**qqElec101Base: Quick Quasar Electron Base App**

[Introduction 1](#_Toc123822381)

[Resources 2](#_Toc123822382)

[My Doco on Quasar V2 with new Elements 2](#_Toc123822383)

[Elements 2](#_Toc123822384)

[Objectives 2](#_Toc123822385)

[User Interface 2](#_Toc123822386)

[Steps 3](#_Toc123822387)

[Problems 4](#_Toc123822388)

[src-electron > preload.js 4](#_Toc123822389)

[src-electron > electron-main.js 5](#_Toc123822390)

## Introduction

This is the first Quick Quasar app using the new Quasar V2 and infrastructure: TS, Vite, Pinia, Eslint. Also, it is the first time developing the app from the start as a GitHub repository for the organizaiton QuickQuasar. So all quite new!

Log

|  |  |
| --- | --- |
| **Date** | **Comments** |
| 12/7/22 | Program dusted off for additional processint in prep for EU ESCO zoom 12/15  Doing Term analysis using OWL as list; not good – need to use JS Object Hierarchy which keeps each node info together (need to exclude non-detailed occup labels) |
| 10/9/22 | Started this program using qq003TsPinia as base model. Also used GitHub capabilities in order to start deploying QQ apps to the QuickQuasar organization on GitHub; github folder was moved from C: users / to D:\github |
| 1/5/23 | Resuming: Updating GitHub project page and uploading this app to  <https://github.com/QuickQuasar/qqElec101Base> |
|  |  |
|  |  |
|  |  |

ToDo:

1. See Steps 12/9
2. Term Analysis: Current processing of labels uses OWL text as array / list – does NOT work well as each node is on multiple lines and we need to focus on “leaves” – detailed occupaitons only and skip BFO, skills and other info
   1. Review the JS Object already created and see if it can be used withoout change
   2. Find a good way to loop through leaves
   3. Find a good set of regular expressions to parcel out label content, such as “of \* worker\*”
   4. Further refine the categorization of occupation terms, tokens and ngrams
3. Results of Term analysis:
   1. What is best form for this – do we need an XLS output? Is one there now?
   2. Need characterization of each term: has or, has comma, compound or simple etc
   3. Distinguish tokens that are simple occup vs tokens only used in combination
4. do a scan and determine what the major nodes are and their content.
5. PUSH TO GITHUB!!

## Resources

This app was developed from a Danny Connell course – recap the development sometime….

Initial App: qqElec207ElecOcco plus FixFile and FixOwl

ESCO work, comparison among the four taxonomies

Current Project / App: qqElec207ExcelOccoONet << What is status of that app??

### My Doco on Quasar V2 with new Elements

Quasar 2 Vue 3 Pinia Vite for QQ (2022-09-21).docx in All Development > Quasar

## Elements

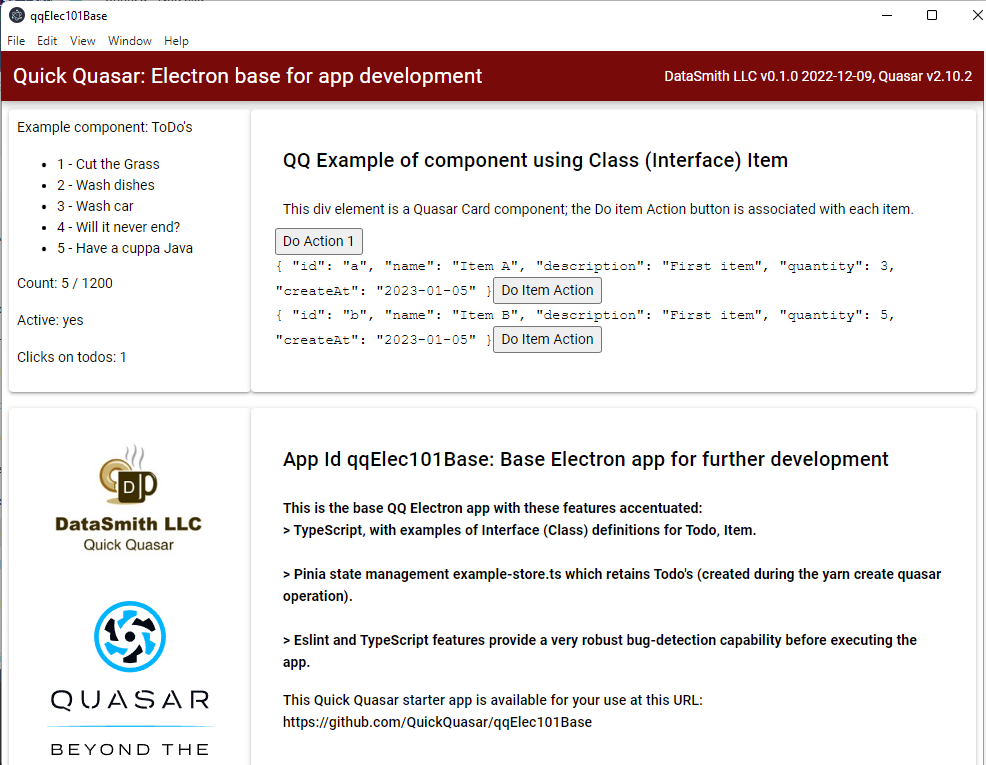
The elements of this app include those that are common among QQ apps: Quasar V2 (with Vue V3), Pinia, Eslint and optionally TypeScript, which is used in this app, and the D3 system of graphics and other capabilities, such as IO. Because D3 IO is aimed at web applications and does not deal with local files, it is not used in this Electron app, but its powerful graphics capabilites will be accentuated in oter QQ Electron apps.

\*\*\* is this still the case? Really a mess getting to work with Prettier: they don’t agree on single quotes; having the autocomplete is often a pain; …

## Objectives

\*\*\* Complete this; also: inputs, outputs, processing and UI Display

## User Interface



## Steps

1. Note: Using the D: drive for this series of developments
2. Go to parent directory D:\All\_Prog2\_QuickQuasar\QQv2\_Electron for Electron apps
3. Command: **yarn create quasar** and enter the following prompts
   1. Quasar CLI
   2. Project folder name qqElec101Base
   3. Quasar V2
   4. Typescript Yes (first time – see what headaches there are!!!...)
   5. Vite (not Webpack)
   6. Package name (lower case)
   7. Description Quick Quasar Electron Base for further development of specialized apps
   8. Author default
   9. SASS with SCSS syntax
   10. ESLINT << get used to this! But not yet!!
   11. State mgt: Pinia << yes – always use this! \*\* THIS DID NOT TAKE; no stores folder!
   12. Axios << NO
   13. Formatter: Prettier
4. **Cd to project folder Forget this every time!**
   1. Execute **yarn quasar dev -m electron** (quasar dev did not work)
   2. NOTE: Electron will be implemented following this command
   3. …
5. NOTE for first GitHub: Rather than cophy to D: Github at the outset, go ahead and deveop the app and get it running on the D:\QuickQuasar folder. Once working, copy over (all including NM folder or not?) to Github…
   1. What other steps?
6. Run the app: yarn quasar dev -m electron
   1. What do things look like – how is TS different?
   2. What can be done to enhance the functionality of the base app (without file read)?
7. Customize to the QQ format
   1. CSS colors and heading styles
   2. Remove left Drawer, button, essential links
   3. Add title and so forth; description
   4. Add DS logo >> also preserve Qusar logo and use small version (see qqElec173…)
   5. What else?
8. ADD Pinia $ quasar new store <store\_name = appData> [--format ts]
   1. Quasar new store appData
   2. … did this work? Try yarn quasar new store
   3. 12/7 first try did not work… Error message below indicates some start is needed?
   4. …yarn
9. Change the CSS to DataSmith / Hexplore standard
10. Added 12/9 – Basic operation successful!!
    1. Initial development in D:\QuickQuasar folder – move code (not NM) to d:\github
    2. Do the initial commit to GitHub and inspect how it appears on QuickQuasar organization
    3. Start updating the Readme
    4. Consider the project / organization wiki
    5. Delete C: folder and check total disk capacity
    6. What else? …..

## Problems

1. Got single-quote fix working… not sure how
2. It does not recognize variables in DOM and in script?
3. Reconcile Composiiton patterns: Vue <script setup> versus Quasar <script> setup()
4. Prettier / eslint problems – see means to disable edits by line or for all
5. Big problem: Quasar Dev Tools window disappears when activate xml commands
   1. Recall: it seems like fs does work??
   2. What is it about xml – is it a Node package while fs is within JavaScript?
   3. …
   4. Trying to see if use of Pinia store will avoid this problem

Pinia related bugs:

### src-electron > preload.js

What the heck does this do? Alert command in it does not work. It defines functions.

const getXlsx = () => {

console.log("I'm getting Excel .xlsx!");

window.myAPI.getXlsx(); // in preload.js

};

### src-electron > electron-main.js

This is actually where the main logic is! All my processing and output is here, which is easiest to do via the console rather than communicate with a .vue/.ts module with display output.

(How can results in main.js get passed back to the display modules?)

app.whenReady().then(() => {

createWindow();

console.log("app is ready");

let outObj = {}; // the output data as JS/JSON Object

// Open input file

const myPath = projectFolder + subFolder + inputFilename;

console.log("Processing Workbook file: " + myPath);

console.log("Processing Worksheet: " + inputSheetname);

const wb = XLSX.readFile(myPath);

const wksht = wb.Sheets[inputSheetname];

const myJson = XLSX.utils.sheet\_to\_json(wksht);

https://medium.com/jspoint/a-beginners-guide-to-creating-desktop-applications-using-electron-824da5665047

browserWindow ...

\_\_dirname

app.whenReady function: ...??

Instead of app.on('ready', handler) approach, you can also use app.whenReady() method which returns a promise. In the then() handler of this promise, you can do what we are doing in the above main.js file.

// More info: https://v2.quasar.dev/quasar-cli-vite/developing-electron-apps/electron-preload-script

app.on("activate" << what is this?

https://www.chromium.org/developers/design-documents/multi-process-architecture/