Building XML-based content for Eclipse Help: a real experience

Christian Kurzke
Gustavo de Paula
Daniel Moura
Agenda

• MOTODEV for JavaME Tools
• Documentation Requirements
• Documentation Structure
• Content Source
• Content Transformation
• Eclipse Help Structure
• Conclusion
MOTODEV

- MOTODEV is the Motorola developer network

- Provides tools, documentation and support to developers

- Targets all different Motorola devices, from mobile devices to set-top boxes

- One of the main platforms supported by MOTODEV is mobile Java (JavaME)
MOTODEV Tools for JavaME

- Full JavaME IDE
- Based on Eclipse Platform+ JDT + EclipseME
- Includes an UEI JavaME Emulator
- Support most of MIDP 2.0 Motorola devices
- Includes complete documentation about the Devices and the APIs
- Include external tools that support the development
- Includes demo of JavaME APIs

- SDK that can be integrated on any UEI compliant IDE
- Based on Eclipse RCP
- Includes an UEI JavaME Emulator
- Support most of MIDP 2.0 Motorola devices
- Includes complete documentation about the Devices and the APIs
- Include external tools that support the development
- Includes demo of JavaME APIs
Documentation Requirements

• There are different types of documentation all of which must be included with the tools
• Most of the documentation is shared between both tools
• Same content source must be used to generate the PDF versions of the documentation to be placed on MOTODEV Website (deployment format independency)
Documentation Structure

• Several distinct types of documentation were identified
  • API Guides
  • JavaDocs
  • Device Matrix
  • Spec sheets
  • User guides
  • Tips and Tricks
  • Cheat sheets

• Some of the types can be reused in both tools
Documentation Structure

- Documentation was broken into two main groups
  - Tool specific documentation
  - Device specific documentation
- Device specific documentation is common to both tools
- Tool specific documentation change from tool to tool
- Each tool has its own documentation feature (docs.studio and docs.sdk)
- Device common documentation has its one feature (docs.device)
- Each type of documentation is pack in its own plugin
Content Source

• Some of content types sources are already defined
  ✷ i.e. Javadocs

• Other types need to be independent from deployment format
  ✷ API Guides
  ✷ Spec sheets

• Necessary to identify a way to represent the content sources
Content Source

• Solution is to use an XML-based format to represent those content sources
• There are already some solutions for that
  - DITA
  - Docbook

• We used an in-house developed pre-existing template framework
  - Originally used only to generate PDFs
  - Later extended to generate HTMLs

• The framework defines
  - An XML schema
  - A set of templates (transformations) for each type of content that generates PDFs
  - A set of templates (transformations) for each type of content that generates HTMLs
Content Transformation

Transformation is executed during Tool build process

- It is also possible to call it separately during “development”
Content Transformation – Directory View

Each content type has its own build
XML and images represent the content

Transformation framework
Used by each content that needs transformation
PDE build is then used to generate plug-ins
Eclipse Help Structure

- Device specific documentation (common)
- Tool specific documentation
Conclusion

• XML based content is a good way to write content to Eclipse Help

• There are existing solutions that can be used to maintain and transform existing content to fit the Eclipse Help structure

• The content structure is quite complex to organize and the effort to create Help content must not be underestimated