



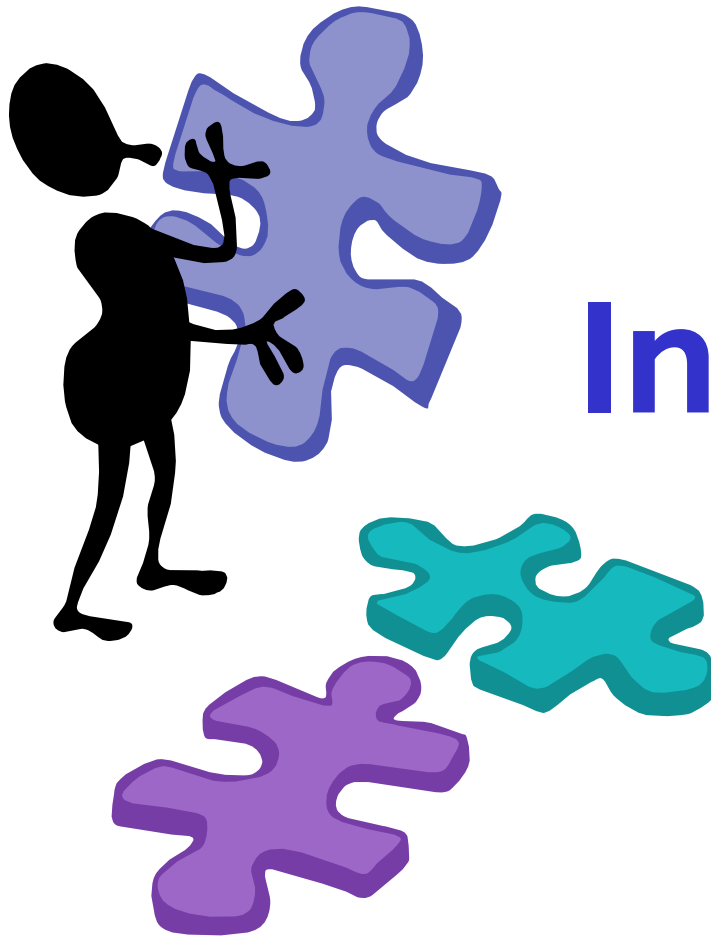
Hello AspectJ

oliver.boehm@agentes.de

Inhalt



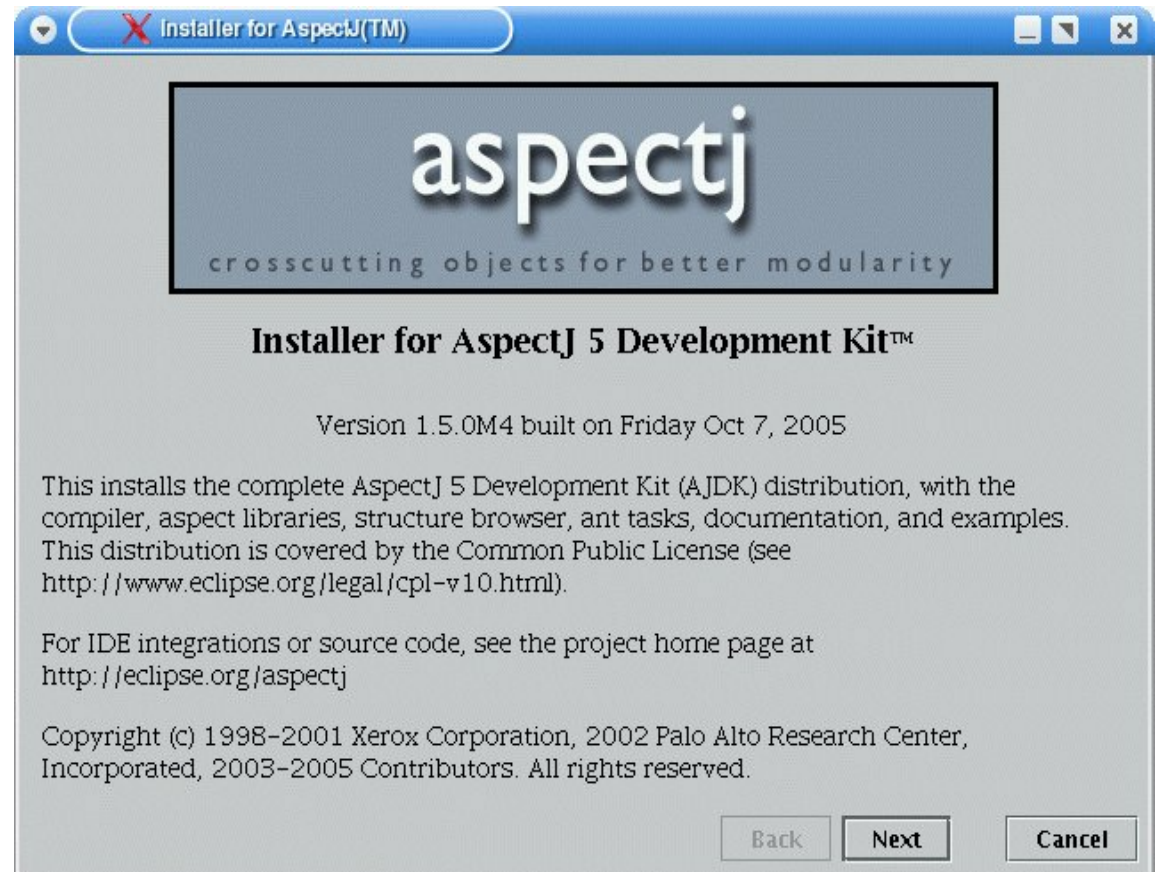
- ☉ **Installation**
- ☉ **Hello IDE**
- ☉ **Zusammenfassung**



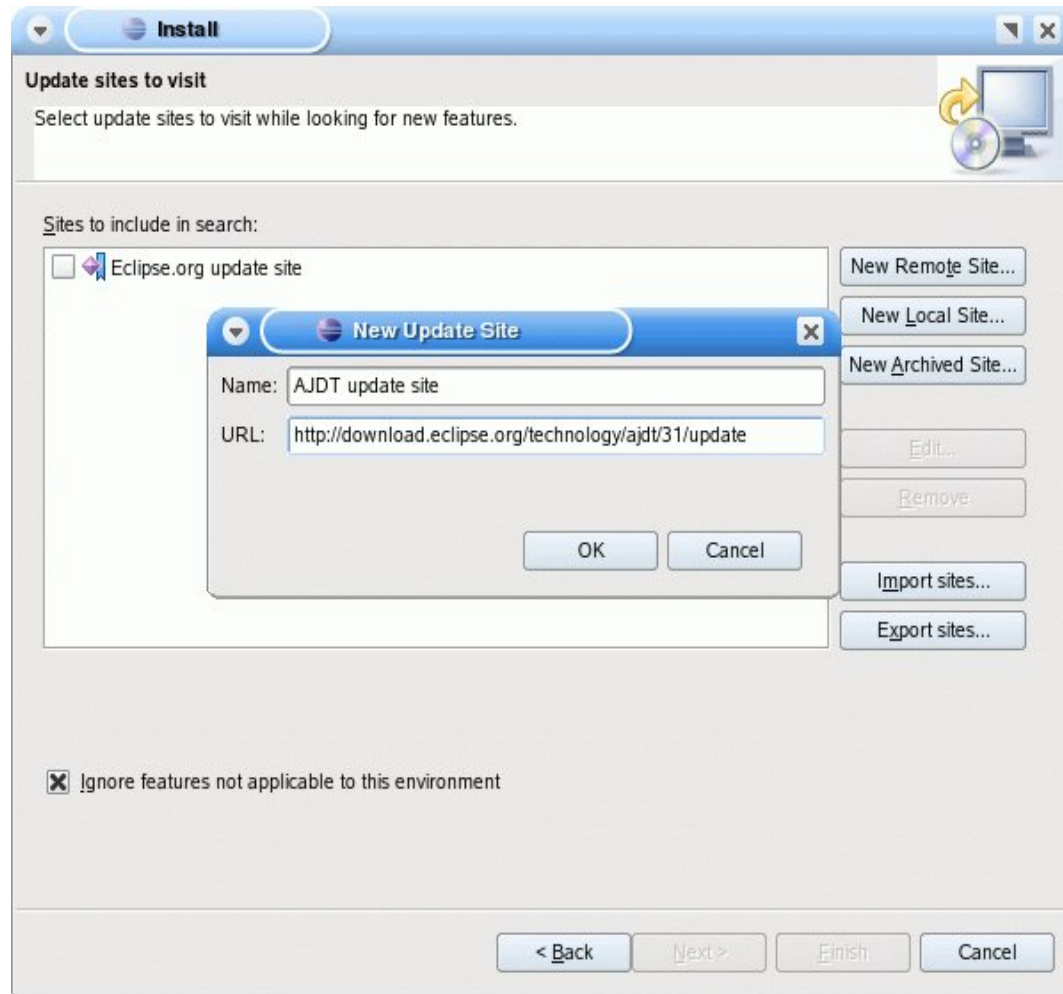
Installation

Compiler installieren

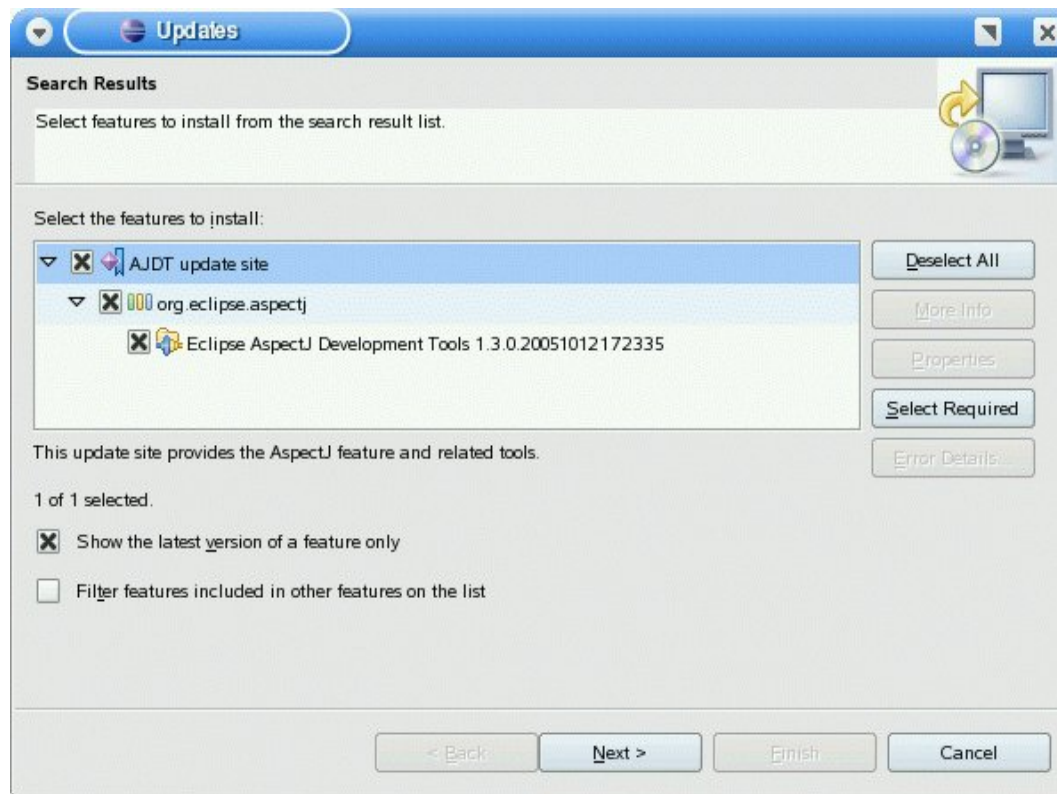
- <http://www.eclipse.org/aspectj/>
- `java -jar aspectj.jar`
- **Wozu?**
 - Aufruf von Hand
 - Build-Prozess
- **Test: `ajc -help`**



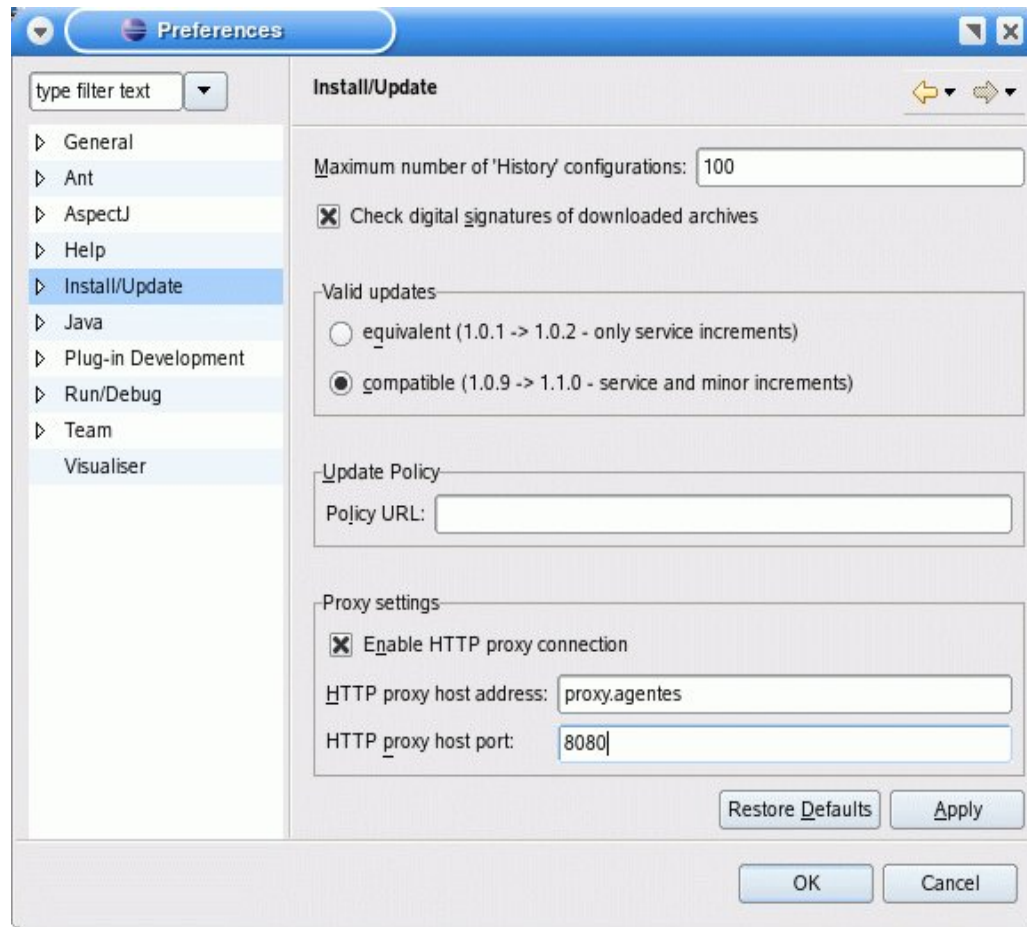
AJDT installieren (Update-Manager)



AJDT als neues Feature auswählen



Proxy-Einstellungen



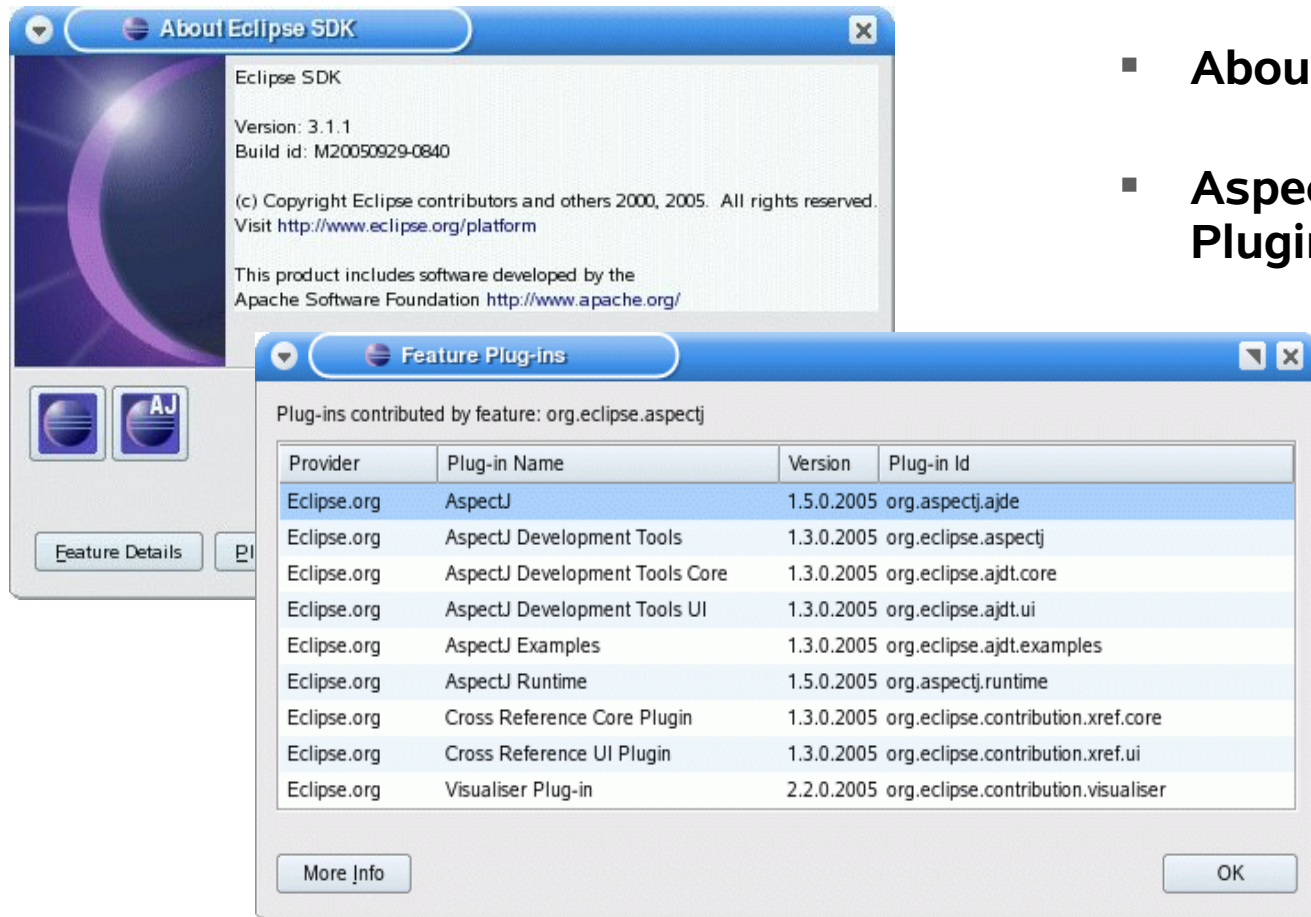
Manuelle Installation

- **Zip-Datei herunterladen**
- **ins Eclipse-Verzeichnis gehen**
- **auspacken:**

```
> unzip ajdt_1.3.0_archive.zip
Archive:  ajdt_1.3.0_archive.zip
  creating: features/org.eclipse.aspectj_1.3.0/
 inflating: features/org.eclipse.aspectj_1.3.0/aspectjLogo.gif
 inflating: features/org.eclipse.aspectj_1.3.0/cpl-v10.html
 inflating: features/org.eclipse.aspectj_1.3.0/feature.properties
 inflating: features/org.eclipse.aspectj_1.3.0/feature.xml
 inflating: features/org.eclipse.aspectj_1.3.0/license.html
  creating: plugins/org.aspectj.ajde_1.5.0/
  creating: plugins/org.aspectj.ajde_1.5.0/META-INF/
 inflating: plugins/org.aspectj.ajde_1.5.0/META-INF/MANIFEST.MF
 inflating: plugins/org.aspectj.ajde_1.5.0/about.html
 inflating: plugins/org.aspectj.ajde_1.5.0/ajde-src.zip
 inflating: plugins/org.aspectj.ajde_1.5.0/ajde.jar
```

...

Installationscheck



- About Eclipse
- AspectJ-Plugins

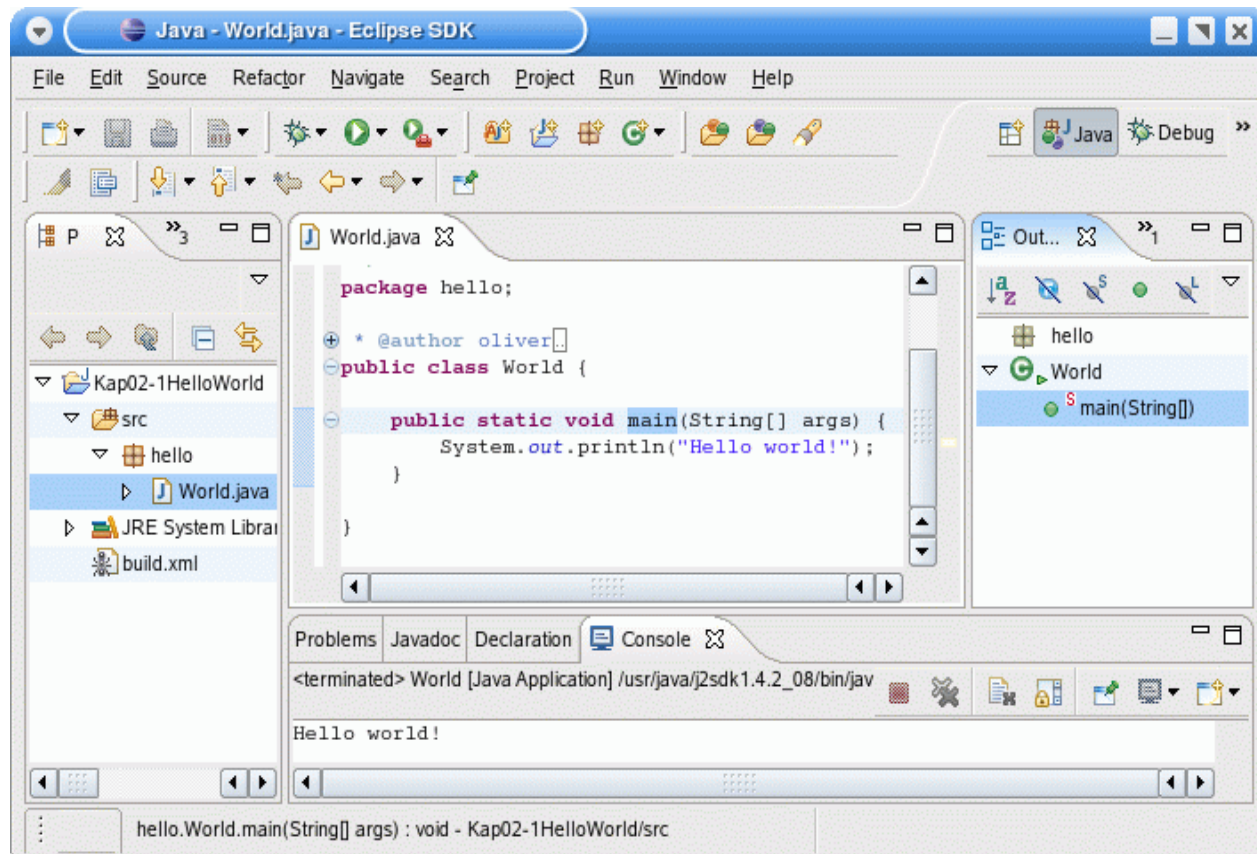


Hello IDE

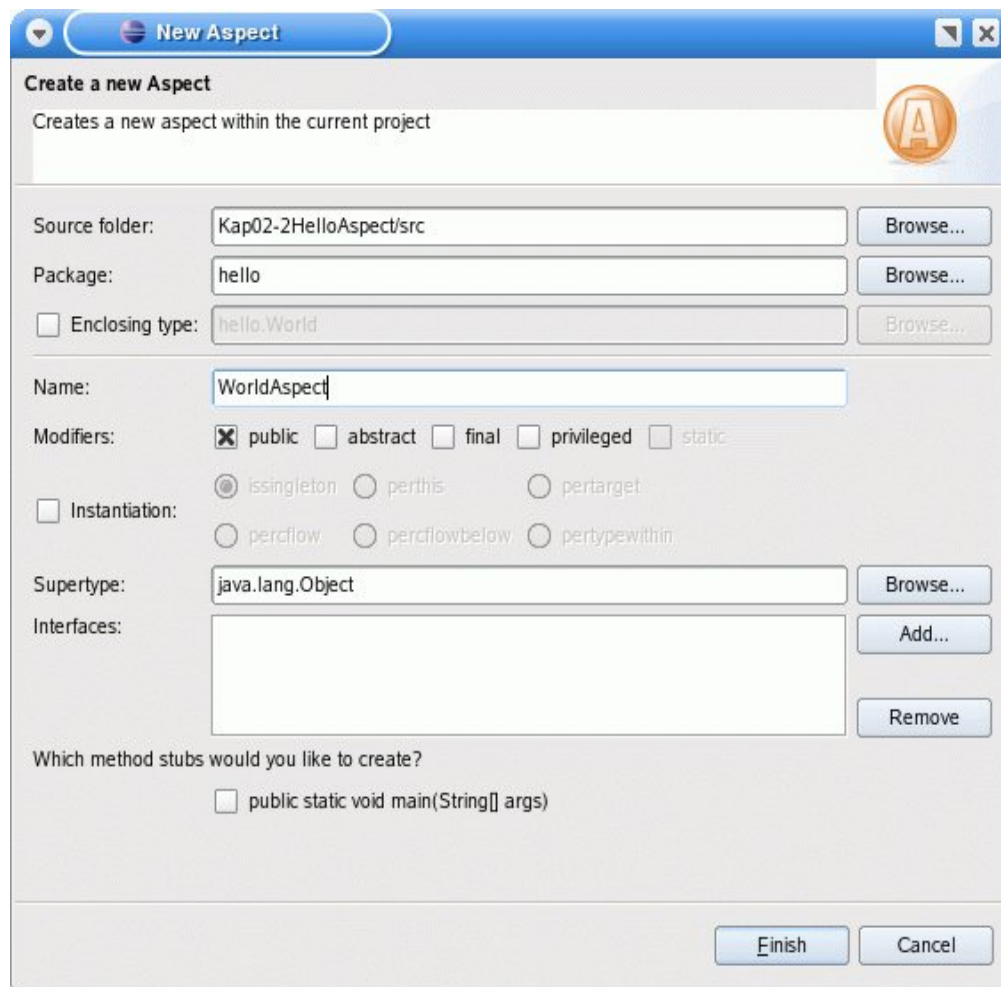
Java-Projekt anlegen



World.java



Aspekt anlegen



Create a new Aspect
Creates a new aspect within the current project

Source folder:

Package:

Enclosing type:

Name:

Modifiers: public abstract final privileged static

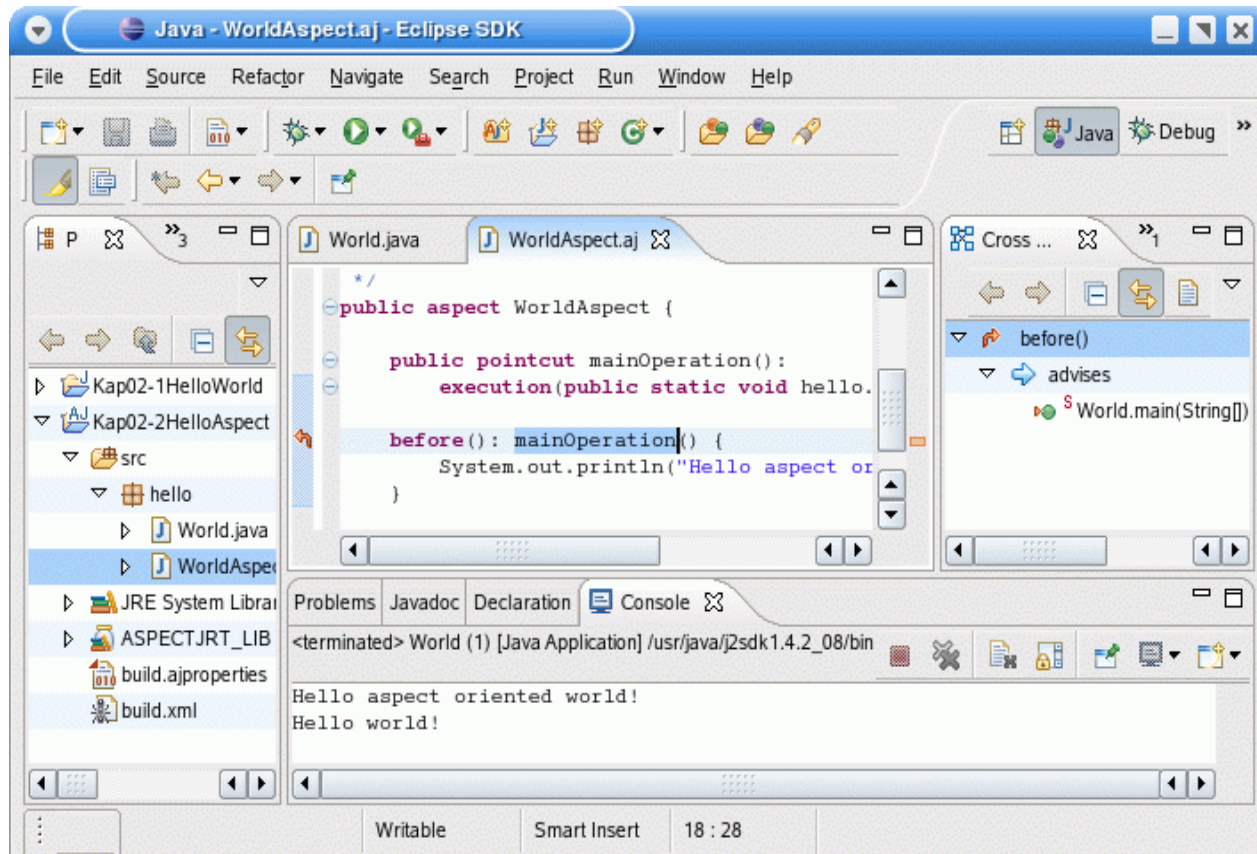
Instantiation: issingleton perthis pertarget
 perflow perflowbelow pertypewithin

Supertype:

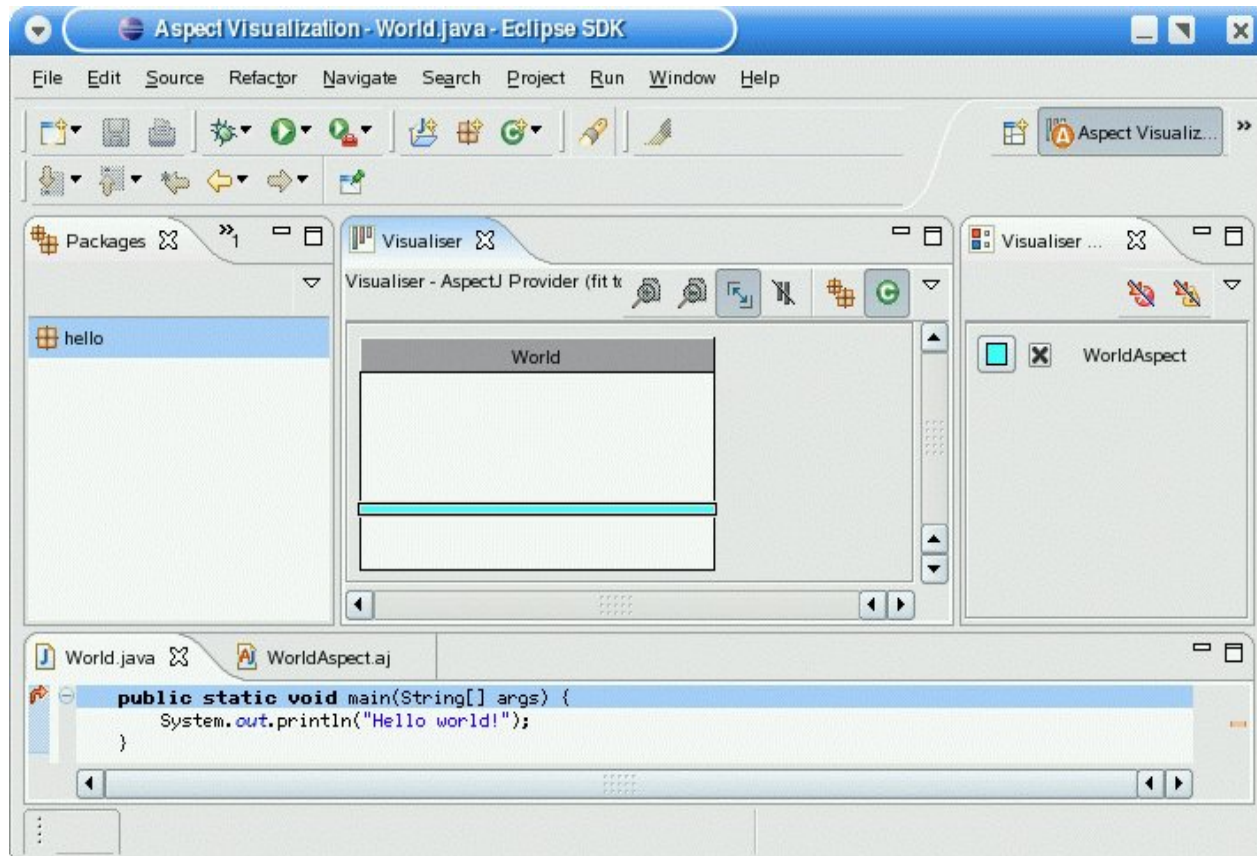
Interfaces:

Which method stubs would you like to create?
 public static void main(String[] args)

WorldAspect.aj



AspectVisualization-Perspektive



inkrementelle Compilierung

- **standardmäßig aktiviert**
- **deaktivieren: Project ▷ Build ▷ Automatically**
 - sinnvoll: bei großen Projekten
- **manueller Build:**
 - Project > Build
 - Build--Button



Zusammenfassung

- **AspectJ setzt auf Java und dem Java-Compiler auf**
- **Aufruf: ajc**
- **es existieren Plugins für diverse IDEs**
- **AJDT: Plugin für Eclipse**
- **Installation:**
 - über Update-Manager
 - manuell
- **Installations-Check: Plugin-Liste überprüfen**
- **Java-Projekte können in AspectJ-Projekt umgewandelt werden**
- **Visualisierungs-Hilfen:**
 - Outline-View
 - Crossreference-View
 - Aspect-Visualization-Perspektive

- Implementieren Sie eine einfache Konto-Klasse mit den Methoden einzahlen(), abheben() und ueberweisen(). Legen Sie es als Java-Projekt an.
- Konvertieren Sie das Projekt in ein AspectJ-Projekt
- Legen Sie einen "LogAspect" im selben Paket wie die Konto-Klasse mit folgendem Inhalt an:

```
package bank;  
public aspect LogAspect {  
    pointcut executeAbheben() :  
        execution(public void Konto.abheben(int));  
    before() : executeAbheben() {  
        System.out.println("starte abheben()");  
    }  
}
```

- Passen Sie Angaben wie Paket-, Klassen- oder Methoden-Namen (abheben) an Ihre Gegebenheiten an. Achten Sie vor allem darauf, dass die Signatur der Methode übereinstimmt.