Migrating Eclipse-based Tools/Plugins to Eclipse Theia or VSCode

Jonas Helming
EclipseSource
Migrating tool to 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)
Note that you can single-source most tools features between cloud and desktop.

=> Many vendors follow a dual strategy: Web-based on the desktop first, add cloud offering as a second step.

<table>
<thead>
<tr>
<th>Desktop</th>
<th>Online</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users get what they are used to</td>
<td>Easy set-up and onboarding for users</td>
</tr>
<tr>
<td>Modern rendering, but still all desktop features</td>
<td>Potential for new business models (SaaS or &quot;device farms&quot;)</td>
</tr>
<tr>
<td>Local Set-up necessary</td>
<td>Significant effort for the infrastructure for the provider (creation and maintenance)</td>
</tr>
<tr>
<td></td>
<td>Might conflict with security and IP governance</td>
</tr>
<tr>
<td></td>
<td>On-premise version might be necessary</td>
</tr>
</tbody>
</table>
The underlying tool platform

Theia vs. VS Code (see also this article)
Conceptional assessment of your Eclipse tool/plugin
Conceptional assessment

- Redesign your workflows
- Remove unnecessary features
- Redesign your UX
Technical assessment of your Eclipse plugin
Technical assessment
Technical assessment - examples

- A code generator
- A custom DSL (e.g. with XText)
- A diagram editor
- Dedicated process / CLI
- Language Server Protocol (LSP)
- Eclipse GLSP + Custom server interfaces (e.g. REST or Sockets)
Preparing the architecture of your Eclipse plugin

- **Eclipse desktop**
  - UI
  - Headless

- **Web-based Tool**
  - Client (HTML, JS/TS, CSS)
    - Client Extension
  - Server (Node.js, Java, CLI, etc.)
    - Server Extension

- **reimplement**
- **reuse!**
Technology selection for the web-based version
## Technical assessment - examples

<table>
<thead>
<tr>
<th>Eclipse desktop</th>
<th>Eclipse Theia / VS Code</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The coffee editor
What extension mechanism to use?
VS Code extensions vs. Theia extensions
Define a Strategy
Strategies for migrating your tool

- Evaluate technology choices
- Build PoCs
- Define MVPs
- Go Iterative
Questions?

Today 17:00: Spotlight session

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

Wed 27.10., 13:50, CDT.cloud? C/C++ tooling in the web

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

Thu 28.10., 16:10, Papyrus UML - the first stage of a journey to the cloud
Evaluate this Session:

- 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 session 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.
Backup slides
Web-based vs. cloud-based tools

● Web-based:
  ○ Use web technologies, especially in the UI, such as: HTML, CSS, JavaScript/TypeScript
  ○ Use a “browser” for UI rendering (real browser or Electron)
  ○ Usually split UI and a “backend” server for the business logic, even when deployed on the desktop

● Cloud-based:
  ○ Backend is deployed remotely (in the cloud)
  ○ UI is accessed via: Browser or Desktop tool connecting to the backend

● Cloud-based tools are usually web-based, web-based tools are usually prepared for the cloud!