Making PT accessible

Implementing PT in TypeScript

Petter Sæther Moen

Thesis submitted for the degree of
Master in Informatics: Programming and System Architecture
60 credits

Department of Informatics
Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Autumn 2020

Making PT accessible

Implementing PT in TypeScript

Petter Sæther Moen

Making PT accessible

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

Abstract

Acknowledgements

Contents

List of Figures

List of Tables

Preface

Part I

Introduction

Chapter 1

Introduction

1.1 What is PT?

1.2 Purpose of implementing PT in TS

Chapter 2

Background

2.1 Package Templates

2.2 TypeScript

Part II

The project

Chapter 3

Planning the project

3.1 Choosing the right approach

Before jumping into a project of this size it is important to find out what approach to use. The end goal of this project is to extend TypeScript with the Package Templates language mechanism, this can be achieved as following:

- Making a fork of the TypeScript compiler
- Making a preprocessor for the TypeScript compiler
- Making a Babel plugin

To get a better understanding for which approach to use we will be implementing a small language feature with each of the approaches named above.

3.1.1 TypeScript Compiler Fork

Possible, however not as accessible as other alternatives and will make upkeep expensive.

3.1.2 Preprocessor for the TypeScript Compiler

3.1.3 TypeScript Compiler Plugin

Not possible at the time as the TypeScript compiler wiki specifies "TypeScript Language Service Plugins ("plugins") are for changing the editing experience only." [2]. This might be possible in the future...

3.1.4 Babel plugin

Babel isn't strictly for TypeScript, but for JavaScript as a whole. Will however make it very accessible as most web-projects use Babel, and the upkeep is cheap, as plugins are pretty independent from the core.

Saying that this is strictly a Babel plugin wouldn't be entirely true, as we would have to fork the Babel parser in order to include our custom

syntax[1]. However this is all hidden away from the user, as this custom parser is a dependency of our Babel plugin.

TODO: Does it support having multiple custom parser? E.g. babel-plugin-typescript + our custom babel plugin?

Part III

Conclusion

Chapter 4

Results

Bibliography

[1] Babel. *@babel/parser*. URL: https://babeljs.io/docs/en/babel-parser (visited on 25/09/2020).

[2] Microsoft. *microsoft/TypeScript*. URL: https://github.com/microsoft/TypeScript/wiki/Writing-a-Language-Service-Plugin.