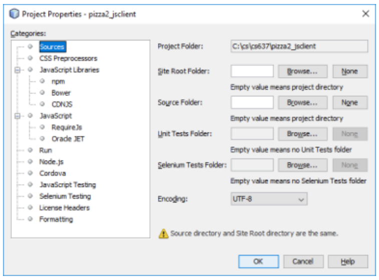
Developing and debugging Javascript

Setting up pizza2_jsclient

- Be sure to turn off caching in Chrome: Inspect>Network>Disable cache
  - ✓ Otherwise, you may find that your newly edited JS is quietly ignored by Chrome
- Edit the baseUrl in app.js to match your working pizza2_server, itself running from an Apache server (needed for its Apache URL routing via .htaccess.)
  - – At http://localhost/cs637/user/pizza2_server/api if it's using your XAMPP installation's Apache
  - – At http://localhost:8000/cs637/user/pizza2_server/api if it's using pe07's Apache server via our tunnel to pe07's port 80
- Set up a project in Netbeans, with following configurations…

Netbeans setup: Sources



Project Properties - pizza2_jsclient

Categories:

- Sources
- CSS Preprocessors
- JavaScript Libraries
  - npm
  - Bower
  - CDNJS
- JavaScript
  - RequireJs
  - Oracle JET
- Run
- Node.js
- Cordova
- JavaScript Testing
- Selenium Testing
- License Headers
- Formatting

Project Folder: C:\cs\cs637\pizza2_jsclient

Site Root Folder: Empty value means project directory

Source Folder: Empty value means project directory

Unit Tests Folder: Empty value means no Unit Tests folder

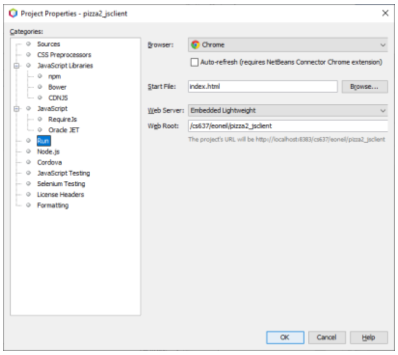
Selenium Tests Folder: Empty value means no Selenium Tests folder

Encoding: UTF-8

⚠ Source directory and Site Root directory are the same.

← Or under XAMPP's htdocs

Netbeans: Run



Project Properties - pizza2_jsclient

Categories:

- Sources
- CSS Preprocessors
- JavaScript Libraries
  - npm
  - Bower
  - CDNJS
- JavaScript
  - RequireJs
  - Oracle JET
- Run
- Node.js
- Cordova
- JavaScript Testing
- Selenium Testing
- License Headers
- Formatting

Browser: Chrome

☐ Auto-refresh (requires NetBeans Connector Chrome extension)

Start File: index.html

Web Server: Embedded Lightweight

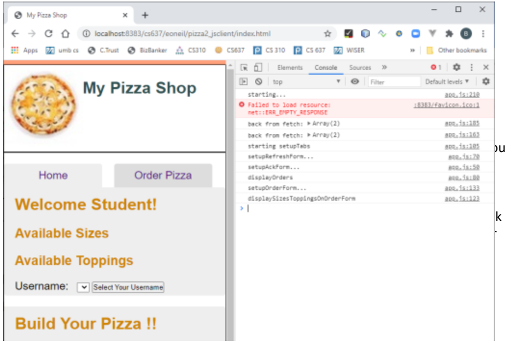
Web Root: /cs637/eoneil/pizza2_jsclient

The project's URL will be http://localhost:8383/cs637/eoneil/pizza2_jsclient

Running the project from Netbeans

- With this setup, right-click on index.html in the pizza2_jsclient project in Netbeans, select Run File, see loaded page
- Page URL: http://localhost:8383/pizza2_jsclient/index.html
- You see that the Netbeans "embedded lightweight server" at port 8383 is serving the page
- See Console output on the next slide: This is for the provided jsclient setup, running with a working pizza2_server.
- This is using CORS, because the browser is using two different servers, one for index.html and another for the web services. Luckily, our Slim server can do the needed CORS headers.
- The following assumes a working pizza2_server, but you can run it without that and see additional errors on the fetches.

Run from Netbeans (right-click index.html> Run File)



localhost:8383/cs637/eoneil/pizza2_jsclient/index.html

My Pizza Shop

Home | Order Pizza

Welcome Student!

Available Sizes

Available Toppings

Username: Select Your Username

Build Your Pizza !!

```
starting...                                              app.js:210
Failed to load resource:                          :8383/favicon.ico:1
net::ERR_EMPTY_RESPONSE
back from fetch: ▸Array(2)                               app.js:185
back from fetch: ▸Array(2)                               app.js:163
starting setupTabs                                       app.js:185
setupRefreshForm...                                       app.js:70
setupAckForm...                                           app.js:50
displayOrders                                             app.js:80
setupOrderForm...                                        app.js:133
displaySizesToppingsOnOrderForm                          app.js:123
```

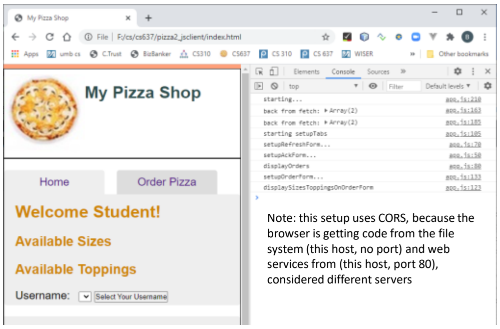
out

k

Console output notes

- The pink error is for the missing favicon icon. If curious, see Stackoverflow 4269695
- Be sure to check out any *additional* pink errors like this!
- The rest of the Console log is coming from the program. Check it out.

Run from filesystem (type the file path of index.html in Chrome's address bar)

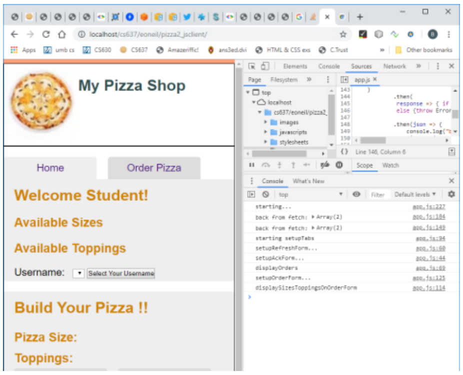


```
starting...                                      app.js:210
back from fetch: ▸Array(2)                       app.js:163
back from fetch: ▸Array(2)                       app.js:185
starting setupTabs                               app.js:105
setupRefreshForm...                              app.js:70
setupAckForm...                                  app.js:50
displayOrders                                    app.js:80
setupOrderForm...                                app.js:133
displaySizesToppingsOnOrderForm                  app.js:123
```

Note: this setup uses CORS, because the browser is getting code from the file system (this host, no port) and web services from (this host, port 80), considered different servers

Single-server Execution at Home

- Have pizza2_jsclient in XAMPP's site, at say
  htdocs/cs637/username/pizza2_jsclient
- Browse to it at http://localhost/cs637/username/pizza2_jsclient
- It will use this URL for web services: http://localhost/cs637/username/pizza2_server/api/...
- So both URLs are at port 80 on localhost, i.e., the same server.
- No CORS needed in this case
- You don't need to do it this way on your home system, but it does come much closer to what happens on pe07, where we have only one web server to use.



```
143        }
144                .then(
145                 response => { if
146                 else {throw Error
147
148                .then(json => {
149                    console.log("t
150
```

```
starting...                                      app.js:227
back from fetch: ▸Array(2)                       app.js:184
back from fetch: ▸Array(2)                       app.js:149
starting setupTabs                               app.js:94
setupRefreshForm...                              app.js:60
setupAckForm...                                  app.js:44
displayOrders                                    app.js:69
setupOrderForm...                                app.js:125
displaySizesToppingsOnOrderForm                  app.js:114
```

Executing on pe07

- We have only one web server on pe07, so the single-server scenario is the way to go.
- However, it's hard to run Chrome or even Firefox on pe07! All we have is lynx.
- We can execute client code *served* from pe07 by browsing with Chrome to localhost:8000/cs637/username/pizza2_jsclient using our tunnel to pe07's port 80.
- Note that it's baseUrl in pizza2_jsclient that determines the URL used for web services
- So we also need to change the baseUrl to http://localhost:8000/cs637/username/pizza2_server to get it to use only pe07, i.e., the delivered code.
- Please do this as part of delivery. Just replace localhost with localhost:8000 in baseUrl of pizza2_jsclient.

Debugging

- Add console.log's to your code just like echos in PHP.
- At some point, an error will make the page reload, grabbing away all your console log output.
  - ➢ Help for this: click the gear in the top right corner of Inspect and select "preserve log upon navigation"
- To protect against problems, save your work (i.e. app.js) in various files as you go, and test your code frequently.
- Note how you can use ls –l or dir to see which file is newest
- Netbeans can diff files: select one, then hold down control and right-click the other, select Tools>Diff…
- Then, in the worst case, you can back up to the previous file, and then carefully add newer code back to it, thus locating the error. Use netbeans diff to help with this.
- Or, use the debugger to creep up on the error. See https://developers.google.com/web/tools/chrome-devtools/javascript