Quasar humble beginnings

So you see we have some quasar directives starting with v-\* that you can use.we can use them on any DOM element under control of vue and not just quasar ones so keep that in mind.

As far as it is understood we can use quasar component by referring them in the confg.js file Otherwise ofcourse we had to import them and register them locally.

Difference between plugins and components is that you can def use plugins even outside vue files that means even in simple js files.

Also we have special methods for quasar comonents these can be like

this.$regs.someref.specialmethod()

so far as it is understood when we install quasar we have two packages one is quasar cli and the other is quasar app package the main pakage is ofcourse quasar app package we can only install it and then in package.json file,write scripts like dev:“quasar dev” and then run them through npm or we can also install quasarcli package and globally do these task.

App internatiolization means for example you want to make a website for both urdu and English users, this serves the purpose.

So in quasar.conf we see that we are exporting a function which would gain a ctx object as a parameter and return an object.the value of keys or fields of ctx object are decided depending upon how you are running the app for example with simple quasar dev command we are running an SPA app but with quasar PWA or some syntax like that ,we are running the app as PWA too, so as a result the mode property in ctx would also contain PWA option which by default only contains SPA one.Based on this context we could deliver multiple functionalities depending on the context.

We also have a prefetch feature and a prefetch hook which is specifically somehow attached to component based on current route.

The PreFetch is a feature (**only available when using Quasar CLI**) which allows the components picked up by Vue Router (defined in /src/router/routes.js) to:

* pre-fetch data
* validate the route
* redirect to another route, when some conditions aren’t met (like user isn’t logged in)
* can help in initializing the Store state

visit documentation page it gives step by step guide how does route work.

We have plenty of options in quasar config.js that we can certainly optimize and put to good use.

A lot of options espectially build property has.

So as you can lazy load routes (route compoenents) you can also lazily load certain components.

Assets VS Static

All the assets that we want the webpack to process are palced inside regular assets folder.for example if we have logo then webpack could inline it or encode in base64 which saves a http request.

Files in the “assets” folder are only included in your build if they have a literal reference in one of your Vue files. Every file and folder from the “statics” folder are copied into your production build as-is, no matter what.

So that means that things like logos e.t.c should be placed there but important images be placed in static assets.

In order to execute some code before vue instance is instatiated we have the concept of boot files which get executed before vue instance is initiated.

* Your Vue plugin has installation instructions, like needing to call Vue.use() on it.
* Your Vue plugin requires instantiation of data that is added to the root instance - An example would be [vue-i18n](https://github.com/kazupon/vue-i18n/).
* You want to add a global mixin using Vue.mixin().
* You want to add something to the Vue prototype for convenient access - An example would be to conveniently use this.$axios inside your Vue files instead of importing Axios in each such file.
* You want to interfere with the router - An example would be to use router.beforeEach for authentication
* You want to interfere with the Vuex store instance - An example would be to use vuex-router-sync package
* Configure aspects of libraries - An example would be to create an instance of Axios with a base URL; you can then inject it into Vue prototype and/or export it (so you can import the instance from anywhere else in your app)

In order to better understand how a boot file works and what it does, you need to understand how your website/app boots:

1. Quasar is initialized (components, directives, plugins, Quasar i18n, Quasar icon sets)
2. Quasar Extras get imported (Roboto font – if used, icons, animations, …)
3. Quasar CSS & your app’s global CSS are imported
4. App.vue is loaded (not yet being used)
5. Store is imported (if using Vuex Store in src/store)
6. Boot files are imported
7. Boot files get their default export function executed
8. (if on Electron mode) Electron is imported and injected into Vue prototype
9. (if on Cordova mode) Listening for “deviceready” event and only then continuing with following steps
10. Instantiating Vue with root component and attaching to DOM

Ok then we arrive at CSS portion we have grid styles.

Important point to note is that we have of course .col class which occupies all the space available of horizontal space.

.col-auto is important when used with breakpoints.

QLAYOUT concepts:

This is important to understand the view prop in qlayout is significant in the sense that we control the view use documentation you shall understand it