bachmann.

create-atvise-app

Overview

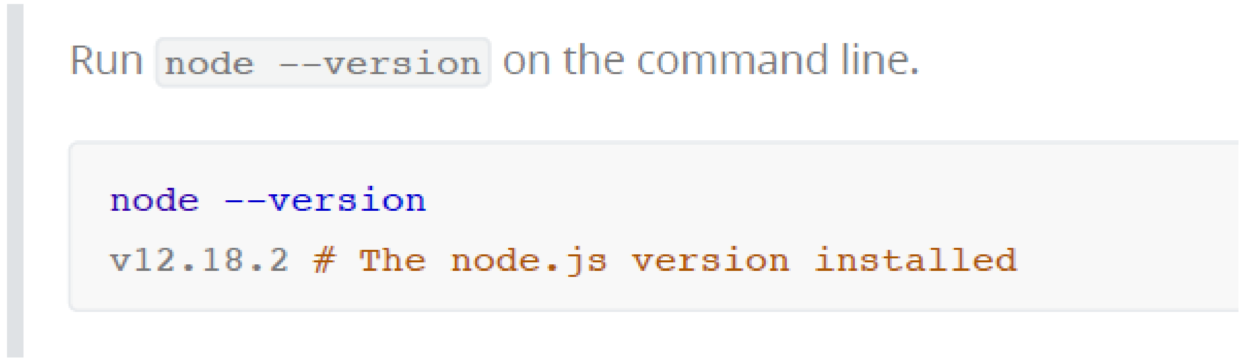
- **create-atvise-app** is a tool which helps atvise® users to build modern web applications based on the React Framework
- Technically the tool is built upon facebooks **create-react-app** and works in the same way, with the only difference that **create-atvise-app** will set up a ready to use development environment for creating react based atvise® applications

Quickstart

Install node.js

First, make sure you have node.js (version 12 or newer) installed.

How?



Run `node --version` on the command line.

```
node --version
v12.18.2 # The node.js version installed
```

If this command fails or prints a version number lower than 12 download and install the latest LTS (long term support) version from nodejs.org.

Quickstart II

Run create-react-app

Next, run *create-react-app* to create a new project:

> Don't worry about installing *create-react-app*, `npx` (npm package runner, shipped with node.js) does this for you 😀

```
npx create-react-app \
  # Use the atvise template
  --template @atvise/cra-template \
  # Use an adapted version of react-scripts
  --scripts-version @atvise/react-scripts \
  # The folder to create the app in
  my-app

cd my-app
```

Quickstart III

Project Structure

Running *create-react-app* will generate the initial project structure and install all the tools and resources required to build and run your app. These are the most important files:

```
my-app
├── README.md                # Contains information on how to run the app
├── node_modules             # Contains the app's dependencies
├── package.json             # The app's manifest file
├── .atviserc.json           # Project configuration
├── public                   # Put your static files in here, e.g. images
│   └── index.html           # The file you see when you open your browser
└── src                      # Contains your project's source files
    ├── App.js               # The App component (React)
    └── index.js             # Renders the 'App' component
```

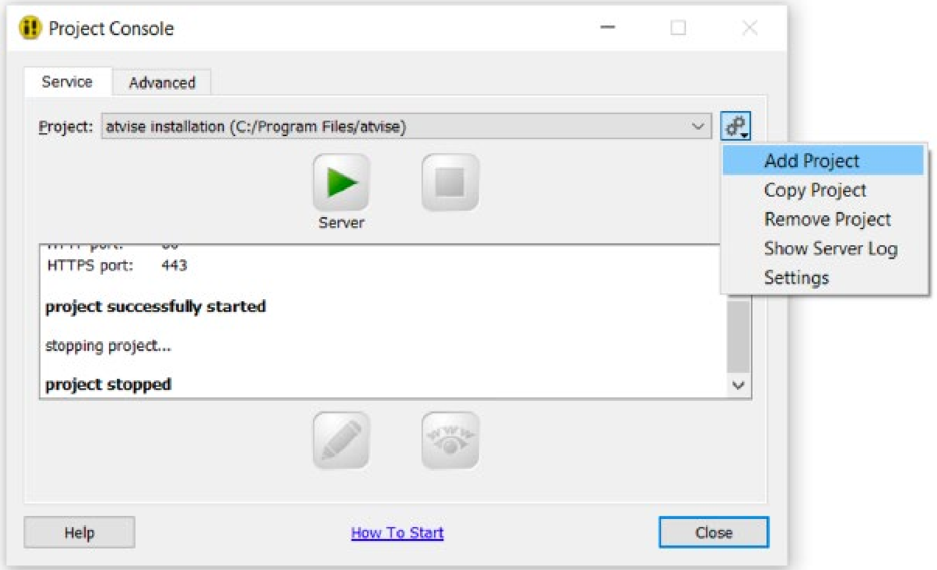
All in all, we have a fully functional React/atvise app at this point, no more configuration required.

Quickstart IV

Test the setup

First, you have to start an atvise® server in the *atserver* directory. The easiest way to do so is via the atvise® Project Console.

- Create a new project



Project: atvise installation (C:/Program Files/atvise)

```
HTTPS port:    443
project successfully started
stopping project...
project stopped
```

- Select the *atserver* directory as it's path
- Click OK to create it and
- Make sure that the ports *4844* and *8084* are selected and start atvise® server via the ▶ -Button

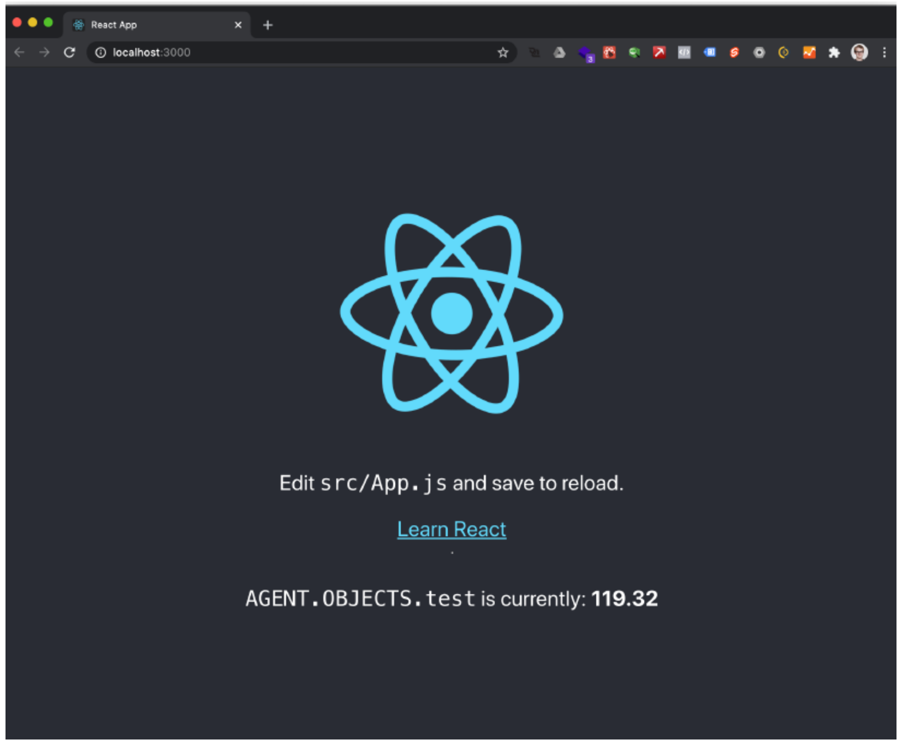
Quickstart V

Start the development server

Now we're ready to start the React development server:

```
npm start
```

This will open your browser and show a fully functional React app, displaying the value of an atvise variable:



Edit `src/App.js` and save to reload.

Learn React

`AGENT.OBJECTS.test` is currently: **119.32**

Development

This project makes very little assumptions on how you will develop your React app:

You can use almost all UI libraries, routers, internationalization frameworks etc. that work with react.

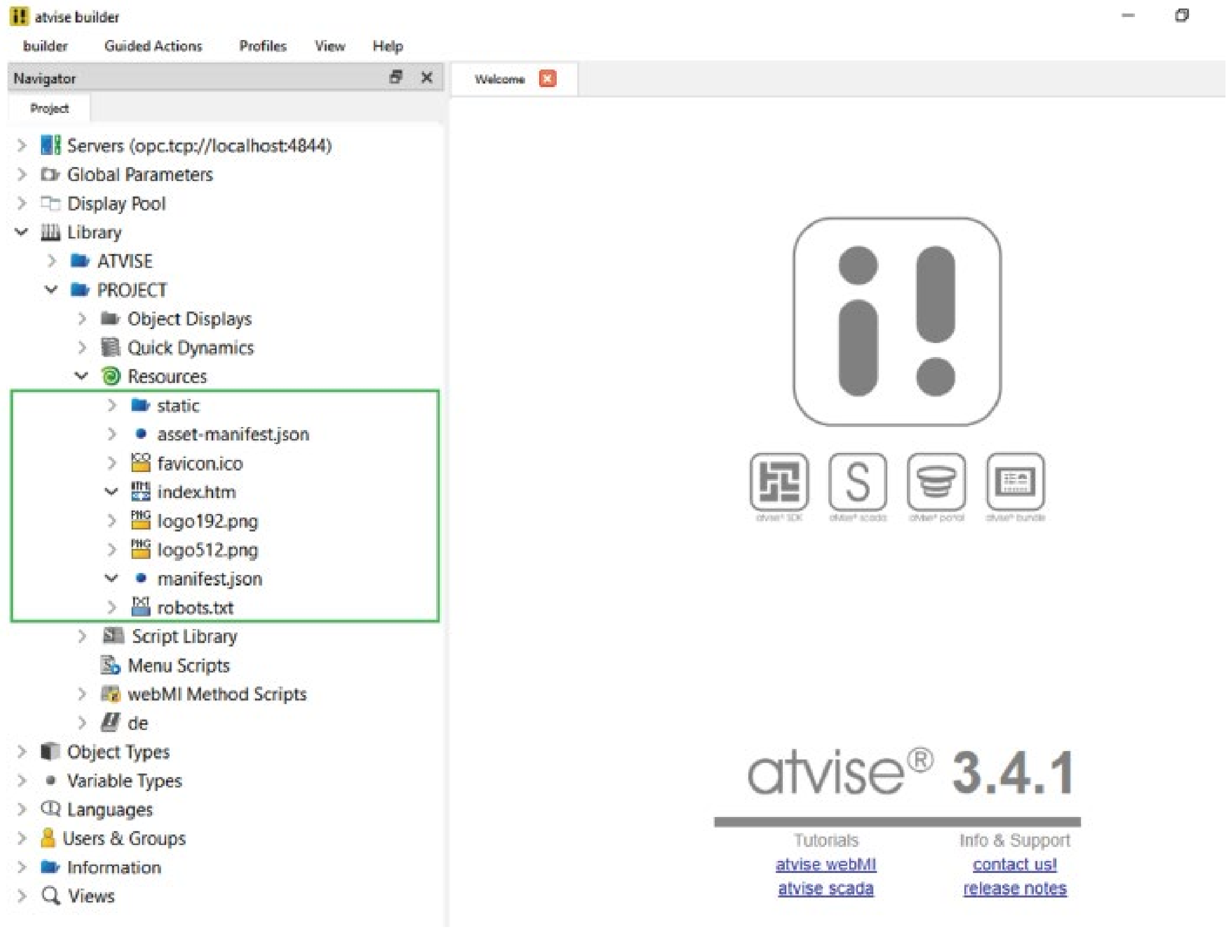
Additionally, the *@atvise/webmi-react* package provides data bindings between your react app and atvise server. For example, you can use the *useSubscription* hook to display a variable's value:

```
import { useSubscription } from '@atvise/webmi-react';

/**
 * @example
 * <MyLabel address="AGENT.OBJECTS.test" />
 */
function MyLabel(props) {
  const { loading, data, error } = useSubscription(props.address);

  if (loading) return <i>loading...</i>;
  if (error) return <i>An error ocurred</i>;

  return <p>AGENT.OBJECTS.test currently is: {data.value}</p>;
```

Deployment

You can deploy it to your atvise® server by running *npm run deploy*. This command will create a new production build (by running *npm run build*) and upload it to your atvise® server's project resources. Afterwards, you should see your react app when you open your atvise® server in the browser (usually available at localhost:8084).

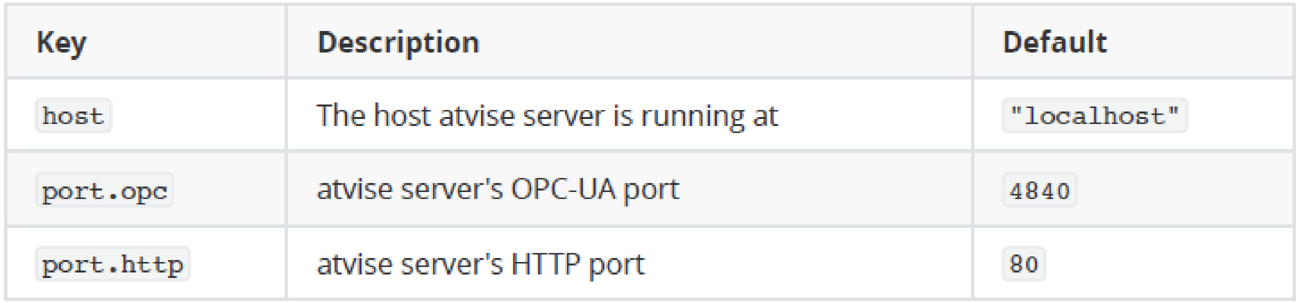
You can view the deployed files in atvise® builder by navigating to the project library's resources:



- Servers (opc.tcp://localhost:4844)
- Global Parameters
- Display Pool
- Library
  - ATVISE
  - PROJECT
    - Object Displays
    - Quick Dynamics
    - Resources
      - static
      - asset-manifest.json
      - favicon.ico
      - index.htm
      - logo192.png
      - logo512.png
      - manifest.json
      - robots.txt
    - Script Library
    - Menu Scripts
    - webMI Method Scripts
    - de
- Object Types
- Variable Types
- Languages
- Users & Groups
- Information
- Views

Configuration

You can adjust some project settings inside the *.atviserc.json* file. It contains a reference to it's schema, so autocompletion should be available in modern IDEs. These are the most important properties:



Key	Description	Default
`host`	The host atvise server is running at	`"localhost"`
`port.opc`	atvise server's OPC-UA port	`4840`
`port.http`	atvise server's HTTP port	`80`

Limitations

Using React instead of regular atvise displays and components comes with some drawbacks, too.
These are the most important:

- **Regular atvise displays cannot be used** alongside the react app.
  In addition the *webMI.display* API does not work as expected.

- **The graphics functions won't work**
  Don't use *webMI.translate* or *webMI.gfx* . You can do almost everything in CSS instead.

This project is currently under development so there may be additional incompatibilities we don't know about yet - Feel free to *create an issue* if you find one.

bachmann.