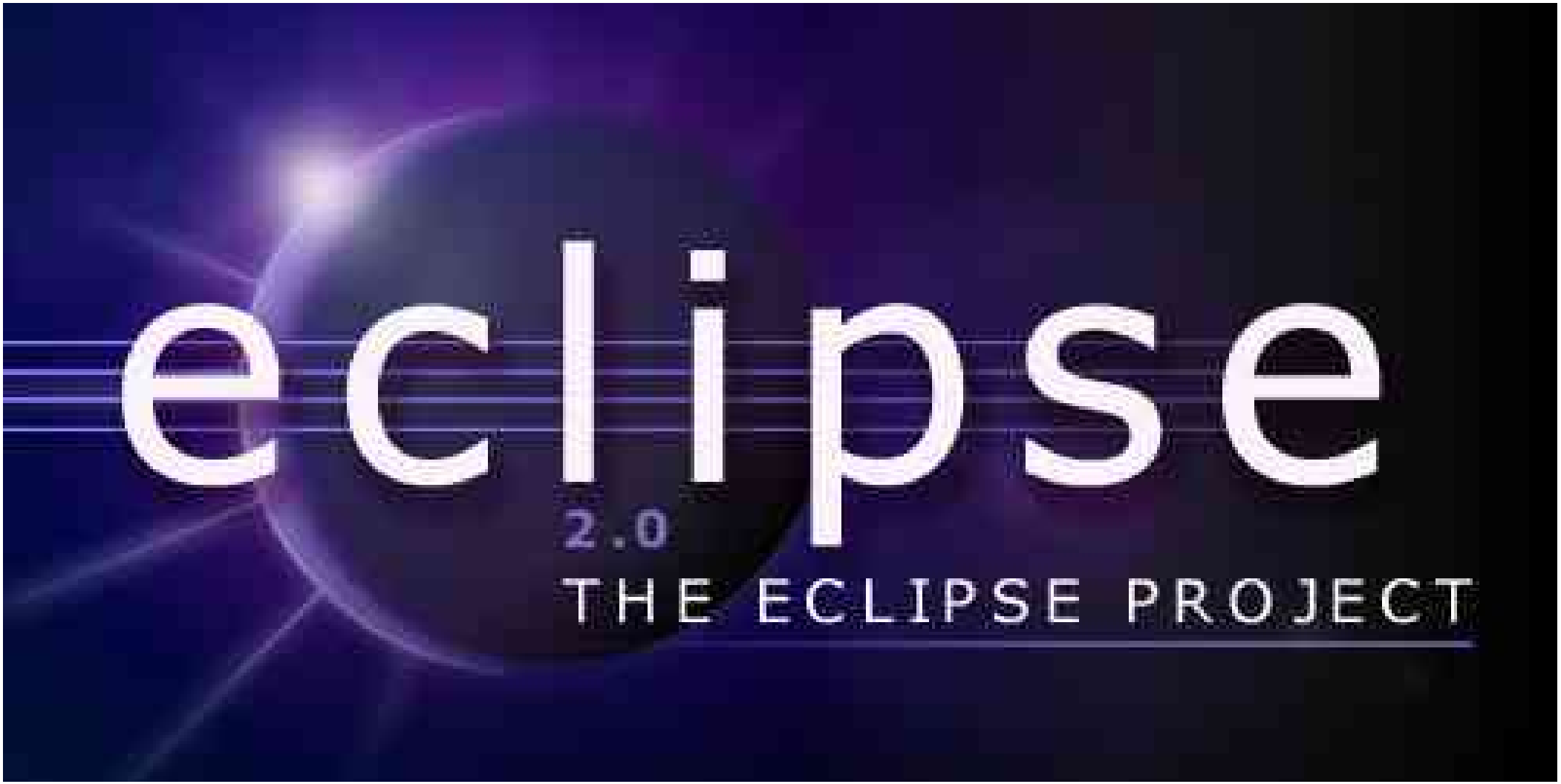

eclipse
2.0
THE ECLIPSE PROJECT

# Gliederung 1/2


- 0.1 Einleitung
- 0.2 Eclipse – Was ist das?
- 0.3 Entstehung
- 0.4 Entstehungszeitraum
- 0.5 Zielsetzung der Entwickler
- 0.6 Systemvoraussetzungen
- 1.0 Eclipse – Drei Projekte
- 1.1 Das Eclipse Projekt
  - 1.1.1 Plattform Überblick
  - 1.1.2 Die Eclipse Plattform
  - 1.1.2.1 Workspace / Resource API
  - 1.1.2.2 Workbench
    - 1.1.2.2.1 Editoren, Sichten, Perspektiven
    - 1.1.2.2.2 SWT, JFace
  - 1.1.2.3 Weitere Komponenten

# Gliederung 2/2


- 1.1.3 JDT – Java Development Tools
  - 1.1.3.1 JDT – Editor Funktionen
- 1.1.4 PDE – Plug-in Develoment Environment
- 1.2 Das Eclipse Werkzeug Projekt
- 1.3 Das Eclipse Technologie Projekt
- 2.0 Quellen

# 0.1 Einleitung


- Eclipse zu dt.: Sonnenfinsternis
  - Entworfen von IBM
  - Rein zufällig heißt IBM's Konkurrent: SUN
- IBM
  - “open source”: Eclipse
  - kommerziell: Websphere Studio Application Developer
- SUN
  - “open source”: Netbeans
  - kommerziell: Sun Java Studio

# 0.2 Eclipse – Was ist das?


- Art universelle Plattform bzw. Rahmenwerk zur Integration von Werkzeugen zur Anwendungsentwicklung
- ist offen und erweiterbar für “Alles”
  - Entwickler können sich ihre Werkzeuge selbst schreiben, oder andere schon existierende Werkzeuge verwenden
- komplett in Java geschrieben
- auf vielen Systemen wie Linux, QNX, Mac OSx und Windows verbreitet
- läuft unter dem "open source" Gedanken unter CPL (Common Public License)
- **Ziel**: Förderung der Verbreitung der Plattform und Entwicklung entspechender Werkzeuge
- die Eclipse Stiftung ist ein nicht profit-orieniertes Unternehmen

# 0.3 Entstehung


- Erschaffen durch IBM und OTI Teams, die schon an der Entstehung von IBM Visual Age beteiligt waren
- Anfangs mit 40 Entwicklern
- Entwicklungsteams befanden sich an verschiedenen Orten (OTI Ottawa, OTI Minneapolis, IBM Toronto, ...)
- Alle Erfolge gingen in das “open source” Eclipse Projekt ein
  - Anfangs war IBM federführend in der Entwicklung der Plattform

# 0.4 Entstehungszeitraum


- April 1999: Arbeitsbeginn durch IBM und OTI Teams
- June 2001: Eclipse 0.9 fertig
- November 2001: www.eclipse.org geöffnet
- Juni 2002: Eclipse 2.0 fertig
  - Jetzt: Eclipse 3.0 M7

# 0.5 Zielsetzung der Entwickler


- Entwicklung einer offenen Platform um Werkzeuge für die Anwendungs-entwicklung zu erstellen
  - lauffähig auf den meisten Betriebssystemen
  - grafische Benutzeroberfläche (GUI)
- Sprachneutral
  - unbeschränkte Inhaltstypen
  - HTML, Java, C, JSP, EJB, XML, GIF, ...
- Unkomplizierte Werkzeugintegration
  - in der Benutzeroberfläche und tiefer
  - Neue Werkzeuge in bereits vorhandene Produkte einbinden
- Entwicklergemeinschaft heranziehen
  - unabhänige Software Händler mit hineinziehen
  - Auf der Popularität von Java aufbauen um Tools zu entwickeln

# 0.6 Systemvoraussetzungen


- Rechner mit mind. 400MHz Taktfrequenz, 256 MB Hauptspeicher und Windows 98+, Mac OS/X, Linux(x86), Solaris (SPARC), QNX, AIX oder HP-UX als Betriebssystem
- Java 2 Standard-Edition Laufzeitumgebung (J2SE JRE) ab Version 1.4 (Java-SDK ist nicht nötig, da Eclipse u.a. einen eigenen Compiler mitbringt)
- Eclipse - entweder das Platform Runtime Binary Archiv + JDT Runtime Binary oder das Eclipse SDK

# 1.0 Eclipse – Drei Projekte


- 1.1 Das Eclipse Projekt
- 1.2 Das Eclipse Werkzeug Projekt
- 1.3 Das Eclipse Technologie Projekt

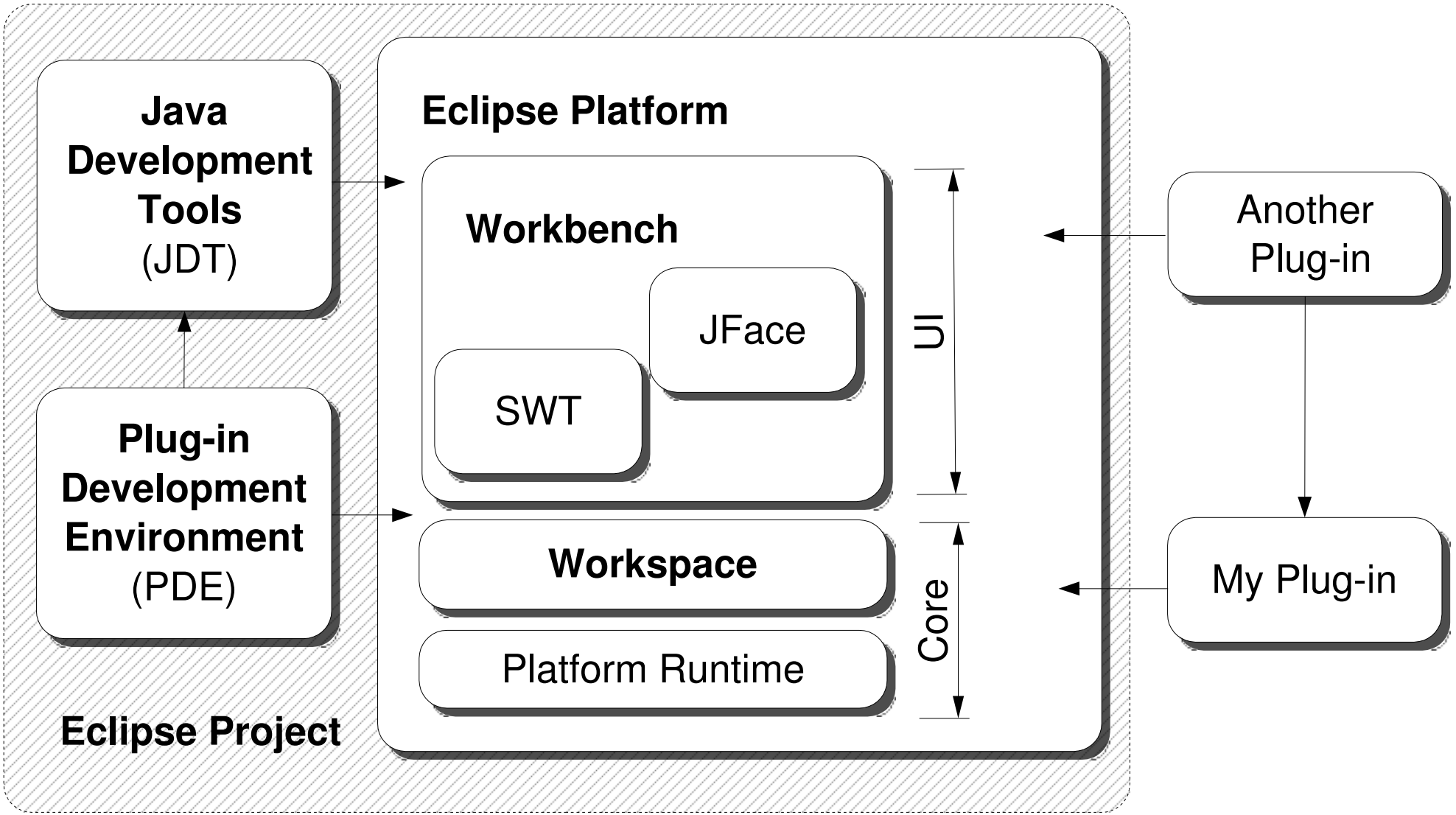
# 1.1 Das Eclipse Projekt 1/2


- "open source" Software-Projekt mit den Zielen:
  - Einem kommerziellen Produkt im Sinne von Qualität, Robustheit und Funktionalität in Nichts nachstehen
  - Anpassung an die Bedürfnisse der Entwicklergemeinde
  - Etablierung zum Industiestandart für die Entwicklung von hoch integrierten Werkzeugen

# 1.1 Das Eclipse Projekt 2/2


- Drei Komponenten:
  - Eclipse Plattform
  - JDT - Java Development Tools
  - PDE - Plug-in Development Environment

# 1.1.1 Plattform Überblick



# 1.1.2 Die Eclipse Plattform


- Definition von Rahmenwerken und Diensten zur Schaffung einer ganzheitlichen Plattform zur Integration verschiedenster Werkzeuge
- Darunter fällt:
  - die Workbench (User Interface)
  - das Projektmodel (Ressourcenverwaltung)
  - portierbare native "Widgets"
  - UI-Bibliotheken
  - sprachunabhänige Debug Infrastruktur, etc.

# 1.1.2.1 Workspace / Resource API 1/2


- ein spezielles Verzeichnis im Dateisystem
- enthält 1 oder mehr Hauptprojekt(e)
  - Projekte werden auf Verzeichnisse des Dateisystems abgebildet
  - Baum aus Ordnern und Dateien (Navigator)
- Dateien, Ordner und Projekte werden als **Ressourcen** bezeichnet
- Alle Werkzeuge arbeiten auf Ressourcen in der Workspace
- Plug-ins erlangen Zugriff durch den Workspace und die Ressource API

# 1.1.2.1 Workspace / Resource API 2/2


- Erlaubt eine schnelle Navigation durch den Workspace Ressourcen Baum
- Ressourcen “Change Listener” zur Überwachung von Aktivitäten
  - d.h. Veränderung einer Ressource durch eine Anwendung aus Eclipse, impliziert die Benachrichtigung anderer Anwendungen
- begrenzte Historie von bearbeiteten / gelöschten Dateien
- Workspace Lebenszyklus (Speichern, Beenden, Wiederherstellen)

# 1.1.2.2 Workbench


- Der grafische sichtbare Teil der Eclipse Plattform
- Benutzer arbeitet immer in ihr
- Komponenten:
  - Perspektiven
  - Sichten
  - Editoren
- Bausteine:
  - SWT
  - JFace

# 1.1.2.2.1 Editoren, Sichten, Perspektiven 1/3


- Erscheinen im Workbench Editor Bereich
- stellen spezielle Befehle für die Menü- und Werkzeugleisten zur Verfügung
- dienen zum Bearbeiten von Ressourcen
  - strikter Laden-Verändern-Speichern Lebenszyklus
  - erst nach Speicherung einer im Editor veränderten Ressource, können die Benachrichtigungsmechanismen
- Plug-in:
  - neue Arten von Editoren möglich (Erweiterungspunkte)
  - z.B.: JDT bietet einen Java Quellcodeeditor
  - z.B.: Eclipse Plattform bietet einen einfachen Texteditor

# 1.1.2.2.1 Editoren, Sichten, Perspektiven 2/3


- Machen Informationen über Objekte sichtbar
- Direkte Kopplung mit einem Editor möglich (nicht mit der Ressource)
  - z.B.: “Outline View” ist an den Java-Quelltexteditor gekoppelt
  - Fasst den Inhalt des Editors zusammen
- Direkte Kopplung mit anderen Sichten
  - z.B.: “Properties View” beschreibt eine  Auswahl
- Plug-in:
  - Neue Arten von Sichten möglich (Erweiterungspunkte)
  - Eclipse Plattform enthält viele Standart Sichten (Resource Navigator, Outline, Properties, Tasks, ...)

# 1.1.2.2.1 Editoren, Sichten, Perspektiven 3/3


- Bestimmen das konkrete Aussehen der Workbench
- Sind Zusammenstellung von Sichten und Editoren
- Verschiedene Perspektiven für unterschiedliche Aufgaben
- Schnelles Umschalten zwischen einzelnen Perspektiven möglich
- Plug-in:
  - Neue Arten von Perspektiven möglich (Erweiterungspunkte)
  - Eclipse Plattform enthält viele Standart Perspektiven (Ressource, Debug, Java, ...)

# 1.1.2.2.2 SWT, JFace 1/3


- SWT = Standard Widget Toolkit
- Allgemeine Grafiken und das “GUI widget set” (Knöpfe, Listen, Texte, Menüs, Bäume, ....)
- Einfach, klein und schnell
- betriebssystemunabhängige API
- Benutzt die Basis-Steuerelemente wenn vorhanden
- Emuliert die Steuerelemente wenn nicht vorhanden

# 1.1.2.2.2 SWT, JFace 2/3


## Warum SWT?

- **Meinung:** Entwickelung professionell aussehender Anwendungen ist zu schwer mit AWT und Swing
- SWT bietet:
  - Enge Integration mit dem bestehenden Fenster System
  - Authentischer “Look and Feel”
  - Gute Leistung und Portierbarkeit
  - Gute Basis zur Entwicklung von robusten Benuzteroberflächen
- “...the proof of the pudding is the eating...”

# 1.1.2.2.2 SWT, JFace 3/3


- Anzahl UI-Rahmenwerke für allgemeine UI-Aufgaben
- Entworfen um in Verbindung mit SWT genutzt zu werden
- Stellt Klassen zur Handhabung von allgemeinen UI-Aufgaben bereit
- API und Implementation ist unabhängig vom  Fenster System

# 1.1.2.3 Weitere Komponenten 1/2


- **Team Komponente**
  - Versions- und Konfigurationsleitung (VCM)
  - Teilen von Ressourcen im Team mittels eines Repository
  - Eclipse Plattform enthält verschiedene CVS Repository Anbieter
- **Ant Komponente**
  - Verwendet Apache Ant
  - Erzeuger Werkzeug in Java mit XML-basierten Scripten

# 1.1.2.3 Weitere Komponenten 2/2


- **Debug Komponente**
  - Verschiedene Startkonfigurationen
  - Allgemeines Debug Modell (Ereignissen, Aktionen, Haltepunkten,...)
  - z.B.: JDT – bringt eigenen Starter und Debugger mit
- **Help Komponente**
  - Präsentiert sich in einem Standard-Webbrowser
  - Eclipse Plattform: “Workbench User Guide, Plattform Plug-in DG, ...
  - JDT und PDE: bringen eigene Hilfen mit
  - Hilfe Mechanismen sind allen Plug-ins zugänglich
  - Hilfe Suchmaschine basiert auf Apache Lucene
  - Hilfe Server (kopflos) basiert auf Apache Tomcat

# 1.1.3 JDT - Java Development Tools


- Werkzeug (Plug-in)
- Implementation einer Java IDE für die Entwicklung von Java-Anwendungen bzw. eigener Eclipse Plug-ins
- enthält eigene Perspektive mit verschiedenen Ansichten, Editoren, Wizards, Code merging, Refactoring
  - eine Entwicklungsumgebung für sich selbst
- Erweiterbarkeit durch andere Werkzeuge
- Ziele:
  - Nummer 1 unter den Java IDE's
  - “..to make programmers smile...”

# 1.1.3.1 JDT – Editor Funktionen


- Tooltips (bei Bewegungen über Bezeichner)
- “fliegender” Sprachtest
- Inhaltsassistent
  - Methoden Vervollständigung
  - Kode Schablonen Hilfe
  - Vorschlag von Variablennamen
  - JavaDoc-Assisten
- Parameter Hinweise
- Kode Formatierer
- Quellkodeanzeige bei binären Bibliotheken
- Refactoring
  - Organisieren von Importen
  - Umbenennen, Bewegen, Extrahieren, ...

# 1.1.4 PDE – Plug-in Development Environ


- Enthält Editoren und Sichten um die Entwicklung von Eclipse Plug-ins zu vereinfachen
- **Plug-in Erzeugung**: Erstellung einer Plug-in Manifest-Datei (XML)
  - Festlegung der Plug-in Laufzeitumgebung, anderer verwendeter Plug-ins
  - Definition von Erweiterungspunkten (Extension points)
  - Zuordenen von XML Schema Dateien zu den Erweiterungspunkten
- "...the PDE makes integrating plug-ins easy and fun."

# 1.0 Eclipse - Drei Projekte


1.1 Das Eclipse Projekt

- 1.2 Das Eclipse Werkzeug Projekt

1.3 Das Eclipse Technologie Projekt

# 1.2 Das Eclipse Werkzeug Projekt


- Schwerpunkt auf Werkzeugentwicklung
  - Qualitätsgarantie für Werkzeuge der Eclipse Plattform
- Förderung einer breiten Werkzeugpalette
- Verhinderung von versehendlichen Überlappungen oder Verdopplungen der Werkzeuge
- Erzeugung von gemeinsamen Komponenten voranstellen
- Kompatibilität der Werkzeuge untereinander gewährleisten

# 1.3 Eclipse - Drei Projekte


1.1 Das Eclipse Projekt

1.2 Das Eclipse Werkzeug Projekt

- 1.3 Das Eclipse Technologie Projekt

# 1.3 Das Eclipse Technologie Projekt


- Angebot  neuer Richtungen für die "open source" Entwickler um sich ihrer Beteiligung an der Weiterentwicklung von Eclipse zu sichern
- Unterteilung in 3 Projekte:
  - Forschung: in Eclipse-bezogene Gebiete (Progr.-sprachen, IDE, Werkzeuge)
  - Brutkasten: kleine Projekte, neue Fähigkeiten für die Eclipse Software Basis
  - Ausbildung: Bereitstellung von Aus- bzw. Fortbildunggskursen, Tutorials

# 2.0 Quellen


- www.eclipse.org
- www.3plus4software.de/eclipse/index.html [Tutorial]
- Javamagazin 5.2004 [Eclipse vs. Netbeans]