eclipseCON 2004

- ▶ Home
- ▶ Program
- ▶ Registration
- ▶ Hotel
- ▶ Sponsors
- ▶ Be A Sponsor
- ▶ EclipseCon Friends
- ▶ Promote EclipseCon
- ▶ Press Room

2004 Gold Sponsors:

hp invent

IBM

SAP

WIND RIVER

2004 Silver Sponsors:

ASC

!Candle

catalyst SYSTEMS CORPORATION

ETRI

ILOG
Changing the rules of business™

INNOOPRACT

intel

mks

MontaVista SOFTWARE

ontometrix

PARASOFT

QNX SOFTWARE SYSTEMS LTD.

Scapa Technologies

serena
Automating Change

slickedit

SoftLanding

Descriptions

Keynotes:

Eclipse: State of the Union

The Eclipse journey continues! In this keynote we survey the latest progress on the Eclipse 3.0 platform and look into the evolution of the Eclipse community. We wrap up with a glance into the crystal ball.

Erich Gamma, Eclipse JDT lead   **John Wiegand**, Eclipse platform lead

---

The Eclipse.org Tipping Point

Every once and a while, people forget that it is just not possible to create a universal framework for integrating software development tools, and another grand project is announced. These projects inevitably fail for the same reasons they have always failed: the scope is too limited to be interesting, the scope is too large to be implemented, the tools require developers to change certain habits, or the tools do not present enough standardization for integration to be observed. Those, and a thousand other reasons, have frustrated attempts to improve tooling, and therefore developer efficiency, beyond narrow technology and industry segments.

Eclipse.org appears to be a tipping point in the making. This talk will address how and why this happened, and what the implications are for developers and the industry.

Michael Tiemann, RedHat

Aspect-Oriented Programming

Aspect-oriented programming (AOP) is an important new development in programming languages and tools. This talk will comprise a range of AOP material, from introduction, to advanced implications. I will also talk about why the most important AOP development are still to come, and what we can expect to see at the programming language, design tool, and process levels. I will also discuss why Java and Eclipse are the leading AOP platforms, and what needs to be done to retain that lead.

Gregor Kiczales, University of British Columbia

From IDE to XDE to CDE

Developing software-intensive systems always has been, is, and will remain essentially hard. As such, the entire history of software engineering can be characterized by a rise in level of abstractions, which is manifest in our programming languages, our methods, and our tools. This presentation will examine the role of tools in this spectrum of abstraction, from command line tools to IDEs and beyond. We'll examine the seminal development along the way, developments that represented a state change in the developer experience, and project out to the developer experience we expect to see over the coming years.

Grady Booch, IBM Rational

The Business of Open Source

Any successful rebellion has to face the prospect of growth and success, and the open source movement is no different. Initially perceived as being about free stuff for pioneers, the need to scale to market-sized proportions looms large. This keynote will consider Benkler's view that open source is 'commons-based peer production' and explore possible futures for open source adoption.

Simon Phipps, Sun Microsystems

Tutorials:

Dwight Deugo, Espirity Inc

Getting Started with Eclipse

This tutorial is for people that have not yet worked with Eclipse. You'll explore Eclipse's architecture and become familiar with the plug-in mechanism. We will start with an overview of Eclipse and how to install and run it. Next, we will describe Eclipse's Workbench and its resources, views and perspectives. Next we will examine Eclipse's Java Development Tools that implement a Java IDE supporting the development of Java applications and plug-ins. We will then focus on how testing and debugging is done in Eclipse. Finally, we will introduce Eclipse's architecture and how it supports plug-ins. With this knowledge,

you will be able to navigate through Eclipse, be able to create your own Java projects, packages and classes, and be ready to learn how to write plug-ins.
Tutorial participants are invited to bring their laptops to get started with Eclipse by working on hands-on exercises.

Who should attend: Java programmers interested in a tour of the eclipse platform and IDE, its features and architecture.

Kai-Uwe Maetzel,
Eclipse Platform
Text Lead

Contributing to Eclipse: understanding and writing plug-ins

Eclipse was designed from the ground-up with extensibility in mind. A scalable plug-in mechanism and a rich set of APIs enables developers to develop and explore new tools quickly, without having to start from scratch.

This tutorial illustrates the full plug-in development cycle by way of an example. You'll explore the Eclipse architecture and become familiar with the basic plug-in mechanism. With this knowledge, you'll write and debug your first plug-in using Eclipse's Plug-in and Java Development Environments. You'll also add extension points to your plug-in to enable others to extend it. You will then package your extensible plug-in as a Feature and publish it with the built-in Eclipse Update Mechanism. Finally, you'll set up and manage an Eclipse Update Site, a place for other Eclipse users to explore new features as well as finding upgrades.

During all these steps, you will learn the underlying Eclipse concepts and design ideas. You'll learn the rules you need to know to make your plug-ins good Eclipse citizens. In addition, you will receive many interesting insights on design challenges in large scale plug-in architectures.

Tutorial participants are invited to bring their laptops and deepen their understanding by implementing practical exercises.

Who should attend:
Java programmers and experienced eclipse users interested in learning to write plug-ins. Developers with an interest in large-scale plug-in architectures will also benefit from the insights into Eclipse.

Steve Northover,
Eclipse SWT Team Lead
Silenio Quarti,
Eclipse SWT Technical
Lead

SWT in Depth

This tutorial is aimed at experienced, Java programmers who want to use the Standard Widget Toolkit to build graphical user-interface (GUI) based, applications.

The Standard Widget Toolkit is a Java class library for creating graphical user-interfaces. It was created, as part of the Eclipse project, to make it possible to build efficient, portable applications that directly access the user-interface facilities of the operating system. Using SWT it is possible to create applications in Java that are indistinguishable from native applications on the desktop.

By breaking the task of building a graphical user-interface based application into component parts, and by showing how these are modeled in SWT and then providing a series of examples, we will provide a guided tour through the toolkit, focusing on what application programmers need to know.

Requirements: You'll need to bring a laptop.

Michael van Meekeren,
IBM

J2ME and Eclipse

While Eclipse provides support for Java program development, such as editing, compiling and debugging, and it is readily extensible through its plug-in mechanism. We have been involved in the development of a set of plug-ins that support the building and launching of embedded applications (with support for various platforms, such as J2ME/MIDP, PocketPC and PalmOS). We will show how applications can be developed, compiled, analyzed and compressed to fit on really small devices. We will demonstrate how to debug applications running in either an emulator or on a real device.

We will also discuss how these Eclipse plug-ins were developed, what trade-offs we encountered, and what lessons we learned, and we will offer suggestions that will benefit (future) Eclipse plug-in writers.

This technical tutorial will include lots of actual code and reports on practical experience. We will provide background information on developing Java applications for resource-constrained environments, such as PalmOS, and explain what Java standardization processes are under way in this area.
Attendee background

Prerequisites: Attendees must have basic experience with Java and any integrated development environment (IDE).
Format - Lecture and demonstrations

Technology Exchanges:

Darin Wright,
Eclipse Debug Lead

Debugger Implementors

This Technology Exchange will focus on the facilities provided in the Eclipse Debug Platform for building debuggers and launching applications. Topics of discussion will include the debug model, launch configurations, and scalability issues in the debug perspective. Those who are building integrated debuggers should attend.

Jean-Michel Lemieux
Eclipse Team Component Lead
Michael Valenta,
Eclipse Platform Team Component Committer

Implementing Repository adaptors

This Technology Exchange will deal with the integration of Team support into Eclipse. The current APIs will be discussed along with tips and tricks for using them. In addition, new 3.0 APIs will be presented and discussed. The new APIs includes a pluggable Synchronize view and the possibility of including the JSR-147 Workspace Version and Configuration Management (WVCM) API into Eclipse. Also of interest is non-versioning team support and other team related tools (e.g. change management and collaboration).

Jeff McAffer ,
Eclipse Equinox Project Lead
Nick Edgar,
Eclipse Platform UI Technical Lead

Implementing Rich Client Applications

Ever wanted to use Eclipse to write a non-tooling application but were scared by the bulk? Didn't want a tool bar? Wanted to have dynamic plug-ins but were afraid to ask? Well, now is your chance to learn about writing Rich Client Applications using Eclipse. Less is more and we have trimmed all of the IDE related features and functions out of the base workbench. The runtime now offers more configurability and features such as dynamic plug-ins. So whether you are a plug-in guru or have never implemented a plug-in before, come and discuss your requirements and explore how the Eclipse Rich Client Platform addresses them.

Frank Budinsky,
Eclipse EMF Project Lead,
Ed Merks,
Eclipse XSD Project Lead,
Jim D'Anjou,
IBM
Eric Chaland,
IBM

EMF Technology in practice

The Eclipse Modeling Framework (EMF) has been available for more than a year now, and its use as a rapid development tool and data integration framework for Eclipse plug-in developers is steadily increasing. The goal of this Technology Exchange is to share experiences and techniques between plug-in developers that are currently using EMF, and to get a sense of the range of applications that are being developed using EMF. Identification of shortcomings of the framework or generator, suggestions for future enhancements, effective techniques for working with EMF, are all welcome topics of discussion. Since one of the main purposes of EMF is to facilitate data integration between tools in Eclipse, experiences that either support or refute this goal would be particularly welcome. Experiences using the underlying EMF template-based generator framework (JET) are also welcome.

Margaret-Anne Storey,
University of Victoria

Eclipse and Education

The goal of this Technology Exchange is to explore how Eclipse is currently being used and how it can be further leveraged as a platform for educational tools in computer science and software engineering. Currently, Eclipse is seeing some use in the classroom and workplace for introductory programming and software engineering courses, as well as for more advanced courses on testing, software design and evolution. One hypothesized advantage of this approach is that students and educators can learn within a relevant context while having access to a set of integrated and powerful tools. In this exchange, we will consider both the technical and socio-technical aspects of this trend and brainstorm about new possibilities. We invite people to share their experiences, ideas and enthusiasm as we attempt to improve education in computer science.

John Duimovich,
Tools Project Lead

IDE Implementors

The IDE Implementors Technology Exchange is intended for people who are writing programming language IDEs on top of the eclipse platform in Java, C/C++, Smalltalk, Python, Ruby, Scheme, Pascal or any other language! Come to this working session to discuss the common issues in writing language IDEs. If you have written an eclipse plug-in for your favorite language or even if you plan to write one and want to talk to others who have done it, this session is for you.

Michael Burke,
IBM Research
Brian Barry,
Bedarra

eTx: eclipse and research

Eclipse offers the promise of common infrastructure for researchers in areas such as programming languages, tools and environments. As extensible and freely available open source software, the Eclipse platform facilitates collaboration and sharing of software artifacts. The last year has seen explosive growth in both interest and activity within the academic research community. The goal of this technology exchange is to bring together researchers working with Eclipse, and to explore the benefits, problems and possibilities of the Eclipse platform in this context. A number of research teams representing a broad cross-section of research areas will provide brief experience reports. We hope to use this event to facilitate a lively discussion on such questions as:

- How do I get started with Eclipse?
- Is Eclipse the right choice for me and my research team?

- How can I as a researcher work with other Eclipse projects and the Eclipse community?
- Are there opportunities for synergies between my project and other researchers?

Michael Norman,
Scapa Technologies
Antony Miguel,
Scapa Technologies
Richard Duggan,
IBM
Harm Sluiman,
IBM

Tracing, Logging and Monitoring Performance in Eclipse

As Eclipse usage matures it is natural for users to require additional tooling to analyze the performance and behavior of systems they have built in the IDE, and indeed to use such tooling at all stages of the lifecycle from development through test and into production. To facilitate this, the Eclipse Hyades project provides data collection infrastructure and analysis tooling for traces, logs and systems performance data. There are APIs so that it is extensible at both ends: you can write additional user interface components to provide deeper analysis of the data, and you can write additional data collection agents to instrument a specific system, or to interface with an existing data collection infrastructure such as PerfMon, SNMP or an enterprise management framework.

This Technology Exchange brings together practitioners who are currently building the open-source Hyades infrastructure layers, agents and tooling, and those building additional open-source and commercial tooling on the infrastructure, and deals with the practical issues of binding to the infrastructure, using the correlation service to manage event stream interrelationships, passing data through the infrastructure as XML fragments which may be based on Common Base Event (CBE), constructing effective loaders to pass the data into Hyades EMF models, overcoming scalability issues in EMF for large data models, and extending the existing Hyades user interface framework and tools.

Technical Talks:

Dirk Baeumer,
Eclipse JDT
UI Lead

Philippe Mulet
Eclipse JDT
Core Lead

Manipulating Java programs

The Java Development Tooling (JDT) adds Java program development capabilities to the Eclipse platform. In addition to the environment itself a set of APIs are provided to further extend the JDT. In particular, it offers services to introspect and manipulate Java source code.

This presentation will take you on a tour through the services involved in implementing a certain refactoring. We will see how the operation uses the Java model, the search engine, the abstract syntax tree, the code rewriting infrastructure, and the refactoring framework itself. The presentation will also outline how JDT's services have evolved to allow other Java-like language tools to participate in operations like search, refactoring and quick fix.

Kai-Uwe Maetzel,
Eclipse Platform
Text Lead

Text editors and how to implement your own

This talk introduces the world of Eclipse text editors to you. It will start with a brief architectural and conceptual overview helping you to understand the functionality provided by the Eclipse Platform and where it is implemented. It will then dive into the details such as syntax highlighting, content assist, rulers, annotations, hovering, and quick diff thereby pointing out differences between the current development stream and previous versions.

Tod Creasey,
Eclipse platform
UI Committer

Scaling large eclipse applications progressively

Many Eclipse based products have a huge amount of functionality making them very complete but intimidating and cluttered. Through progressive disclosure an Eclipse application can show functionality as the user discovers it and filter content until the user is ready to use it. In this talk we explore how to make a less cluttered initial user experience, tailoring an Eclipse product for different users and gradually making functionality available as the user learns the product.

Jim des Rivieres,
Eclipse Platform
Committer

Eclipse APIs: Lines in the Sand

Useful and stable APIs are an important aspect of open systems like Eclipse. Designing good APIs and maintaining them is an interesting and ongoing challenge. Like the proverbial drawing of a line in the sand, the process of putting in an API starts in a wide open space of possibilities, and abruptly transitions into something quite constrained that must be held and defended. Drawing on our experience with the Eclipse APIs, this talk will discuss effective ways to draw the line, to make the line visible to all, and to move the line in response to a changing environment.

John Arthorne
Eclipse Platform
Core Committer

Writing responsive UIs using the Eclipse 3.0 concurrency architecture

One of the goals of Eclipse 3.0 is to make the UI appear more

Jean-Michel Lemieux,
Eclipse Team
Component Lead

responsive. One approach to addressing this involves moving time-intensive operations into background threads to allow the user to continue working. This required the creation of a concurrency infrastructure to manage interactions between background and foreground operations. New UI facilities were also needed for reporting progress on things happening in the background, adding indicators to views containing information in flux, and resolving situations where user activity collided with ongoing background operations. This talk will introduce the new concurrency architecture and UI facilities, giving examples of how to apply them in your own plug-ins. You should attend this talk if you are an intermediate to advanced Eclipse developer looking to write new plug-ins or adapt old plug-ins to play well in a more concurrent environment.

Dejan Glozic,
Eclipse
Update/Install Lead

Share this plug-in with the ones you love: using PDE to create an Eclipse plug-in and publish it on the update site

During this presentation, you will be taken through the process of using the Plug-in Development Environment to create a lowly 'Hello, World' plug-in, creating an installable feature for it, creating an Eclipse update site to share it with others, and using Eclipse Update to find it and install it into another Eclipse application. We will then create an update for it, and let automatic background update search find it. Finally, we will show how local administrators can mirror the site behind the firewall to conserve bandwidth and minimize download failures.

Karsten Schmidt,
SAP AG

Keep on swinging - productivity layers on top of SWT

We introduce some concepts that aim at increasing the productivity of tool development and at supporting the development of homogeneous UIs across an Eclipse based application - in our case SAP NetWeaver Developer Studio.

For Swing programmers, SWT / JFace is a low-level UI approach lacking many of the higher-level Swing concepts. Integrating Swing UIs into Eclipse is not an option if you want to achieve a consistent Look and Feel. We introduced a software layer on top of SWT / JFace that carries some Swing concepts to the SWT world. It supports the container / control approach based on add methods. Instead of directly assigning Windows style bits to public member attributes, you use swing-style methods to configure your UI parts. Different Look & Feels are possible by delegating the creation of widgets to a Widget Factory. Finally, Swing table and tree models can be used to create JFace tables / trees. This keeps the strict model / view separation and allows the reuse of existing models. All of this sits on top of SWT instead of replacing it. Thus native SWT programming is still possible. We also introduce ways how to ensure simple, well-designed usage patterns for a consistent UI approach across any application. Reusable UI components are the building blocks of such patterns. The predefined set of components includes: a Method editor, a Tree Selector, and several standard dialogs like a Message Dialog. Finally, we explain advanced techniques for image handling. We distinguish between globally available images and those belonging to one plug-in but also meant to be used by others. The reuse of both sorts is supported through a central access and administration layer.

Steve Northover
Eclipse SWT Team Lead

Inside SWT

This session will provide the definitive overview of SWT from the point of view of its original designer and the current leader of the development team.

Richard Wilson,
IBM Lotus

Lotus Workplace: Rich Client Platform

Come learn how IBM is using the Eclipse platform as part of the Lotus Workplace offerings. A major goal of Workplace is to support of a variety of client access points such as browser, mobile and rich client. This session will showcase how Lotus is using the Eclipse platform as the basis for the Lotus Workplace Rich Client Platform and will include demonstrations of some of the offerings using this platform. Finally, we will cover how we are working closely with the Equinox and Eclipse teams on the development of the RCP theme in Eclipse 3.0.

Todd Williams
Genuitec

Eclipse-based Applications - Java on the Desktop Revisited

The traditional approach to achieving "Java on the desktop" has resulted in a long journey toward an unfulfilled goal. Each successive release of Java technology has failed to easily enable the intrinsic qualities that are required for building successful and endearing business applications. The Eclipse Platform, while best known as an IDE, defines an extensible architecture, a rich set of frameworks, and unique user-interface capabilities that make it possible for Java applications to finally compete head-to-head with applications built using platform-native technologies.

This presentation will cover:

- The advantages of using an application framework
- Why is Eclipse the best framework for desktop Java applications
- The business case for using Eclipse as a framework
- Case Study - Building an Eclipse-based product
- The impacts of Eclipse 3.0's Rich Client Platform

Frank Gerhardt,
SENS
Chris Wege,
DaimlerChrysler AG

Experiences with rich client application development

We share our experiences in developing the GDFSUITE rich client application for processing geographic data. GDFSUITE is designed for extensibility and uses the Eclipse plug-in mechanism itself for extending the suite. Currently we use Eclipse 2.1 as basis for our application. We discuss issues related to the migration to the Rich Client Platform of Eclipse 3.0. Our development process is similar to the development process of the Eclipse team.

Nick Edgar,
Eclipse Platform UI
Technical Lead

Eclipse Rich Client Applications - Overview of the Generic Workbench

One of the key changes in Eclipse R3.0 is that the Eclipse Platform is being refactored to remove its IDE personality, allowing Eclipse to be used not just as "an open extensible IDE for anything and nothing in particular", but broadening its reach to be an open extensible platform for any application. This presentation will give an overview of the opportunity this presents, the technical details on the refactoring of plug-ins, and the impact on existing Eclipse-based products.

Jeff McAffer,
Eclipse Equinox
Project Lead

Inside the RCP Runtime: Dynamic Plug-ins and Beyond

The Eclipse community is increasingly interested in using Eclipse in non-tooling scenarios -- as a so-called "Rich Client Platform". These scenarios challenge the Eclipse runtime to support dynamic plug-ins, increased configurability and security without undue changes in API and level of function. This talk details how the Eclipse 3.0 runtime addresses these challenges.

Andy Clement,
Eclipse AJDT
Project Committer
Mik Kersten,
Eclipse AJDT
AspectJ Project
Committer

Getting started with aspect-oriented programming in Eclipse

AspectJ is a seamless aspect-oriented programming (AOP) extension to Java. It can be used to cleanly modularize the crosscutting structure of concerns such as exception handling, multi-object protocols, synchronization, performance optimizations, and resource sharing. When implemented in a non-aspect-oriented fashion, the code for these concerns typically becomes spread out across entire programs. AspectJ controls such code-tangling and makes the underlying concerns more apparent, making programs easier to develop and maintain.

The AspectJ Development Tools Plug-in for Eclipse (AJDT) fully integrates the AspectJ language into Eclipse. This session will provide an introduction to using AJDT, demonstrating how it is easy to progress from traditional Java development to a situation where you can exploit AspectJ. During the session we'll look at mining crosscutting concerns from some existing Java code base, debugging aspects, using aspects to enforce rules, and extending the core function of programs using unpluggable aspects. The latest AJDT provides tighter integration with the JDT features that Java programmers are accustomed to, along with new capabilities that surface the aspect-oriented structure of a system.

David Orme,
Visual Editor Project
Lead, ASC

The Visual Editor Project - Flexible GUI Building for Eclipse

You've likely spent time working with or looking Swing-based UI applications. Perhaps you've heard about the Standard Widget Toolkit (SWT), and the fact that it's a cross-platform, open source GUI toolkit for the Java programming language. But the most recent news is that Eclipse has just added a GUI builder project for Eclipse! In this session, David Orme, the leader of the new Eclipse Visual Editor project will take you through the ins and outs of the GUI builder being created for Eclipse, and how it will impact graphical user interface developers.

Sridhar Iyengar,
IBM
Ed Merks,
Eclipse XSD Project
Lead

Rapid plug-in development and integration using EMF

Many programmers, especially the experienced ones, often have little or no use for modeling. Maybe a class diagram or two to fill out the documentation, but other than that, it simply doesn't seem to help. Well, what if there was a framework/toolkit that brought you the benefits of modeling with a very low cost of entry? Enter the Eclipse Modeling Framework (EMF). EMF is a framework and code generation facility for building robust applications based on surprisingly simple models. Models can be defined in several different ways - Java interfaces, XML Schemas, UML/EMOF - from which EMF will generate a large part of the application. The generated code is clean, efficient, and easily hand

modified. You can even regenerate the model after changing the code, without wiping out your changes. This talk will describe and demonstrate EMF, showing how it brings the benefits of modeling to the mainstream Java programmer and how EMF and modeling facilitate tool/application integration. The talk will also give a brief overview of MDA (Model Driven Architecture) and explain how EMF forms the foundation of an Eclipse MDA tools platform.

David Frankel,
David Frankel
Consulting

The Eclipse Modeling Framework and MDA: Status and Opportunities

The Eclipse Modeling Framework (EMF) and the OMG's Model Driven Architecture (MDA) are pushing model-oriented strategies and techniques forward in the industry. This session will examine the relationship between these two initiatives. It will also explore the potential for this relationship to work synergistically with other model-centric movements including Generative Programming, Domain-Specific Languages, Product Line Practices, and Model-Integrated Computing.

Michael G. Norman,
Scapa Technologies

Why integrate test, trace, log and performance tools via Hyades?

Hyades is an Eclipse Tools Sub-project providing an infrastructure for integrating test, trace, log and performance tools at both the user interface and the data level. Hyades applies across the lifecycle from development through test and into production and is also applicable to systems integration in environments where no source code is present. Hyades provides a range of tools and tool components which operate inside that infrastructure, and extension points where vendors can add higher-value commercial plug-ins from profiling, testing, through log analysis, to systems performance and autonomic management. This talk explains the benefit that the unprecedented level of tools integration provides to the user of the tool, to the tools vendor seeking to provide high-value innovative tooling to customers, and to the vendor or integrator of the system to which the tools interface.

Randy Hudson,
Eclipse GEF Project
Lead

Building Applications with Eclipse's Graphical Editing Framework (GEF)

Checkbox tables and monospaced fonts don't get you excited? Come learn about the graphical editing framework and get a sneak peak of features in 3.0. Targeted towards developers using Eclipse, the session starts with an architectural overview and tutorial outlining the steps required to go from a domain model to a fully-functional graphical editor. Learn techniques for visualizing different types of documents, as well as some of the less well-known features like graph layout and Section 508 compliance (accessibility). Preview some of the upcoming Eclipse 3.0 features. Questions will be taken at the end.

Sebastian Marineau
CDT Project Lead,
QNX

Using Eclipse CDT for C/C++ Development

The open-source CDT project provides a cross-platform C/C++ tooling environment for multiple development hosts and target systems, allowing even large development teams to standardize on one IDE. Moreover, its widely supported plug-in architecture makes it possible to integrate tools from multiple vendors; developers can, in effect, take a best of breed approach to their tool selection.

This presentation provides an introduction to the CDT architecture and components, and uses many live examples to show how to best take advantage of its features. Participants will also get an overview on how to integrate other tools with the CDT.

Christopher Judd,
Judd Solutions

Eclipse for PHP Developers

Learn how the open source Eclipse IDE and PHP plug-ins can increase PHP developer productivity. This session will explain installation and configuration of the PHP plug-ins as well as specific PHP wizards, editors and debugger.

John Duimovich,
Eclipse Tools
Project Lead

How to build your favorite language IDE

The Eclipse IDE is definitely not just for Java. This talk will describe the creation of a Smalltalk IDE on the eclipse platform and some of the architectural decisions that went into building it. We will explore the levels of integration that a language IDE implementor should consider when bringing their language to the eclipse platform. If you have a favorite language and you have been thinking about writing an eclipse plug-in to support it, this talk is for you.

Dave Thomson,
IBM

A Different Shade of Blue: Moving Eclipse from Closed to Open Source

Learning to work in an open source project is a challenge for individuals. Multiply that challenge by 100 fold, for a large development team in a large blue company. This talk explores the journey of the eclipse technology and team from a (somewhat) traditional internal software project through the launch of the eclipse open source project. We'll talk about how we initially structured the project and the reasons for those choices. We'll take a look at the evolutionary path for the project and

some of the issues we met along the way.

Kevin Haaland,
Eclipse Platform PMC

JIT software development: Inside the Eclipse development process

Shipping releases on time for a large, geographically distributed software team with a complex, established codebase, defies the natural rules of people, geography and software development. While we haven't changed any natural laws, the eclipse development process has found ways to help mitigate the problems working in this environment. In this talk we'll describe the eclipse release process, some of the basic principles of how our teams work together and the important elements of the planning process.

Joshua Kerievsky,
Industrial Logic
Somik Raha,
Industrial Logic

Being Extreme with Eclipse

Eclipse is the most effective IDE for practicing Extreme Programming (XP). Its design embodies XP's values of Simplicity, Feedback and Communication. Its tools provide powerful support for test-driven development, refactoring, continuous integration, collective-code ownership and coding standard. This session will demonstrate how XPers use Eclipse to efficiently and effectively practice XP. We will also give a sneak peek of some of the latest XP development tools for Eclipse.

Boris Magnusson,
Lund University

Supporting Refactoring in Eclipse - needs and experiences from an XP-teaching setting

Eclipse supports performing basic refactorings, but merging parallel development including refactorings still frequently go wrong. Our experience from using Eclipse for teaching XP in projects show that refactorings still are best done when nobody else changes the system in parallel. This is contrary to how we want the students to practice XP with tight interactions and short development cycles. The situation can, however, be much improved by using advanced Configuration Management techniques. We demonstrate how storing information on which refactorings have actually been applied, can support merging refactorings with other edits and provide a consistent result. This also when the refactorings and edits interfere with each other. There is thus a potential to make refactorings a light-weight operation rather than a pre-planned activity also in a multi developer environment.

The talk will also describe how we teach XP in very intense projects with 10 students working concurrently using Eclipse. It will also discuss how the CM support for refactorings may improve the situation for changing published interfaces, a problem today with grave consequences for long-lived systems.

Claire Rogers,
HP

Developing Web Services with Open Source and Eclipse

Over the past year, Web services have been positioned as a key enabler to application integration and B2B integration. Companies such as Amazon.com and Google have already deployed web services, with real, demonstrated business value. Many software development vendors have made large investments in supporting the web services development process. However, for some companies just beginning to investigate the value of web services, the cost required to begin might pose a huge barrier. How then can development shops begin to explore this new and emerging technology? If cost is an issue during the investigation stage, high-priced development tools may not be an option, and teams may often have to look to Open Source to get started. This session takes an in-depth look at the web services development process, and the tools that can used to get started quickly. Tools such as Apache Axis, Ant, and Tomcat will be integrated with the Eclipse programming environment, demonstrating how a Java-based component can be exposed and accessed as a web service.

Jochen Krause
Innoopract

Web Development with the Eclipse Platform

Eclipse and Web development have not been a natural fit in the past. This session gives an overview about the variety of open source and commercial tools available for J2EE development, Web development frameworks and app server infrastructure support. Future developments of the Platform and resulting opportunities for web development will be explored.

Michael Bechauf,
SAP AG

Building Large-Scale Enterprise Applications with Eclipse

Eclipse and its unique open platform extensibility has enabled dramatic improvements in the Java development process used by large-scale enterprise solution providers and has propelled Java application development productivity to unprecedented levels. The key to these breakthroughs has been tightly-integrated tools created on top of the Eclipse framework that have supported developers in all phases of the software development life-cycle essential for production of fully integrated solutions. These tools have employed declarative, model-driven development techniques that have allowed developers to focus on the essential domain-specific problems thereby improving efficiency of development and the quality of solutions. The future of the Eclipse

platform will rely on the real-world feedback about advanced tools concepts that can best be provided by application solution developers. Their critical role and tight integration into the fabric of the Eclipse open source eco-system will lead to adoption by a wide range of industry sectors and drive continued technology innovation.

Karl Kessler
SAP, AG

Overview of SAP NetWeaver Developer Studio and Java Development Infrastructure

The Developer Studio is SAP's integrated development environment for Java applications that are targeted for SAP's NetWeaver application and integration platform. Based on Eclipse SAP offers a highly model driven tool set for a variety of different program models such as J2EE, Java Persistence and Web Dynpro, SAP's MVC based architecture for web user interfaces. The Developer Studio is tightly integrated into the Java Development infrastructure which is the basis for efficient team development offering repository, central build, deployment and distribution services based on SAP's component model for business applications written in Java.

Bjorn Freeman-Benson,
University of Washington

UrbanSim: An Open-Source Tool for Integrated Land Use, Transportation and Environmental Modeling

UrbanSim is "SimCity done for real" - an open source simulation model for integrated planning and analysis of urban development, incorporating the interactions between land use, transportation, the environment, and policy. Our agent-based model is behavioral and thus the operation of UrbanSim is fairly simple to understand, yet is still able to capture complex interactions in the markets for land, development, and transportation. It is a valuable tool for improving the level of understanding of how a metropolitan region is developing and how various combinations of land use and transportation policies and investments are likely to shape these trends. Some of the issues of interest, such as affordable housing, are easily within the scope of the model, since it deals with predicting housing prices, and disaggregates households by income as well as other characteristics, and can capture the affordability impacts of alternative scenarios.

UrbanSim is interesting to the EclipseCon audience because UrbanSim is one of the first non-code IDE uses of Eclipse; UrbanSim uses Eclipse as an integrated development environment for urban simulations and simulation scenarios. Urban simulations consists of four major phases: base year data preparation, scenario design, simulation, and results analysis. The base year database contains a description of the municipal area's land parcels, households, employment, zoning, transportation links, etc. A scenario is a delta on that base year - for example, "what if we build a new freeway?" or "what if we change the zoning in these neighborhoods to allow mixed use development?". Once the simulation run is complete, UrbanSim provides an extensible mechanism for evaluating the "goodness" of the results - in the planning community, these evaluations are known as "indicators" as in "an indicator of housing affordability" or "an indicator of employment density".

UrbanSim uses the Eclipse resource navigator along with UrbanSim-specific resources, editors, views, buttons, wizards (and all the rest of the Eclipse stuff) to provide the user with an IDE for urban simulation. We have user tested our IDE and have found generally good acceptance of the Eclipse user model amongst our urban planning user community - computer savvy, but not Java knowledgeable, senior planners at regional planning agencies.

In this talk, I will start by describing UrbanSim, the political issues arising from our customer's uses of UrbanSim and how open source helps mitigate those issues. I will continue with our mostly positive experiences creating a non-code IDE on top of the code-IDE-centric Eclipse platform. I will conclude with gracious acknowledgement of our sponsors and perhaps finish with a witty quip.

Heung-Nam Kim,
ETRI

Esto : An Eclipse based Embedded Software Development Tool

In this talk we will introduce the activities of ETRI, the national research institute of Korea. We'll focus on the work of the Embedded Software Research Center which includes digital home solutions such as home server set-top, PDAs and embedded media players. One of the center's primary projects has been the development of Esto (Embedded Systems Tool), an eclipse based IDE that features a Remote Debugger, Project Manager, Tracer, and the first Power Monitor and DDK (Device Driver Development Kit) plug-in tools for Eclipse.

Aaron Levinson,
Intel

The Intel VTune Performance Analyzer: Insights into Converting a GUI from Windows to Eclipse

Intel's VTune Performance Analyzer tool is currently available on Microsoft Windows platforms with both a GUI and command line user interface and on Linux with a command line interface. This presentation provides details and insights gathered by the VTune analyzer engineering team as it converted the VTune analyzer to the Eclipse 2.1 framework. Previously, the team had undertaken the task of implementing a command line interface for a formerly GUI-only Windows product and making this interface available on both Windows and Linux, thereby providing the industry with the first native version of the VTune analyzer

running on Linux. The next step was to provide a GUI for the product on Linux while continuing to support existing Windows GUI code, code that includes the ability to plug-in to Visual Studio .NET.

Eclipse was eventually chosen as the GUI framework for Linux, requiring the conversion of code written predominantly in MFC to Eclipse-centric Java. High-level source code organization, translation of existing Windows GUI behavior to Eclipse, and the use of undocumented Eclipse features are just some of the insights presented in this talk.

Emeka Nwafor,
IBM Rational

Using UML Views of J2EE Platform Elements to Improve Developer Productivity

This presentation discusses how UML can be used to provide an alternate view of code that facilitates the developer in gleaning and communicating important aspects of a J2EE application - aspects that aren't easily surfaced by a developer when working with the standard Java editors and Deployment Descriptor editors. The presentation will focus on technology being developed jointly by the Rational and WebSphere teams for the Studio tools for the Java platform. The presentation will include a technical overview of how the Eclipse JDT, EMF, and GEF technologies have been leveraged together with UML 2.0 to provide the developer with tools that facilitate the navigation, understanding, analysis, and communication of enterprise applications built on the J2EE platform. A demo of these capabilities will also be provided.

Tan Phan,
SlickEdit Inc

Eclipse Integration Lessons from the Trenches

The Eclipse framework provides many opportunities for developers to create plug-ins and make modifications to further integrate advanced functionality into the Eclipse workbench. In this session, Tan Phan will demonstrate specific integration challenges and solutions, along with sharing experiences so that others may accelerate their own development. Examples that are applicable across many types of integrations will be covered, such as overriding the Eclipse workbench key bindings, addressing logging errors, and removing or fitting duplicated functionality. Any developer writing plug-ins or using the Eclipse framework to develop other Eclipse-based products will want to integrate with the workbench help system. Tan will demonstrate how far SlickEdit was able to go with the online help system, including help content, searching, indexing, tutorials, and product documentation. Finally, branding considerations will be discussed, focusing on providing a cohesive user experience for all integrated products.

Brad Van Horne,
MKS Inc.

Java Application Lifecycle Management-Development through to Deployment

Organizations running mission critical Java applications have a growing need for a managed, repeatable process to reliably develop, build and deploy these applications. This presentation, targeted at development managers and production deployment specialists will address the following concerns:

- How can development teams work on multiple versions of their application, while guaranteeing no unplanned changes are introduced into production when fixes are required?
- How can an Eclipse development team reduce the time spent on destabilizing manual source code merging and unproductive ANT scripting?
- How can an engineer clearly see the status of every file in their Java project allowing them to make informed development decisions?
- How do you deploy what was tested while taking into account that different environment variables are likely needed for test and production?

The attendee will learn how to implement a repeatable software development process that provides control over costs and quality without inhibiting their developer's creative talents.

Eric Clayberg,
Instantiations
Dan Rubel,
Instantiations

Battlefield Experiences with Eclipse: Supporting Multiple Eclipse Versions Simultaneously

Many developers, especially tool developers who write code for Eclipse itself like ourselves, prefer to be using for the latest bleeding edge code, but large IT departments and other paying customers tend to prefer to lag the edge, using older more established software. We need to develop products that run on multiple versions of Eclipse so that we can support the widest range of customers as possible. The Eclipse Platform provides good backward compatibility, but in the case of Eclipse 3.0, some of the public API has changed and many great new features have been added. We have developed several techniques for dealing with version differences like these in such a way that we can build multiple product binaries from a single source base while still supporting the latest Eclipse features.

Steve Taylor,
Catalyst Systems Corp

Lessons Learned in Plug-in Development

This session will walk through the development process of an Eclipse Plug-in from initial design to the migration to Eclipse version 2.0. Swing

developers who need clarification on the use of SWT will particularly benefit from this "Lessons Learned" session. Programming guidelines for using SWT will be reviewed, with examples on mapping between Swing and SWT. In addition, the Eclipse APIs used for Plug-in development will be identified and reviewed.

The Openmake Eclipse Plug-in will be used as the example for this session. This plug-in example demonstrates the use of the Eclipse Builder (JDT) as well as the plug-in User Interface. The use of plug-in objects such as Progress Bars, Buttons and Wizards will be reviewed. The following technical details will be covered:

- Designing your plug-in
- Understanding SWT if you're a Swing programmer
- Review of the critical Eclipse APIs for plug-in development
- Understanding the Eclipse Builders
- Interacting with the Plug-in User Interface

Steps for migrating from Eclipse V1 to V2.

Margaret-Anne Storey,
University of Victoria

Gild: An environment to facilitate collaboration in teaching and learning

The objectives of the Gild project are to improve the teaching and learning of introductory computer programming courses by providing a flexible learning and development environment that facilitates collaboration and community building. Using requirements gathered from students and instructors, we are designing a set of plug-ins for both students and instructors within Eclipse. In this talk, we present how a simplified student perspective provides access to integrated course materials facilitating more interactive and collaborative opportunities for programming. We will also discuss how the tool provides support for instructors through improved management of shared course materials as well as semi-automated support for marking assignments in Eclipse. Gild is currently deployed in an introductory course at the University of Victoria. We will share some of the results from our initial evaluation of Gild and Eclipse in the classroom.

Gary Brunell,
Parasoft

Stop Chasing Errors - Prevent Them by Performing Unit Testing in Eclipse

It is unimaginable how many times software development organizations find and fix the same errors over and over again. Our industry must mature the software development process. It is no longer reasonable to address software quality issues by simply "testing" applications, "chasing" errors one by one, and providing "bug fixes." The cost of this process and the resulting faulty software is too high.

However, the concept of Automated Error Prevention (AEP) can help break this cycle. AEP is a concept that advocates finding an error once and implementing practices to prevent the entire class of errors from ever happening again. The ultimate goal of AEP is to reduce the cost of software development and increase overall product quality and reliability. This concept, with automation, introduced into each phase of the software development lifecycle will prevent entire classes of errors from ever entering the source code.

The availability of tools that help developers perform unit testing within Eclipse makes it easier than ever before for users to enjoy the benefits of implementing AEP into this phase of the software lifecycle. Developers can simultaneously improve software reliability and reduce development time by performing thorough unit testing as soon as they complete each application component. When unit testing is performed immediately after every unit is compiled, developers not only detect errors that would evade other levels of testing, but also prevent errors from occurring in the first place.

This presentation describes the concept of Automated Error Prevention and the logistics of implementing unit testing in Eclipse. It begins by explaining how AEP can be implemented into the development lifecycle and then proceeds to show how developers can perform thorough, immediate unit testing within Eclipse.

Greg Stein
CollabNet, Inc

Subversion and Eclipse

Subversion is a brand new version control system, intended to replace CVS. It provides an easy, fast, and capable version control system for your every-day needs. No longer do you need to suffer with CVS' foibles -- switching from CVS to Subversion is quite easy to do. This talk will describe Subversion, compare and contrast it against CVS, and discuss and demonstrate Subclipse, a plug-in that provides Subversion integration into Eclipse.

Kip Harris
IBM

Making Eclipse technology accessible to people of all abilities

Successful access to information and use of information technology by people who have disabilities is known as "accessibility". Accessibility is an important feature, not only because it enables people with disabilities to work with IT, but also because of US federal regulations which place accessibility requirements in procurement policies. Similar regulatory requirements are expected from local, state, and national governments

in the US and other geographies. Furthermore, we expect accessibility requirements will cascade from the public sector into private industries, starting with the large companies. For example, IBM currently includes accessibility requirements in its purchasing specifications.

Eclipse has provided important Accessibility features in each of the 2.0, 2.1, and 3.0 releases. Developers using Eclipse technology need to be aware of these features in order to ensure that their applications are enabled for accessibility and meet the requirements just referenced. In this session, we will discuss what the developer needs to know about accessibility in the Eclipse framework. We will also demonstrate Eclipse working with an assistive technology.

Steve Forsyth
HP

Integrating Software Development Kits into the Eclipse Platform

Do you want to provide Eclipse integration for a different library ? Most Java code libraries, for example Jakarta's Log4J, are not integrated into a development environment. Many libraries come with only an ant script and/or a bat file to help the end-user explore code samples that utilize the library's API. After manually building library integration plug-ins several times, we decided to leverage Eclipse to provide a more productive environment for working with these libraries.

We constructed an integration tool which produces plug-ins that expose the library's sample code as Eclipse projects. The plug-ins integrate with the Eclipse help system to provide library documentation (e.g. its Javadoc) as well as plug-in usage instructions. The generated plug-ins can be utilized as is or may act as the starting point of a more full-featured plug-in.

Michael Valenta,
Eclipse Platform Team
Component Committer

Integrating Team Tools into Eclipse

The Eclipse platform provides API that allows repository vendors to integrate richly into Eclipse. This allows repository providers a means to surface their functionality to users in a way that makes sense for each repository type. This API is successful in that it provides repository vendors a level of integration that is not possible in other IDEs. However, the cost of entry is high in the sense that each repository vendor must implement all their own GUI components and workflows. Also, although the API provides mechanisms for surfacing repository operations in the workbench GUI, there is no API defined for programmatically invoking repository functionality. This talk will discuss integrating support for repositories into Eclipse including plans in Eclipse 3.0 for lowering the bar of entry for implementing repository providers. In addition, the possibility of promoting a generic Team API for use by 3rd party tools will also be discussed.

Jacob Lehrbaum,
MontaVista Software

Tux in Tool-land: Building an Eclipse-based development environment for Embedded Linux

This talk will focus on the use of the Eclipse based MontaVista DevRocket development environment in doing embedded systemsdevelopment. It will cover the unique aspects of embedded systems as well as specific DevRocket
methods for doing embedded systems development. Areas covered include cross development, kernel configuration, file system trimming, application tuning, cross debugging, and deployed image generation.

Panels:

Chair: Mike Taylor
President, CEO of
Instantiations, Inc.

Eclipse in the enterprise...its not just for development tools anymore!

This panel will explore the use of Eclipse as an application building platform. An increasing number of organizations are using Eclipse not as a software tools platform, but as a general application construction environment. Panel members from a wide range of industries will share their technical and political experiences as they brought Eclipse into their organizations and used it to build applications for their internal or external customers.

Chair: John Wiegand,
Eclipse Platform Lead

Open Q&A with the eclipse development team

The Q&A session will be designed by you, the EclipseCon attendees! We'll provide cards in advance throughout the conference for you to jot down your Q&A questions or topic suggestions. Before the session, we'll try to structure the topics in a way that we can create a 'custom' discussion for the audience with members of the eclipse development team. This will be your chance to ask questions or raise topics that we haven't covered in the sessions, that may have arisen from a session or simply something you heard in a discussion that would be interesting to all participants!

Chair: Michael Bechauf,
SAP AG

Tools Interoperability - Benefit or Burden ?

Even though Eclipse has grown to be a strong force in the developer tools market, today it is still a reality for add-in providers that they need to implement their plug-ins multiple times to fit other IDEs as well. Since most add-in providers target their products for multiple IDEs, this results

in significant additional development costs. Recently, standardization activities such as JSR 198 are attempting to define cross-IDE APIs which will reduce the effort or even completely eliminate porting plug-in code from one platform to another. However, there are many points to consider to determine whether or not standardization is truly beneficial to the Java tools community. For example:

- Does the current platform diversity foster innovation? Would this benefit be lost or reduced with standardization?
- While tools interoperability would be a benefit for add-in providers who target multiple IDEs, will the lack of a unified API will be more of a burden for the Java tools market?
- Can we really create a complete set of cross-IDE APIs? If so, will the resulting implementation simply be another universal tool integration platform?

This discussion is important for the industry overall and the expert panel will summarize the essential positions. The goal of the panel is to inform conference attendees of recent trends and to increase understanding of the various viewpoints. The panel will not attempt to draw a final conclusion, but will present facts and findings. The jury is still out there: It is up to the audience to make their decision.

Jay Parikh,
Akamai Technologies

Edge-Computing Toolkit for WebSphere Studio

A presentation of the Edge-Computing Toolkit for WebSphere Studio that simplifies development, testing, and deployment of applications for Akamai Technologies' on-demand edge-computing service, called Akamai EdgeComputing(SM) powered by WebSphere. The service, which combines IBM's WebSphere Application Server with Akamai's 15,000 servers, intelligent workload management, and dynamic provisioning, provides a globally distributed, responsive platform for executing Web applications. Applications designed for the Akamai EdgeComputing service adhere to the standard Web programming model of J2EE (Servlets and JavaServer Pages), but the distributed nature of Akamai's platform implies additional steps for testing and deployment. Those additional steps can be carried out manually, but a developer's experience could be improved through an integrated solution with WebSphere Studio. The Edge-Computing Toolkit for WebSphere Studio addresses these issues in several ways: 1) A new server type that represents a developer's provisioned EdgeComputing service. This new server allows a developer to publish, with wizard-driven interactions, Web applications to the EdgeComputing service. 2) Step-by-step documentation on how to configure WebSphere Studio for testing "split-tier" applications, which are those that are executed cooperatively between the Akamai EdgeComputing service and the customer's own data center.