CDT.cloud? C/C++ tooling in the web

Stefan Dirix
EclipseSource
Why C/C++ tooling in the web/cloud?

How to provide C/C++ tooling in the web/cloud?
Typical architecture of a web/cloud-based tool

Client
- Browser engine / HTML
- Tool frontend

Cloud
- Node.js / Java / C / etc.
- Tool backend

Server

Electron (Desktop)
C/C++ tooling in the web

Desktop

Web/Cloud

CDT.cloud Blueprint

- Template for a web-based C/C++ development IDE based on Theia
- Includes selected Theia and VS Code extensions

Early state available. See cdt-cloud.io for more information.
The underlying tool platform
Desktop vs Cloud

Many vendors follow a dual strategy:

Web-based on the desktop first, add cloud offering as a second step

CDT.cloud Blueprint: Theia based Desktop application (cloud coming soon)
Theia vs VS Code

- VS Code is an extensible editor
- Theia is a framework for web-based editors and tools

VS Code extensions can be used both in Theia and VS Code

Some vendors follow a dual strategy:

Some features as a VS Code extension, full tool based on Eclipse Theia
Language support

- Historically: Built into the IDE (e.g. Eclipse CDT)
- Today: Based on LSP (Language Server Protocol)
C/C++ language servers

CPP tools (ms-vscode.cpptools)
- Very popular and mature
- Includes language server and debug support
- Restrictive license -> Can’t be used outside of VS Code

clangd (llvm-vs-code-extensions.vscode-clangd)
- Part of the LLVM project
- Fully-fledged alternative
- Can be used in Theia (and other tools)
clangd

De facto choice for all IDE vendors

To note:

- Requires build information per source file ("compile_commands.json")
  - Consequence on build integration
- Multi-project capabilities under investigation in the embedded SIG
Compilation

- Traditionally very project-specific

Some options:
- Terminal (CLI)
- Scripts / Tasks
- Cmake tools (ms-vsc ode.cmake-tools)
- Own UI
Debugging

Accomplished via DAP (Debug Adapter Protocol)
Debug Adapters

- CPP Tools *(ms-vscod.cpptools)*
- CDT GDB *(eclipse-ctd.cdt-gdb-vscode)* from Eclipse CDT
- CodeLLDB *(vadimcn.vscod-lldb)*

Embedded:

- Native Debug *(webfreak.debug)*
- Cortex-Debug *(marus25.cortex-debug)*
Debug Memory Inspector

Recently contributed to Theia

- Currently Theia only
- Tested with CDT GDB adapter but should work with any adapter which can provide memory information
- Part of “@theia/cpp-debug” but this will probably change in the future
Tracing

Eclipse Trace Compass

- Refactored architecture to use TSP (Trace Server Protocol)
- Theia integration exists
  - Theia-independent React integration is available
Conclusion

● Fast moving platform with a lot of potential
● C/C++ but also other areas
● Many embedded vendors adopting Theia
● Many use cases of C/C++ development are already covered

See cdt-cloud.io to stay up to date
Other sessions

Tue 26.10., 13:50, Getting started with Eclipse Cloud Dev Tools

Tue 26.10., 14:30, Diagram editors with Eclipse GLSP - 1.0

Tue 26.10., 16:50, Migrating Eclipse-based Tools/Plugins to Eclipse Theia or VSCode

Tue 26.10., 17:30, Spotlight session

Wed 27.10., 16:10, Model validation, diffing and more with EMF.cloud

**Wed 27.10., 18:15, BoF - Building web-based tools with Eclipse**

EclipseSource Booth
Evaluate this Sessions

● Please help by leaving feedback on the sessions you attend!
● To rate a session, you must be registered for it in Swapcard BEFORE the talk starts.
● Swapcard will prompt you to leave feedback after the end of each session.
● You may also rate a talk by locating the session from the “Agenda” or “My Event” buttons on the Event Home page. Click on the session and look for the “Give your feedback” box.
Questions?