TAURI

@quasar/tauri - v1.0.0-alpha.0

https://github.com/quasarframework/tauri

Create tiny, fast and secure cross-platform native apps with the ease of VueJS and the power of the Quasar Framework

Lifecycle

This document seeks to provide visual and conceptual insight into the flow of Tauri App development and bundling. It assumes that all prerequisites for Tauri development have been met (i.e. Node, Quasar, Rust etc.) For the sake of brevity we have not detailed app extensions, transpiling to the AST or linting in the diagram.

DEV
The process of constructing a development environment begins with the Quasar CLI command: **quasar dev**
This command collects the config (CFG) and context (CTX) and prepares the environment by sending the transpiled source code through Babel and Webpack. Then the Rust dependencies are built fresh (if never done before) and devland code is compiled. A Webview is constructed, the Webpack server connects to it and the app is rendered and fully interactive.

HMR
When some part of the UI code changes, Chokidar will alert Webpack to recreate chunks, which are then injected into the receiving Webview. If Rust code changes, the Webview will be destroyed and rebuilt.

BUILD
The build command will create a finished app (based upon the configuration in **quasar.conf.js**). Node is used as a part of the toolchain and is not shipped. Webpack bundles the assets for production, Rust acquires and builds its dependencies and then bundles for the target platform.