Model validation, diffing and more with EMF.cloud

Maximilian Koegel, EclipseSource
What is EMF.cloud?

- Framework for building web-based (modeling) tools
- EMF.cloud is not EMF but offers similar concepts
- Provides components
  - covering frequent use cases
  - encapsulating best practices
- Provides architecture blueprints
  - for building web-based modeling tools
  - demonstrating EMF.cloud component usage
  - demonstrating usage of other components
What do we mean by “web-based”? 

- **Web-based technology:**
  - HTML, JS/TS, CSS, SVG
  - React, node, inversify, …

- **Orthogonal aspects**
  - desktop-based vs. browser-based
  - offline vs. online
  - on-premise vs. off-premise/cloud-based
The foundation of web-based (modeling) tools

- Eclipse Theia: Extensible cloud IDE platform
- Eclipse EMF.cloud: Web-based modeling tools
- Eclipse GLSP: Web-based diagramming
- Eclipse Che and Kubernetes: Cloud deployment
Architecture Blueprint: Coffee Editor

- **Purpose**: Example product to model coffee makers
- **Demo Path**:
  - Workbench with Workspace
  - Forms and Trees
  - Diagrams
  - Textual DSLs
  - **Comparison and Versioning**
  - Multi-Editor Support
  - **Validation**
  - Code Generation, Coding and Debugging
  - Model Analysis
Demo: Workbench with Workspace
Underlying platform: Eclipse Theia

- Windowing
- Workspace
- Menus and Status Bar
- Integrated Terminal
- LSP Support
- Many Extensions
  - Compatible with VS Code
  - Debug
  - Git
  - Problems View
  - ....

Browser / Electron
Theia frontend
Generator UI

Node.js
Theia backend
Generator Launcher

JVM: Java
Code Generator

© 2021 EclipseSource | https://eclipsesource.com | Dr. Maximilian Koegel | Model validation, diffing and more with EMF.cloud
Demo: Form-based editing
Involved components: Form-based editing

- **Tree**: EMF.cloud Tree Component
- **Forms**: JSON Forms
  - Declarative:
    - JSON Schema
    - UI Schema
  - Highly extensible
  - Multi-platform:
    - React
    - Angular
    - Vue.js
    - Mobile (Ionic)
    - Vanilla
Demo: Graphical Modeling
Involved components: Graphical Modeling

- Graphical Language Server Platform (GLSP)
  - LSP for Graphical Editors
  - GLSP client:
    - generic
    - renders graphical visualization
  - GLSP server:
    - specific to DSL
    - maps model to graphical visualization
  - Based on Eclipse Sprotty
Demo: Textual Modeling

```
1 machine: SuperBrewer3000
2 workflow: BrewingFlow
3 4 probabilities
5  low : 0.1
6  medium : 0.5
7  high : 0.75
8  assertions
9  Preheat => Brew, Preheat =>
```

- Brew
- Check drip tray
- Check Water
- ffsdfs
- Preheat
- Push
- Refill water
- Water Ok
Involved components: Textual Modeling

- Text Editing: Theia Editor (Monaco)
- LSP Server: XText LSP Server
Demo: Versioning and Comparison
Involved Components: Comparison and Versioning

- **EMF.cloud Compare: alpha**
  - Tree-based Compare UI
  - GLSP Compare UI
  - Compare Backend
    - Compare CLI
      - Compare/Highlight/Merge
      - Based on EMF Compare
    - Git Integration

- **Theia Git Extension**
Demo: Multi Editor Support
Involving Components: Multi Editor Support

- **EMF.cloud Model Server**
  - Defines a common API and protocol
  - Facilitates reusing EMF-based models
  - Encapsulates EMF dependency

- **Features**
  - Fetch and load models
  - Command-based change interface
  - Notification mechanism via sockets
  - REST APIs for clients (Java, TS/JS)
Demo: Validation
Involved Components: Validation

- **EMF.cloud Validation:**
  - Validator Extensions:
    - EMF Validation
    - Custom
  - Feature of Model Server
  - Integration with GLSP
- **Theia Markers Extension**
  - Problems View
Demo: Code Generation, Coding and Debugging
Involved components: Code Generation

- Generator framework: Eclipse Xtend
- Generator: jar build with Maven
- Integration:
  - Launched on demand via CLI
  - Generates into selected Theia workspace folder
Involved components: Coding and Debugging Java

- **Code Editor via LSP:**
  - Frontend: Theia Code Editor
  - Backend: jdt.ls (JDT)

- **Debugging via DAP:**
  - Frontend: Theia Debug Extension
  - Backend: Java Debug Server
Involved components: Coding and Debugging C++

- Code Editor via LSP:
  - Frontend: Theia Code Editor
  - Backend: `clangd` (clang)

- Debugging via DAP:
  - Frontend: Theia Debug Extension
  - Backend: GDB
Demo: Model Analysis
Involved components: Model Analysis

- Visualization: D3.js
- Analysis: Custom Kotlin code
Summary

- Web-based tools are the new normal
- Numerous building blocks available
- Reuse and migration facilitated
- Architecture Blueprints in EMF.cloud
  - Coffee Editor: Github
  - Ecore Editor: Github

→ How to start?: Build POC based on Blueprint

→ Feel free to contact me for questions: coffee@eclipsesource.com

→ Join our BOF today 6:15 pm CET on “Building web-based tools”
Evaluate the 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.

WITH 5 stars