

Documentation

Client Side:

The intent of this section is to explain how to get the client side of the app working in PhoneGap:

- 1.) Create a GitHub account if you don't have one already.
- 2.) Open up Terminal (Mac) or Putty (Windows)
- 3.) Type “cd desktop”, to ensure you're in the desktop folder.
- 4.) Type “git clone <https://github.com/rgc292/mobileApp.git>”, which will create a mobileApp folder on your desktop.
- 5.) Create a folder on your Desktop and label it “HelloWorld”
- 6.) Open the PhoneGap desktop app and create a new project. Select the default Hello World template.

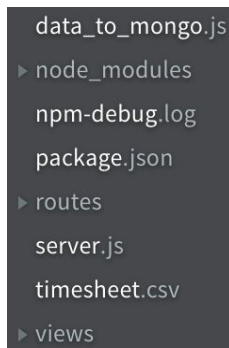
- 7.) For the local path, use the “HelloWorld” folder you’ve created on your Desktop, and then click Create Project.
- 8.) Open the “HelloWorld” folder on your Desktop, and then open the “www” folder.
In a separate window, open the “mobileApp” folder.
 - a.) Under the “mobileApp” folder, open the “path_mobile_client” folder.
 - b.) Copy the entire contents of this folder, but not the folder itself, into the “www” folder under “HelloWorld”. The contents should include: [‘css’, ‘Images’, ‘index.html’, ‘jquery.mobile.custom’, ‘jquery.mobile.prior’, and ‘js’]
- 9.) Run the Hello World app in the PhoneGap desktop app.
- 10.) Open PhoneGap on your mobile device.
 - a.) Enter the ip address the server is running on into the box provided
 - b.) Press Connect, and the app should run.

Server Side:

The intent of this text is to familiarize you with the server side files, with how to start or stop the server, and with how to check its functioning.

The folder containing the server side files is called *path_mobile_server*. Right now, the server should be up and running on port 7004, and you could verify that by typing the following URL on your browser: <http://websys3.stern.nyu.edu:7004>. If this is not the case, you will be able to have it working at the end of this guide.

The contents of the *path_mobile_server* folder has the following structure:



```
data_to_mongo.js
▶ node_modules
  npm-debug.log
  package.json
▶ routes
  server.js
  timesheet.csv
▶ views
```

The path “/var/www/html/websysF16GB/websysF16GB4/” at *websys3* should be hosting the *path_mobile_server* folder. If this is not the case, the path hosting the running server should be “/var/www/html/websysF16GB/websysF16GB4/public_html/websys/”. As a last resort, the file can be downloaded from <https://github.com/rgc292/mobileApp>. In order to verify if the server side is running you can type the command “ps -eaf” on the *websys3* command line. After that, you can look for the name “SCREEN -t Path_Mobile” at the rightmost column called “CMD”. If this name is not there, you will need to start the server on your own, and we will take care of this soon.

Inside the *path_mobile_server* folder, some files and folders are necessary. Based on the previous picture, *timesheets.csv* is the file containing the timetables that will be stored into MongoDB. *Data_to_mongo.js* has the code that will load the data from the .csv file into MongoDB. The folder called *node_modules* contains all libraries or modules needed to have all server side code functioning correctly. The file *package.json* has the dependencies the code needs for the server to be up and running. The *routes* folder has three files inside. One is *timesheets.js* which queries the timetable data from MongoDB, and makes it available on port 7004 through ExpressJS. Another is *index.js* which renders the file *index.html* under the folder

views for checking on the web if the server is running under <http://websys3.stern.nyu.edu:7004>. Finally, *alerts.js* parses the alerts from the Path RSS feed into a JSON object making it available to port 7004 using ExpressJS as well. The file *npm-debug.log* keeps a log tracking all functioning or malfunctioning on the server side. The *server.js* file wraps up the whole functioning of the server side.

Once you have the *path_mobile_server* folder available in a directory, you need to make sure that you have NodeJS, and MongoDB up and running. After that, you need to be inside the *path_mobile_server* folder, and then you can type the command “npm install” for making sure that node has all dependencies installed. After that, you can type the command “screen -t Path_Mobile”. This should open a new terminal window. This is an alternative way to run the server side on the background with an identifiable name. In sequence, type the command “node server”. At this point, the server should be running, and you should see the message “Server running on port 7004”. Once you see this, you will need to send the process to the background pressing “Control+A” or “CTRL+A”, and then letter “D”. Pressing letter “D” after “CTRL+A” will send the process to the background, and you should see the message “[detached]” on the screen. If everything worked, you should see the name “screen -t Path_Mobile” available after typing the command “ps -eaf” on the rightmost column. Assuming this is the case, if you go to your browser and try to access the following URLs, some content should be there:

<http://websys3.stern.nyu.edu:7004>

<http://websys3.stern.nyu.edu:7004/api/timesheets>

<http://websys3.stern.nyu.edu:7004/api/alerts>

Remember that maybe the alert address will render this “{“alert”:[]}” for not existing any alert at that moment.

For stopping the server, you should verify it is running typing the command “ps -eaf” as mentioned before. After that, you should verify which line has the name “screen -t Path_Mobile”. Having identified it, the second column (called “PID”) on that line has a number. Right now the number is 5526, but it can be a new one in the future. In sequence, type the command “kill -15 5526”. The *-15* option is a safe one for stopping the main process from running along with other processes instantiated when the main one started.

One last comment, the initial Excel spreadsheets called *path_table.csv* went through a data cleaning process in order to have the file *timesheets.csv* ready for using. The code used for that was a command line one as follows:

```
cut -f 2- -d',' path_table.csv | awk -F ',' ' $1!="" { print $0}' | awk -F ',' ' $4!="" { print $0}' > timesheet.csv
```