readme.md for @push.rocks/early

minimal and fast loading plugin for startup time measuring

Install

To install `@push.rocks/early`, run the following command in your terminal:

```
npm install @push.rocks/early --save
```

This will add `@push.rocks/early` to your project's dependencies and prepare it for use.

Usage

In this guide, we'll explore how to use `@push.rocks/early` to measure startup times in your application. This tool is designed to be minimal and fast, providing accurate measurements of how long it takes for parts of your application to load.

Getting Started with `@push.rocks/early`

First, ensure that your project is set up with TypeScript and that you are familiar with working with ES modules. `@push.rocks/early` is built with TypeScript in mind, offering excellent IntelliSense support in compatible editors.

Importing `@push.rocks/early`

To begin, import the library into your module:

```
import * as early from '@push.rocks/early';
```

Starting Measurement

The first step in using `@push.rocks/early` is to start the measurement. This should be done as early as possible in your application's startup sequence:

```
early.start('myModuleName');
```

Replace `'myModuleName'` with a string identifier for the module or application part you are measuring. This is purely for identification purposes and aids in debugging or logging.

Performing Your Loading Operations

After starting the measurement, proceed with the operations or loading tasks you wish to measure. This could involve loading modules, reading files, or any setup tasks required for your application to start:

```
// Example loading operations
await myModule.initialize();
await myOtherModule.loadConfig();
```

Stopping Measurement and Retrieving Results

Once your loading operations are complete, stop the measurement to retrieve the loading time:

```
early.stop().then((loadingTime: number) => {
  console.log(`Loading time: ${loadingTime} milliseconds`);
});
```

This will output the total loading time in milliseconds, giving you a precise measure of the startup performance.

Advanced Features

`@push.rocks/early` allows for more advanced use cases, such as measuring multiple segments individually or in parallel. Each `start` can be matched with a corresponding `stop` to measure different parts of your application independently:

```
early.start('partOne');
// Load operations for part one...
early.stop().then((time: number) => console.log(`Part one loading time: ${time}`));

early.start('partTwo');
// Load operations for part two...
```

```
early.stop().then((time: number) => console.log(`Part two loading time: ${time}`));
```

High Resolution Time Measurement Class

For more granular control or to measure very short operations, `@push.rocks/early` offers a `HrtMeasurement` class for high-resolution time measurements:

```
import { HrtMeasurement } from '@push.rocks/early';

const hrtMeasurement = new HrtMeasurement();
hrtMeasurement.start();
// perform short duration tasks
hrtMeasurement.stop();

console.log(`Operation took ${hrtMeasurement.milliSeconds} milliseconds`);
```

This can be particularly useful for micro-benchmarks or when measuring the performance impact of optimization efforts.

Conclusion

`@push.rocks/early` is a powerful yet minimalistic tool for measuring startup times and performance in Node.js applications. Its straightforward API, coupled with TypeScript support, makes it an excellent choice for developers looking to optimize their applications or simply understand their loading characteristics better.

By integrating `@push.rocks/early` into your development process, you can gain valuable insights into your application's performance, helping you make informed decisions about optimizations and improvements.

License and Legal Information

Trademarks

This project is owned and maintained by Task Venture Capital GmbH. The names and logos associated with Task Venture Capital GmbH and any related products or services are trademarks of Task Venture Capital GmbH and are not included within the scope of the MIT license granted herein. Use of these trademarks must comply with Task Venture Capital GmbH's Trademark Guidelines, and any usage must be approved in writing by Task Venture Capital GmbH.

Company Information

Task Venture Capital GmbH
Registered at District court Bremen HRB 35230 HB, Germany

For any legal inquiries or if you require further information, please contact us via email at hello@task.vc.

By using this repository, you acknowledge that you have read this section, agree to comply with its terms, and understand that the licensing of the code does not imply endorsement by Task Venture Capital GmbH of any derivative works.

Revision #3
Created 2026-03-28 11:09:51 UTC by foss.global Team
Updated 2026-03-28 12:16:26 UTC by foss.global Team