



# JTube

## Lightweight Support for Structuring and Documenting Java Source Code

December 13<sup>th</sup>, 2005  
Oberseminar "Methoden und Theorie der Software-Entwicklung"

Patrick Nepper  
([patrick.nepper@cdtm.de](mailto:patrick.nepper@cdtm.de))

## Agenda

### 1. Project Summary

- Project Description
- Overview of the Results

### 2. Problem Domain

- Our use cases
- Our motivation

### 3. JTube Features

- Method Groups
- Submethods
- External Documentation

## Project description



Working Title: **JTube: Lightweight Support for Structuring and Documenting Java Source Code**

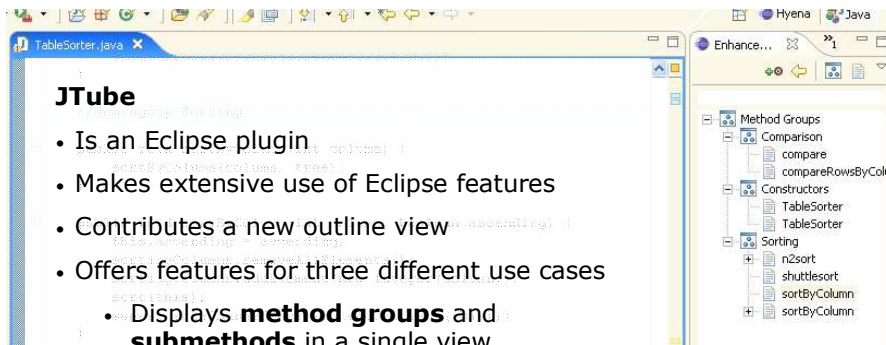
Purpose: Create an Eclipse Plugin which

- supports using Java comments for **grouping methods**,
- visualizes **submethod dependencies** and
- makes **Java source code** positions available for reference within **Hyena**.

Student: **Patrick Nepper**

Supervisor: **Axel Rauschmayer**

## Result: JTube, a lightweight Eclipse tool



### JTube

- Is an Eclipse plugin
- Makes extensive use of Eclipse features
- Contributes a new outline view
- Offers features for three different use cases
  - Displays **method groups** and **submethods** in a single view
  - Creates links between Java source code and **external documentation** (Hyena)

## We looked at three different use cases

### Method Groups

```
//----- Meta-Containers
private HyenaContainer _metaContainer;
public HyenaContainer getMetaContainer() {...}
//----- Containers
public HyenaContainer makeContainer(String path) {...}
public HyenaContainer makeContainer() {...}
```

### Sub-Methods

```
public void createControl(Composite parent) {
    _composite = new Composite(parent, SWT.NULL);
    createLeftToolBar(_composite);
    buildSearchField(_composite);
    createTopRightToolBar(_composite);
}
```

### External Doc

```
/**
 * @see review folder for pending review
 */
public int getSubjCount() {...}
```



## JTube is based on an Agile approach

### Features focus on the developer

- No new tools - extension of existing tool chain
- No reorientation necessary - matches common practices

### JTube's outline view

- Is easy to use
- Allows to filter information
  - Overview first
  - Zoom and filter
  - Then details-on-demand

## JTUBE visualizes method groups

The screenshot displays a Java IDE with two main panes. The left pane shows the source code of a class with several methods. The right pane shows a tree view of method groups.

```

public void addInfoEntry(String key, Object obj) {
    _info.put(key, obj);
    fireInfoChanged();
}

public Object getInfoEntry(String key) {
    return _info.get(key);
}

public Set<String> getInfoKeys() {
    return Collections.unmodifiableSet(_info.keySet());
}

private List<InfoListener> _infoListeners = new ArrayList<InfoListene:
public void addInfoListener(InfoListener listener) {
    _infoListeners.add(listener);
}

public void removeInfoListener(InfoListener listener) {
    _infoListeners.remove(listener);
}

private void fireInfoChanged() {
    for (InfoListener listener : _infoListeners) {
        listener.infoChanged();
    }
}

//----- Services
/**
 * Find a container that has a BeansService component in it
    
```

The tree view on the right shows a hierarchy of method groups:

- main
  - Container component management (no
    - registerComponentImplementation
  - Containers
    - getContainer
    - getContainer
    - getContainerNameTree
    - hasContainer
    - hasContainer
    - makeContainer
    - makeContainer
    - makeContainer
    - makeUniqueChildContainer
    - removeContainer
    - setNodeContent
  - Info: for broadcasting global informatio
    - addInfoEntry
    - addInfoListener
    - fireInfoChanged
    - getInfoEntry
    - getInfoKeys
    - removeInfoListener
  - Meta-Containers
    - getMetaComponent
    - getMetaContainer
  - Services
    - doDelete
    - doGet
    - doPost
    - doPut
    - findService

## JTUBE recognizes submethods

The screenshot displays a Java IDE with two main panes. The left pane shows the source code of a class with a public method and a private submethod. The right pane shows a tree view of method groups.

```

private void sort(Object sender) {
    // @category Sorting
    checkModel();

    compares = 0;
    // n2sort();
    // qsort(0, indexes.length-1);
    shufflesort((int[]) indexes.clone(), indexes, 0, indexes.length);
    // System.out.println("Compares: "+compares);
}

public void n2sort() {
    // @category Sorting
    for (int i = 0; i < getRowCount(); i++) {
        for (int j = i+1; j < getRowCount(); j++) {
            if (compare(indexes[i], indexes[j]) == -1) {
                swap(i, j);
            }
        }
    }
}
    
```

The tree view on the right shows a hierarchy of method groups:

- compare
- compareRowsBy
- Constructors
- TableSorter
- TableSorter
- Sorting
  - n2sort
  - shufflesort
  - sortByColumn
  - sortByColumn
  - sort

At the bottom, the IDE's 'Call Hierarchy' tab is active, showing members calling 'sort(Object)' in the workspace:

- sort(Object) - TableSorter
- sortByColumn(int, boolean) - TableSorter

## JTUBE teams-up with Hyena for external doc

The screenshot shows the JTube IDE interface. The top window displays the source code for `TableSorter.java`:

```
private void sort(Object sender) {
    // @category Sorting
    checkModel();

    compares = 0;
    // n2sort();
    // qsort(0, indexes.length-1);
    shufflesort((int[]) indexes.clone(), indexes, 0, indexes.length);
    // System.out.println("Compares: "+compares);
}
```

The bottom window shows the external documentation for the selected node:

Node: eh:/JHCI Shop Application/src/TableSorter.java/1343b651\_107eafc0ea8\_-7fff

Property	Value
jube:resource	"JHCI Shop Application/src/TableSorter.java"
jube:line	"192"
jube:marker_id	"1343b651_107eafc0ea8_-7fff"
jube:instance_of	jube:JTubeMarker

At the bottom of the IDE, there is a table with the following content:

Node	Triple	Prefixes	Filter: 0	Columns	Vodules	SubGraphs	Server: on	Prefs
------	--------	----------	-----------	---------	---------	-----------	------------	-------

Thanks for sharing a ride.