C'est 2 - A Standard C++ Library with `constexpr` Extensions

Tools and Libraries for Compile-time Software Engineering
HiPEAC Conference 2024
Munich, Germany

Paul Keir [1]
Joel FALCOU [2]

[1]School of Computing, Engineering & Physical Sciences
University of the West of Scotland, Paisley, UK

[2]Le Laboratoire Interdisciplinaire des Sciences du Numérique (LISN)
Université Paris-Saclay, Paris, France

January 17th, 2024

Overview

- Motivation
- C'est 2 versus C'est
- C'est 2 Design
- Features
- `std::shared_ptr` and `std::function`
- Metamath
- Using C'est 2
- Hands On

Motivation

- C++ Template Metaprogramming is powerful and expressive
- Numerous impressive projects are implemented within this idiom
- But its syntax is not that of standard C++ runtime code
  - ...while the syntax of the `constexpr` idiom *is*
- As more features become `constexpr`-friendly, opportunities arise to:
  1. Repurpose decades of existing C++ runtime programs;
  2. Utilise the knowledge of traditional C++ runtime developers

C'est

- The C++ standard library may in time be entirely constexpr
  ...meanwhile, the C'est library can be used today
- Verify string-based embedded DSLs; explore code generation
- Support (incomplete) for: `forward_list`, `list`, `set`, `map`, `queue`, `deque`, `istringstream`, `unique_ptr`, `shared_ptr`, `exception` & `function`
- C'est is not standalone: libstdc++ is required; and its code is used
- Established constexpr entities from libstdc++ are wrapped within the cest namespace; e.g. `<algorithm>`, `<numeric>`, `std::vector`, `std::string`, `std::array`, `std::optional`, `std::pair`, `std::variant` ...
- C'est supports recent versions of g++ and clang++
- The project's Github repository: https://github.com/pkeir/cest

Acknowledgements

UWS UNIVERSITY OF THE WEST of SCOTLAND

RSE The Royal Society of Edinburgh
KNOWLEDGE MADE USEFUL