the command line while in the ‘jaa-website’ folder. ‘watchBrowser’ is the default command meaning it can be run using the command ‘gulp’. The other two can be run using ‘gulp compile’ and ‘gulp export’. Running ‘gulp compile’ will compile the contents of the ‘src’ folder meaning it will modify them in the ways described in Source Code Structure and What Compiling It Does and put the new versions into the ‘docs’ folder. Running ‘gulp’ will run the compile command and also use a Node.js module called Browsersync to launch a developer server and open it in the default web browser. The ‘gulp’ command will not exit after running as it tries to update changes to the code in the browser by recompiling and refreshing the browser. Changes to ‘gulpfile.js’ or any of the files in the ‘gulp’ folder will need a restart of gulp to take place. This can be done by pressing Ctrl+C, then running the ‘gulp’ command. Running ‘gulp export’ will also first run the compile command and then move the contents of the ‘docs’ folder to a zip file in the ‘deploys’ folder. This zip file can then be moved to the server to update the website.

Mailing List Form

The responses to the form to add yourself to the alumni mailing list can be found in the new gmail account Anahit has made. They go into the ‘Mailing List Data’ sheet you can find in the google drive of that account. Instructions on how this was done can be found at github.com/jamiewilson/form-to-google-sheets.

Adding News or Events

‘src/public/js/news.js’ and ‘src/public/js/events.js’ store all the news and events on the page. The system for adding these are the same. We will describe here how to add a news item.

All of the news items are in the news array in ‘news.js’. Below is an example news item.

```
{
   date: {
       large: "May",
       small: "2020",
       sub: "",
       title: "May 13, 2020",
       datetime: "2020-5-13",
   },
   blurb: "This is a news story that you should read",
   imageUrl: "../public/assets/images/news/a-cool-img.jpg",
   link: {
       title: "Read more about this news story",
       url: undefined,
   },
   modal: {
       heading: "A News Story",
       content: [
           `Welcome to our news story`,
           `Welcome to the second paragraph`,
       ]
   },

},
```

The above shows all the fields of the news story. For the date, ‘date.large’ is the big value in green while ‘date.small’ is the smaller value in black in the date item layout. ‘date.sub’ is the text that goes underneath both of those. For news stories, we used the three letter month for ‘large’ and the year for ‘small’ while ‘sub remained empty’. For event’s we recommend using the day for ‘large’, the month for ’small’ and the day of the week written out for ‘sub’. ‘date.title’ will show on a hover of the date. We used the fill written out date as this is a format which could not be confused like the american mm/dd/yy format could be. The ‘datetime’ is added for accessibility reasons. This value will help screen readers read the date. Examples of the format of this value can be seen here: developer.mozilla.org/en-US/docs/Web/HTML/Element/time#Valid_datetime_Values.

The ‘blurb’ is the text on the page which can be clicked on to show the load the full item. In most cases this should be the headline or title of the item.

‘imageUrl’ is the link to the image used on both the page and the modal. This can be either a url from the internet or the server.

The ‘link’ key holds the title for the link which is shown on mouse hover and should say something like “Read more”. The ‘url’ key is the link should be the link to the news article if one is available. If not, it should be undefined which will mean a modal is opened on click instead of a link. Modals can be read about here: w3schools.com/w3css/w3css_modal.asp.

The content of the modal that is opened is defined in the ‘modal’ key. ‘heading’ is the heading of the modal which in many cases could be the same as the blurb as the article title. The ‘content’ is used to store the content of the modal if the content is just paragraphs of text (the image will also be present on the modal). If the content needs to be more complicated than this, html can be used instead though the ‘html’ key of ‘modal’. Below is an example of this

```
{
   blurb: "This is an event without a image or date but it's got html!",
   link: {
         title: "Read more",
         url: undefined,
   },
   modal: {
         heading: "HTML!",
         html: `<h3>This is a special item</h3>
<div><p>It uses divs and <a href="#">links</a></p></div>`
   },

},
```

Another feature shown here is that this element doesn't have an image or date. If this happens the blurb will keep the left and right padding but on mobile there will not be any empty space shown from the missing elements.

If a link is used, the ‘modal’ key does not need to be defined. Shown below, is an example news item using a random article: reuters.com/article/us-fish-hibernation/scientists-find-hibernating-fish-in-antarctic-idUSL049441120080305.

```
{
   date: {
       large: "Mar",
       small: "2007",
       sub: "",
       title: "March 4, 2008",
       datetime: "2008-3-4",
   },
   blurb: "Scientists find hibernating fish in Antarctic",
   imageUrl:
"https://s1.reutersmedia.net/resources/r/?m=02&d=20080305&t=2&i=3399576&r=2008-
03-05T013138Z_01_L0494411_RTRUKOP_0_PICTURE0&w=640",
   link: {
       title: "Read more at reuters.com",
       url:
"https://www.reuters.com/article/us-fish-hibernation/scientists-find-hibernatin
g-fish-in-antarctic-idUSL049441120080305",
   },
},
```
The ‘imageUrl’ here is taken from the first image shown with the news article. This will make the loading of the news and events slightly slower so it would be a good idea to copy this image to the server instead and link to it there.

How to Change the Logo

Open file ‘src/views/partials/includes/nav-bar.hbs’, towards the top, you can see a line that says ‘<div class="logo">’ (‘div.logo’). Here, you can see the highlighted code is referring to pictures of logos for both English and Armenian, and you can see where the picture is stored.

To change the logo, simply go to the folder where the picture is stored, which is ‘src/public/assets/images/jaa-logo-resize’. You can replace the images with the names ‘jaa-logo-en.png’ and ‘jaa-logo-hy.png’. However, if you would like to use a different name, you will also need to change the html of the nav bar. This can be done by changing the img tag ‘src’ attributes with the parent of ‘div.logo’. Remember to still use the logo suffix of ‘-size-l’ described in How to Resize Images. Also if you update the logo to a new design, you should also update the favicon so they are consistent. This is described in the Updating the Favicon section.

```
<div class="left-menu">
   <div class="logo">
       <div>
           {{#if_eq lang "en"}}
               <a href="index.html" title="Home">
                   <img src="../public/assets/images/<NEW NAME
HERE>-size-l.png" alt="JAA">
               </a>
           {{else if_eq lang "hy"}}
               <a href="index.html" title="Գլխավոր էջ">
                   <img src="../public/assets/images/<NEW NAME
HERE>-size-l.png" alt="JAA">
               </a>
           {{/if_eq}}
       </div>
   </div>
</div>
```

Updating the Favicon

The favicon is the image you see in the top left corner of a web browser tab. It is meant to help the user tell which tab is which more easily. The favicon used for the website was generated from a version of the JAA logo without the text. This file can be found in the website repository at `src/public/assets/icons/jaa-logo.png`. The logo was generated by realfavicongenerator.net. Besides making the image for the web browser, it also makes images files used by iOS and Android if the user adds the site to their home screen, Windows Metro where the site is pinned to your desktop, and macOS Safari for the pinned tab and touch bar. The following settings were used for this:

- Desktop Browsers
  - Use the original image as is.
- iOS Homescreen
  - Add a solid, plain background to fill the transparent regions.
    - Color: #fdb614 (ja primary amber)
    - Margin size: default of 4
- Android Homescreen
  - Add a solid, plain background to fill the transparent regions.
    - Color: #fdb614 (ja primary amber)
    - Margin size: default of 8
  - App name: JA Armenia
  - Theme color: #fdb614 (ja primary amber)
- Windows Metro
  - Color: Yellow
  - Use the original favicon as is.
- macOS Safari
  - Turn your picture into a monochrome icon.
    - Threshold: default (whatever it takes to get the yellow triangles to go away)
  - Theme color: #00763d (ja primary forest green)
- Generator Options
  - I cannot or I do not want to place favicon files at the root of my website. Instead I will place them here
    - ../public/assets/icons

You can change one of these options by redoing the process with the new options and generating. Once on the `Install your favicon` page, click on the `HTML5` tab and follow the instructions to download the files. Once extracted, they should replace their counterparts in the ‘src/public/assets/icons’ folder. Then, copy and paste the code given over the previous version in ‘src/views/layouts/base-layout.hbs’. It should go directly after the viewport tag.

Adding and Changing a Heading / Subheading

To add a heading or subheading, you need to use a text editor and open ‘gulp/nav.js’. Inside of this file there is an array called ‘nav’ which stores the text on the nav bar.

```
const nav = [
   {
       page: "index",
       heading: {
           en: "Home",
           hy: "Գլխավոր էջ"
       }
   },
   {
       page: "about-us",
       heading: {
           en: "About Us",
           hy: "Մեր մասին"
       },
       subheadings: [
           {
               en: "History",
               hy: "History hy",
               refId: "history"
           },
           {
               en: "Our Vision and Mission",
               hy: "Our Vision and Mission hy",
               refId: "mission"
           },
           // ...
       ]
   }
]
```

This is responsible for generating the headings and subheadings displayed in the navbar. ‘page: “index”’ means the heading will link to the file ‘index.html’ generated from ‘index.hbs’ (the home page). ‘heading: {en: “Home”, hy: "Գլխավոր էջ"}’ are the words used the navbar for a heading. ‘en’ and ‘hy’ correspond to English and Armenian subheading names as they are the language codes for english and armenian respectively.

To add or change a heading, simply go to the correct place for the headings (it is in order). You can use the existing code as a template, write or change the code accordingly. Remember, if you want to create a new heading, you need to add a new page to and link the heading to the page to make sure the website functions correctly. More detail could be found in section How to Add a New Page.

To add a subheading, you can use the 'programs' page as an example. You can change or add the subheading here.

```
{
   page: "programs",
   heading: {
       en: "Programs",
       hy: "Ծրագրեր"
   },
   subheadings: [
       {
           en: "Student Programs",
           hy: "Student Programs hy",
           refId: "studentPrograms"
       },
       {
           en: "Program Success Stories",
           hy: "Program Success Stories hy",
           refId: "progSuccessStories"
       },
       {
           en: "Alumni Success Stories",
           hy: "Alumni Success Stories hy",
           refId: "alumniSuccessStories"
       },
       {
           en: "Testimonials",
           hy: "Testimonials hy",
           refId: "testimonials"
       },
       {
           en: "Teacher Programs",
           hy: "Teacher Programs hy",
           refId: "teacherPrograms"
       },
   ]
},
```

‘refId’ is the id of the heading to go to. ‘refId’ could be anything, as long as there is a matching id in the html of the page linked to. For example, in ‘program.hbs’ in the ‘src/views/en’ folder. You can find the corresponding id to the ‘Student Programs’ subheading is ‘studentPrograms’.

```html
<h2 id="studentPrograms" class="header">Student Programs</h2>
```

How to Change or Add a Photo

Changing a photo is easy, if you know which photo you want to change, you can simply go to folder ‘src/public/assets/image’. All photos, icons and design elements are in the folder. You can change the existing photo and replace it with a new one with the same name and file extension.

If the file extension is different or you want it to have a different name, you can go to the html to change the section which refers to the photo in the ‘images’ folder.

To add a new photo, you need to go to the code segment for content in .hbs files for the desired page. For example, if you want to add a photo for a new employee, you can go to ‘about-us.hbs’. At the bottom of the page, we can find the description and a link to the photo for the employee. You can use the existing code as template, copy and paste at the bottom.

```html
<article class="img-content-wrapper">
   <div class="img-content-img-wrapper">
       <img src="../public/assets/images/about-us/team/Narine
Grigoryan-size-icl.jpg" loading="lazy" alt="Narine Grigoryan">
   </div>
   <div class="img-content-content">
       <p><strong>Narine Grigoryan</strong> has been JAA's representative in
<strong>Vayots Dzor</strong> region since 2019. Prior to joining the
organization, she worked in Yeghegnadzor's No. 1 school as a history and social
studies teacher. She has participated in many educational trainings and
community initiatives.</p>
   </div>
</article>
```

Drop the picture you want to add in the desired folder. Here, the desired folder is the ‘src/public/assets/images/about-us/team/img-content-resize’ folder, where all the pictures of the team belong.

Change the file path in the img src from ‘Natalya Tumanova’ to whatever is in the name of your picture. Of course here we would also want to edit the description of the employee if you want. You can find more detail about how to edit the written content in the How to Edit the Written Content section.

After you make any changes, save the file and use gulp to view the result. For more detail about how to use gulp, see Getting Started section.

How to Edit the Written Content

Changing the written content on the website is easy. If you know which sentence or paragraph you want to edit, you can simply go to the .hbs file for that specific page, in the 'src/views/en' folder.

For example, if I want to change the description for Economics Program under programs page. I need to open the 'programs.hbs' file in a text editor. By scrolling down or searching the text you will find the code below. It contains the header, an picture of the economics program and the actual description of the program.

```html
<h3 class="header">Economics Program</h3>
<article class="img-content-wrapper">
   <div class="img-content-img-wrapper">
       <img src="../public/assets/images/programs/economics-size-icl.jpg"
alt="Economics">
   </div>
   <div class="img-content-content">
       <p>Junior Achievement of Armenia's (JAA) economics program teaches
students to understand and appreciate free enterprise and entrepreneurism. As
high school students begin to position themselves for their future, there are
many unanswered questions about what lies ahead. JAA's economics course helps
students make informed, intelligent decisions about their future, and fosters
skills that will be highly useful in the business world and global economy.</p>
       <p>JAA accomplished their goals through curriculum-aligned textbooks,
community outreach, and extracurricular programs. Students use Junior
Achievement's internationally-recognized Applied Economics textbook that has
been translated into Armenian and edited to complement JAA's curriculum and
represent the local experience. Volunteer business people, including local
Rotarians, are brought into the classroom to share their real-life experiences
with students. Each year, JAA's acclaimed summer camp is offered to the
program's highest achievers, and the summer study abroad program is offered to
Armenia's top high school students.</p>
   </div>
</article>
<hr>
```

To change the description of the program, simply edit the white text. However, one thing to notice is that each paragraph needs to be started with <p> and end with </p>, so the program knows the text is a stand alone paragraph and will be displayed on the website. You should also be aware that by default, whitespace (spaces, tabs, newlines) doesn't work as usual. Inside a paragraph, any of these whitespaces will just show as a single space on the page.

How to Add a New Page

To add a page in the website, we need to add both a .hbs file in the correct folder and a new header in the navbar. To find more detail in how to add a new header in the nav bar, please see the Adding and Changing a Heading / Subheading section. You need to add a new object to the nav array . For example, here is a new header called extra page which will appear on the nav bar between alumni and partners.

```
{
   page: "alumni",
   heading: {
       en: "Alumni",
       hy: "Շրջանավարտներ"
   }
},
{
   page: "extra-page",
   heading: {
       en: "Extra Page",
       hy: "Extra Page hy"
   },
},
{
   page: "partners",
   heading: {
       en: "Partners",
       hy: "Գործընկերներ"
   },
},
```

The other thing is to add a new .hbs file in the correct folder, in this case, ‘src/views/en’ and ‘hy’. Create a new file with .hbs extension, and the file name should be the same with the name you put in ‘page:”extra-page”,’ line. In this case, the name of the file should be called ‘extra-page.hbs’.

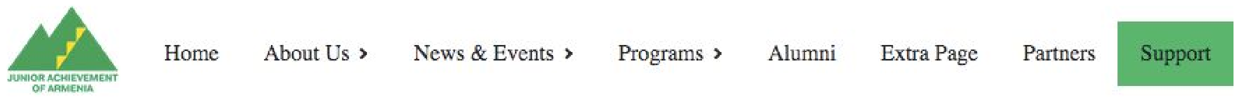
Now, to display something to see if you successfully added a new page in the website. Paste the following line to the ‘extra-page.hbs’ file.

```
{{#> base-layout
        title="extrapage" }}
    {{#*inline "content"}}
        extrapage

    {{/inline}}
```

```
{{/base-layout}}
```

After saving all the changes, you can restart gulp and view the changes on the website.



To add more content in the new page, you are welcome to copy the existing code from another .hbs file and modify it. You can find how to change and add an image and written content in other sections.

How to Add a Link

Seen below is a basic example of a link. The link (a tag) is in a p tag which defines a paragraph and contains the text ‘here’. On the website it will show up as “here” and when somebody clicks that word, it will bring them to the link. The href attribute is the link that will be traveled to on click of `here`. In this case the href is a directory relative link as. This means the link will start with the full path of the parent folder so if the link was on the page ‘jaarmenia.org/en/index.html’, the link would be relative to the `en` folder and go to ‘jaarmenia.org/en/about-us.html#team’. The `#team` will scroll down the page until the heading (h1, …, h6 tags) with the id of `team` is at the top of the page. Adding descriptive text in the title tag is important to have on every link. This text will show up when hovering over the link but the reason it is important is it’s the text that screen readers use when reading out links.

To learn more about relative links visit
developer.mozilla.org/en-US/docs/Web/HTML/Element/a#Linking_to_relative_URLs

```html
<p>See our team
   <a href="about-us.html#team" title="Our team">
       here
   </a>.
</p>
```

If the link you are adding is not a link on the jaa website, it should be an absolute link; meaning it starts with `http://` or `https://`. Https should be used if available so try changing it to https in a web browser and seeing if it works. The reason https is prefered besides just security is it also provides the user with a sense of security from seeing the browser show the security symbol. For external sites, you may want the link to open in a new tab. To do this you can add `target=”_blank” as seen below. When having a link that opens in a new tab it is important to also add `rel="noreferrer noopener"`. Adding this to the rel attribute is for security and performance reasons that are described here:
developer.mozilla.org/en-US/docs/Web/HTML/Element/a#target

```html
<p>You can visit our Facebook group
   <a href="https://www.facebook.com/groups/137502986335101/"
      target="_blank"
      rel="noreferrer noopener"
      title="Visit our Facebook group">
       here
   </a>.
</p>
```

Documentation on links in html can be found here:
developer.mozilla.org/en-US/docs/Web/HTML/Element/a

Adding the Armenian Version

Follow the instructions for SEO (**S**earch **E**ngine **O**ptimization) support.google.com/webmasters/answer/189077?hl=en to add links to the headers of pages to show the alternate version. This will involve adding text into the header per each page. As this is in the template the way to do that would be to add a block in the template by adding the following to the end of 'base-layout.hbs'.

```
{{#> alt-links}}
{{!-- Links to alternate language versions --}}
{{/alt-links}}
```

Then in each page you can add the following for each page in the 'en' and 'hy' folders.

```
{{#*inline "alt-links"}}
  <link rel="alternate" hreflang="lang_code" href="url_of_page"/>
{{/inline}}
```

One thing we have already built for the armenian version is the button to switch between languages. This was in the footer. To add back the button you can add the following to 'src/views/partials/includes/footer.hbs' inside '#footerContent' between the social logos the contact information.

```
<div>
   <h3>
       {{#if_eq lang "en"}}
           <a href="#" class="text-link language-switch"
data-lang="hy">Հայերեն</a>
       {{else if_eq lang "hy"}}
           <a href="#" class="text-link language-switch" title="View this page
in English" data-lang="en">English</a>
       {{/if_eq}}
   </h3>
</div>
```

To make the button actually work, some js code is needed. Since this js code will need to be on every page, a new js file should be created. The name we recommend for this is 'main.js'. This should be created in 'src/public/js'. Then to make the new js file loaded in every page, in 'src/views/layouts/base-layout.hbs', add the following towards the bottom right before the js block is defined with handlebars syntax.

```
<script src="../public/js/main.js"></script>
```

Now that the file will be loaded it's time to add the contents to the file. The function below, when added to 'main.js', will determine what the current language of the page is from the html tag ‘lang’ attribute. Then it will make a function called changeLanguage edit the url of the page to change from ‘en/my-page.html’ to ‘’hy/my-page.html’. It does this by simply removing the ‘en’ and replacing it with ‘hy’ or the other way around if the current page is in armenian. The next step is to find the button just added to the footer, then to add a call changeLanguage function while passing in the language the button says it should switch to.

```
(function() {
   // get current language from document html tag language attribute
   const curLang = document.querySelector("html").getAttribute("lang");

   function changeLanguage(lang) {
       if (curLang !== lang) {
           location.pathname = location.pathname.split(curLang).join(lang);
       }
   }

   document.querySelector("#footer a.language-switch")
       .addEventListener("click", event => {
           event.preventDefault();
           changeLanguage(event.target.dataset.lang);
           return false;
       })
})();
```

For the actual content of the website armenian, this can be done by copy and pasting the contents of the files in ‘src/en’ to those in ‘src/hy’, then pasting in the armenian translations. Some of the content needed to be translated are in other locations. The news and events content is in ‘events.js’ and ‘news.js’ in ‘src/public/js’. The footer can be edited in ‘src/views/partials/includes/footer.hbs’. And the nav bar can be edited in ‘gulp/nav.js’.

Right now the ‘hy’ folder is not being compiled. To add compiling for ‘hy’, go to ‘gulp/compile-templates.js’ and change the return of the ‘compileTemplates’ function.

```
return merge(
   compileTemplatesLang('en'),
   compileTemplatesLang('hy')
)
```

How to Resize Images

All images on the website are currently resized automatically to make page loads faster. It's the folders that end in '-resize' that allow this to happen. When compiling, images in folders with this suffix are considered to be marked for resizing. The rest of the folder name describes what the image should be sized for. For example, images used in the image content layout have a maximum size of 200x150. Therefore, the image on the server should also be that size. This is done by the node module 'sharp' (sharp.pixelplumbing.com/api-resize) in the 'compileResizeImages' function in 'gulp/compile-assets.js'. The definitions for how to size certain images are in 'gulp/image-sizes.js'. This file stores two arrays. 'classes' stores information on resizes that are used multiple times (often for a css layout) which use the folder names: 'img-content-resize', 'grid-img-resize', 'date-img-resize', 'social-icons-resize', 'big-img-resize', and 'jaa-logo-resize'. 'names' stores information on resize types that are only done to one image. These resizes also use the filename to assign the size to the image hence the name.

To add a new image size you will use multiple times, you will need to add a new object to the classes array. The following is an example of adding a new item which describes the sizing for a new layout.

```
{
   name: 'my-new-layout-resize',
   suffix: '-size-mnl',
   options: {
       width: undefined,
       height: 100,
       fit: 'inside',
       withoutEnlargement: true,
   }
},
```

Here, the 'name' key defines the name of the folders within which images should be resized to these specifications. The value of 'suffix' is added to the end of all image names (before the file extension) after they are compiled. The format of the suffix is '-size-' followed by any abbreviation you like to describe the image size. While typing in longer file names with the suffix isn't fun, this makes it easy to know which files are which size. This also avoids the problem that would be caused by resizing the same image twice in the same location. Since the '-resize' folder is only defining image resizing and not meant to organize the files, the folders ending with 'resize' are not used when adding the images to the 'docs' folder. Without the suffix describing the image size, two folders with the same size would overwrite each other. Next the 'width' and 'height' define the width and height to resize to in pixels. When one of these is not set like width is here, then that dimension of the image can be any value. When being resized the images will keep their aspect ratio, not be cropped, and will also not expand past their quality. These options are defined with the ‘fit’ and ‘withoutEnlargement’ keys.

If there is an image that needs to be resized to a unique size, then a very similar object should be added to the ‘names’ array.

```
{
   name: 'name-resize',
   basename: 'my-img',
   suffix: '-size-n',
   options: {width: 200...}  // same as before
},
```

The main difference here is the addition of the ‘basename’ key. This should be set to the filename of the image without the extension. Also different now is that all images resized in this way should be in a folder named ‘name-resize’ and the suffix used for these are ‘-size-n’ but again the suffix can be anything to describe the image size. The options work the same way.

Potential Change to the Alumni Page

On the alumni page below the section describing Junior Achievement Alumni of Armenia and to the right of the Join the Alumni Group form, we planned two sections. One which would link to the several regional alumni groups and another that would link to the job opportunity postings for JAA alumni. The following html could be added to the alumni page (src/views/en/alumni.hbs) inside the div with id of `alumniBottom` (div#alumniBottom) and after the current div with the form.

```
<div>
   <h2 class="header">Connect with Others</h2>
   <div id="regionalAlumniGroups">
       <a href="https://www.facebook.com/groups/<THE YEREVAN ALUMNI
GROUP>/" target="_blank" rel="noreferrer noopener" title="Visit the
Yerevan Facebook group">
           <img src="/public/assets/images/facebook-logo.png"
loading="lazy" alt="Facebook">
       </a>
<!--         repeat for each group-->
   </div>
   <h2 class="header">Find Employment Opportunities</h2>
<!--    link job opportunities that people have posted about-->
   <p>describe the page and link <a href="">here</a>.</p>
</div>
```

The following css should also be added to the alumni section of main.css while the current css under the selector of `#alumniBottom` should be deleted.

```
/* desktop specific styles */
@media only screen and (min-width: 48em) {
   #alumniBottom {
       /* Move sections to side by side */
       flex-direction: row !important;
   }
}

#alumniBottom {
   display: flex;
   flex-direction: column;
}
```

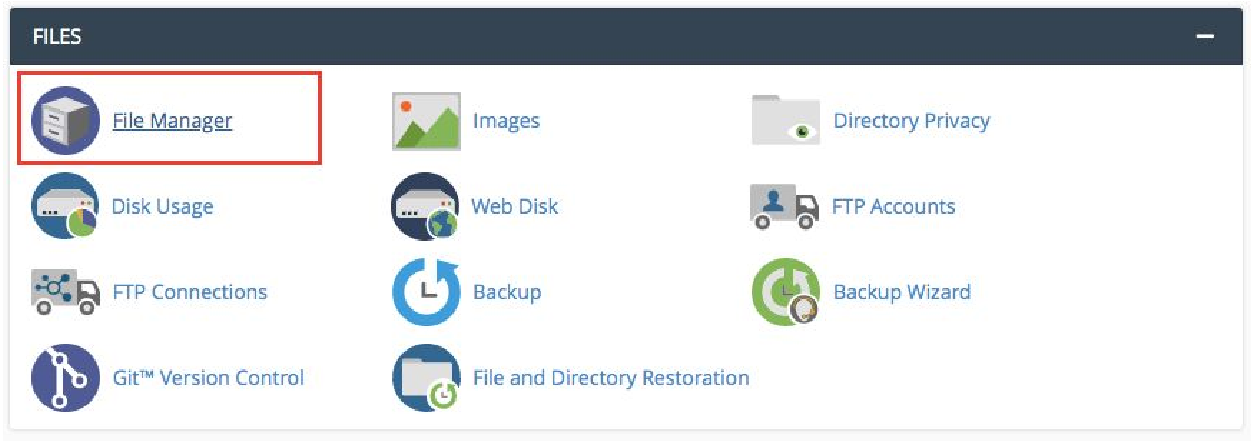
```
#alumniBottom div:nth-child(1) {
   margin-right: 2em;
}

#regionalAlumniGroups img {
   /* This is only needed if other images are used. The facebook
image is already 2.5em (40px) */
   max-width: 2.5em;
}
```

Deploying the Website to the Server

Once changes are ready to be put onto the website, you will need to upload the compiled files to the server through cPanel. It is best to upload the files to the server in a zip file. To get this you can either run the 'gulp compile' command then zip the contents of the 'docs' folder or simply run the 'gulp deploy' command which will do the same thing and put the zip file in the deploys folder.

Once you have the zip file, open cPanel and go to the File Manager.



Then go to the 'public_html' folder. This folder is where the 'docs/en' and 'docs/public' folder should go. Our website is static, meaning the server just serves files to the user and doesn't need to do any compiling. Currently the server is being used in a very different way. The server current has configuration files that restrict the files accessible on the internet. The current files will need to be removed but of course they should be backed up first.

The standard way to backup on cPanel is to make a zip file of everything in 'public_html' including the previous backup. You can see there is already a zip file here called 'jaa.zip' which is an old backup. To do this select all the files, right click, and select 'compress'. Choose 'Zip Archive' as the compression type and use the filename '/public_html/jaa<date>.zip' with the date filled out. You can add a date to this filename by using '/public_html/jaa_may_10.zip' for example. This will create a zip file of everything which you can unzip later if you need the old version. Once you press 'compress file' and the progress is done, you can close the popup window. To see the new zip file you will need to press the 'Reload' button which is above the files. Now you can delete all the files that are outside the new zip file. The 'Delete' button is at the top-center of the page.

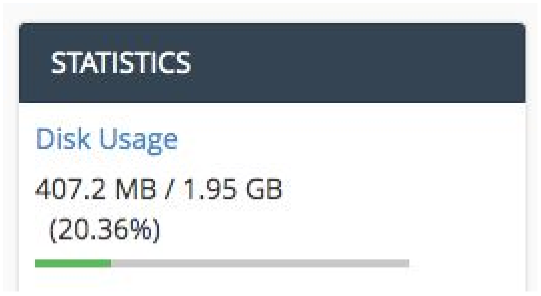
Now you can upload the zip file of the updated version. Do this by pressing the 'Upload' button and either dragging and dropping it or using 'select file' to find it. Once it says 'complete' below the progress bar, you can close the tab. Make sure it says complete because the progress bar can turn green and say 100% before the upload is actually complete. If you close before it is done the upload will end and nothing will be uploaded.

Now that the zip file is on the server, extract it by right clicking on it and pressing ‘extract’. Make sure the location is ‘/public_html’ then press ‘Extract Files’. You can close the extraction results window when it comes up. Then press ‘Reload’ again to see the new files. Now that the files are here, they should be accessible through the url ‘jaarmenia.org’. Even though the pages are in ‘en’, the loose ‘index.html’ file will redirect this url to the ‘en/index.html’ file

Managing Server Space

Check Storage Capacity

If the server fills up, you may stop getting emails through the contact email address and you will have trouble uploading updates to the server. Thankfully, both the new and old version of the website take very little space. One thing that could become a problem, however, given enough time are emails. Storing the spam emails JAA gets over the years can add up to a significant percentage of the servers capacity. You can check the capacity of the server on the right side of the home page of cPanel under the statistics heading.



STATISTICS

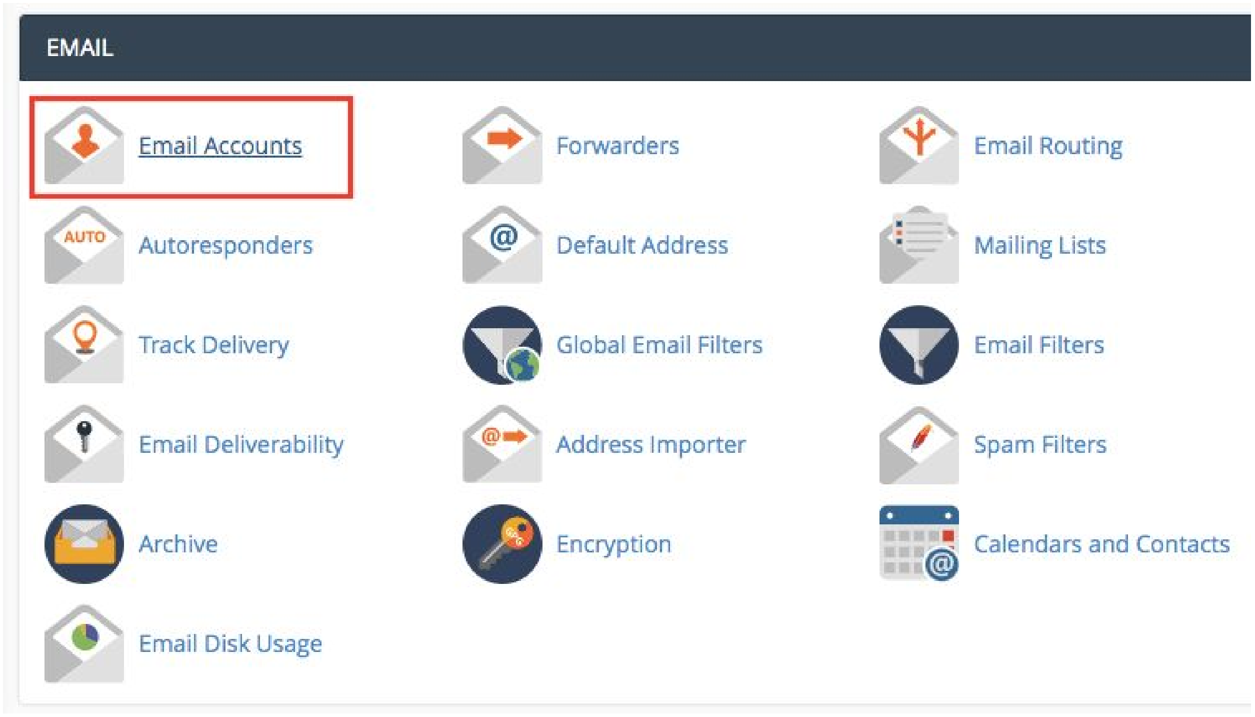
Disk Usage

407.2 MB / 1.95 GB (20.36%)

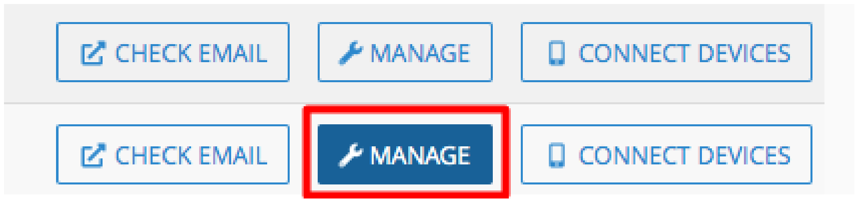
We have cleared emails recently so this should not be a problem for several more years given that it took from 2011 to now to fill up around 60% of the storage.

Clearing Old Spam Email

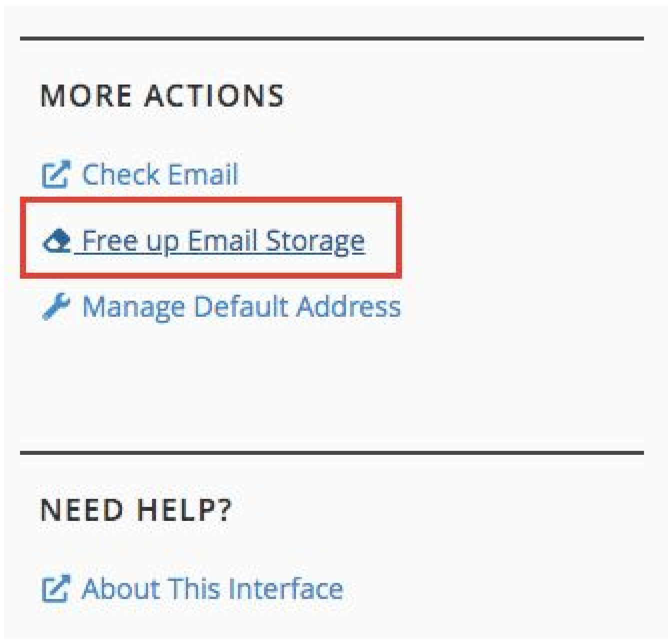
Once logged into the cPanel, click on Email accounts under Emails



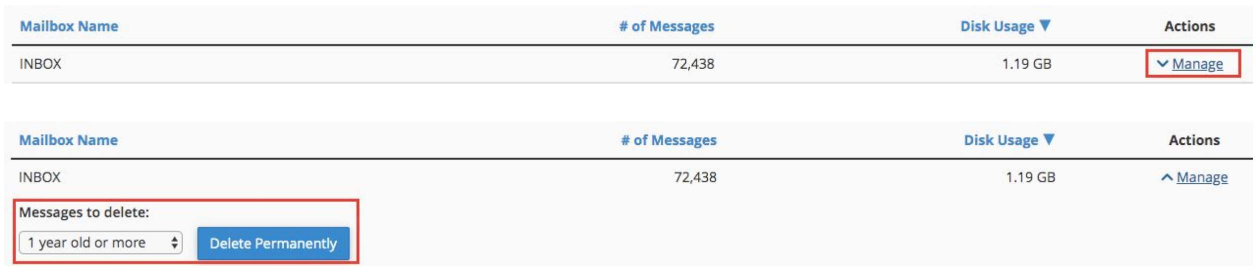
Then you can go to the 'manage' option on either account but it will probably be the default account which is taking up space. Being the default account means that any emails sent to the jaarmenia.org domain get sent here if there is not an account with the name. For example, if someone emailed 'nowhere@jaarmenia.org', this email will end up in the default email account but if someone emailed 'contact@jaarmenia.org', this would go to the contact account.



After opening the mange page, click on 'Free up Email Storage'.



This will bring up a page which details all the folders in the email account. The 'INBOX' folder is likely the only one to have emails. The next steps are to click on 'Manage' and then select the drop down option you want (we chose '1 year or more'), and finally press the 'Delete Permanently'.



Mailbox Name	# of Messages	Disk Usage ▼	Actions
INBOX	72,438	1.19 GB	Manage

Mailbox Name	# of Messages	Disk Usage ▼	Actions
INBOX	72,438	1.19 GB	Manage

Messages to delete: 1 year old or more

If you need to clear emails on the contact email account, you can do this without worrying about losing emails as deleting emails in this way would not affect the gmail account which is linked to the contact email address.

How Often Should You Update

We recommend that the website content be updated as changes happen to JAA. To stay on a schedule, we suggest updating the testimonials and events monthly, and updating everything else yearly as there are not many changes to the other aspects of the website. We suggest monthly updates as it will give enough time for the content to be viewed without making the website look outdated. When a user realizes part of a website is outdated, they can lose faith in the rest of the content.

Common Words and Their Definitions

HTML

HTML describes the structural elements of the website.

Handlebars

Handlebars is a templating engine that makes it easier to write HTML. You can define HTML using control flows like if-else or looping over an object as well as being able to import the contents of a file into another.

JS

JavaScript is the programming language of the HTML and web, it programs the behavior of the page. You can find a guide in w3schools.com/js/.

CSS

CSS provides style to HTML elements such as colors, padding, and alignment. One difficult thing to remember while writing CSS is CSS selectors. Here is a guide to them w3schools.com/cssref/css_selectors.asp

Command Line

One important command is to use “cd” to change the directories. In other words, to open and exit folders and files. “Cd folder_name” will let you go to the folder you want to open. “Cd ~” will put you at the home directory, “cd -” will return you to the previous directory, and “ls” will list all the files and folders in the directory.

Git

Git is a tool for allowing team collaboration with code as well as version control (backups). Any command in the command starting with git will initiate a git command, such as status, pull, add, commit, push, and merge.

Node.js

Node.js is a JavaScript runtime environment that executes JavaScript code outside of a web browser. It allows developers to run script server-side to produce dynamic web page content before it is sent to a webpage.

Gulp

Gulp.js is a toolkit built on Node.js and npm, used for automation of time consuming and repetitive tasks involved in web development like minification, optimization.

Additional Sources

Git / GitHub

guides.github.com/activities/hello-world/

help.github.com/en/github/creating-cloning-and-archiving-repositories/cloning-a-repository

developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/GitHub

HTML

developer.mozilla.org/en-US/docs/Web/HTML

w3schools.com/html/

JavaScript

developer.mozilla.org/en-US/docs/Web/JavaScript

w3schools.com/js/

CSS

drafts.csswg.org/selectors-3/

developer.mozilla.org/en-US/docs/Web/CSS

w3schools.com/cssref/css_selectors.asp