Entwicklercamp 2008

IBM Lotus Design Tools

Stefan Neth, Senior IT Specialist, Channel Tech. Sales , IBM

Agenda

- **How we got here: Lotus Domino Designer Retrospective**
- More Tools in use today
- Where do we go from here: Lotus Domino Designer Roadmap
- What's next?
- Futures
- First Steps
- Q&A

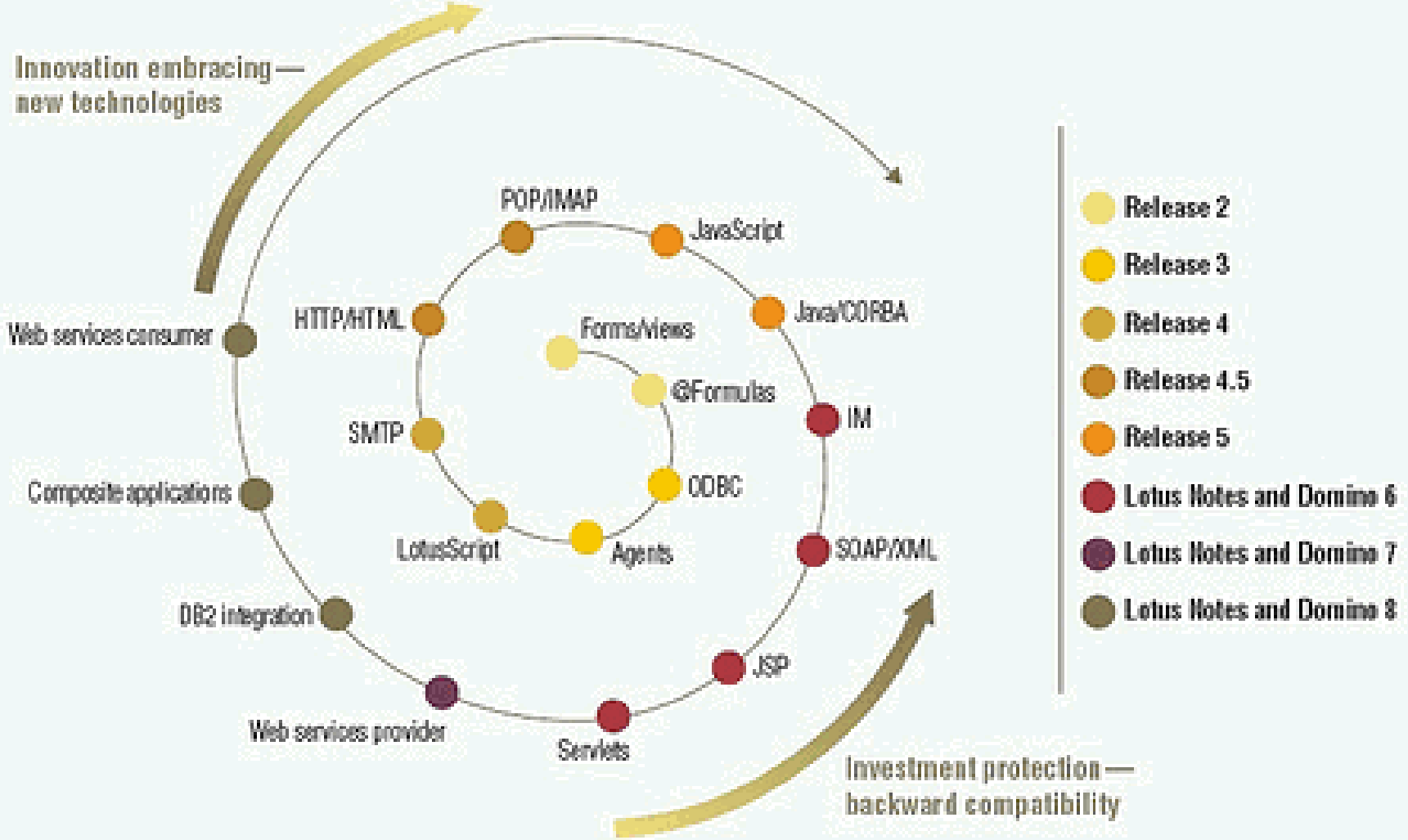
Lotus Domino Designer Benefits

Maximize Return on Investment and respond quickly to business demands	Rapid Application Development ▸ Template Driven Designs ▸ Reusability ▸ Built-in directory, security and data storage support
Reduce Total Cost of Ownership (TCO)	Ease of Deployment and Administration ▸ Changes synchronized through replication ▸ Backwards compatibility
Security-rich options to protect business data	Security ▸ From the server, to the database, to the form, to the field ▸ Built in PKI encryption
Leverage existing investments in developer skills and external systems	Open Access ▸ Program in a variety of languages ▸ Access enterprise data

What can you build with Lotus Domino Designer?

- Collaborative applications that support interaction and innovation
- Dynamic applications with content based on user name, user profile, permissions, or other variables
- Workflow applications that route information
- Tracking applications that monitor processes, projects, performance, and tasks
- Data integration applications that work with relational databases, transactional systems and other enterprise systems

Embracing Innovation while Protecting Investment



Lotus Domino Designer Roadmap

- 8.0
  - Released in August 2007
  - Focused on Composite application and Web service consumer features
- 8.0.1
  - Continued work on Composite applications
    - Page access control
  - Property broker usability improvements

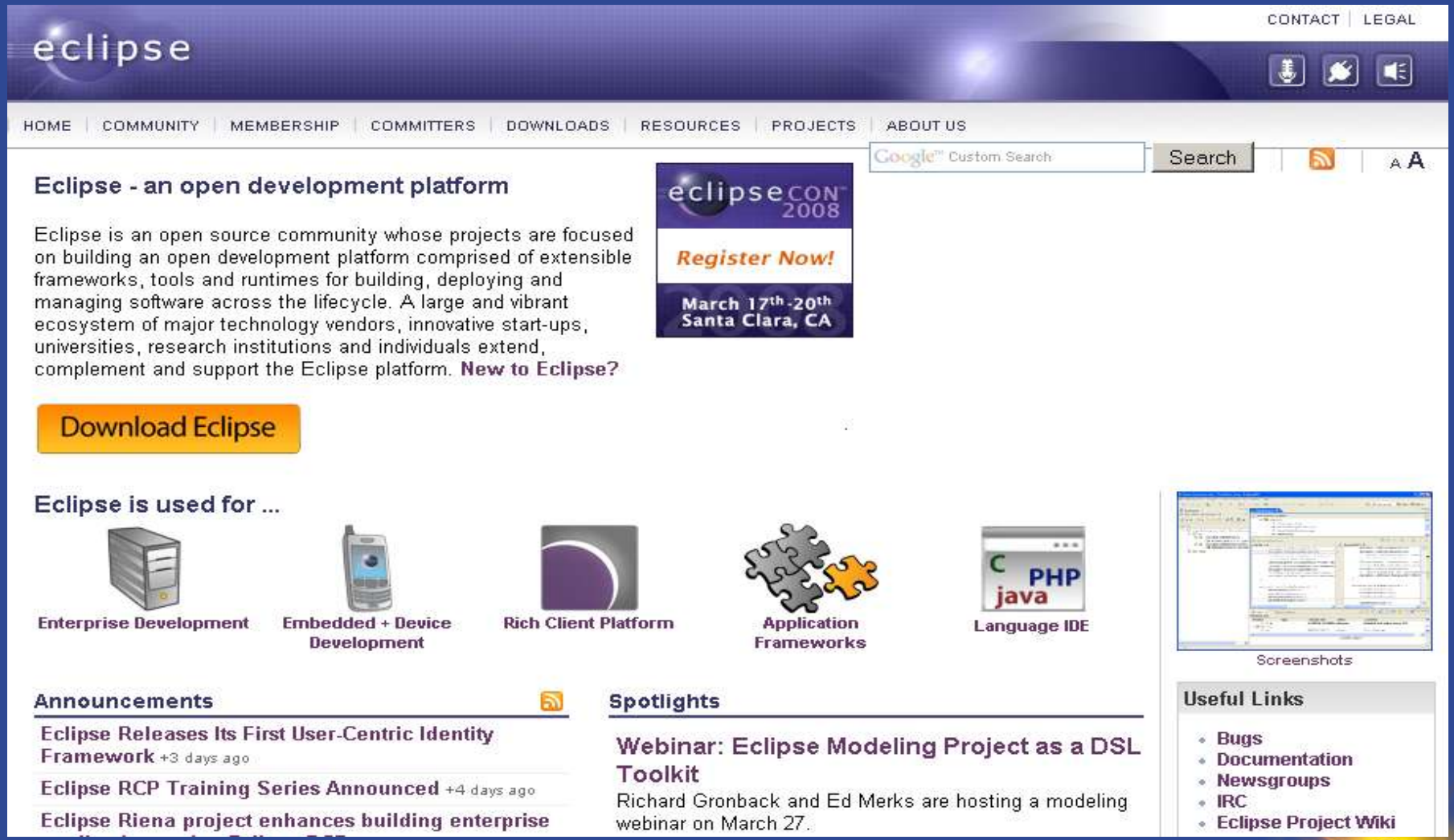
Agenda

- How we got here: Lotus Domino Designer Retrospective
- **More Tools today**
- Where do we go from here: Lotus Domino Designer Roadmap
- What's next?
- Futures
- First Step
- Q&A

More Tools in use today

- Eclipse
- IBM Lotus Expeditor Toolkit
- IBM WebSphere Portlet Factory Designer
- IBM Lotus Component Designer
- IBM Rational Application Developer

Eclipse


eclipse
CONTACT | LEGAL
HOME | COMMUNITY | MEMBERSHIP | COMMITTERS | DOWNLOADS | RESOURCES | PROJECTS | ABOUT US
Google™ Custom Search
Search
A A
Eclipse - an open development platform
Eclipse is an open source community whose projects are focused on building an open development platform comprised of extensible frameworks, tools and runtimes for building, deploying and managing software across the lifecycle. A large and vibrant ecosystem of major technology vendors, innovative start-ups, universities, research institutions and individuals extend, complement and support the Eclipse platform. New to Eclipse?
eclipseCON 2008
Register Now!
March 17th-20th
Santa Clara, CA
Download Eclipse
Eclipse is used for ...
Enterprise Development
Embedded + Device Development
Rich Client Platform
Application Frameworks
C PHP java
Language IDE
Screenshots
Announcements
Eclipse Releases Its First User-Centric Identity Framework +3 days ago
Eclipse RCP Training Series Announced +4 days ago
Eclipse Riena project enhances building enterprise
Spotlights
Webinar: Eclipse Modeling Project as a DSL Toolkit
Richard Gronback and Ed Merks are hosting a modeling webinar on March 27.
Useful Links
Bugs
Documentation
Newsgroups
IRC
Eclipse Project Wiki

Benefits of Eclipse

- World class script editing and debugging
- Built-in editors for standard elements such as style sheets, xml, html
- Leverage continuing improvements in Eclipse "for free"
- Partners can easily extend the tool through Eclipse extension points
- Framework maintenance reduced, allows us to focus on your feature requests (tables with pixels, etc.)
- Allows you to work with multiple IBM tools in a single container

IBM Lotus Expeditor Toolkit

IBM Lotus Expeditor Toolkit provides a complete, integrated set of tools that allows you to develop, debug, test, package, and deploy client applications to IBM Lotus Expeditor V6.1, IBM Lotus Sametime V7.5.1, and IBM Lotus Notes V8. You can use the toolkit to develop the following types of client applications:

* Eclipse Rich Client Platform (RCP) applications (desktop client only)
* Eclipse embedded Rich Client Platform applications
* Web applications
* Embedded transaction applications
* Portlet applications (desktop client only)
* Database applications
* Messaging applications
* Web services applications

WebSphere Portlet Factory Designer

- IBM WebSphere Portlet Factory includes an easy-to-use graphical tool for creating, viewing, and running portlets: the IBM WebSphere Portlet Factory Designer. The WebSphere Portlet Factory Designer provides simplified rapid application development of custom portlets for IBM WebSphere Portal that leverage existing enterprise applications, data and systems - including IBM Lotus Domino®, SAP®, PeopleSoft®, DB2® and Web Services, among others. It does this all without requiring Java" 2 Enterprise Edition (J2EE") development expertise, and without developers needing to implement and learn application and portal APIs by automating portlet development with reusable wizard-like components, called Builders. This automation speeds custom portlet development to reduce development time and cost while accelerating portal deployment for incremental ROI.

IBM Lotus Component Designer

Lotus software

Lotusphere 2008

IBM Rational Application Developer

Helps Java™ developers rapidly design, develop, assemble, test, profile and deploy high quality Java/J2EE™, Portal, Web, Web services and SOA applications.

* Increases productivity and shortens the development and test cycles as it extends the Eclipse 3.2 environment.
* Features a flexible installation process designed to provide developers with only the functions they need.
* Is integrated and optimized for IBM WebSphere® Application Server and IBM WebSphere Portal Server products and includes test environments for these products.
* Shortens the Java learning curve through visual design, with auto-synchronized code modeling.
* Uses robust capabilities to create SOA applications, including the ability to automatically create necessary SOA components such as WSDL and WSIL files.
* Applies visual portal development techniques to speed development.
* Update! version 7.0.0.3 provides features for assembling and deploying reliable, asynchronous, secure, and interoperable Java Web services for application components.
* Operating systems supported: Linux, Windows

Agenda

- How we got here: Lotus Domino Designer Retrospective
- More Tools today
- **Where do we go from here: Lotus Domino Designer Roadmap**
- What's next?
- Futures
- First Step
- Q&A

Lotus Domino Designer Roadmap

- 8.0
  - Released in August 2007
  - Focused on Composite application and Web service consumer features
- 8.0.1
  - Continued work on Composite applications
    - Page access control
  - Property broker usability improvements
- 'Next' will be on the Eclipse platform!!!!!
  - We can finally say when!!!

Benefits of Eclipse

- World class script editing and debugging
- Built-in editors for standard elements such as style sheets, xml, html
- Leverage continuing improvements in Eclipse "for free"
- Partners can easily extend the tool through Eclipse extension points
- Framework maintenance reduced, allows us to focus on your feature requests (tables with pixels, etc.)
- Allows you to work with multiple IBM tools in a single container
- **Lotus Domino Designer on Linux®, Mac "in the fullness of time"**

Do I have to learn Java?

NO!!*

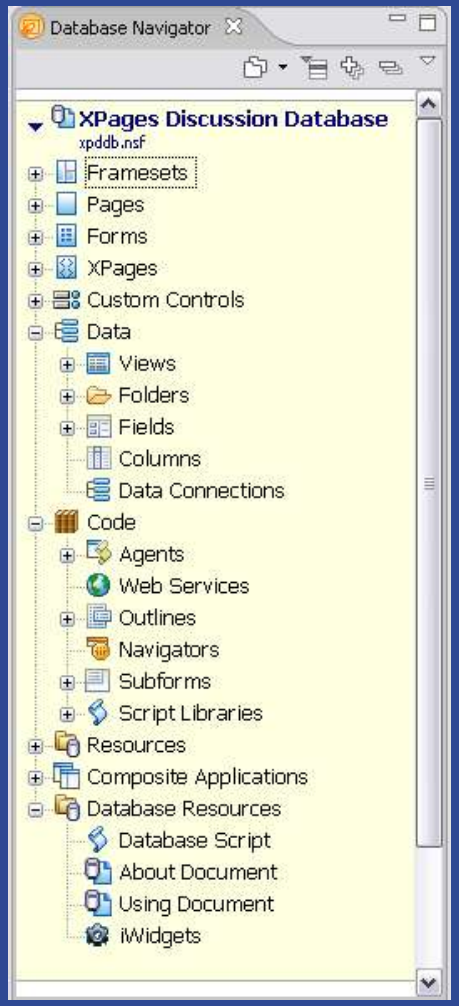
* But if you want to, it's a lot easier to use now

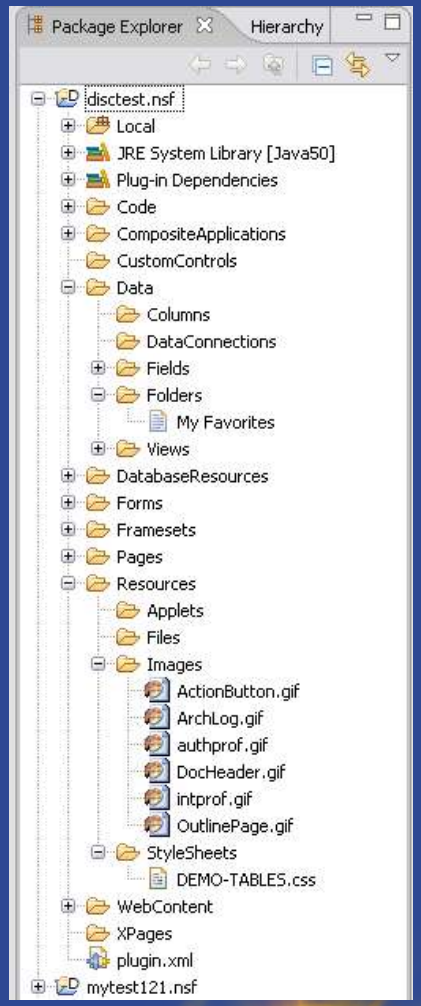
Agenda

- How we got here: Lotus Domino Designer Retrospective
- More Tools today
- Where do we go from here: Lotus Domino Designer Roadmap
- **What's next?**
- Futures
- First Step
- Q&A

Navigating Lotus Domino Designer 'Next'

- Familiar, but better!
- Bye-bye bookmarks
- Hello Working sets and navigation
- Designer perspective
- Java (or other) perspectives
  - NSF as file system





Editing in Lotus Domino Designer 'Next'

- A brand new LotusScript editor
- A nice JavaScript editor
- Java editor – same one as in Eclipse
- Editing files and style sheets

More Developer Options

- New options on existing design elements to support web server improvements
- New design elements to revolutionize application building
  - XPages
  - Custom controls
  - Server-side JavaScript libraries
  - Connections (DCRs finally get some friends!)

What's an XPage?

- New design element for building beautiful and accessible UIs
  - Full style sheet support
  - Source mode available to users who want full control
  - Standard JSF extended for Domino
  - Business logic through JavaScript and simple actions
  - Separation of data and presentation layers
  - External data access via Web service, REST, or ATOM
  - Built-in and custom Ajax behaviors
  - Custom controls
    - Extensible control model
  - Localizable pages
- Demo

Agenda

- How we got here: Lotus Domino Designer Retrospective
- More Tools today
- Where do we go from here: Lotus Domino Designer Roadmap
- What's next?
- **Futures**
- First Step
- Q&A

Futures

- Will we get it all done in Lotus Domino Designer 'Next'?
  - Wish so, but not…
- Eclipse-based LotusScript debugger
- Eclipse-based Formula editor and debugger
- More extension points for BP/ISV solutions
- Property panels for more design elements
- Palettes on forms and views
- Improved UI

Lotus Domino Designer

- Long-standing commitment to your world class business solutions
- Innovative design elements for attractive and compelling user-facing applications
- Usability enhancements that improve developer experience
- New integration options and opportunities for business partners

Agenda

- How we got here: Lotus Domino Designer Retrospective
- More Tools today
- Where do we go from here: Lotus Domino Designer Roadmap
- What's next?
- Futures
- **First Step**
- Q&A

First Step

- Recap introduction to Lotus Expeditor - the foundation of Lotus Notes 8
- Why should I care about Lotus Expeditor?
- How do I build Lotus Expeditor applications in Lotus Notes 8?
- Getting Started
- Resources

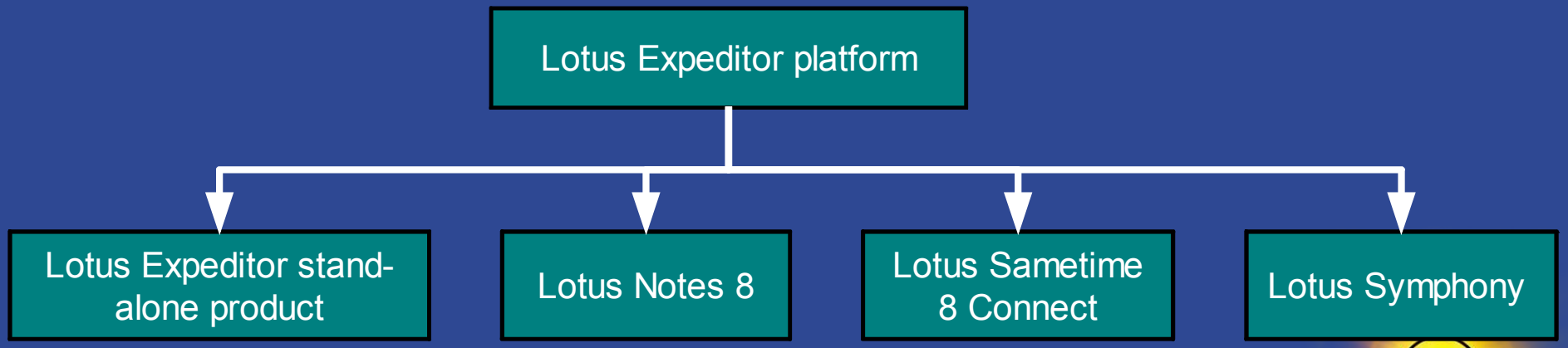
Brief introduction to Lotus Expeditor - the foundation of Lotus Notes 8

Lotus software

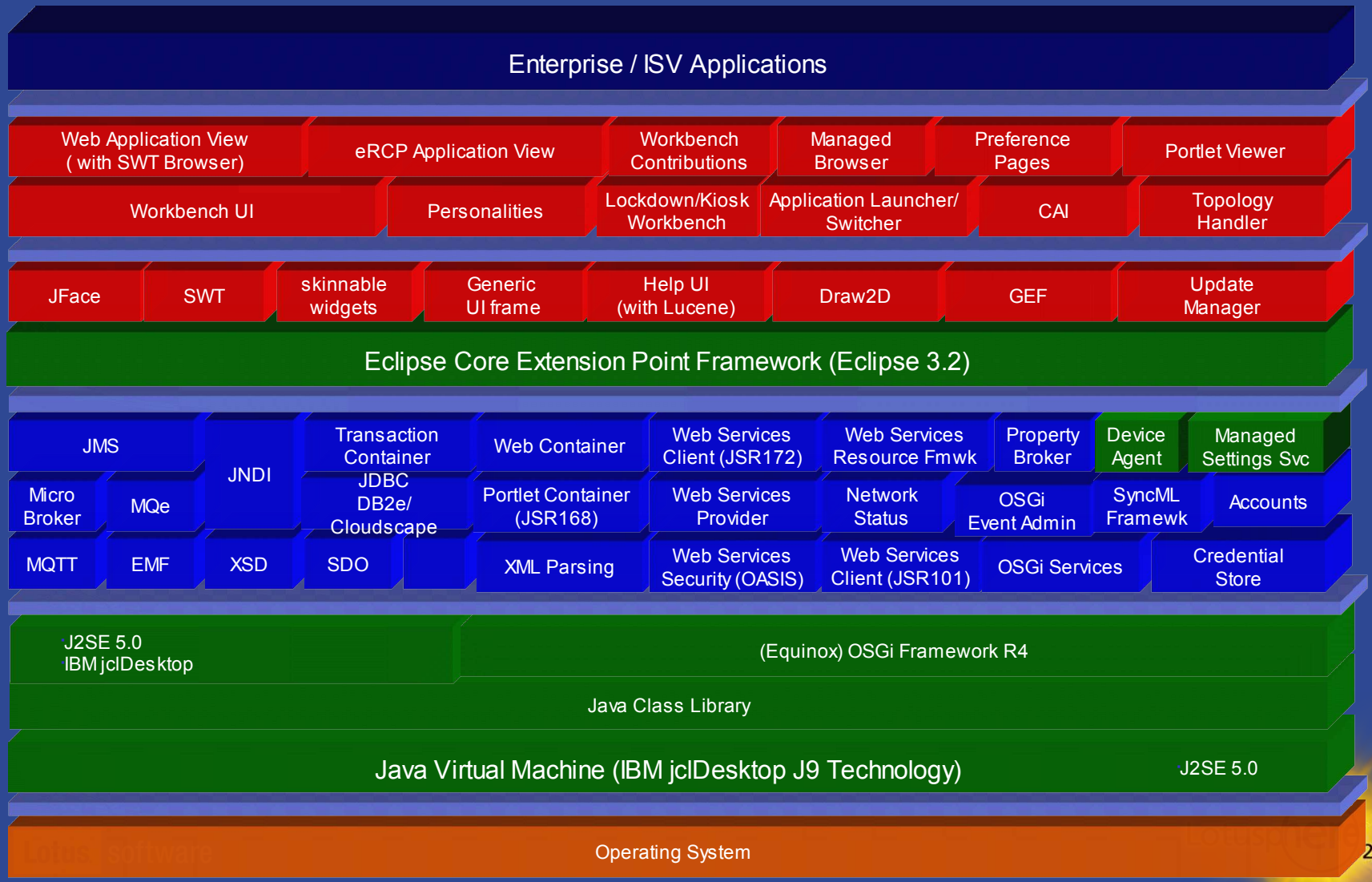
Lotusphere 2008

Remember Lotus Expeditor?

- Common foundation for next-generation Lotus client-side products
- Core runtime and comprehensive set of services for building business applications
- Lotus Notes 8, IBM Lotus Sametime 8, IBM Lotus Symphony use a subset of the full Lotus Expeditor platform
  - Reduced footprint
  - May extend platforms with additional Expeditor features

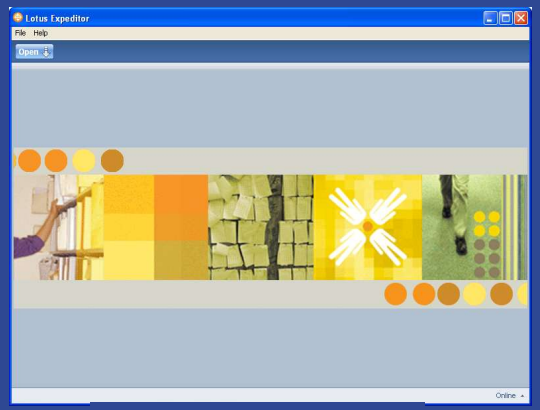


Lotus Expeditor Platform



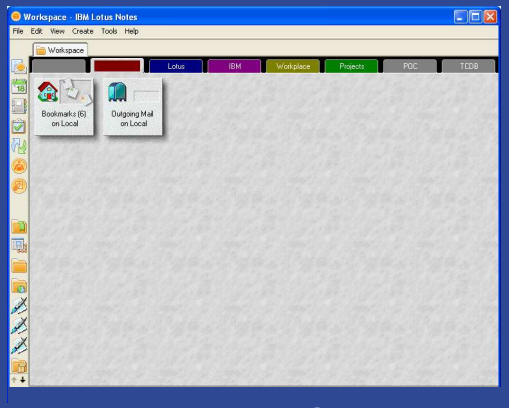
What is Lotus Notes 8?

- Innovative product that ensures compatibility while providing a new desktop application platform – Lotus Expeditor
- Lotus Notes 8 C++ client (nlnotes.exe) is wrapped invoked from a plugin inside Lotus Expeditor
- Lotus Notes 8 C++ client UI is wrappered (re-parented) as Expeditor components
- End result: all capabilities of Lotus Notes surfaced on a new, extendable platform



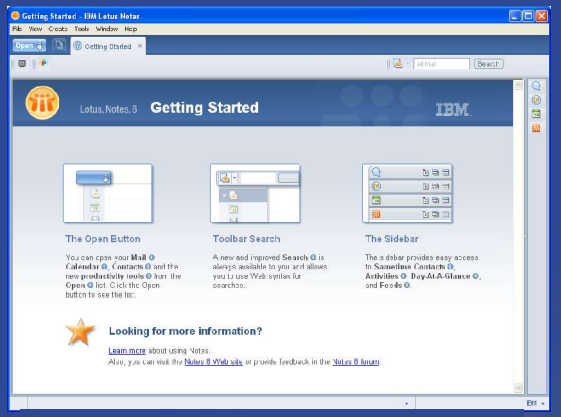
Lotus Expeditor

+



Lotus Notes C++ client (nlnotes.exe)

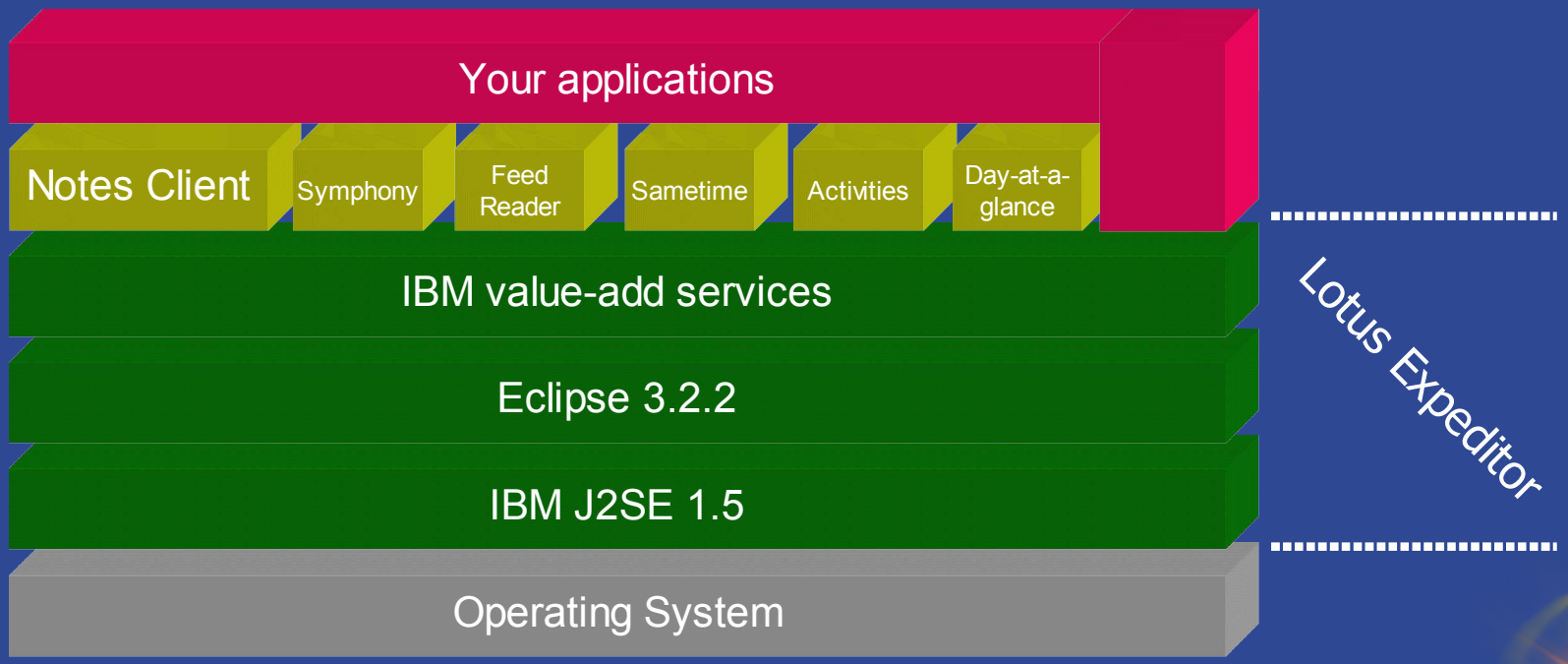
=



Lotus Notes 8 Standard (notes.exe)

Lotus Notes 8 Software Stack

- Build applications using Lotus Notes or Lotus Expeditor platforms

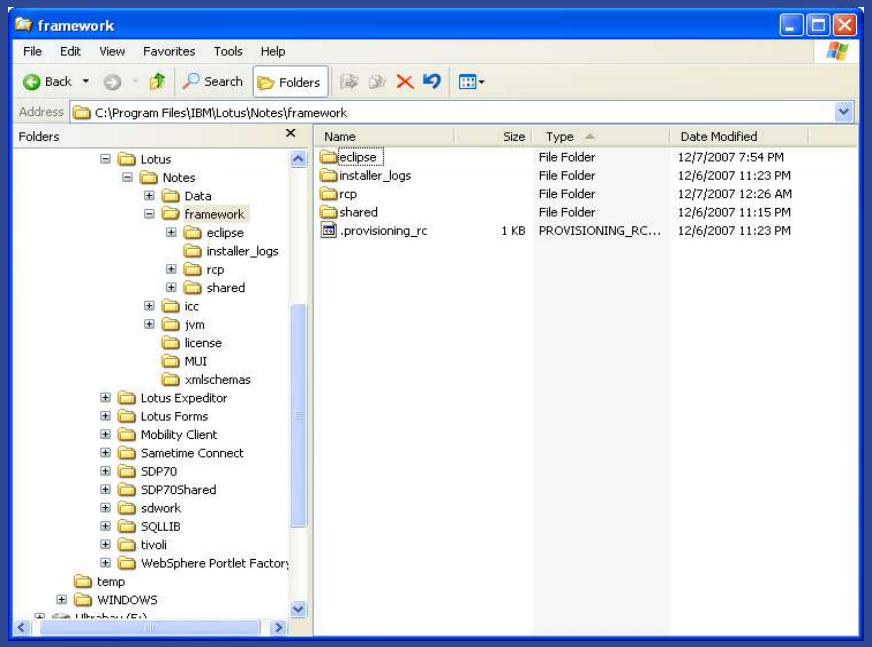


Lotus software

Lotusphere 2008

Lotus Expeditor Directories in Lotus Notes 8

- Lotus Notes 8 directory structure is same as previous releases
- Lotus Expeditor platform resides in "<notes_program_dir>/framework" directory
- Lotus Expeditor data directory resides in "<notes_data_dir>/workspace" directory



Lotus software

Lotusphere 2008

Lotus Notes 8 compared to full Lotus Expeditor platform

- Lotus Notes 8 uses a subset of the full Lotus Expeditor platform
  - Features from the full platform can be added to Lotus Notes 8

Lotus Expeditor features in Lotus Notes 8

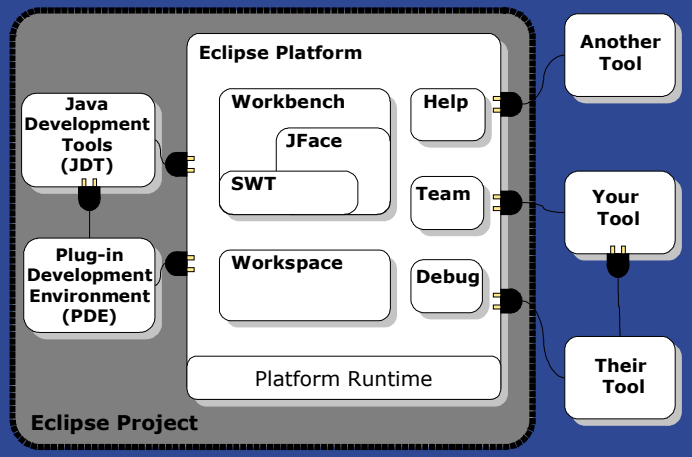
Apache Derby
Spell checker
J2EE Web container
JSR168 portlet container
WSRP portlet consumer
Network Awareness
Offline manager
Web services consumer
Accounts
Enhanced browser
Managed settings
Rich text editor
Synchronization manager
Enhanced SWT widgets
Composite application infrastructure
Property Broker
Personality framework
Sidebar
Portal-managed applications

Additional features in full the Lotus Expeditor platform

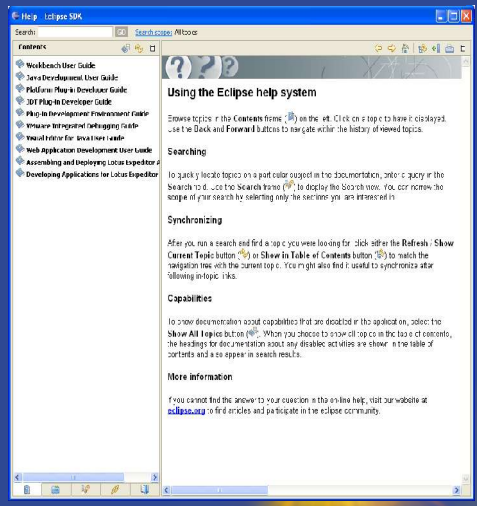
DB2 Everyplace (DB2e)
DB2 Everyplace iSync client
MQ Everyplace (MQe)
Microbroker (pub/sub messaging using MQTT)
Web Services Resource Framework (WSRF)
Embedded transation container (EJB subset)
Web services security
SyncML4J framework
Web services provider

Key Eclipse Concepts - 1

- Lotus Expeditor is an extension of Eclipse
  - Since Lotus products run on Lotus Expeditor, Eclipse concepts are important to understand
- OSGi platform
  - Industry standard for multiple Java apps in 1 JRE
  - Pluggable architecture - Eclipse (and thus Expeditor) is a collection of plugins on a thin OSGi runtime
  - Defines plugin packaging, metadata, dependency management, public/private code, etc
  - Provides plugin protection and isolation
- Extension framework
  - Enables 1 plugin to contribute data or functionality to another plugin with runtime binding
  - Examples include help content, views, menu actions, toolbar icons, side bar panels, etc.



Eclipse Help system is good extension point example



Key Eclipse Concepts - 2

- Comprehensive Platform
  - Standard Widget Toolkit (SWT) – use native operating system widgets for familiar look & feel
  - Window management
  - Threading
  - Progress framework
  - JFace (wizards, model-driven lists, presentation utilities, etc)
  - Help system
  - Preferences
  - …
- Community and Ecosystem
  - Get help
  - Get news
  - Most 3rd party Eclipse plugins will run on Lotus Expeditor

Lotus. software

Why should I care about Lotus Expeditor?

3 Reasons to Care About Lotus Expeditor

- You can build a new breed of rich desktop applications
- You can build composite applications
- You can extend the client itself

Lotus Expeditor programming model

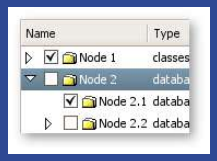
- Data store
  - Apache Derby (local relational database)
  - Lotus Notes database via Lotus Notes Java APIs
  - Web services
- Application logic
  - Java 2 Standard Edition (J2SE) 1.5
- Presentation
  - Eclipse views rendered with SWT, Swing or AWT
  - Browser-based user interfaces:
    - Java 2 Enterprise Edition (J2EE) using the local web container
    - Portlets using local JSR 168 portlet container

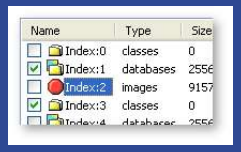
Lotus Notes 6/7 vs Lotus Expeditor

	Data Store	Logic	User Interface
Typical Lotus Notes 6/7 development	▪ Non-relational document store (NSF)	▪ Lotusscript, Java or @function ▪ Java is for backend logic only	▪ Build user interface visually using high-level constructs – Fields, Forms, Pages, Views, Folders, Navigators, Outlines
Typical Lotus Expeditor development	▪ Access Lotus Notes document store via Notes Java APIs ▪ Use local Apache database ▪ Web services ▪ Other remote API	▪ Java 2 Standard Edition (J2SE) 1.5	▪ Construct panels from a library of widgets (tables, trees, combo-boxes, buttons, etc) ▪ Visual editor available ▪ Express user interface as 1 or more Eclipse views ▪ Layout pages with 1 or more Eclipse views ▪ 3rd party tools available ▪ Build local J2EE web application ▪ Build local JSR168 portlet

Benefits of new programming model - 1

- New user interface possibilities - not constrained to form/view/page/navigator/etc paradigms
  - Rich set of native widgets
  - Drag-and-drop
  - Add controls to status bar
  - Add system tray icons and actions
  - Web-based widgets, e.g. Google Gadgets
  - …





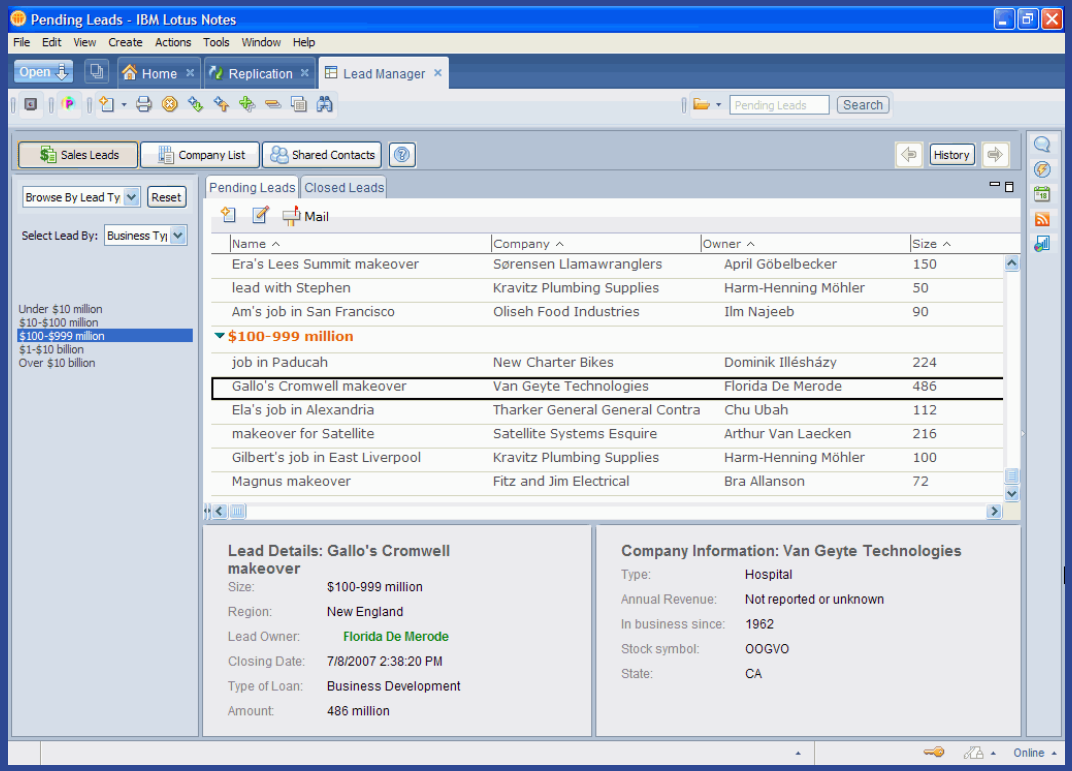
- Leverage non-Notes skills
  - For example, J2EE developer can build a user interface for a desktop application by building a local web application targeted for the Lotus Expeditor web container
- Mix Java and UI (unable to do this in Notes!)
- Reuse existing assets
  - J2EE web applications
  - JSR 168 portlets
  - Java/Swing apps

Benefits of new programming model - 2

- Reuse code across applications
  - In Lotus Notes, application code cannot be used outside the NSF it resides in
  - In Lotus Expeditor, application code resides in plugins, and plugins can be used by other plugins
  - Build reusable services using OSGi Services Framework

Composite Applications

- "Mashups" across application types
  - Lotus Notes
  - Swing/AWT
  - Web (local and remote)
  - RCP/SWT
  - Portlet (local and remote)
  - Host access
  - Reparent native applications



Lead Manager – public sample from Lotus Sandbox

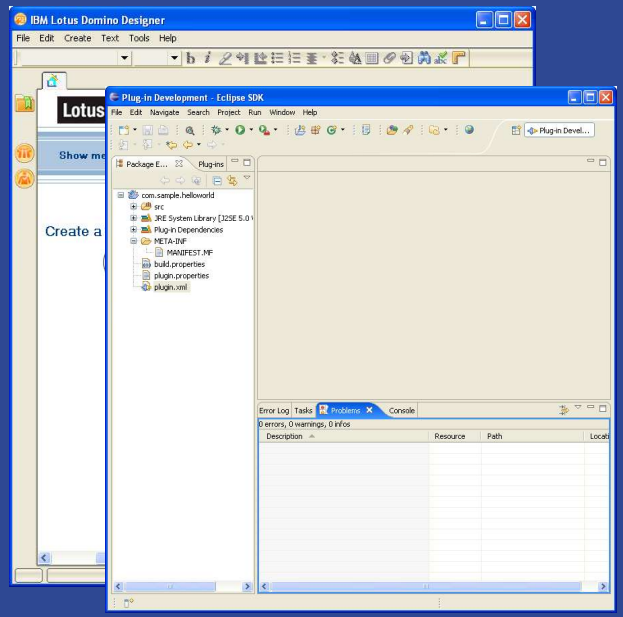
Lotus software

Lotusphere 2008

Why Composite Applications?

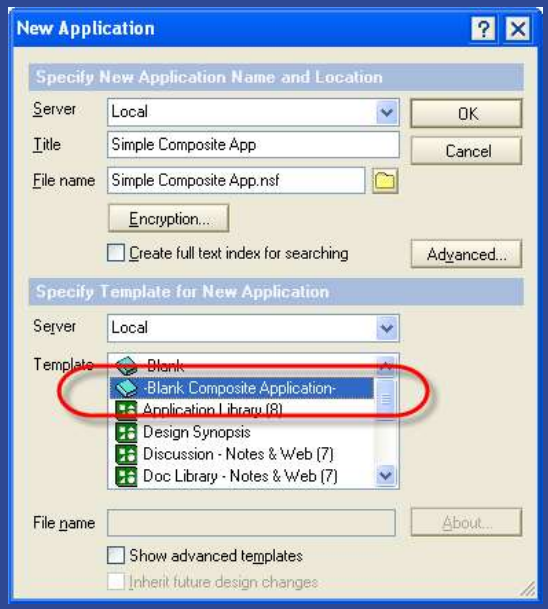
- Integrate disparate applications/technologies into a single user experience
  - Benefits
    - Improved productivity (less context switching)
    - Potential synergy among the applications
    - Stepping-stone to solution redesign
- Build a new application from a palette of reusable, well-defined components
  - Benefits:
    - Knowledge workers, not necessarily developers, can assemble applications from components
    - An individual component can be updated without updating other components. This is in contrast to deploying new versions of an entire monolithic application to deliver an update
    - New components can be added incrementally
    - Application layout and contents can be adjusted without changing code

How are Composite Applications Built?

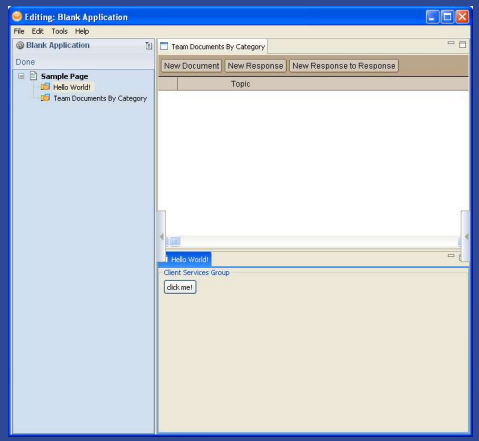


1. Build NSF and Lotus Expeditor components

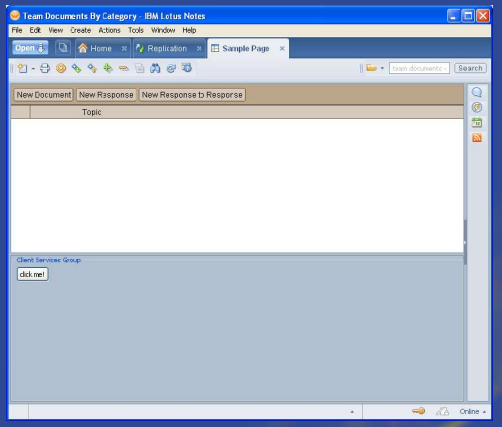
2. Create a new database using the "Blank Composite Application" template



3. Assemble and wire components using Composite Application Editor



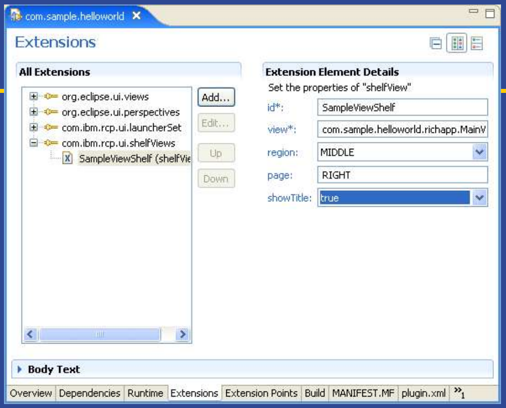
4. Done



Lotus software

Extend Lotus Notes itself

- Extend Lotus Notes 8 at the Lotus Expeditor layer
  - Lotus Notes 8 is a Lotus Expeditor application
  - Lotus Expeditor, with its Eclipse foundation, is very extendable
- Examples
  - Add a panel to the side bar
  - Add a menu item to a right-click context menu
  - Add a global action to the toolbar or top menu bar
  - Create a custom theme
  - Create plugins that can be reused across multiple applications
    - In contrast to NSF model where code cannot be reused across NSFs



Add to Notes side bar using an extension point

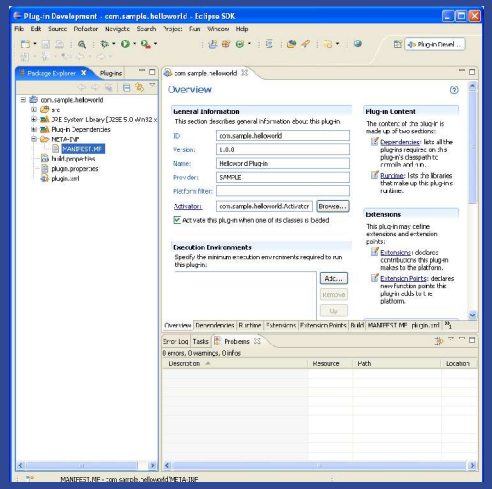
How do I build Lotus Expeditor applications in Lotus Notes 8?

Lotus Expeditor development overview

- Lotus Notes-only applications built same as before using IBM Lotus Domino Designer
- If you want to include Lotus Expeditor-based components in your application, you build 1 or more plugins
  - Lotus Expeditor is built on Eclipse
  - Eclipse is a set of plugins running on a thin OSGi runtime
  - Therefore, a Lotus Expeditor application is a set of 1 or more plugins
- What does a plugin do?
  - Whatever you want
  - 1 plugin can contain your entire application, or application can be divided among multiple reusable plugins
  - Some typical plugin roles:
    - Monolithic application
    - Model and business logic-only
    - Presentation-only
    - Utility
    - Provide a framework
    - Wrapper 3rd party code
    - Wrapper legacy code
    - Libraries
    - Provide content
    - Provide a concrete implementation for an extension point

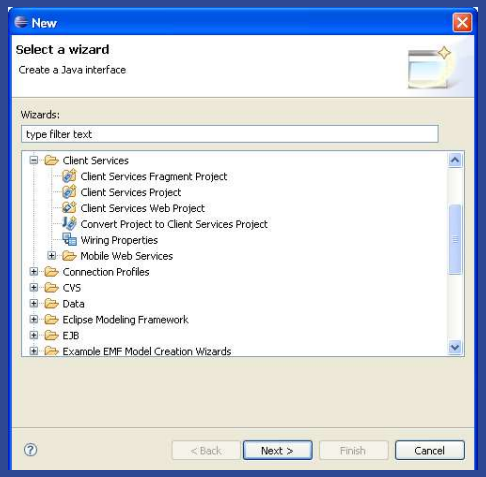
Development Environment

- Plugins are built, tested and packaged using:
  - Eclipse Plugin Development Environment
  - Lotus Expeditor Toolkit
  - Lotus Notes 8 client (provides runtime environment)



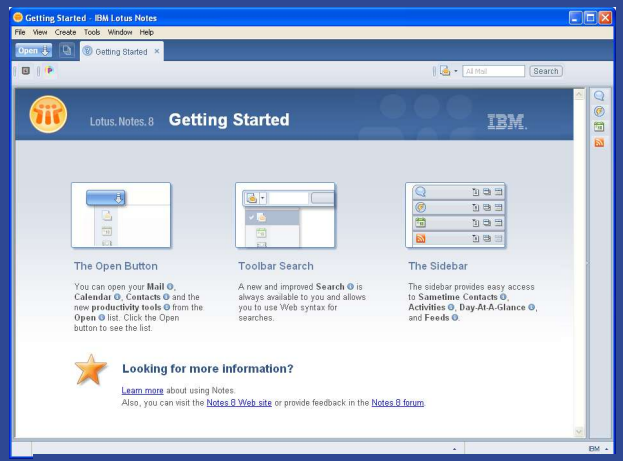
Eclipse-based Plugin Development Environment (PDE)

Build, test, deploy plugins



Lotus Expeditor Toolkit

Extends PDE with Expeditor-specific tooling, e.g. new project wizards



Lotus Notes 8

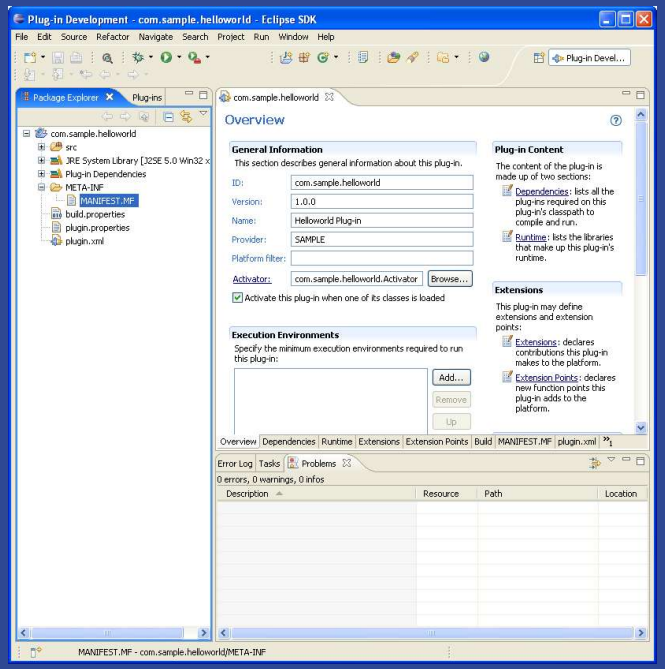
Provides runtime environment for testing your plugins

Lotus software

Lotusphere 2008

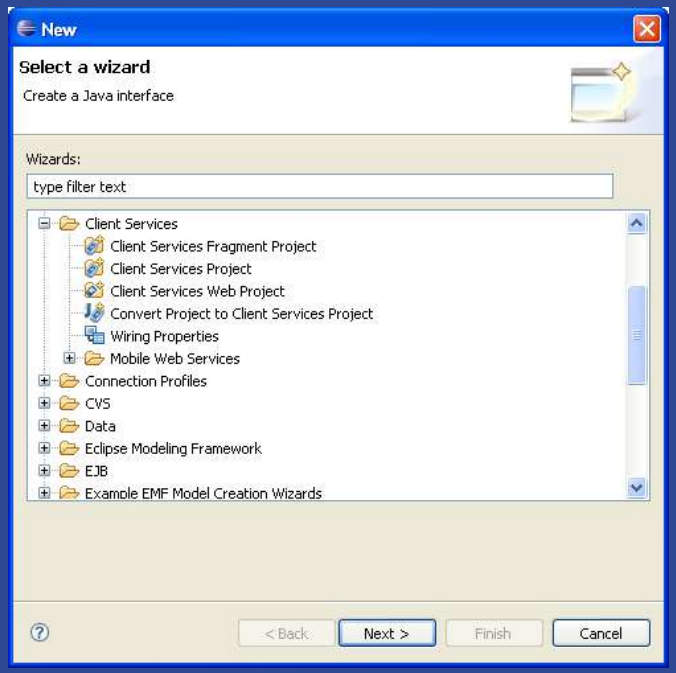
Eclipse Plugin Development Environment

- Eclipse plugins are built using the Eclipse Plugin Development Environment (PDE)
- To build plugins for the Lotus Expeditor platform, use either
  - Eclipse 3.2.2 SDK (free download from eclipse.org)
  - IBM Rational Software Development Platform 7 (commercial product)
- Consider Rational when:
  - Building complex web applications for the J2EE web container
    - Rational offers better web tooling and improved JSF support
  - Building portlets for Expeditor's portlet container
    - Rational provides tools for building portlets
  - Building secure web services (full Expeditor platform only)
  - Building projects for the embedded transaction container (full Expeditor platform only)



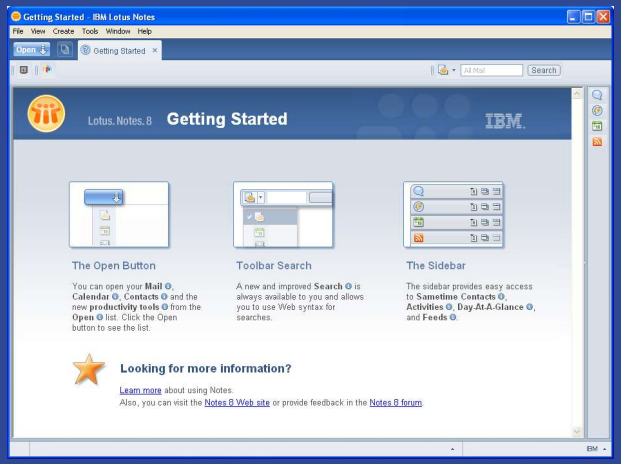
Lotus Expeditor Toolkit

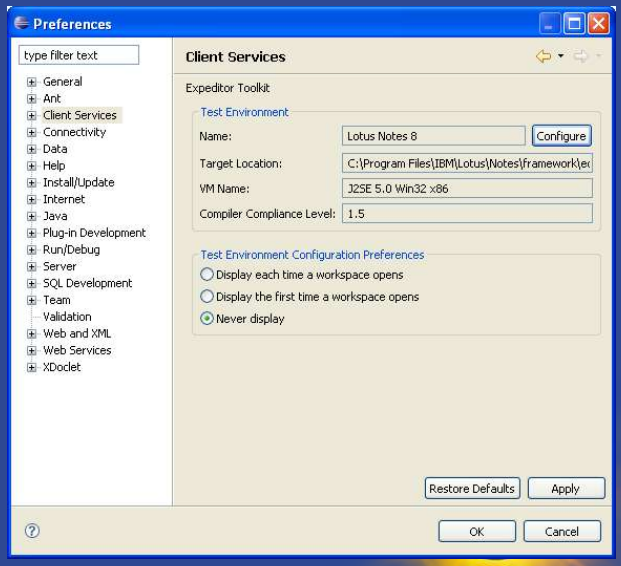
- Lotus Expeditor Toolkit extends the Eclipse Plugin Development Environment with additional tools:
  - New Client Services project types
    - RCP, web, portlet, embedded transaction container
  - Bundle developer kit
    - Additional tools for editing and compiling bundle manifests
  - Web service wizards
  - Embedded transaction container tools
    - EJB tooling; requires RAD
  - Launchers
    - Launchers are used to launch the Expeditor platform
  - Web tools integration
    - Expeditor web container becomes another local test environment for testing web bundles



Lotus Notes 8 Client

- Plugins execute in a "target platform"
  - Target platform is the collection of plugins that comprise the platform the plugin will execute in
  - If building a plugin for Lotus Notes 8, Lotus Notes 8 is the target platform
  - If building a plugin for Lotus Sametime 8, Lotus Sametime 8 is the target platform, and so on
- Lotus Notes 8 client is required to build and test plugins targeted for Lotus Notes 8
  - Configure toolkit to where Lotus Notes 8 is installed
- When ready to test your plugin, PDE will launch an instance of Lotus Notes 8 which will include your plugin





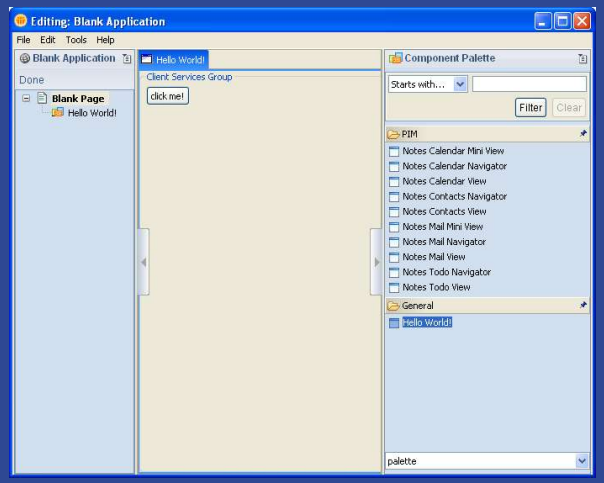
Lotus. software

Build

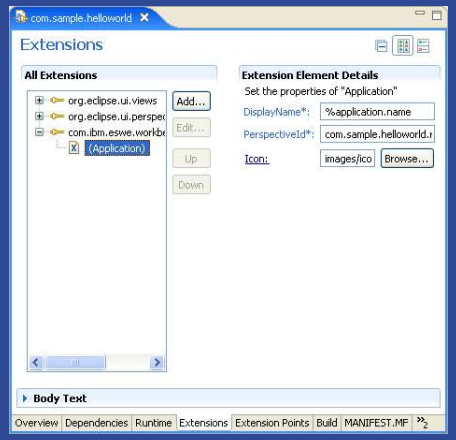
- Create a new Client Services project
  - Out-of-the-box templates available for basic applications
  - Teams can add their own templates (Eclipse IDE is very extensible)
- Implement your logic and UI
  - Java 1.5
  - Optional: use Eclipse's Visual Editor to layout UI
  - Optional: use 3rd party plugin development tools
  - Declare dependencies on other plugins to use their public packages
    - Example: declare a dependency on the "com.ibm.rcp.accounts" plugin to use the Accounts API

Test

- In plugin development, you launch an instance of your target platform (Lotus Notes 8, Lotus Sametime 8 or Lotus Expeditor stand-alone) to run test plugin
  - For Lotus Notes 8 development, launch an instance of the Lotus Notes 8 client from the IDE
- To run your plugin inside Lotus Notes 8,



Option 1 – Add to a composite app (easiest)



Option 2 – Add directly to the "Open" menu via extension point

Lotus. software

Deploy

- How does a plugin get from the IDE to the end user client?
  - In Lotus Notes 8, Expeditor plugins can only be installed from an update site
  - Update site is a catalog of deployable features and plugins available for download
  - Eclipse PDE tooling automates creation of deployable features, plugins and update sites
- Update sites
  - Update sites are simply a directory structure
  - Update sites can be hosted on file system, HTTP server or by a NSF
- NSF-based update sites
  - Install plugins from a Notes database
  - Use the update site template
  - Populate by importing deployable artifacts from file system
  - Benefits
    - Leverage benefits of Notes infrastructure to distribute code
      - Replication, fail-over, security, logging
    - Can package plugins in same database as application

Best Practices – Multi-tier Architectures

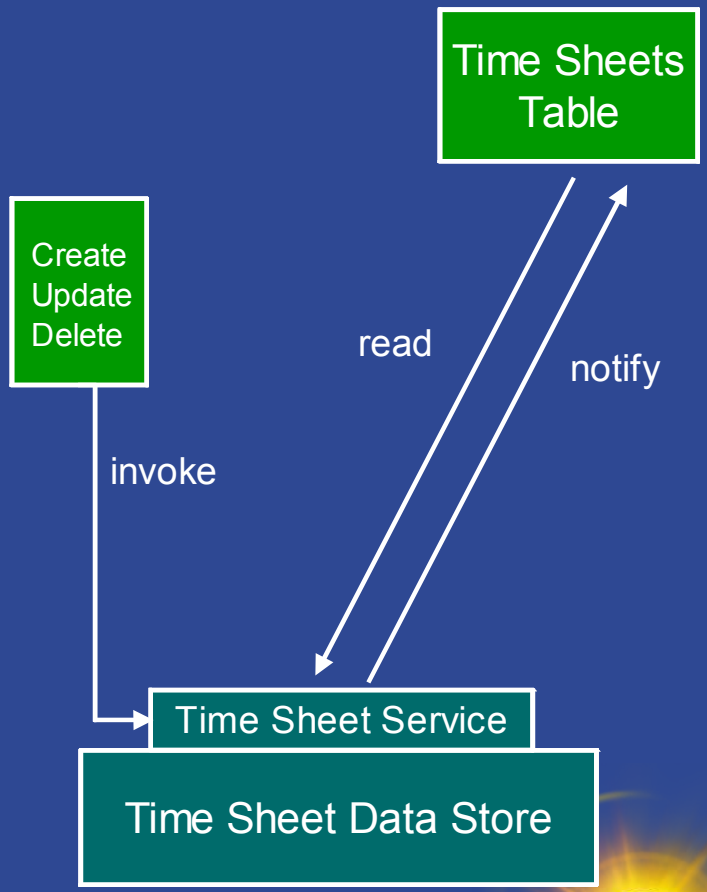
Role of tier		Example
JFace, SWT, Forms, Actions, model event listeners	Application Client	· Delete Ticket menu action · Tickets view
Business logic, calls DAO for persistence, event notification Use Service Locator to obtain the service	Service	· Check preconditions on backend · Call DAO to perform ticket deletion
DAO provide access to persistent stores	Data Access Object (DAO)	· Create JDBC connection · Execute SQL to delete ticket
Persistent data store	Database	· Local Derby Database

Best Practices – Multi-tier Architectures

- Unlike Lotus Notes 7 development, Lotus Expeditor development basically starts with a blank slate
  - Lotus Notes manages access to its data store (NSF) and provides a nice document-centric API on top
  - Lotus Notes also provides UI widgets (views, forms) which are bound to the data store
  - In Lotus Expeditor, this infrastructure does not already exist
    - But the trade-off is flexibility and control
- When designing an application that interacts with a data source, design in tiers.
- Benefits
  - Proven pattern
  - Encapsulation - implementation changes in 1 tier do not affect another tier as long as the interfaces remain unchanged
  - Easy for others to understand your code

Best Practices – Observer Pattern

- In Lotus Notes 7 and web applications, when something (Notes action, JavaScript button) updates the model, that something normally needs to update any affected UI elements
- Eclipse offers a better model through the observer pattern
- In the observer pattern,
  - Anything interested in knowing when the model changes registers with the model as an observer
  - When the model changes, all registered observers are notified
  - Actions then only need to update the model. UI (and potentially other components) are notified by the model and react accordingly, e.g. refresh a UI table.
- Java provides utility classes to assist in implementing the observer pattern



Getting Started

Lotus software

Lotusphere 2008

Getting Started

- Get pre-requisite skills
  - Java
  - Eclipse plugin development
- Setup your development environment
  - IDE
    - Eclipse 3.2.2 SDK (free from eclipse.org)
    - IBM Rational Software Development Platform 7
  - Lotus Expeditor Toolkit 6.1.2 (free from developerWorks)
  - Lotus Notes 8 client
- Play
  - Install and run the samples
  - Follow the tutorials
  - Create a plugin using a template then make incremental modifications, e.g. add a button or a rich text editor
- Leverage education resources (see resources)

Getting Started - Java

- What do I need to know about Java?
  - Understand object-oriented concepts in Java
  - Understand exception handling
  - Understand Java interfaces
  - Understand anonymous classes
  - Understand observer pattern
  - Understand collections
  - Understand API specifications (Javadoc)
  - Understand how to log using java.util.logging concepts instead of System.out.println()

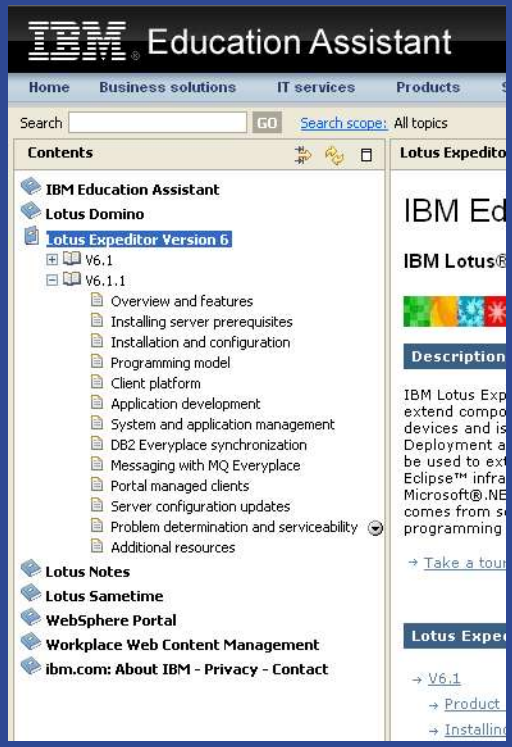
Getting Started – Eclipse plugin development

- What do I need to know about Eclipse plugin development?
  - Create and test a bundle (plugin) that uses:
    - View
    - Perspective
    - Preference Page
    - Action button
    - SWT widgets
    - SWT layout manager
    - JFace list control
  - Understand plugin deployment
    - Features
    - Update sites

Resources

Resources - Education

- Lotus Expeditor Client and Toolkit workshop
  - https://www-304.ibm.com/jct09002c/isv/spc/events/description.jsp?event=256A8C62AD5DDE21862572590065B201
- Lotus Notes 8 and Lotus Domino 8 workshop
  - https://www-304.ibm.com/jct09002c/isv/spc/events/description.jsp?event=3838C2D6038A598486257322005530F4
- IBM Education Assistant
  - Excellent resource
  - http://www-306.ibm.com/software/info/education/assistant/
- Virtual Innovation Center
  - For business partners and IBMers
  - Access from IBM Partner World
- Custom education programs available - contact your sales representative



Lotus. software

Legal

The information contained in this publication is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this publication or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

References in this presentation to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in this presentation may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth or other results.

IBM, the IBM logo, Lotus, Lotus Notes, Notes, Domino, Quickr, Sametime, WebSphere, UC2, PartnerWorld and Lotusphere are trademarks of International Business Machines Corporation in the United States, other countries, or both. Unyte is a trademark of WebDialogs, Inc., in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Lotus software

Lotusphere 2008