TypeFox

LangDev CON 2024

Introducing Typir for Type Checking in the Web

LangDev'24

Johannes Meier

TypeFox

Checks in language engineering

1. Parser errors
2. Linking errors
3. Language-specific validations:
   - Syntactic checks
   - Semantic checks at development time
   - Semantic checks at runtime

Type checking:
- Annotate AST with types
- Checks on these types

Motivation for type checking

- Validation of type errors
  - Type-related constraints
  - Assignability including sub-typing, casting, …
- Resolving cross-references
- Generators
- … with helpful error messages!

Live-Demo: LOX*

Motivation for Typir

- Utilities for easier type checking
- … for the web!
- Support language engineering and modelling projects
  - Easy application of type checking
  - Reuse

Introducing Typir!

Library for Type checking

- Core features
- Reusable types

TypeScript

API for Language engineers

Typir

langium -binding

Open source

Pragmatic

Early state

- since 2024
- by TypeFox

Pragmatic type checking

- More pragmatic than formal
- Default implementations for recurring problems
  - Kinds of types: primitives, classes, functions, …
  - Algorithms: Circular type definitions, performance/caching, …
  - Meaningful error messages
- Internal type graph

Ready for the web

- Written in TypeScript
- Runs in web browsers, e.g. web app, …
- Runs in Node.js, e.g. web server, CLI, …
- Runs in desktop applications, e.g. VS Code, Theia, …

Open source

- Source code is open: https://github.com/TypeFox/typir/
- MIT license
  - ✓ Commercial projects
  - ✓ Closed projects
  - ✓ no fees, no contracts
- Collaboration with community
  - GitHub Discussions
  - We are open for contributions and funding!

What about Langium?

$$\begin{tikzcd}
\text{Typir} & & \text{langium} \\
& \text{Typir-Langium} \arrow[ul] \arrow[ur] & \\
\text{Your DSL} \arrow[uu] & & \text{Your langium DSL} \arrow[ul] \arrow[uu]
\end{tikzcd}$$

Customization

… by dependency injection:

- Services
- … with default implementations
- Exchangeable implementations
- Parts of default implementations can be overridden

… by using custom types:

- Registry for kinds
- Type graph accepts any types

Live demo

Application of Typir to LOX*

AE9R

Core features

- Assignability: Type x Type → Boolean
- Subtype check: Type x Type → Boolean (nominal vs. structural)
- Coercion/casting: Type x Type → {Implicit, Explicit, None, Self}
- Equality: Type x Type → Boolean

- Inference: (node: unknown) → Type
  - Langium: AstNode → Type
- Validation: (model: unknown) → ValidationHints
  - Langium: AST → ValidationHints

Supported types

- Primitive types
- Fixed parameter types, e.g. List<T>, Map<K, V>
- Functions, incl. overloading
- Classes: super-classes, fields and methods
- Top type ("any")
- Bottom type ("never")
- Operators (mapped to Functions)

Roadmap

- Typir release v0.1.0 probably in November 2024
- Iterative development
  - Adding new features
  - Applying Typir to commercial projects
- Bindings for other language workbenches: …, LionWeb, … ?
- Transition to Eclipse Foundation (?)

Features for the near future

- Lambdas
- Union, intersect, except, …
- Generic types
- Enumeration types
- Type aliases
- Scopes, type assertions
- Cyclic type definitions, e.g. for trees (Node { children: Node[] })
- More performant APIs for registering validations and inference rules
- …

Summary: Benefits of Typir

- Predefined types to reuse + customizations
- Builtin solutions for …
  - Cyclic type definitions
  - Caching
  - Useful error messages for users of the DSL
- Bindings for language workbench
- … in the web!

TypeFox

LangDev CON 2024

Discussions & Ideas !

typir.org

typefox.io

AE9R

ProxyHands