Application development today

- I'm an experienced Caché user and I need to develop a modern, slick application (or need a complete make-over of a legacy app)
- What are my options?
  - Go native (Java, .NET/C#, Xcode/Swift, Android Studio/Java, …)
  - Web-based: SPA, PWA, web app
- Remember:
  - You must support a range of devices & operating environments
  - Mobile use is becoming more important than desktop!
- Most popular technologies

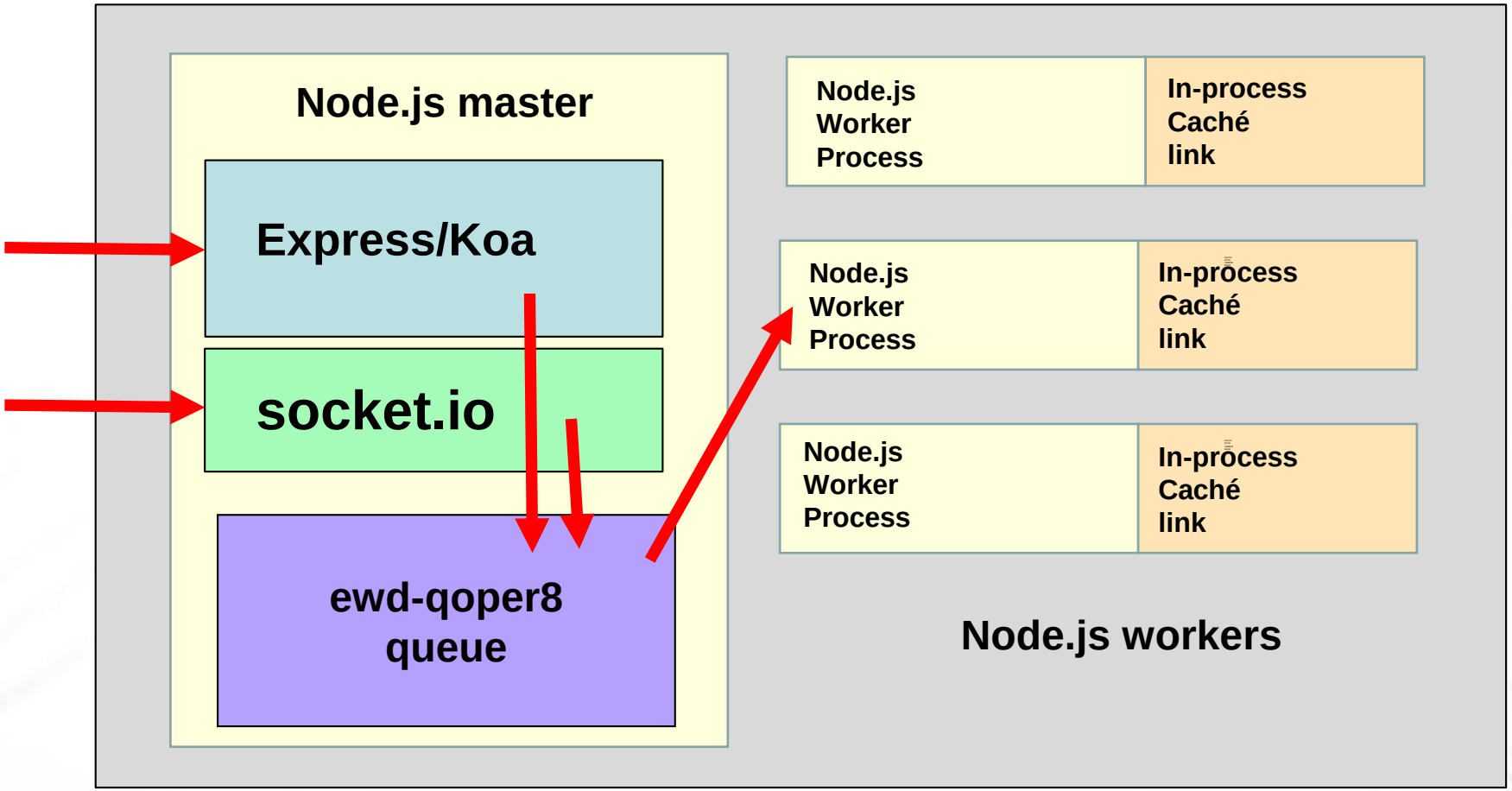
Application development today (cont'd)

- Using native development: can quickly become very costly (you'll need to write the same app multiple times!)
- Need for development environment that covers ALL devices … re-use code!
- One solution for that … the JavaScript ecosystem! Or use Java …
- BUT: don't lock yourself into a technology or framework:
  - Clear separation of concerns: model – view – state (store) – logic (actions, mutations), define & use consistent development methodologies
  - Make your application "data-driven"
  - Service oriented back-ends: write all your logic as service endpoints
  - Limit your dependencies because technology changes fast (be prepared to replace your front-end without the need to rewrite all your code)
  - A big consolidation took place in the JavaScript ecosystem lately

Application development today (cont'd)

- How can you make your code future-proof?
  - Caché: use extrinsic functions (wrappers) where you write your COS code – expect a JSON parameter global in & create an outgoing JSON result global (you can use every feature of COS like Objects, SQL, ...!)
  - Caché: or define CSP/REST endpoints and write your COS code in %CSP.REST classes
  - Middle tier: use a QEWD/Node.js application server with WebSocket requests
  - Middle tier: or define REST endpoints/microservices handlers using the qewd-cos helper module to call your extrinsic functions in COS (not applicable for CSP/REST)
  - Front-end: send QEWD WebSocket messages using this.$qewd.send(...)
  - Front-end: call your REST endpoints using e.g. the axios module (also with CSP/REST)
- As a result, you are much less dependent on changes in (framework) technology: only using standards like COS, WebSockets, REST (or even better, if you want to return complex data, use GraphQL)

Middle tier: QEWD/Node.js application server with WebSocket or REST requests



Vue.js

- OK, I want to start developing modern web applications …
- I only know basic HTML, a little JavaScript, CSP/Zen, …
- What do I need from this huge Web & JavaScript ecosystem?
- I'm overwhelmed …
- I want to be productive quickly!
- No huge learning curves please …
- What's an excellent option to start? Vue.js!

Why is Vue.js different?

- Stays very close to plain HTML & vanilla JavaScript
- Very flat learning curve, easy to learn
- Very lightweight (and performant)
- Ecosystem is very well defined, no « overchoice »
- Much easier than other popular choices: React, Angular, …
- Very popular: almost as much stars on GitHub vs React!

Vue.js history

- Modern, recent front-end framework (2014)
- Created by an ex-Angular developer @Google (Evan You)
- Learned from the « mistakes » made in other frameworks
- Used by big companies: Alibaba, Baidu, Adobe, IBM, …
- Very well suited for SPA's, PWA's and websites (SEO)
- Vue.js = view layer, Vuex = app state management, Vue Router = routing, Nuxt.js = create full websites

- Very simple to start: Hello World example

Vue.js live example

- Plain single-file HTML + JavaScript example
- Modular example using vue-cli module (using Webpack & interfacing ag-Grid with Caché classes & SQL)

- WebSockets server binding using QEWD + vue-qewd module (the Vue.js equivalent of react-qewd for React)
- REST calls using the axios module connecting to a QEWD/REST server or a Caché CSP/REST server

- Enough talk now, let's code ...

Useful links

- Vue.js (website)
- Vue.js course (free)
- Vue.js introduction video (starts at 31 min., see also slides)
- StackBlitz (online IDE, coming shortly: Vue.js support)
- Visual Studio Code (IDE for Windows, Mac, Linux)
- GitHub (online repositories with simple examples)
- Recipe app using Vue,js and Vuex (extensive example)
- Awesome Vue.js (resource collection)
- Growth of Node.js
- RealWorld example apps (compare frameworks & back-ends)