Visual Editor Project 1.1 Release Review

Slide deck revision 3.0

-- VE Project Leaders

 July 22, 2005 | http://www.db4o.com

VE 1.1 Highlights

- We did what we promised to do:
  - Improved performance and scalability
  - Began documenting APIs: Eclipse Corner article
- ...and we did even more!
  - Improved Swing and SWT developer experience
  - Supported natively editing RCP Views and Editors

Themes and Plan Items

- Theme: Platform Maturity
  - Committed plan items
    - Platform performance – The Visual Editor is now 40-60% faster on startup, along with other improvements
    - API documentation – We published our first Eclipse Corner article describing how to customize the Visual Editor

Themes and Plan Items

- Completed plan items that had been marked proposed or deferred in our plan document
  - Evolve RCP Support – We introduced support for directly editing Eclipse Views and Editors
  - More coding pattern support – We now support JBuilder and NetBeans code generation patterns
  - Cut/Copy/Paste – now fully supported

New and Noteworthy—M1 and M2

- Much performance work
- SWT codegen now produces more idiomatic SWT code
- Improved font property editor
- Improved color property editor
- Improved TableColumn support
- Better PDE development support
- Ability to edit RCP views
- Ability to edit RCP editors
- Ability to launch RCP views or editors separately from the RCP
- Preferences-aware RCP View tabs
- Improved support for parsing complex method arguments
- Support for new SWT controls
- Enhanced New Java Visual Class wizard
- SWT Shell re-parenting support
- Grouping of generated expressions
- New detailed VE extension tutorial
- Cut/Copy/Paste
- GridLayout tooling enhancements
- New visual GridLayout editor
- X/Y layout enhancements
- Prompt for bean name
- Turn off prompt for bean name
- Rename multiple beans at once—improved dialog
- Improved SWT container support
- SWT Custom Controls support
- Improved editor state persistence
- Improved Java Bean customizer support

 |

Performance work

- VE startup performance between 40% and 60% faster
- VE now caches background JVMs
  - When VE needs to restart the background JVM, it uses a cached VM, then starts a new one in the background
  - The user perceives the restart to be instantaneous
- Lots of other caching added...
- See also
  - http://dev.eclipse.org/viewcvs/indextools.cgi/%7Echeckout%7E/vep-home/WebContent/docs/performance/index.html

 |

Non-Code Aspects

- Increased competition from other Java-based visual editors has strategic importance for Eclipse
- New! Support for directly editing RCP views and editors

 |

Non-Code Aspects

- VE self-published an article on extending VE
- This article recently was promoted to being a full Eclipse Corner article
- We have begun to see significant input from the community
- 4th most popular download at Eclipse.org!

 |

Non-Code Aspects

- VE itself has its strings fully externalized and has been translated into German, Spanish, French, Italian, Japanese, Korean, Portuguese (Brazil), Traditional Chinese and Simplified Chinese
- VE's dialogs are all accessible
- VE is otherwise as accessible as makes sense for a GEF-based visual editor
- VE's generated code inherits Platform's excellent support for internationalization, localization, and accessibility

Platform Quality API

- The initial code contribution was a tool with no defined API boundaries
- We have initiated an iterative process to define our API
  - In this release cycle, we have tentatively defined some API boundaries
    - Result: we have moved from 100% internal API to a “friends” API
      - (www.eclipse.org/org/processes/Eclipse%20Quality%20APIs%20v2.pdf)
  - In this release cycle, we have gone from zero documentation to publishing an article explaining how to extend VE—explaining the new APIs
  - As we see how people use these APIs and what issues they encounter, we will be able to solidify them into Eclipse Platform quality APIs
  - Lather, rinse, repeat.

Architectural Issues

- VE 1.0 was not considered usable by a large portion of the community due to its lack of performance
- We aggressively added caching to many of the communication paths used inside the VE to improve this situation
- The caching work caused significant internal rewrites and a certain amount of API breakage to our internal APIs

 |

End-of-Life

- Since all API was declared internal in v1.0, we have no end-of-life issues.

 |

Bugzilla

- Between December 9, 2004 and July 15, 2005
  - 633 reports created
  - 337 reports fixed
  - 127 resolved without changing code
- Current state:
  - 3 blocking 2 critical
  - 4 P1 and 16 P2 defects

Standards

- J2SE
  - Our tooling is compatible with J2SE 1.4 and 1.5.
    - We generate 1.4-compatible code

 |

Schedule

- We promised M1, M2, M3, and then 1.1 final
- We made a tradeoff for more RCP features instead of an M3 build and slipped the schedule for M1 and 1.1 final

Target	Expected	Actual
M1	Fri, 18 February	Mon, 25 April
M2	Fri, 1 April	Fri, 10 June
M3	Fri, 10 June	*Cancelled*
Final	Fri, 15 July	Fri, 22 July

- Target completion on July 22, or 1 week later than scheduled
  - (but with greatly increased functionality than originally promised)

Process

- The Visual Editor Project is developed using an open, transparent, and inclusive process
- The VE team relies on Bugzilla, the newsgroup, and the project mailing list
  - Due to the small project size, we mainly rely on Bugzilla and the newsgroup
    - (ie: we don't enforce a strict separation on the newsgroup of questions for VE users and VE developers. However, the mailing list is strictly for VE developers.)
- Synchronous project communication is done on an ad-hoc basis as needed; results are documented on the newsgroup

Community

- The VE team actively monitors Bugzilla, the newsgroup, and the mailing list
- The VE team leader blogs about issues related to Eclipse and VE
  - http://www.coconut-palm-software.com
- The VE team presented one session and one tutorial at EclipseCon 2005

IP Issues

- In-bound Licensing
  - All contributors have signed committer agreements
  - So far, we have accepted ideas, but not code from the community
  - All in-bound code licensed by committers under the EPL
- Out-bound Licensing
  - All VE code provided under the EPL
  - Each source module has an EPL attribution (we use a script to do this)
- Due Diligence and Record Keeping
  - We currently use no 3rd-party (non-EPL) components

Conclusion

- On the tool side, we have delivered much more than we promised in only one week more than the original time allotment
- Performance is starting to be good enough that a significant portion of the community can and will use VE
- The initial code contribution was a set of tools that were also chartered with being a framework, that themselves had no defined API boundaries
- We have made excellent progress toward stabilizing the internal architecture of our tools and beginning to define APIs

 |