Developer Manual

Contents

General Concepts and Definitions

This chapter describes the general concepts and definitions of the Server State project.

Server State

An API-first, modular server monitoring solution

Server State is a project by fliegwerk that strives to create an API-first, fully modular server monitoring solution that adjusts to the individual clients' needs.

Related information

fliegwerk Homepage

Data Format (DF)

Standardized data formats for API communication in the Server State ecosystem

Data Formats are standardized interfaces for Data Communication via JSON within the Server State Ecosystem. They can represent things like

- Table-like data (that usually gets visualized in a table)
- XY-Data (that can get plotted in a graph

etc.

Note: Ideally, all formats should include at least one implementation of a viewer in every Frontend implementation of *Server State*.

In the generated type docs is a list of Common Data formats that we can use, allowing us to use standardized FEMs.

Related information

https://server-state.github.io/types/modules/serverstate.dataformats.html

Server Module (SM)

A Server Module is a function that returns JSON-serializable data that can get sent to clients.

Note: You can find technical information regarding the function definition of an SM at https://types.server-state.tech/modules/serverstate.html#smf

Notice: In the early stages of Server State, SM referred to a usage of a function. This function is was formerly called "Server Module Function" and is now just called "Server Module".

A SMF should be a *`readonly`* operation on the system (not changing any files), and

1. Return a JSON-serializable value on success
2. Throw (or, for asynchronous modules, reject with) an `Error` with a descriptive error message (`throw new Error('what went wrong')`)
3. **Not** log to the console or do other "active" operations. Handling error messages etc. is up to the SM's consumer (in most cases, the server-base).

The above guidelines aren't strict rules, but exceptions should be well-justified (for official modules) to allow for consistency and ease of use.

Related information

https://types.server-state.tech/modules/serverstate.html#smf

Frontend Module (FEM)

A Frontend Module provides a visualization of data from a Server Module that can get used in web-based clients like, e.g., the Web Client on page 4.

FEMs are, in the simplest form, the combination of some metadata and a React component that follows a standard that makes it possible to integrate them into client software implementing the required interfaces.

Development Process

FEMs can get developed using either the tooling provided by Server State or custom solutions implementing the required standards, as established by the Server State specifications.

FEMs get packaged into JS-compatible .fem files. These can either get distributed and installed manually or via the official or an unofficial FEM Registry.

Related information

Server State FEM Registry

web-client

Repositories / Subprojects

The Server State ecosystem consists of several sub-projects and/or repositories. You can see an overview of the most important ones below.

Server Base

Web Client

Related information

GitHub Repository

Typescript Definitions

CLI

FEM Template

SM Template

Index