VueJS App Global State with Pinia

Step 1: Adding Pinia to Existing Vue3 project

```
# Do it from your project directory
yarn add pinia      # Version 2.0
```

```
// main.ts
import { createApp } from 'vue'
import { createPinia } from 'pinia'
import App from './App.vue'

const pinia = createPinia()
const app = createApp(App)

app.use(pinia)
app.mount('#app')
```

Step 2: Define The Store

mystore.ts

```
import { defineStore } from "pinia"
import {ref, Ref} from "vue"

export const useAppStore = defineStore('foo', () => {
  const accessCode: Ref<number> = ref(0)
  const favColor: Ref<string> = ref("blue")

  function updateAccess(val: number) {
    accessCode.value = val
  }
  function updateColor(col: string) {
    favColor.value = col
  }
  return {accessCode, updateAccess, favColor, updateColor}
})
```

This looks exactly like a setup script in any VueJS component!

Make these vars & funcs available throughout the app

Stackblitz Playground

Access Store Variables From Components

```
<script setup lang="ts">
import {storeToRefs} from "pinia"
import {useAppStore} from "./mystore"
const myStore = useAppStore()
const {accessCode, favColor} = storeToRefs(myStore)
</script>

<template>
  <p>Your access code is {{accessCode}}</p>
  <p>Your favorite color is {{favColor}}</p>
</template>
```

Stackblitz Playground

Global vars

- accessCode ⇒ number
- favColor ⇒ string

Access Store Functions From Components

```
<script setup lang=”ts”>
import {storeToRefs} from “pinia”
import {useAppStore} from “./mystore”
const myStore = useAppStore()

function doUpdate() {
  myStore.updateColor(‘yellow’)
}
</script>

<template>
  <button @click=”doUpdate”>Update</button>
</template>
```

Global functions

- updateColor(__: string)
- updateAccess(_:number)

StackBlitz Playground