

CSeR - A Code Editor For Tracking & Visualizing Detailed Clone Differences

Ferosh Jacob, Advisor: Daqing Hou

June 2, 2009

Outline

- 1 Introduction
 - Why CSeR?
 - Example Scenario
 - With CSeR
- 2 Design
- 3 Validation
- 4 Conclusion & Future Work

What Is Missing In Current Copy-Paste Scenario?

Copy-Paste

Copy-Paste results in clones, Lack of tool support leads to following problems

- Implicit Relationship.
- Implicit Detailed Differences.
- ⇒ Time Consuming To Recover Detailed Differences Between Clones.
- ⇒ Guide For Similar Copy-Paste Operations in Future.

GUI Comparison of SetFilterWizardPage & ExclusionInclusionDialog

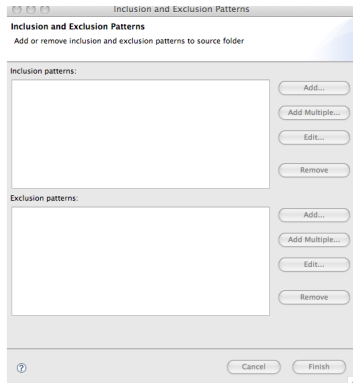
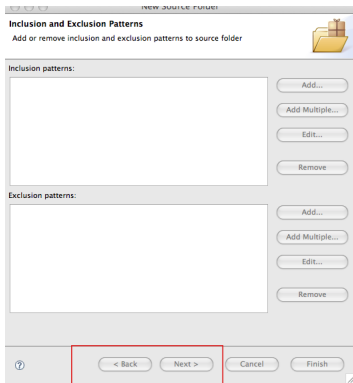


Figure: SetFilterWizardPage & ExclusionInclusionDialog

Detailed Differences: SetFilterWizardPage & ExclusionInclusionDialog



Figure: SetFilterWizardPage & ExclusionInclusionDialog

Source of ExclusionInclusionDialog

```
59 public class ExclusionInclusionDialog extends StatusDialog {
60     private static class ExclusionInclusionLabelProvider extends LabelProvider {
61
62         private Image fElementImage;
63
64         public ExclusionInclusionLabelProvider(ImageDescriptor descriptor) {
65             ImageDescriptorRegistry registry= JavaPlugin.getImageDescriptorRegistry();
66             fElementImage= registry.get(descriptor);
67         }
68
69         public Image getImage(Object element) {
70             return fElementImage;
71         }
72
73         public String getText(Object element) {
74             return BasicElementLabels.getFilePattern((String) element);
75         }
76     }
77 }
78
79 private ListDialogField fInclusionPatternList;
80 private ListDialogField fExclusionPatternList;
81 private CPLElement fCurrElement;
82 private IProject fCurrProject;
83
84 private IContainer fCurrSourceFolder;
85
86 private static final int IDX_ADD= 0;
87 private static final int IDX_ADD_MULTIPLE= 1;
88 private static final int IDX_EDIT= 2;
89 private static final int IDX_REMOVE= 4;
90
91
92 public ExclusionInclusionDialog(Shell parent, CPLElement entryToEdit, boolean focusOnExcluded) {
93     super(parent);
94     fCurrElement=entryToEdit;
95     setTitle(NewWizardMessages.ExclusionInclusionDialog_title);
```

Figure: ExclusionInclusionDialog Source

Tracking and Visualizing Differences

```
59 public class ExclusionInclusionDialog extends StatusDialog { 9 NewElementWizardPage
60     private static class ExclusionInclusionLabelProvider extends LabelProvider {
61
62         private Image fElementImage;
63
64         public ExclusionInclusionLabelProvider(ImageDescriptor descriptor) {
65             ImageDescriptorRegistry registry= JavaPlugin.getImageDescriptorRegistry();
66             fElementImage= registry.get(descriptor);
67         }
68
69         public Image getImage(Object element) {
70             return fElementImage;
71         }
72
73         public String getText(Object element) {
74             return BasicElementLabels.getFilePattern((String) element);
75         }
76     }
77     private static final String PAGE_NAME= "SetFilterWizardPage";
78
79     private ListDialogField fInclusionPatternList;
80     private ListDialogField fExclusionPatternList; 1
81     private CListElement fCurrElement;
82     private IProject fCurrProject;
83
84     private IContainer fCurrSourceFolder;
85
86     private static final int IDX_ADD= 0;
87     private static final int IDX_ADD_MULTIPLE; 2
88     private static final int IDX_EDIT= 2;
89     private static final int IDX_REMOVE= 3
90
91
92     public ExclusionInclusionDialog(6 Shell parent, 5 IContainer sourceFolder, CListElement entryToEdit, 4 boolean onFocusExcluded,
93         7 super(parent), 7 fCurrElement= entryToEdit,
94         setTitle(NewWizardMessages.ExclusionInclusionDialog_title); 8
95     --
```

Figure: SetFilterWizardPage & ExclusionInclusionDialog Differences

Outline

- 1 Introduction
- 2 Design
 - Requirements
 - Implementation
 - An Example
- 3 Validation
- 4 Conclusion & Future Work

Actions And Goals In Editing

- Actions.
 - ⇒ Backspace, Select And Delete, Refactoring tools.
- Goals.
 - ⇒ Renaming a variable, Inserting a field.

Usecases For CSeR

No	Use Case ID	Name
1	UC001	Consistent Tracking
2	UC002	Simple Name
3	UC003	Statements
4	UC004	Arguments
5	UC005	Parameters
6	UC006	Expressions
7	UC007	Comments
8	UC008	Keywords
9	UC009	Fields
10	UC010	Multiple Edit
11	UC011	Delete Operation
11	UC012	Conditional Statements

Implementation Highlights

While Copy And Paste

- Build AST 's.
- Find Ranges of AST's.
- Keep Correspondance Between Ranges.

While Editing

- Keep Track Of These Ranges.
- Build ASTs, Calculate Changes By Comparing AST's.
- Show Changes in Editor.

Correspondance

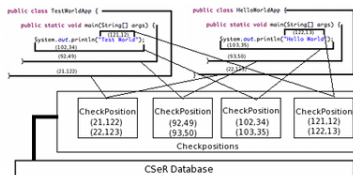


Figure: Correspondance Between HelloWorldApp & TestWorldApp

Keeping Correspondance While Editing

Table: HelloWorldApp & TestWorldApp Ranges

No	Range Description	Orginal	Current	Recent
1	Entire package name - package com. sample	0, 19	0, 19	0, 19
2	First part package - com	8, 3	8, 3	8, 3
3	Second part package - sample	12, 6	12, 6	12, 6
4	Class declaration - public class Hello..	21, 124	21, 123	21, 122
5	Class name - HelloWorldApp	34, 13	34, 12	34, 12
6	Method declaration - public static voi..	54, 89	53, 89	53, 88
7	Method name - main	73, 4	72, 4	72, 4
8	Parameters - String[] args	78, 13	77, 13	77, 13
9	Parameter Type name - String	78, 6	77, 6	77, 6
10	Parameter name - args	87, 4	86, 4	86, 4
11	Method block - { System. out. println(..	93, 50	92, 50	92, 49
12	Expression statement - Syst..World");	103, 34	102, 34	102, 33
13	Method invocation - Syst..elloWorld")	103, 33	102, 33	102, 32
14	Simple name - System	103, 6	102, 6	102, 6
15	Simple name - out	110, 3	109, 3	109, 3
16	Simple name -println	114, 7	113, 7	113, 7
17	String literal - "Hello World"	122, 13	121, 13	121, 12

Correspondance



Figure: Three Type Of Comparisons

Outline

- 1 Introduction
- 2 Design
- 3 Validation**
 - Robustness
 - Comparison With Existing Tools
 - Usefulness
- 4 Conclusion & Future Work

Editing Scenarios

No	Type	Goal Description	Action Description	Implemented
1	Names	Creating a name	Paste or Type	✓
2		Replacing part	Paste or Type	✓
3		Correcting typos	Backspace and Type	✓
4		Replacing name	Backspace, Type or Paste	✓
5		Removing name	Backspace, Delete or Type	✓
6		Splitting a name	Type in between	✓
7		Renaming	Using tools	×
8	Lists	Creating a new list	Type or Paste	✓
9		Inserting a new element	Type or Paste	✓
10		Removing an element	Delete, Type or Backspace	✓
11		Moving an element	Cut and Paste or Copy Paste and Delete	✓
12		Removing entire list	Backspace or Delete	✓
13		Flattening a list inside a list	Backspace or Delete	✓
14	Expressions	Inserting a new expression	Type or Paste	✓
15		Updating an expression	Type or Paste	✓
16		Removing an expression	Delete, Type or Backspace	✓
17		Moving an expression	Cut and Paste or Copy Paste and Delete	✓
18	Comments	Comment code	Type Line or Block comment	✓
19		Creating annotations		×
20		Inside expressions	Type Block comments	✓
21	Keywords	Insert keyword		×
22		Update keyword		×
23		modify keyword		×

Popular Source Comparison Tools

- Text-based.
- AST-based.

Text Based Tools

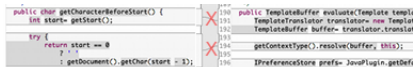
Popular Tools

- diff.
- CompareEditor In Eclipse.
- Version Editor.

In Comparison With CSeR.

- Text based.
- Unable To Distinguish Code And Comments.
- Unable To Find Moved Code.
- Wrong Correspondances.

Text Based Tools Example



```
public char getCharacterBeforeStart() {
    int start= getStart();

    try {
        return start == 0
            ? ...
            : getDocument().getCharCetart = 1);
    }

}

public Templatebuffer evaluate(Template templ
190 public Templatebuffer evaluate(Template templ
191     TemplateTranslator translator= new Templat
192     Templatebuffer buffer= translator.translat
193
194     getContextType().resolve(buffer, this);
195
196     IPreferenceStore prefs= JavaPlugin.getDefi
197
198 }
```

Figure: CompareEditor Showing Wrong Correspondances

AST Based Tools

Popular Tools

- Breakaway.
 - Changedistiller.
-
- Level Mismatch.
 - Batch Mode.
 - Breakaway - For Generalization Tasks
- ⇒ First Match Approach Based On Similarity Threshold.
- ⇒ UnOrdered & Ordered Mismatches.
- Changedistiller - Comparing Version Of A Class
- ⇒ Limited move operations.

AST Based Tools Example

```
protected void acceptSourceMethod(  
    IType type,  
    char[] selector,  
    char[][] parameterPackageNames,  
    char[][] parameterTypeNames  
)
```

→

```
protected void acceptSourceMethod(  
    IType type,  
    char[] selector,  
    char[][] parameterPackageNames,  
    char[][] parameterTypeNames,  
    boolean isDeclaration,  
    int start,  
    int end)
```

=

```
protected void acceptSourceMethod(  
    IType type,  
    char[] selector,  
    char[][] parameterPackageNames,  
    char[][] parameterTypeNames,  
    boolean isDeclaration,  
    int start,  
    int end)
```

Figure: ChangeDistiller And CSeR

Experiment Set Up

Experiment Goal

To recreate the scenario, the programmer went through while creating the classes.

- Identify Clones. Say FileA, FileB Are Clones
- Rename Second File, FileB → OriginalFileB
- Copy And Paste First File With Second File Name, Copy FileA & Paste As FileB
- Make Changes In Pasted File To Make It SecondFile, FileB=OriginalFileB

Experiments Results Overview

Eclipse (19), JavaLobby Community Platform (27), Literature (10)

No	Type Overall Distribution (517)	Description	Internal Distribution
1	Update (52 %, 268)	Variable Name (V_n)	50 %
2		Variable Type (V_t)	26 %
3		Literal (L)	9 %
4		Method (M)	13 %
5		Other (O)	<1 %
6	Delete (32 %, 165)	Statement (S)	40 %
7		Method Declaration (M)	30 %
8		Field Declaration (F)	20 %
9		Expression (E)	8 %
10		Parameter (P)	2 %
11		Class Declaration (C)	< 1 %
12	Insert (14, 72 %)	Statement (S)	37 %
16		Field Declaration (F)	37 %
15		Method Declaration (M)	16 %
13		Expression (E)	4 %
14		Parameter (P)	4 %
17		Class Declaration (C)	2 %
18	Move (2 %, 12)	Method Declaration (M)	58 %
19		Statement (S)	33 %
20		Class Declaration (C)	8 %

Outline

- 1 Introduction
- 2 Design
- 3 Validation
- 4 Conclusion & Future Work
 - Conclusion

Conclusion & Future Work

Conclusion

- Code Editor For Tracking And Visualizing Changes Continously.
- Different From Existing Tools.
- Useful, Robust And Unique.

Possible Extensions In CSeR

- Clone Groups.
- Tracking Code And Identifying Templates.
- Version Control Integration.

Thank You

Questions ?