Posudek diplomové práce

Matematicko-fyzikální fakulta Univerzity Karlovy

Autor práce	Ondřej Roztočil		
Název práce	Type providers for TypeScript		
Rok odevzdání	2024		
Studijní program	Informatika	**Studijní obor**	Softwarové systémy

Autor posudku	Tomáš Petříček	**Role**	Vedoucí
Pracoviště	Department of Distributed and Dependable Systems		

Text posudku:

The thesis presents a design and implementation of the "type providers" feature for the TypeScript programming language. The mechanism, best known from F#, makes it possible to extend the TypeScript compiler with a compile-time/design-time extension that generates statically typed interface for accessing external data based on external schema or sample of the data.

The thesis makes two important contributions. First, it presents the design of the type provider mechanism in the context of the TypeScript ecosystem - the author considers TypeScript design principles (e.g., type erasure), syntactic preferences (compatibility with ES6), implementation requirements (e.g., based on AST generation) and describes a design for type providers that is inspired by the design known from F#, but suitable for implementation in TypeScript. This part results in a series of requirements collected throughout the analytical part of the thesis.

Second, the thesis is accompanied by a realistic implementation that extends the latest version of the open-source TypeScript compiler (and server used by IDEs) with support for the mechanism. The implementation is well-integrated with the (very complex and large) TypeScript codebase and could realistically serve as the basis for an open-source contribution to the language and compiler.

The text of the thesis is focused on documenting the design decisions, the implementation structure and on documenting the developed system. Given the applied nature of the work, this is a reasonable structure. The text itself is well-written, clear and serves its pragmatic purpose.

The accompanying implementation consists of a modified TypeScript compiler/server and a number of example type providers. Modifying the compiler required understanding the (poorly documented) architecture of the TypeScript compiler and finding an appropriate extension point - this is a major undertaking and the result shows that the author has been able to solve this task well (finding an appropriate way that makes it possible to support type providers with a relatively small change to the compiler). The sample type providers are based on existing work in F# and serve as good cases studies to showcase the capabilities of the system.

One thing that I would have appreciated seeing in the text is a more explicit discussion of things that remain problematic or were omitted from the final design - the thesis focuses on the final design, but notes some difficulties (asynchronous providers, lazy code generation) and it would be useful to collect those as aspects that may be conceptually incompatible with the TypeScript ecosystem or compiler implementation.

Overall, the thesis reports a very nice engineering-oriented applied project that adapts a more academic programming language feature to a widely-used context and shows the feasibility of implementing the feature in a real-world open-source project. As such, I have no hesitation in recommending it for a defense.

I have two questions that I would like to see discussed in the defense:

First, based on the experience with designing type providers for TypeScript, are there some aspects of the mechanism that should have been done differently when the mechanism was first described? In other words, was there something about previous type providers that was perhaps too F#-specific and not useful for TypeScript (or something that you thought about adding to the TypeScript version)?

Second, the use cases (CSV, JSON-Zod, XML) are mainly inspired by existing standard F# type providers, but are there some TypeScript-specific areas that you encountered where a type provider would be useful (perhaps outside of the core data access domain)?

Práci doporučuji k obhajobě.

Práci nenavrhuji na zvláštní ocenění.

Pokud práci navrhujete na zvláštní ocenění (cena děkana apod.), prosím uveďte zde stručné zdůvodnění (vzniklé publikace, významnost tématu, inovativnost práce apod.).

Datum 31. 8. 2024 **Podpis**