Eclipse Platform Vision and Strategy

Goal

- To realize a vision for the state of Eclipse developer tools over the next three years
- To create a strategy for implementing that vision

What is “The Platform”?

- The *Developer* Platform
- Developer Tools
- Eclipse Platform, JDT, Web Tools, CDT, EGit, Mylyn, ...
- Che, Orion, Flux

Players

- Wayne Beaton
- John Arthorne
- Dani Megert
- Martin Lippert
- Alex Kurtakov
- Tyler Jewel
- Max Andersen

Vision

Our vision is to build leading desktop and cloud-based development solutions, but more importantly to offer a seamless development experience across them. Our goal is to ensure that developers will have the ability to build, deploy, and manage their assets using the device, location and platform best suited for the job at hand. Eclipse projects, the community, and ecosystem will all continue to invest in and grow desktop Eclipse. Full-function cloud-based developer tools delivered in the browser will emerge and revolutionize software development.

Continued focus on quality and performance, out-of-the-box experience, Java 9, and first class Maven, Gradle, and JVM Languages support also figure prominently in our vision of a powerful developer’s platform.

Vision: Desktop and Cloud

- Desktop Eclipse will remain dominant for the foreseeable future
- Cloud-based developer environments like Che and Orion will revolutionize software development
- Developers can choose the most appropriate tools and environment
  - Move projects from desktop to cloud and back

Vision: Community Investment

- Desktop Eclipse developer tools will gain momentum
- The community will continue to invest in desktop Eclipse

Vision: Java 9, Maven, Gradle, JVM Languages

- Open implementation of Java 9 JSRs
- Maven support is “cost of admission”
  - Java developers won't take us seriously if we don't support Maven
- Gradle is the future (maybe)
  - We must have a good Gradle story
- JVM Languages (e.g. Scala) are growing in popularity

General Strategy

- Build momentum
  - Shift focus to the user
  - Identify and address user-experience concerns
- Maintain Eclipse “coolness”
  - Make a Eclipse a place where users and contributors want to spend to spend time
- Actively recruit resources
- Get Che project running

Focus on the User

- Make “packages” more like products
- Holistic approach to user experience
  - Front to back: website, downloads, install, use, extend
- “Every Detail Matters”
  - Start with the small stuff
  - e.g. “line numbers on by default” was a big win

“Little Things”

- Meaningful feature names and descriptions
- User-focus in simultaneous release repository
  - Functionality over SDKs
- Oomph-based installer (really a big thing/deal)
- Streamline website
- Identify and nurture “important” bugs

Quality and Performance

- Error reporter
- Long running process detection
- Misbehaving feature identification
- Platform/JDT performance tests running on foundation hardware

Reduce Friction for Committers

- Honor the principles
  - Open source rules of engagement, IP Policy
- Everything else is on the table
  - Revise the EDP
  - Streamline processes to keep committers focused on writing code

Marketing Focus

- Focus marketing resources on developer tools
- Developer tools on home page
- Demo camps
- Evangelism

This Won't Just Happen

- The vision is a rallying cry, not a guarantee or promise of delivery
- This will be an iterative process and will take time (especially given limited resources)
- We need to actively recruit development resources
- We need to find new ways to bring resources to bear

Go Forward

- Make it even easier to contribute
  - Oomph
  - Gerrit/Bugzilla/IPZilla automation
  - Revise the EDP and EF processes
- Biweekly meetings to track progress
- Engage the Web Tools PMC
- Engage the broader Eclipse project community
  - IDE discussion group, Commons project, ...
- Evangelism

Conclusion

- Eclipse projects, the community, and ecosystem will all continue to invest in and grow desktop Eclipse
- Full-function cloud-based developer tools delivered in the browser will emerge and revolutionize software development
- Developers will be able to switch between and mix-and-match their tool platform as required
- Quality, performance, functionality, and out-of-the-box experience will to continue to improve