

日本Javaユーザーグループ  
第3回Java基礎セミナー

# Java開発環境基礎

---

日本Javaユーザーグループ  
松前健太郎・川尻剛

# 目次

- 1 . 開発環境・IDEについて
- 2 . Eclipseについて
- 3 . Eclipse+JDTでJava開発
- 4 . WTPでWebアプリ開発

## 休憩

- 5 . TPTPで品質向上
- 6 . Subversive+Mylyn+Tracでタスク指向開発
- 7 . その他のeclipseサブプロジェクト

# はじめに

今日はJavaの開発環境のお話・・・  
ところで、開発環境って何？

# 開発環境といえば・・・

・IDE(統合開発環境)、テキストエディタなど



・バージョン管理システム



VSS(Visual Source Safe)

・バグトラッキングシステム(BTS)、懸案管理システム



# IDEの機能・役割

## (1)プロジェクト管理

ソースや設定ファイルをプロジェクト、ワークスペース、といった単位で管理できる。

## (2)バージョン管理

ソースコードの版管理。

IDEから直接SVN, CVS, VSSとやり取りできる。

## (3)GUIの作成

UIをグラフィカルに定義できる。VisualStudioが得意としているところ。NetBeansもいい。Eclipseはいまいち。

# IDEの機能

## (4)チーム開発

みんな同じ環境で開発するので、作業が進めやすい。  
手順が決まっているのでルールを決めやすい。

## (5)コーディング補助

コードアシスト、シンタックスハイライト、  
インクリメンタルビルドなど。

## (6)ビルド、デバッグ補助

ビルド、デバッグ機能を備えているものが多い。

# いろいろなIDE

- ・Eclipse
  - ・NetBeans
  - ・IntelliJ IDEA
  - ・Borand JBuilder
  - ・JDeveloper(oracle)
  - ・Visual Studioシリーズ
  - ・ReSharper
- などなど

今回はEclipseに着目!

# Eclipseとは

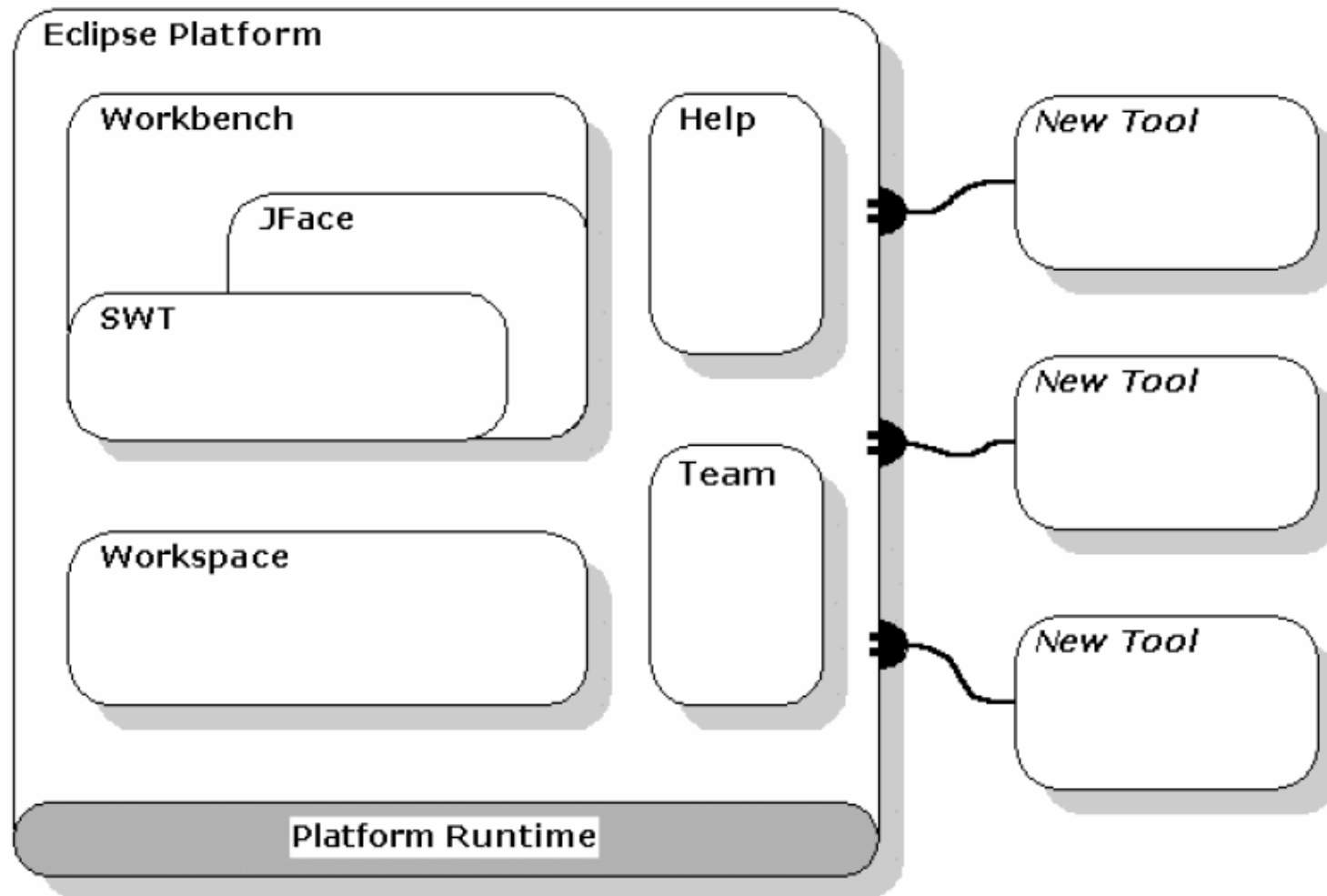
- ・統合開発環境(IDE)のひとつ
- ・プラグインによる機能拡張が可能
- ・Javaで動作する(一部ネイティブ)
- ・オープンソース

# Eclipseの簡単な歴史

- 1999/04 IBM/OTI内部でEclipse開発開始
- 2001/10 Eclipse1.0リリース
- 2001/11 IBMがソースコードをEclipse Foundationに寄贈  
eclipse.orgサイトオープン
- 2002/06 Eclipse2.0リリース
- 2003/03 Eclipse2.1リリース
- 2004/06 Eclipse3.0リリース
- 2005/06 Eclipse3.1リリース
- 2006/06 Eclipse3.2リリース
- 2007/06 Eclipse3.3リリース
- 2007/09 Eclipse3.3.1リリース(予定)

# Eclipseの構造

- ・Eclipseはプラグインの塊  
「開発ツールのプラットフォーム」



# Eclipseプロジェクトの構成

## トップレベル・プロジェクト

- ・ Eclipse Project
- ・ Eclipse Tools Project
- ・ Eclipse Web Tools Platform (WTP) Project
- ・ Eclipse Test & Performance Tools Platform (TPTP) Project
- ・ Eclipse Technology Project
- ・ Eclipse Modeling Project
- ・ Business Intelligence and Reporting Tools (BIRT)
- ・ Data Tools Platform (DTP)
- ・ Device Software Development Project
- ・ SOA Project

# Eclipseプロジェクトの構成

## トップレベル・プロジェクト

- Eclipse Project
- Eclipse Tools Project
- Eclipse Web Tools Platform (WTP) Project
- Eclipse Test & Performance Tools Platform (TPTP) Project
- Eclipse Technology Project
- Eclipse Modeling Project
- Business Intelligence and Reporting Tools (BIRT)
- Data Tools Platform (DTP)
- Device Software Development Project
- SOA Project

# Eclipseプロジェクトの構成

## Eclipse Project

Eclipseの基本的な機能を提供するトップレベルプロジェクト  
-platform

Ant, Core, CVS, Debug, Search, SWT, Team/Compare,  
Text, User Assistance, UI, Update, etc..

-JDT (Java Development Tools)

Java開発支援機能

-PDE (Plugin Development Tools)

Eclipseプラグイン開発

-Equinox (an OSGI framework)

# Eclipseプロジェクトの構成

## トップレベル・プロジェクト

- ・ Eclipse Project
- ・ **Eclipse Tools Project**
- ・ Eclipse Web Tools Platform (WTP) Project
- ・ Eclipse Test & Performance Tools Platform (TPTP) Project
- ・ Eclipse Technology Project
- ・ Eclipse Modeling Project
- ・ Business Intelligence and Reporting Tools (BIRT)
- ・ Data Tools Platform (DTP)
- ・ Device Software Development Project
- ・ SOA Project

# Eclipseプロジェクトの構成

## Eclipse Tools Project

基礎的なツールや共通的なコンポーネントを提供

- C/C++ IDE
  - COBOL
  - PHP Development Tools
  - AspectJ/AJDT
  - Mylyn Project
  - Visual Editor (VE)
  - Graphical Editor Framework
- etc..

# Eclipseプロジェクトの構成

## トップレベル・プロジェクト

- ・ Eclipse Project
- ・ Eclipse Tools Project
- ・ **Eclipse Web Tools Platform (WTP) Project**
- ・ Eclipse Test & Performance Tools Platform (TPTP) Project
- ・ Eclipse Technology Project
- ・ Eclipse Modeling Project
- ・ Business Intelligence and Reporting Tools (BIRT)
- ・ Data Tools Platform (DTP)
- ・ Device Software Development Project
- ・ SOA Project

# Eclipseプロジェクトの構成

## WTP Project

J2EE/JavaEEやWebアプリケーションの開発ツールを提供

-Web Standard Tools (WST)

-J2EE Standard Tools

-Java Server Faces (JSF)

-Dali-ORM (JPA)

-AJAX Tools Framework

# Eclipseプロジェクトの同時リリースの取り組み

・数多くのプラグインがリリース

プラグイン同士の依存関係が複雑化(例:VE EMF WTP)

Eclipseのリリースにあわせて主要なプラグインを同時期にリリースする取り組みを開始。

Eclipse3.2リリース Callisto

(今ココ ) **Eclipse3.3リリース Europa**

Eclipse3.4リリース Ganymede (2008/06/29?)

Eclipse3.5リリース IO ???

( "Callist", "Europa" という名前の製品が存在するというわけではない！ )

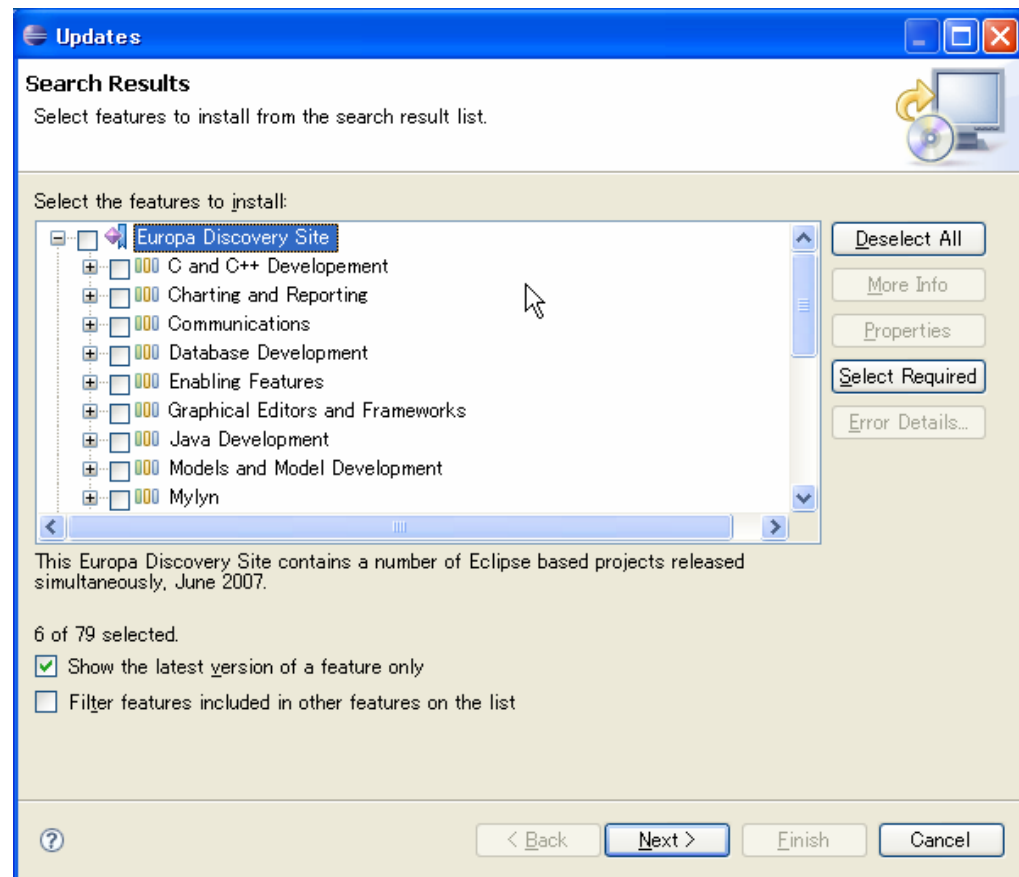
# ディスカバリーサイトの提供

・Eclipseと同時にリリースされた各プラグインを配布する更新サイト(Europa Discovery Site)を公開

(1) Eclipseをダウンロード

(2) ディスカバリーサイトから必要なプラグインをインストール

**容易な導入、安定した動作**



## Europaに含まれる21プロジェクト

AspectJ Development Tools (AJDT)

Business Intelligence and Reporting Tools (BIRT)

Buckminster

C/C++ Development Tools (CDT)

Data Tools Platform (DTP)

Device Software Development Platform - Device Debugging (DSDP.DD)

Device Software Development Platform - Target Management (DSDP.TM)

Dynamic Languages Toolkit (DLTK)

Dash (Eclipse Monkey)

Eclipse Communication Framework (ECF)

Eclipse Platform, JDT, PDE and Equinox.

Eclipse Modeling Framework (EMF)

Eclipse Modeling Framework - Query, Transaction, Validation (MQ, MT, VF)

Graphical Editing Framework (GEF)

Graphical Modeling Framework (GMF)

Model Development Tools (MDT)

Model to Text (M2T) - JET

Mylyn

SOA Tools Platform (STP)

Test and Performance Tools Platform (TPTP)

Web Tools Platform (WTP)

# Eclipse+JDTでJava開発！ の準備

# 開発の準備

http://www.eclipse.org/downloads/ からダウンロード

The screenshot shows the Eclipse Downloads page in Mozilla Firefox. The browser's address bar shows the URL <http://www.eclipse.org/downloads/>. The page content includes a navigation menu on the left, a main content area with an orange banner that says "eclipse europa is here! get it!", and a list of Eclipse packages. A yellow arrow points to the "Eclipse IDE for Java EE Developers" package, which is highlighted with a callout box containing the text "Webアプリ開発ならこれ".

To download Eclipse, select a package below or choose one of the third party Eclipse distros. **You will need a Java runtime environment (JRE) to use Eclipse (Java 5 JRE recommended).** All downloads are provided under the terms and conditions of the **Eclipse Foundation Software User Agreement** unless otherwise specified.

### Eclipse Packages

| Package Name                       | OS                     | Size   |
|------------------------------------|------------------------|--------|
| Eclipse IDE for Java Developers    | Windows, Linux, MacOSX | 78 MB  |
| Eclipse IDE for Java EE Developers | Windows, Linux, MacOSX | 125 MB |
| Eclipse IDE for C/C++ Developers   | Windows, Linux, MacOSX | 62 MB  |
| Eclipse for RCP/Plug-in Developers | Windows, Linux, MacOSX | 153 MB |
| Eclipse Classic                    | Windows, Linux, MacOSX | 140 MB |

完了 Pathtraq 0.1 / 0.7 ノートブックを開く(N) M 1

# 開発の準備

## インストールと起動

アーカイブを展開して、“eclipse.exe”を実行するだけ(もちろんJRE/JDK必須)

## 起動時オプション

eclipseを起動するJVM、メモリ割り当てサイズなど指定しておく快適

```
eclipse.exe  
-clean -vm c:/../jdk1.6.0/javaw.exe  
-vmargs -Xmx256M -Xms256M -XX:MaxPermSize=128M
```

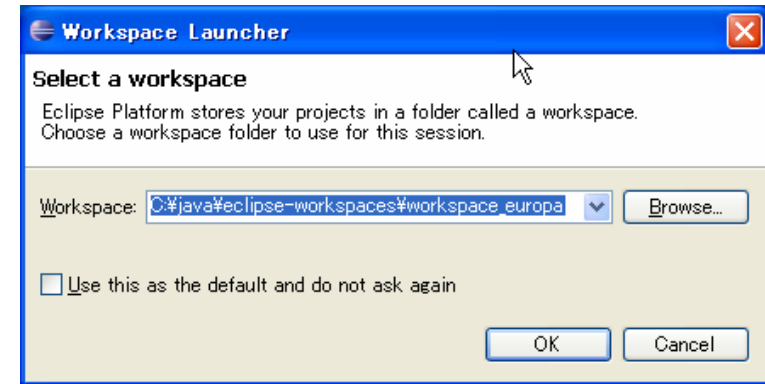
## ヒープ情報の表示

- ・ステータスバーにヒープメモリの状態をリアルタイム表示
- ・[Preferences] [General] [Show heap Status]にチェック



# 開発の準備

## ワークスペース・フォルダの指定



- ・ワークスペース = Eclipseで扱う全てのファイル(ソースコード、ライブラリ、設定ファイル)を格納するフォルダ
- ・Eclipseの各種設定内容もワークスペースに保存される。

- ・自分が関わっているプロダクト、製品、業務ごとにワークスペースを分けておくとよいかも。

例)

仕事A用のワークスペース

仕事B用のワークスペース

- ・eclipseインストールフォルダ以下にワークスペースを作るのはあまりおすすめできない。

```
-ワークスペース/  
-プロジェクトA/  
  -src  
    -javaソースコード  
  -lib  
    -ライブラリ  
  設定ファイル  
  ビルドファイル  
-プロジェクトB/  
-プロジェクトC/
```

# 開発の準備

## Eclipseの設定・これだけは要確認

### インストールJREの設定

- ・JREは起動時に自動的に認識、登録されるが意図したものと違ったものが登録されていないか。
- ・WTP+Tomcatを使う場合にはJREではなくJDK

### コンパイラ準拠レベルの設定

- ・1.3? 1.4? 5.0? 6.0?

[デモ] [cap¥010\\_jre\\_settings.htm](#)

# 開発の準備

## プラグインのインストール

・zipをダウンロード 解凍 pluginsフォルダ上書き・・・(古いやり方)  
最近は・・・

・「**Update site (更新サイト)**」を経由してインストールする

- (1) プラグインのWebサイトに行く
- (2) 更新サイトのURLを探す
- (3) Eclipseを起動し、[Software Update]を実行

インストール済みのプラグインを確認したり、削除したいときは・・・

・「**Product Configuration**」**ダイアログ**を使う

/pluginsフォルダからプラグインフォルダを直接削除・・・なんてことはしない！

[デモ] [cap¥020\\_update\\_site.htm](#)

[デモ] [cap¥030\\_manage\\_configuration.htm](#)

Eclipse+JDTでJava開発！

# その前に

ここからはEclipseを実際に動かしながら、  
お話をしていきますが…

Eclipseを使いこなす第一歩は

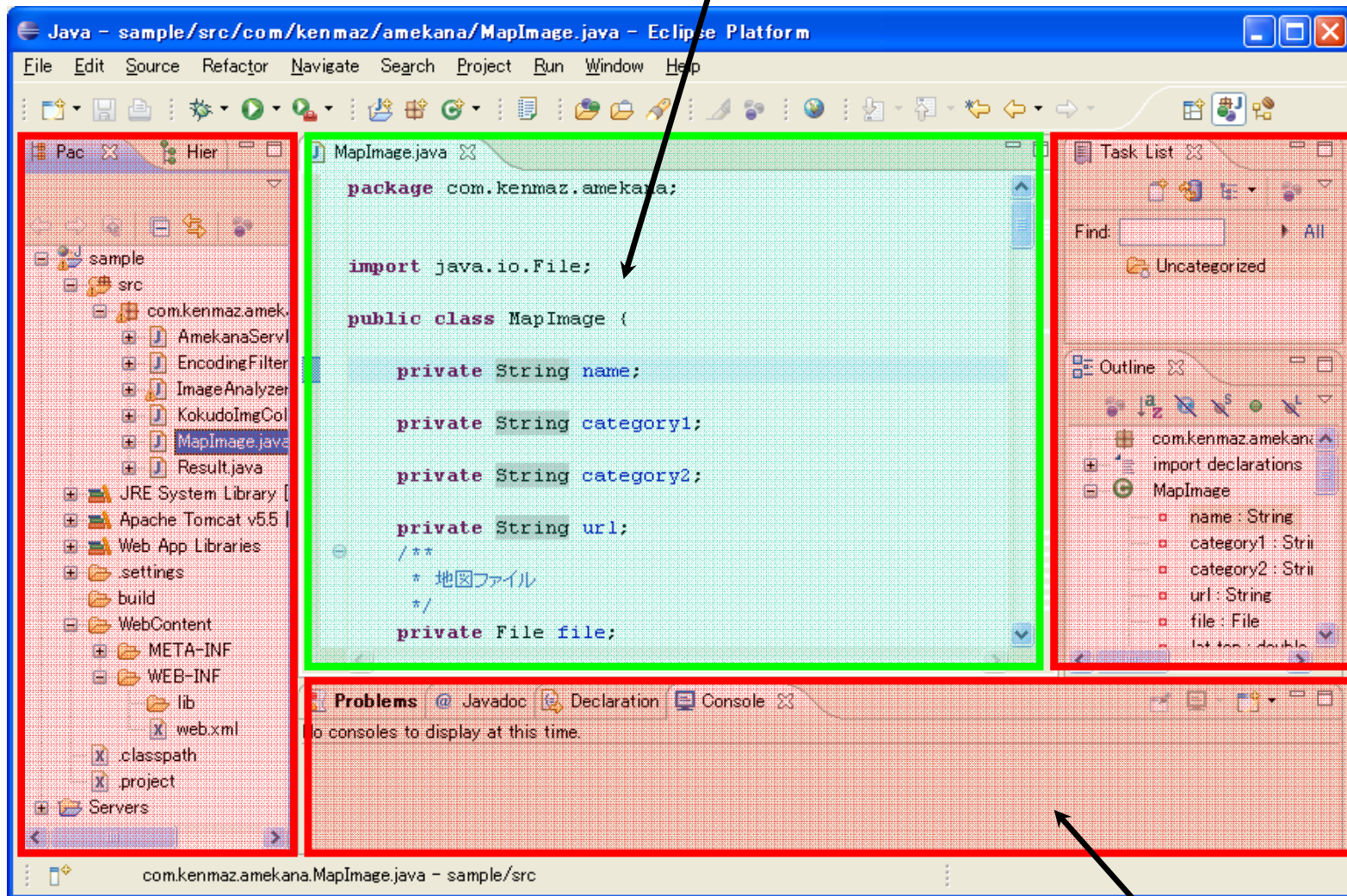
**キーボードショートカットを使いこなすことである！**

…と思っています。

そのあたりにも注意しつつご覧ください。

# Eclipseワークベンチの基礎

エディタ領域



ビュー領域

# Eclipseワークベンチの基礎

## エディタ領域

- ・各種エディタ(Javaエディタ、テキストエディタ、XMLエディタ)を表示する領域
- ・タブ形式で表示
- ・レイアウトは自由に変更可能(タブを並べる、最大化する)
- ・同じファイルを別タブに表示して並べることも可能

## タブの切り替え方法はいろいろ

- ・ Ctrl+PageDown, Ctrl+PageUp (順番に)
- ・ Ctrl+E (エディタ領域ごと)
- ・ Ctrl+F6 (すべてのエディタを一覧表示)
- ・ F12で強制的にエディタにフォーカス

# Eclipseワークベンチの基礎

## ビュー領域

- ・各種ビュー(パッケージエクスプローラ、ナビゲータービュー、アウトラインビュー、問題ビューなど、エディタ以外の画面要素)を表示する領域
- ・画面が狭い人はビューの最大化、最小化
- ・画面が狭い人は高速ビュー

## マウスが嫌いなひとは

- ・ Ctrl+PageDown, PageUp (順番に)
- ・ Ctrl+F7 (すべてのビューを一覧表示 選択)

[デモ] [cap¥041\\_view.htm](#)

# Eclipseワークベンチの基礎

## パースペクティブ

- ・作業目的に応じて、ビューを組み合わせたセット
- ・カスタマイズ、保存が可能。自分の使いやすいように。

### *[Javaパースペクティブ]*

- パッケージエクスプローラ
- 階層ビュー
- アウトラインビュー
- タスクリスト
- 問題
- Javadoc
- 宣言

### *[デバッグパースペクティブ]*

- デバッグ
- 変数
- ブレークポイント
- コンソール

### *[JavaEEパースペクティブ]*

- パッケージエクスプローラ
- サーバー
- データソースエクスプローラ
- スニペット

[デモ] [cap¥042\\_perspective.htm](#)

# JDTの基礎

## JDTのポイント

- ・Javaパースペクティブ
- ・インクリメンタルビルド
- ・Javaコーディング支援機能
- ・Javaソースコード編集支援機能
- ・Javaブラウジング支援機能
- ・Javaデバッグ支援機能
- ・Java実行支援機能

# JDTの基礎 –Javaパースペクティブ

## パッケージエクスプローラ

プロジェクトをJavaの論理構造にしたがって表示するビュー

- ・パッケージごとにソースを表示
- ・ビルドパスをわかりやすく表示。ライブラリに含まれるクラス、ソースを簡単に探索
- ・Javaエディタとの同期

## ナビゲータビューとの違い

- ・ファイルシステム上の構造をそのまま表示
- ・Javaの論理構造にしたがって表示
















[デモ] [cap¥043\\_package\\_explorer.htm](#)

# JDTの基礎 –Javaパーспекティブ

## アウトラインビュー

クラス、フィールド、メソッドの構造を表示するためのビュー

- ・要素の種類、アクセス修飾子をアイコン別に表示
- ・Javaエディタ<->アウトラインビューの連携
- ・アウトラインビュー内でソースの編集も可能
- ・ソート・フィルタ表示(ex. アルファベット順、publicなものだけ表示)
- ・内部クラス、無名クラスが大量にある場合に便利

|  |  |
|--|--|
|  class (public)             |  default field (package visible)    |
|  interface (public)        |  private field                     |
|  enum type (public)       |  protected field                  |
|  annotation type (public) |  public field                     |
|  |  |
|  package visible class    |  default method (package visible) |
|  private class            |  private method                   |
|  protected class          |  protected method                 |
|  |  |
|  public method          |  |

# JDTの基礎 –Javaパースペクティブ

## 階層ビュー

クラス、インタフェースの階層構造を表示するビュー

- ・選択したクラスの親クラス、子クラスの関係を表示
- ・実装しているインタフェースの表示
- ・継承されたメンバーの表示

## 問題ビュー

プロジェクトで発生している問題、警告を表示するビュー

[デモ] [cap¥044\\_hierarchy.htm](#)

# JDTの基礎 –Javaコーディング支援機能

- ・Java構文の強調表示
- ・コードテンプレート
- ・コンテンツアシスト
- ・インクリメンタルビルドと連動したエラー・警告表示
- ・クイックフィックス

# JDTの基礎 –Javaコーディング支援機能

## コードテンプレート

- ・[Ctrl]+[Space]で起動
- ・定型コードを挿入できる
- ・可変部分は[tab]で書き換え

‘main’    public static void main(...){ }

‘syso’    System.out.println();

‘syse’    System.err.println();

‘syst’    System.out.println(<現在のクラス名、メソッド名>);

‘for’     for(int i=0; i<array.length; i++){....}

          for(Iterator itr = collection.iterator(); itr.hasNext()){...}

‘try’     try{ }catch(Exception e){ }

‘switch’    switch(..){ case x: break;.. default: break;}

‘pub’     publicメソッドの雛形

[デモ] [cap¥050\\_template.htm](#)

# JDTの基礎 –Javaコーディング支援機能

## コンテンツ・アシスト

- ・[Ctrl]+[Space]で起動（コードテンプレートと同様）
- ・コードの文脈に合わせて候補を表示、挿入
- ・メソッドを挿入した場合はカーソルは引数位置に移動するので便利
- ・コードテンプレート機能と組み合わせて、より少ないタイピングでコーディング。
- ・あまり頼りすぎると危険かも
- ・たまにはテキストエディタでやってみるのもイイかも

[デモ] [cap¥051\\_contentassist.htm](#)

# JDTの基礎 –Javaコーディング支援機能

## インクリメンタルビルド、エラー表示、クイックフィックス

- ・変更分のみバックグラウンドで再コンパイルされるので高速
- ・問題がある場合は、すぐにエラー、警告表示
  - エラー: 文法エラー、コンパイルエラー
  - 警告: 未使用変数の警告、コーディング形式の警告など
- ・エラー箇所で[Ctrl]+1を押せばクイックフィックス起動
  - 自動的に修正してくれる
  - つづり間違いの修正
  - 未定義メソッド呼び出しの修正

[デモ] [cap¥052\\_quickfix.htm](#)

# JDTの基礎 –Javaソースコード編集支援機能

## ソースコード編集

### ・javadocコメントの生成

- 「/\*」+[Enter]でjavadocを生成
- クラス、フィールド、メソッドに応じた形式で生成される

### ・コメントアウトの入力支援

- 範囲指定 「Ctrl+/'」で一括コメントアウト
- 再度実行でコメントアウトの除去

### ・ソースコードのフォーマット

- Ctrl+Shift+Fでソースコードを自動で整形 (format)
- 設定ダイアログ [java] [コードスタイル] [フォーマッター]で設定
- コミット前のフォーマットは要注意
- [Save Actions]で自動的にフォーマットも可能

**[デモ]** [cap¥053\\_comment.htm](#)

# JDTの基礎 –Javaソースコード編集支援機能

- ・import文の編成

- Ctrl+Shift+oでimport文を再編成 (organize)

- import文の挿入、種別ごとに並べ替え、未使用import文の削除

- ・Getter/Setterの生成

- フィールドのみを定義しておき、コンテキストメニュー [ソース]

- [GetterおよびSetterの生成]を選択

- 挿入されるjavadocの形式は設定可能(設定ダイアログ [Java]

- [コード・スタイル] [コード・テンプレート])

- ・メソッドのオーバーライド/実装支援

- コンテキストメニュー [ソース] [メソッドのオーバーライド・実装]

**[デモ]** [cap¥054\\_import\\_getter\\_override.htm](#)

# JDTの基礎 –Javaソースコード編集支援機能

- ・メソッドパラメータのアサイン
- メソッドパラメータを指定してCtrl+1
- メソッドパラメータをフィールドに代入してくれる

# JDTの基礎 –Javaソースコード編集支援機能

## リファクタリング支援

### ・名前変更

- クラス、メソッド、変数名にカーソルを合わせ、Ctrl+1 もしくは Ctrl+2+R もしくはAlt+Shift+R(Rename)で名前変更。
- コメント文中の該当行も一緒に変更可能

### ・メソッド、変数、の抽出

- メソッド内のコードを範囲選択 Alt+Shift+M (Method)で、選択部分をメソッドとして切り出し。
- 式を選択 Ctrl+1 もしくは Ctrl+Shift+L (Local)でローカル変数として切り出し。

### ・クリーンアップ

- 一括でリファクタリング
- コンテキストメニュー [ソース] [クリーンアップ]

[デモ] [cap¥054\\_refactor.htm](#)

# JDTの基礎 –Javaブラウジング支援機能

## ・宣言を開く

Java要素(クラス、メソッド、変数)にカーソルを合わせる [F3]

その宣言箇所が表示される

## ・参照の検索

Java要素にカーソルを合わせる Ctrl+Shift+G

メソッド呼び出し関係、クラス、変数の参照関係を一覧表示

## ・出現箇所の表示

Java要素にカーソルを合わせるだけ

そのJava要素を参照しているすべての箇所がハイライト表示

# JDTの基礎 –Javaブラウジング支援機能

- ・型を開く(.java、.class)

Ctrl+Shift+T (Type)で、「型を開く」ダイアログ表示 クラス、IF名を入力 クラス、IFのソースコードが開く

- ・リソースを開く(.java,.class以外)

Ctrl+Shift+R (Resource)で、「リソースを開く」ダイアログ表示 ファイル名を入力 リソースが開く

- ・指定行へジャンプ

Ctrl+L (Line) 行番号を入力 ジャンプ

- ・ソースコード・ファイルの検索

Ctrl+H 検索ダイアログが開く

# JDTの基礎 –Javaブラウジング支援機能

- ・クイックアウトライン

- Ctrl+O

- エディタ上で簡単アウトライン表示

- ・クイック・アクセス(Eclipse3.3新機能)

- Ctrl+3

- あらゆる機能にクイックアクセス！

- ・戻る、進む

- Alt+[ ], Alt+[ ]

まとめて[デモ] [cap¥055\\_navi.htm](http://cap¥055_navi.htm)

# JDTの基礎 –Javaブラウジング支援機能

## その他

### ・ローカルヒストリーとの比較・置換

- パッケージエクスプローラなどで、ファイルを選択 メニュー  
比較 ローカルヒストリー
- 対象ファイルの現在の状態と、以前の状態を比較できる
- ソースコード管理システムを使っていなくても単純な履歴管理が可能
- 一部もしくは全体を置換できる

### ・タスク、ブックマーク機能

- コード中に「TODO」「FIXME」「XXX」という特別なコメント文を記述すると、タスクビューに一覧表示される。
- 簡単タスク管理が可能
- エディタのルーラー領域で右クリック ブックマークで、その行をブックマークしておくことができる。

[デモ] [cap¥056\\_task.htm](#)

ちょっと一息



## キーボードショートカットをマスターするために

- ・覚えたら無理して使ってみる 手放せなくなる
- ・Windowsの場合は[Alt]キーも重要  
ex) Alt+W, Alt+Space+X, Alt+Enter, メニュー呼び出し  
などなど
- ・Eclipseは日本語化しないほうがよい？  
英語表示のほうがメニューの指定がラク

いざとなったらCtrl+Shift+L

ショーカットキー一覧表が表示されます。

# JDTの基礎 –Java実行・デバッグ支援機能

Eclipse上でJavaプログラムを実行できる。以下の2モードがある。

- ・Runモード
- ・Debugモード

## Runモード

プログラムを通常どおり起動する。一時停止や値の評価などは不可。  
動作確認や実行速度の測定などで使用

## Debugモード

Javaプログラムのデバッグが可能。ただしデバッグモード実行なので  
パフォーマンスは若干下がる。

# JDTの基礎 –Java実行・デバッグ支援機能

デバッグモードでは以下の機能が利用可能

- ・ブレイクポイントの設置によるプログラムの一時停止、ステップ実行
- ・変数、オブジェクトの状態のチェック
- ・式の評価

## ブレイクポイント

- ・Javaエディタのルーラ領域をダブルクリックすることでブレイクポイントマーカの設置が可能。
- ・デバッグモードで実行すると、その行の実行前にプログラムが一時停止する。
- ・ブレイクポイントマーカを右クリック [ブレイクポイント・プロパティ] で、ブレイクする条件を指定可能(例:n回ごとに停止、変数がnullだったら停止、など)。条件はJavaコードで定義できる。
- ・ブレイクポイントビューで一覧表示

# JDTの基礎 –Java実行・デバッグ支援機能

## ステップ実行の種類

- ・ Step Into : 呼び出しているメソッドに潜り込む
- ・ Step Over : 次の行を実行
- ・ Step Out : 呼び出し元メソッドに戻って実行
- ・ Drop To Frame: メソッド呼び出しのスタックを遡って実行

# JDTの基礎 –Java実行・デバッグ支援機能

## 変数(Variables)ビューによる変数値の調査

デバッグ中のプログラムの変数の値を調べることができる。  
オブジェクトの構造もわかる。  
値を書き換えることも可能。

## 表示(Display)ビューによる式の評価

デバッグ中のプログラムの状態を保持しながら、式の返値を調査したり、  
変数に対して操作ができる。「この瞬間に、このオブジェクトの  
xxxメソッドを呼び出したらどうなるか？」などを調べることができる。

まとめて[デモ] [cap¥057\\_debug.htm](http://cap¥057_debug.htm)

# WTPによるWebアプリケーション開発

# WTPが提供する機能

- ・Webプロジェクト(Dynamic, Static)作成機能
- ・Eclipse上からのサーバーの起動・終了の制御機能
- ・サーバーへのWebアプリのデプロイ機能
- ・Webアプリのデバッグサポート
- ・HTML,JSP,XML,CSSエディタ
- ・Webページエディタ
- ・デプロイメント・ディスクリプタ(web.xml)エディタ
- ・war,earエクスポート

# Webアプリケーション開発の大まかな流れ

1. 開発に用いるサーバーを設定
2. ワークスペースにサーバーを追加
3. Dynamic Webプロジェクトを作成
4. コーディング
5. WTP上でサーバーを起動しWebアプリの動作確認
6. warファイル作成

[デモ(設定編)] [cap¥060\\_wtp\\_setup.htm](#)

[デモ(開発編)] [cap¥061\\_wtp\\_dev.htm](#)

# その他の機能

## Web Page Editor

- ・WTP2.0で実験的に取り込まれた、WYSIWYGなHTML,JSPエディタ。
- ・Webページを視覚的に編集できる。
- ・動作が重く、表示も未完成なため、実用にはもうしばらくかかりそう。

## Data Source Explorer

- ・Eclipse上でDBを操作できる簡単なビュー
- ・DTPの機能

# WTPを使っていて困ったら<Tomcatを使っている場合>

## Webアプリがうまく動かない

とりあえずコンソールビューを確認。Tomcatがエラーを出力していないか？

### ダメなら

以下のフォルダにtomcatのエラーログが出力されるので確認

<workspaceフォルダ>/.metadata/.plugins/org.eclipse.wst.server.core/tmp0/logs/

デフォルトではtmp0フォルダにアプリがデプロイされる。変更したい場合は

[Server]ビューで、サーバーから一旦アプリを削除し、サーバーを開く

[Server Locations]で[use custom location]を選択して、任意のデプロイ先を指定

### まだダメなら

[Server]プロジェクト、[Server]ビュー内のサーバーを一旦削除、登録しなおしてみる

デプロイが失敗してデプロイ先が壊れた可能性あり

# 実プロジェクトでWTPを使用する際のポイント

- ・Tomcatは軽いので、コーディング 動作確認 コーディング …のサイクルが軽快
- ・ただし、Tomcatを本番機で使用することは稀(WebSphere,WebLogicなど商用サーバーが多く、ライセンスも必要)。開発中も本番と同じ環境で動作確認したい。

システムを小さい単位(モジュール・サブシステム)に分け、各担当者は自機のWTP+Tomcat上でさくさく開発。

ある程度出来上がったらテスト機(WebLogicなど)にデプロイ、動作確認。  
最終的に全てのモジュールを統合する。

特に大規模プロジェクトの場合は、モジュールの分け方、統合の仕方を工夫する必要がある。

# 例) とある大規模プロジェクトのEclipseプロジェクト構成

-mainプロジェクト (Dynamic Web Project)

全てのサブシステム、共通ライブラリは、統合用のビルドスクリプトにより、ここにコピーされる。

-subAプロジェクト (Dynamic web Project)

-subBプロジェクト (Dynamic web Project)

-subCプロジェクト (Dynamic web Project)

...

各サブシステムはそれぞれ独立したDynamic Web Project。サブシステムごとに、小規模なチームが割り当てられる。

-commonAプロジェクト (Java Project)

-commonBプロジェクト (Java Project)

共通ライブラリは単純なJava Project。JAR化され、各プロジェクトが参照する。

-ビルド用のプロジェクト (Simple Project)

-build.xml

各サブシステムをまとめる統合用のビルドスクリプトを作成する。テスト機へのデプロイなども一緒に行う。

# 実プロジェクトでWTPを使用する際のポイント

- ・WTPのみでWeb開発を行うのは結構厳しい。
- ・使用するフレームワークをサポートするEclipseプラグインを導入するのが効果的。ただしEclipse.orgでは特定のWebフレームワークを支援するプラグインは提供されていない。

・サードパーティー製のEclipseプラグイン

Struts・・・JBoss Tools, Amateras StrutsIDEなど

Spring・・・SpringIDEなど

Hibernate・・・JBoss Tools

# Eclipse TPTPで品質チェック

# Eclipse TPTP

- ・プログラムが動く「だけ」では不十分。
  - ・品質を確保して「ちゃんと」動かなくてはならない。
  - ・品質を確かめるには？      テスト
- 
- ・TPTP(Test and Performance Tools Platform) Project
    - ・テストやパフォーマンス測定ツールの基盤を提供
    - ・併せて、その上で動く標準的なツールも提供
    - ・<http://www.eclipse.org/tptp/>

TPTPでアプリの品質をチェックしてみましよう

# TPTPが提供する機能

- ・Static Analysis
- ・Unit Test
- ・Manual Test
- ・URL Test
- ・Profile

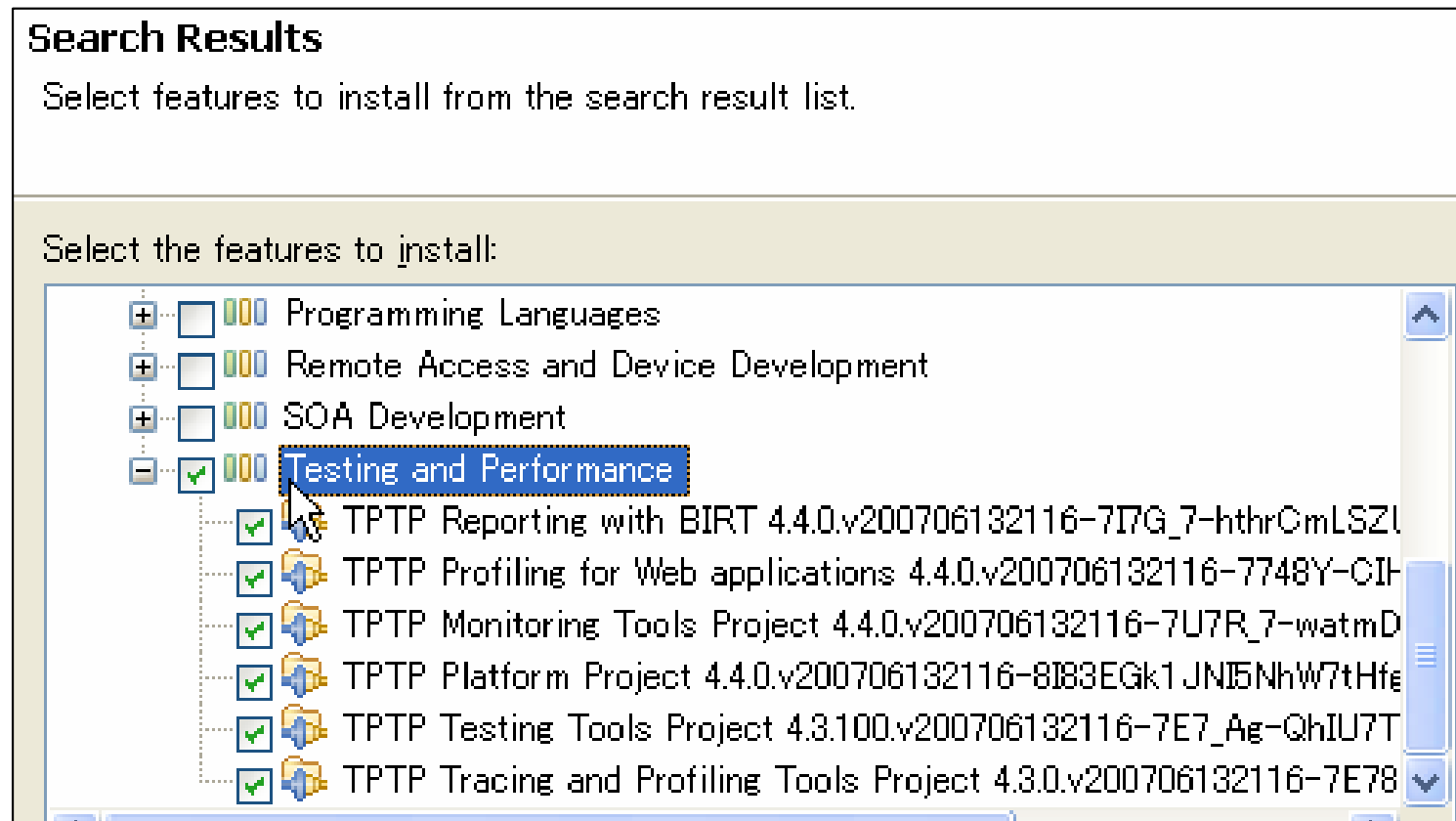
- ・Monitoring
- ・Log & Trace Analyzer
- ・Probekit
- ...

非常に広範囲

今日はこれらの機能を  
順に紹介していきます

# TPTPのインストール

- ・TPTPはデフォルトではEclipseに入っていない
- ・Europa Discovery Siteを使用してインストール  
[Testing and Performance]を選択



# TPTP Static Analysis

- ・Eclipse上のリソースを解析するための仕組みを提供
  - ・デフォルトではコードが標準的なコーディング規約に沿っているかチェックできる
- ・なぜチェックするのか？
- ・コーディングには開発者のクセが出る  
あまりに異なってくると保守が大変

[デモ] [cap¥072\\_static\\_analysis.htm](http://cap¥072_static_analysis.htm)

# TPTP Unit Test

- ・単体テスト(モジュールごとのテスト)を行う機能を提供
- ・Eclipseには標準でJUnitという単体テストのフレームワークが同封
- ・JUnitと比べて何が嬉しいの？
  - ・結果がファイルとして残る
  - ・複数の結果をレポートとして出力することができる
- ・TPTPで単体テストを行うには
  - ・TestSuiteという定義ファイルを作成してテストコードの骨組みを生成
  - ・逆にJUnitのコードからTestSuiteを生成する事もできる

[デモ] [cap¥074\\_unittest.htm](#)

# TPTP Manual Test

- ・手動テストを行う機能を提供
- ・手動テストとは？
  - ・単体テストで確認しづらい部分を補うテスト
    - ・ボタンを押した時の動作など
  - ・通常は紙に書かれた手順に従ってアプリケーションを操作して確認する
- ・TPTP Manual Testを使うと？
  - ・指定した内容がウィンドウに順次表示されていく  
見落としが無くなる
  - ・結果をTPTP Unit Testと同じように見ることができる

[デモ] [cap¥075\\_manual.htm](#)

# TPTP URL Test

- ・Webアプリケーションに対する負荷テストの機能を提供
  - ・ブラウザを操作してリクエストを記憶
  - ・記憶した内容を基にテストを作成する
- ・負荷テストを行うツールは沢山あるが...
  - ・TPTPだとEclipse上で実行できる
  - ・他のテストと同様のフォーマットで結果が残る

[デモ] [cap¥076\\_url.htm](#)

# Profile

- ・プロファイルを行うための機能を提供
- ・プロファイルとは？
  - ・プログラムが予想したスピードを出さない事が良くある  
実行時間やメモリの使用量などを取得して、問題点を特定する作業
- ・TPTPでは？
  - ・メソッドの呼び出し関係や、メモリの使用量などを見る事ができる

[デモ] [cap¥077\\_profile.htm](#)

# その他の機能

- ・Monitoring
  - ・OSのメモリ使用量といった、実行情報をリアルタイムに表示する機能を提供
- ・Log & Trace Analyzer
  - ・ログを読み込んで解析するための機能を提供
- ・Probekit
  - ・ログ出力といった処理をアプリケーションに組み込むための機能を提供

etc …

# Platformに込められた願い

- ・ Test and Performance Tools 「Platform」
  - ・ ツールの共通基盤
  - ・ 様々な拡張ポイントが公開されている

- ・ デフォルトのままでも十分使えるが、プラグインを自作して拡張すると、より使えるものになる。

プラグインの開発に挑戦して、開発スタイルに合ったツールを作成してみてください

[ Eclipse TPTP Project について (2005) ]

<http://www.hitachi-sk.co.jp/research/techdoc/TPTP/>

- ・ Log & Trace Analyzerの拡張方法など
- ・ TPTPでググると2位

# Mylyn+Subversive+tracでタスク指向開発

- Subversionによるバージョン管理
- Tracによるバグ管理
- Mylynでタスク指向開発！

# Subversionによるバージョン管理

- ・ 多人数のチームでアプリケーション開発を行う場合は、バージョン管理システムを使ってソースコードを管理するのが一般的。

## Subversionとは

- ・ 現在最も普及しているCVSの後継ともいえるSCMシステム 比較
- ・ SourceForge, ApacheなどOSSプロジェクトを中心に普及

本セミナーではSubversion本体のセットアップ手順については省略します。

## Subversion+Eclipse

- ・ Eclipse用のSubversionプラグイン・ Subversive、Subclipse
- ・ Subversiveは2006/11にEclipse Technology Projectに承認、移行予定  
Eclipseの今後のバージョンでは標準で組み込まれる予定



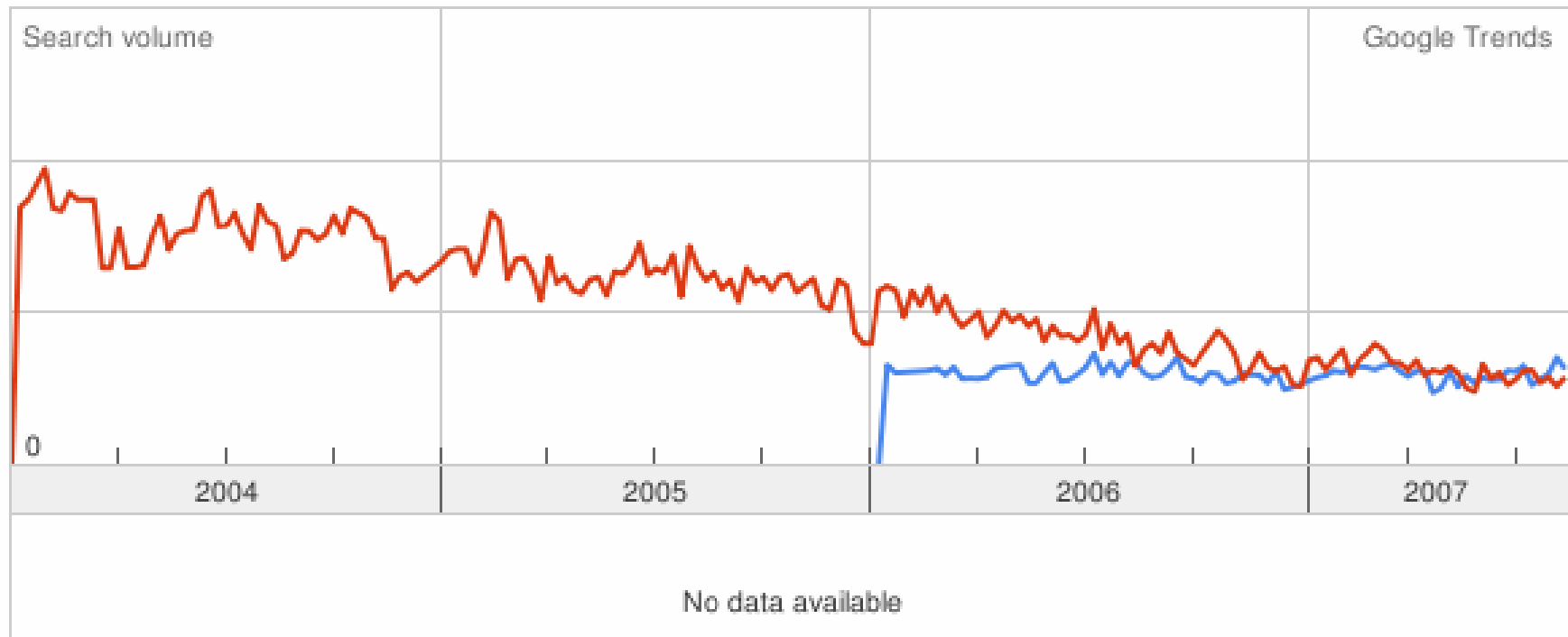
subversion repository ,cvs repository

Search Trends

Tip: You can compare searches by separating with commas.

## Trend history

● subversion repositor... ● cvs repository



# Subversiveの基本

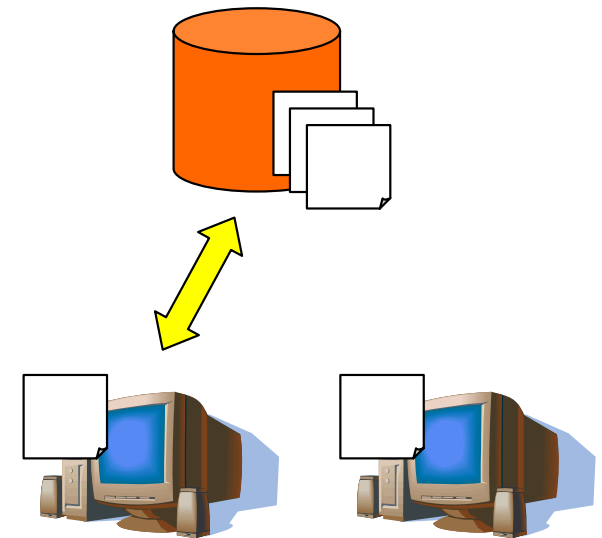
## Subversiveのインストール

更新サイト: <http://www.polarion.org/projects/subversive/download/1.1/update-site/>

## Subversiveを使ったおおまかな作業の流れ

1. SVNリポジトリビューに**SVNリポジトリ**を登録(初回のみ)
2. ワークスペースのプロジェクトをSVNリポジトリに**インポート**(初回のみ)
3. ワークスペースにプロジェクトを**チェックアウト**
- ~ ~
4. プロジェクト全体を**アップデート**
5. ローカルのソースコード(**作業コピー**)を修正
6. リポジトリとの**差分(diff)**をとって修正内容を確認
7. **コミットログ**を記述し、**コミット**

[デモ] [cap¥080\\_svn.htm](#)



# Subversiveの基本

## その他の機能

### [Show Resource History]

指定したファイル、フォルダの履歴(コミットログ、リビジョン、修正内容)を確認

### [Revert]

修正内容を取り消したいときは”Revert”を実行するとリポジトリの状態に復元

### [Cleanup]

updateやcommitがうまくいかない場合は”Cleanup”を実行。不要なロックを解除できる。

### [Lock]

VSSのように、自分が修正しているコードを他人に修正されないようにロックできる。

# バージョン管理システムとうまく付き合うには

- (1) コミット、アップデートはこまめに  
(単機能追加、1バグ修正の単位で)
- (2) 機能追加とリファクタリングを同時に行わない
- (3) 作業コピーのコピーはとらない
- (4) コミットは慎重に (修正箇所の再確認)
- (5) コミットログはきちんと書く

# BTSでバグ管理

## バグ管理システム(バグ追跡システム:BTS)

- 開発中のソフトウェアにどんなバグ・タスクがあるか
  - どれが修正済みで、どれが未着手か
  - だれが修正を担当するのか
  - いつまでに完了するのか
  - どれが重要でどれが重要でないのか
- を管理するシステム。

・バージョン管理システムと同じく、健全なソフトウェア開発プロジェクトのために非常に重要

# Tracの基礎

## Tracとは

- ・オープンソースのBTSのひとつ
- ・使いやすいWebのUI
- ・チケットでタスク・バグを管理、マイルストーンなどを設定可能
- ・Subversionとの連携が強力(チェンジセット、リポジトリレビューなど)
- ・pythonで開発されている
- ・様々なプラグインで機能拡張が可能

[デモ] [cap¥090\\_trac.htm](#)

単体で使っても非常に便利だが...

# Mylynでタスク指向開発！

## Mylyn(まいりん)とは

- ・**タスク指向**のユーザインターフェースをEclipseに付加するプラグイン
- ・Eclipse上での全ての作業を「**タスク**」として定義して管理
- ・現在自分が担当している**アクティブなタスク**に関連するリソース(プロジェクト・ソースコードなど)のみを表示
- ・あるタスクに関連するリソースを**コンテキスト**として保存、参照可能
- ・あるタスクの解決にかかった時間が記録される

# Mylynでタスク指向開発！

## Mylyn(まいりん)とは

- ・Bugzilla, Trac, JIRAなどのBTSに登録されたタスクを、MylynのタスクとしてEclipseに取り込むことが可能(ローカルにタスクを保存することもできる)。
- ・Eclipse上でBTSに登録されたタスクを閲覧、編集するためのUIも提供
- ・SubversionやCVSとの連携も可能

# Mylyn+Trac+Subversionの環境構築

## 1 . Subversion、Tracをサーバーにインストール

-Mylynのフル機能を使うにはXmlRpcPluginをTracに要インストール

## 2 . Mylyn+Trac Connectorをインストール

-Mylynは標準でインストール済み

-[Mylyn Connector: Trac]はEuropa Discovery siteからインストール

## 3 . Subversiveのインストール

## 4 . Subversive Mylyn Integrateをインストール

-必須ではないがSubversive+Mylyn連携は便利

# Mylynによるタスク指向開発の基本的な流れ

1. 開発中のソフトウェアにバグ発見！
2. [Task List]ビューで新規タスクを追加 Tracにチケットが自動登録
- …(1~2を繰り返す)
3. バグ修正に取り組む準備ができたなら、該当タスクの状態をアクティブにして、バグ修正開始！
4. 作業途中で別のタスクが割り込んできた場合は、一旦現時点のタスクをパッシブにして、別のタスクをアクティブに切り替える
5. バグ修正が完了したら、修正したソースをSubversionにコミット。  
このときコミットコメントに自動的にタスクのID(チケット番号)が挿入される！
6. コンテキスト情報をTracにアタッチ(添付)
7. タスクをクローズにする=Tracのチケットも自動的にクローズ
8. タスクをパッシブにする

# Mylynを使うと何がうれしいか？

- (1)全ての作業に対し「タスク」が割り当てられるため作業内容を明確化できる。
- (2)タスク1が終わったら、タスク2、タスク3、というパターンで仕事肅々と能率的に片付けていくことができる( GTD)。
- (3)コンテキストの仕組みにより、大量のソースコードの中からタスクに関連するものを即座に把握できる。
- (4)Subversionのコミットログに自動でTracのチケット番号を埋め込んでくれる(地味だが便利)。

参考:GTD(Getting Things Done)

David Allen氏が提唱した時間管理法。自分になすべき仕事を小さなタスクとしてリストに記録しておき、頭の中をすっきりさせた状態で効率的に仕事をする。Lifehackのひとつ。

# その他の注目eclipseサブプロジェクト

## Visual Editor

- ・AWT/Swing/SWT/Jface/EclipseRCPなどのGUIプログラミングを支援するプラグイン
- ・すごく高機能
- ・重い
- ・最近の開発が停滞。

## JSF/Dali-ORM (from WTP)

- ・JSF,JPA開発をサポートするプラグイン
- ・そろそろJSF

# その他の注目eclipseサブプロジェクト

## GMF (Graphical Editing Framework)

- ・Eclipse上でグラフィカルなエディタを開発するためのフレームワーク
- ・業務で使うことはまず無い
- ・でもかなり面白い

## DLTK (Dynamic Language ToolKit)

- ・動的言語のためのベースとなるプラグイン
- ・Ruby,Python,Tclの開発環境が実験的に公開されている。Perl,PHPにも対応予定
- ・流行りそう？

# Eclipseプロジェクト以外のおすすめプラグイン

## PMD

- ・コードの静的解析(CheckStyle, FindBugsと同等)
- ・コピーアンドペーストした可能性のあるコードを検知してくれる！

<http://pmd.sourceforge.net/eclipse>

# その他ハマりどころなど

## eclipseが起動しない/プラグインがインストールできない

[Error Log]ビューを確認。怪しいプラグインをインストール Eclipse起動時ロードに失敗 その他のプラグインがロードされずにEclipseが起動、ということもある。

## アップデートサイトに接続できない

ネットワークの設定を見直し。ネットワーク環境によってはプロキシの設定が必要な場合もある。([Preferences] [General] [Network Connection])

# 情報源

Eclipseのヘルプ!

かなり充実している。

Eclipse.org: Announcements, Community News

公式アナウンスやニュース <http://www.eclipse.org/>

Eclipse Resources

かなりまとまった技術文書 <http://www.eclipse.org/resources/>

Eclipse Plugin Central

サードパーティのプラグイン情報 <http://www.eclipseplugincentral.com/>

IBM developerWorks IBM's resource for developers - Japan

日本語記事多数 <http://www-06.ibm.com/jp/developerworks/>

EclipseWiki

日本のコミュニティ <http://www.eclipsewiki.net/eclipse/>

ブログ、ニュースサイト等

# おまけ:本の宣伝



「Eclipse 実践開発入門」

出版社: 技術評論社 (2007/5/15)

¥3,360円

大型本: 432ページ

商品の寸法: 22.8 x 18.4 x 2.6 cm

Amazon.co.jp ランキング: 本で47,968位

“Eclipse”で検索すると1位<sup>のときもある</sup>

- 1章 Eclipseとは
  - 2章 Eclipseのインストールおよびセットアップ
  - 3章 Eclipseの使い方
  - 4章 JDTによるJavaアプリケーション開発支援
  - 5章 JDTによるソースコードの編集支援
  - 6章 プロジェクトを開始する前に
  - 7章 WTPを使ったWebアプリケーション開発
  - 8章 TPTPを使ってプログラムの品質を上げる
  - 9章 AspectJ + AJDTによるアスペクト指向プログラミング
  - 10章 次世代Web開発 JSF, Ajax, JPA
- 付録A WTPを活用したWebアプリケーションの開発  
付録B キーボードショートカットを活用した高速開発

# ご清聴ありがとうございました！

ご意見、ご感想、質問等はこちら・・・

松前: kentaro.matsumae@gmail.com (<http://d.hatena.ne.jp/kenmaz>)

川尻: taken.kz@gmail.com

## 商標類

- ・ Javaおよび全てのJava関連の商標は、米国およびその他の国における米国 Sun Microsystems, Inc.の商標または登録商標です。
- ・ Apacheは、米国およびその他の国における米国Apache Software Foundationの商標です。
- ・ Microsoft、Visual Studio、Visual Source Safe、Windows は、米国 Microsoft Corporationの米国及びその他の国における登録商標または商標です。
- ・ Windowsの正式名称は、Microsoft Windows Operating Systemです。
- ・ その他記載の会社名、製品名は、それぞれの会社の商標もしくは登録商標です。