

RefactorHub:

A Commit Annotator for Refactoring



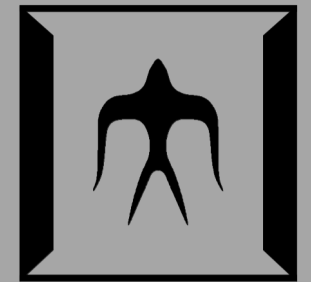
[github.com/
salab/RefactorHub](https://github.com/salab/RefactorHub)

Ryo
Kuramoto

Motoshi
Saeki

Shinpei
Hayashi

Tokyo Tech.
Japan



Empirical studies on refactoring

We need
refactoring data

Google Scholar

refactoring "empirical study"

Articles

About 12,200 results (0.04 sec)

Any time

Since 2021

Since 2020

Since 2017

Custom range...

Sort by relevance

Sort by date

☐ include patents

☒ include citations

Create alert

Search-based refactoring: an empirical study

M O'Keeffe, MÓ Cinnéide - Journal of Software Maintenance ..., 2008 - Wiley Online Library

Object-oriented systems that undergo repeated addition of functionality commonly suffer a loss of quality in their underlying design. This problem must often be remedied in a costly **refactoring** phase before further maintenance programming can take place. Recently search ...

☆ Cited by 158 Related articles All 2 versions

When does a refactoring induce bugs? an empirical study

G Bavota, B De Carluccio, A De Lucia... - 2012 IEEE 12th ..., 2012 - ieeexplore.ieee.org

Refactorings are-as defined by Fowler-behavior preserving source code transformations. Their main purpose is to improve maintainability or comprehensibility, or also reduce the code footprint if needed. In principle, refactorings are defined as simple operations so that ...

☆ Cited by 150 Related articles All 10 versions

[PDF] A multidimensional empirical study on refactoring activity.

N Tsantalis, V Guana, E Stroulia, A Hindle - CASCON, 2013 - users.encs.concordia.ca

In this paper we present an **empirical study** on the **refactoring** activity in three well-known projects. We have studied five research questions that explore the different types of refactorings applied to different types of sources, the individual contribution of team ...

☆ Cited by 59 Related articles All 11 versions

The impact of refactoring changes on the szz algorithm: An empirical study

EC Neto, DA Da Costa... - 2018 IEEE 25th ..., 2018 - ieeexplore.ieee.org

SZZ is a widely used algorithm in the software engineering community to identify changes that are likely to introduce bugs (ie, bug-introducing changes). Despite its wide adoption, SZZ still has room for improvements. For example, current SZZ implementations may still flag ...

☆ Cited by 25 Related articles All 2 versions

An empirical study to improve software security through the application of code refactoring

H Mumtaz, M Alshayeb, S Mahmood, M Niazi - Information and Software ..., 2018 - Elsevier

Context Code bad smells indicate design flaws that can degrade the quality of software and can potentially lead to the introduction of faults. They can be eradicated by applying **refactoring** techniques. Code bad smells that impact the security perspective of software ...

☆ Cited by 25 Related articles All 6 versions

Behind the intents: An in-depth empirical study on software refactoring in modern code review

M Paixão, A Uchôa, AC Bibiano, D Oliveira... - Proceedings of the 17th ..., 2020 - dl.acm.org

Code refactorings are of pivotal importance in modern code review. Developers may preserve, revisit, add or undo refactorings through changes' revisions. Their goal is to certify that the driving intent of a code change is properly achieved. Developers' intents behind ...

☆ Cited by 9 Related articles All 5 versions

Empirical study on refactoring large-scale industrial systems and its effects on maintainability

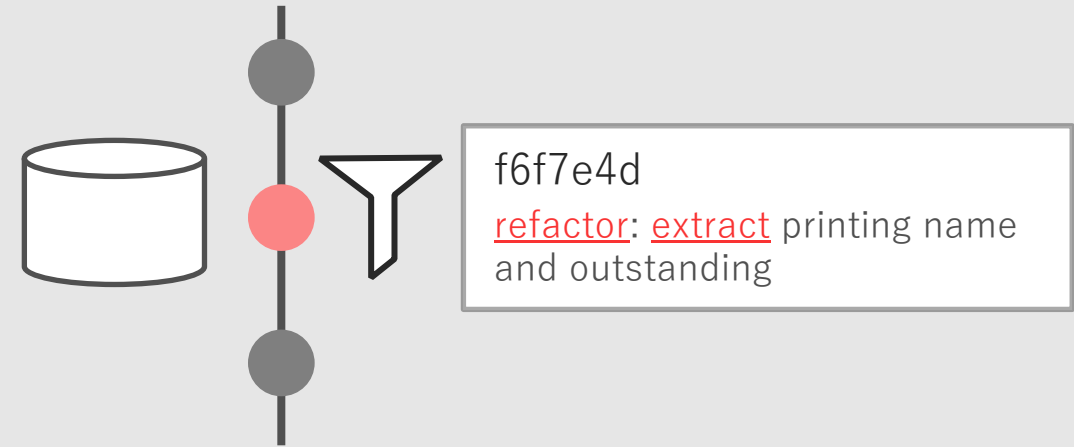
G Szőke, G Antal, C Nagy, R Ferenc... - Journal of Systems and ..., 2017 - Elsevier

How to collect refactoring data



Using detection tools
(e.g., RefactoringMiner, RefDiff)

- ✓ Accuracy
- ✗ Type coverage
- ✓ Detail information



Using self-affirmed commits
(commit message filtering)

- ✗ Accuracy
- ✓ Type coverage
- ✗ Detail information



RefactorHub:

A Commit Annotator for Refactoring

Annotation Approach



RefactorHub

```
1 public class Printer {
2     void printOwing() {
3         pringBanner();
4
5         println("name: " + name);
6         println("amount: " + getOutstanding());
7     }
8 }
```

extracted code

```
1 public class Printer {
2     void printOwing() {
3         pringBanner();
4         printDetail(getOutstanding());
5     }
6     void printDetail(double outstanding) {
7         println("name: " + name);
8         println("amount: " + outstanding);
9     }
10 }
```

invocation

extracted method

Input (Commit)

f6f7e4d

refactor: extract printing name and
outstanding

✗ Accuracy
✗ Detail information

Output (JSON)

```
{
  "type": "ExtractMethod",
  "extracted method": {
    "type": "MethodDeclaration",
    "path": "src/main/Printer.java",
    "range": {
      "startLine": 6, "startColumn": 2,
      "endLine": 9, "endColumn": 3
    }
  }
}
```

✓ Accuracy
✓ Detail information

Refactoring info.

refactor
Extract Method

Description

Extract Method package clearCallbacksAndListener() : void extracted from private checkCurrentDimens() : void in class

com.bumptech.glide.request.target.ViewTarget.SizeDeterminer

bumptech / glide / [0d4b279](#)

Clear callbacks from ViewTargets in onLoadCleared.

Created by MOE: <http://code.google.com/p/moe-java>

MOE_MIGRATED_REVID=95763399

judds committed on 2015/6/12 4:11:07

Refactoring
parameters
(before refactoring)

Refactoring
parameters
(after refactoring)

```
86 int currentHeight = getViewHeightOrParam();
87 if (!isSizeValid(currentWidth) || !isSizeValid(currentHeight)) {
88     return;
89 }
90 notifyCbs(currentWidth, currentHeight);
91 // Keep a reference to the layout listener and remove it here
92 // because the observer remove itself because the observer
93 // we add the listener to will be almost immediately merged into
94 // another observer and will therefore never be alive. If we instead
95 // get a reference to the listener and remove it here, we get the
96 // current view tree observer and should succeed.
97 ViewTreeObserver observer = view.getViewTreeObserver();
98 if (observer.isAlive()) {
99     observer.removeOnPreDrawListener(layoutListener);
100 }
101 layoutListener = null;
102 }
103
104 public void getSize(SizeReadyCallback cb) {
105     int currentWidth = getViewWidthOrParam();
106     int currentHeight = getViewHeightOrParam();
107     if (isSizeValid(currentWidth) && isSizeValid(currentHeight)) {
108         cb.onSizeReady(currentWidth, currentHeight);
109     } else {
110         // We want to notify callbacks in the order they were added and we only expect one
111         // callbacks to
112         // be added a time, so a List is a reasonable choice.
113         if (!cbs.contains(cb)) {
114             cbs.add(cb);
115         }
116     }
117     if (layoutListener == null) {
118         final ViewTreeObserver observer = view.getViewTreeObserver();
119         observer.addOnPreDrawListener(new Runnable() {
120             @Override public void run() {
121                 // We want to notify callbacks in the order they were added and we only expect one
122                 // callbacks to
123                 // be added a time, so a List is a reasonable choice.
124                 if (!cbs.contains(cb)) {
125                     cbs.add(cb);
126                 }
127                 cb.onSizeReady(currentWidth, currentHeight);
128             }
129         });
130     }
131 }
```

```
194 int currentHeight = getViewHeightOrParam();
195 if (!isSizeValid(currentWidth) || !isSizeValid(currentHeight)) {
196     return;
197 }
198 notifyCbs(currentWidth, currentHeight);
199 clearCallbacksAndListener();
200 }
201
202 void getSize(SizeReadyCallback cb) {
203     int currentWidth = getViewWidthOrParam();
204     int currentHeight = getViewHeightOrParam();
205     if (isSizeValid(currentWidth) && isSizeValid(currentHeight)) {
206         cb.onSizeReady(currentWidth, currentHeight);
207     } else {
208         // We want to notify callbacks in the order they were added and we only expect one
209         // callbacks to
210         // be added a time, so a List is a reasonable choice.
211         if (!cbs.contains(cb)) {
212             cbs.add(cb);
213         }
214     }
215     if (layoutListener == null) {
216         final ViewTreeObserver observer = view.getViewTreeObserver();
217         observer.addOnPreDrawListener(new Runnable() {
218             @Override public void run() {
219                 // We want to notify callbacks in the order they were added and we only expect one
220                 // callbacks to
221                 // be added a time, so a List is a reasonable choice.
222                 if (!cbs.contains(cb)) {
223                     cbs.add(cb);
224                 }
225                 cb.onSizeReady(currentWidth, currentHeight);
226             }
227         });
228     }
229 }
```

Code difference

Modified files

Element name

ElementType

multiple

...m/bumptech/glide/request/target/ViewTarget.java

...umptech/glide/request/target/ImageViewTarget.java

...m/bumptech/glide/request/target/ViewTarget.java



...umptech/glide/request/target/ImageViewTarget.java



...m/bumptech/glide/request/target/ViewTarget.java

After Elements

* extracted code

Code Fragment

Add Location

* invocation

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Method Declaration

Annotating Extract Method

Before:

- extracted code
- target method

After:

- extracted code
- invocation
- extracted method
- target method

Refactoring Type

Extract Method

Description

Extract Method protected setMaxHeight(maxHeight int) : void extracted from protected wrapMenuBar(menuBar ToolbarMenuBar) : Widget in class

rstudio / rstudio / [cb49e43](#)

allow dynamic max height for scrollable toolbar popup menu

JJ Allaire committed on 2015/6/8 21:53:53

Before Elements

* extracted code

☐ CodeFragment[]

Add Location

target method

☐ MethodDeclaration

~

element name

ElementType

☐ multiple

Add Element

Lock

```
1 /*
2  * ScrollableToolbarPopupMenu.java
3  *
4  * Copyright (C) 2009-12 by RStudio, Inc.
5  *
6  * Unless you have received this program directly from RStudio,
7  * to the terms of a commercial license agreement with RStudio,
8  * this program is licensed to you under the terms of version 3
9  * of the GNU Affero General Public License. This program is distributed
10 * WITHOUT ANY EXPRESS OR IMPLIED WARRANTY, INCLUDING THOSE OF NON-
11 * MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Please refer to the
12 * AGPL (http://www.gnu.org/licenses/agpl-3.0.txt) for more details.
13 *
14 */
15 package org.rstudio.core.client.widget;
16
17 import com.google.gwt.dom.client.Style.Overflow;
18 import com.google.gwt.event.logical.shared.HasSelectionHandlers;
19 import com.google.gwt.event.logical.shared.SelectionEvent;
20 import com.google.gwt.event.logical.shared.SelectionHandler;
21 import com.google.gwt.event.shared.HandlerRegistration;
22 import com.google.gwt.user.client.ui.*;
23 import org.rstudio.core.client.dom.DomUtils;
24 import org.rstudio.core.client.theme.res.ThemeStyles;
25 import org.rstudio.core.client.widget.ToolbarPopupMenu;
26
27 public class ScrollableToolbarPopupMenu extends ToolbarPopupMenu
28 {
29     @Override
```

...e/client/widget/ScrollableToolbarPopupMenu.java

Lock

```
1 /*
2  * ScrollableToolbarPopupMenu.java
3  *
4  * Copyright (C) 2009-12 by RStudio, Inc.
5  *
6  * Unless you have received this program directly from RStudio,
7  * to the terms of a commercial license agreement with RStudio,
8  * this program is licensed to you under the terms of version 3
9  * of the GNU Affero General Public License. This program is distributed
10 * WITHOUT ANY EXPRESS OR IMPLIED WARRANTY, INCLUDING THOSE OF NON-
11 * MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Please refer to the
12 * AGPL (http://www.gnu.org/licenses/agpl-3.0.txt) for more details.
13 *
14 */
15 package org.rstudio.core.client.widget;
16
17 import com.google.gwt.dom.client.Style.Overflow;
18 import com.google.gwt.event.logical.shared.HasSelectionHandlers;
19 import com.google.gwt.event.logical.shared.SelectionEvent;
20 import com.google.gwt.event.logical.shared.SelectionHandler;
21 import com.google.gwt.event.shared.HandlerRegistration;
22 import com.google.gwt.user.client.ui.*;
23 import org.rstudio.core.client.dom.DomUtils;
24 import org.rstudio.core.client.theme.res.ThemeStyles;
25 import org.rstudio.core.client.widget.ToolbarPopupMenu;
26
27 public class ScrollableToolbarPopupMenu extends ToolbarPopupMenu
28 {
29     @Override
```

...e/client/widget/ScrollableToolbarPopupMenu.java

After Elements

* extracted code

☐ CodeFragment[]

Add Location

invocation

☐ MethodInvocation

~

extracted method

☐ MethodDeclaration

~

target method

☐ MethodDeclaration

~

element name

ElementType

☐ multiple

Add Element

Annotating Extract Method

Before:

- extracted code
- target method

After:

- extracted code
- invocation
- extracted method
- target method

SaveDiscard

Extract Method

Extract Method protected setMaxHeight(maxHeight int) : void extracted from protected wrapMenuBar(menuBar ToolbarMenuBar) : Widget in class org.rstudio.core.client.widget.ScrollableToolBarPopupMenu

Before Elements

* extracted code

CodeFragment[]

Add Location

target method

MethodDeclaration

element name

ElementType

☐ multiple

Add Element

Lock

```
59 public int getSelectedIndex()
60 {
61     return menuBar_.getSelectedIndex();
62 }
63
64 @Override
65 protected Widget wrapMenuBar(ToolBarMenuBar menuBar)
66 {
67     scrollPanel_ = new ScrollPanel(menuBar);
68     scrollPanel_.addStyleName(ThemeStyles.INSTANCE.scrollPanel);
69     scrollPanel_.getElement().getStyle().setOverflowY(Overflow.SCROLL);
70     scrollPanel_.getElement().getStyle().setOverflowX(Overflow.SCROLL);
71     scrollPanel_.getElement().getStyle().setProperty("maxHeight", getMaxHeight());
72     return scrollPanel_;
73 }
74
75 protected int getMaxHeight()
76 {
77     return 300;
78 }
79
80
```

```
59 public int getSelectedIndex()
60 {
61     return menuBar_.getSelectedIndex();
62 }
63
64 @Override
65 protected Widget wrapMenuBar(ToolBarMenuBar menuBar)
66 {
67     scrollPanel_ = new ScrollPanel(menuBar);
68     scrollPanel_.addStyleName(ThemeStyles.INSTANCE.scrollPanel);
69     scrollPanel_.getElement().getStyle().setOverflowY(Overflow.SCROLL);
70     scrollPanel_.getElement().getStyle().setOverflowX(Overflow.SCROLL);
71     setMaxHeight(getMaxHeight());
72     return scrollPanel_;
73 }
74
75 protected int getMaxHeight()
76 {
77     return 300;
78 }
79
80+ protected void setMaxHeight(int maxHeight)
81+ {
82+     scrollPanel_.getElement().getStyle().setProperty("maxHeight",
83+         maxHeight + "px");
84+ }
85+
```

```
81 protected class ScrollableToolBarMenuBar extends ToolbarMenuBar
82     implements HasSelectionHandlers<MenuItem>
83 {
84     public ScrollableToolBarMenuBar(boolean vertical)
85     {
86         super(vertical);
87     }
88
89     public HandlerRegistration addSelectionHandler(Handler<MenuItem> handler)
```

...e/client/widget/ScrollableToolBarPopupMenu.java

Lock

```
59 public int getSelectedIndex()
60 {
61     return menuBar_.getSelectedIndex();
62 }
63
64 @Override
65 protected Widget wrapMenuBar(ToolBarMenuBar menuBar)
66 {
67     scrollPanel_ = new ScrollPanel(menuBar);
68     scrollPanel_.addStyleName(ThemeStyles.INSTANCE.scrollPanel);
69     scrollPanel_.getElement().getStyle().setOverflowY(Overflow.SCROLL);
70     scrollPanel_.getElement().getStyle().setOverflowX(Overflow.SCROLL);
71     setMaxHeight(getMaxHeight());
72     return scrollPanel_;
73 }
74
75 protected int getMaxHeight()
76 {
77     return 300;
78 }
79
80+ protected void setMaxHeight(int maxHeight)
81+ {
82+     scrollPanel_.getElement().getStyle().setProperty("maxHeight",
83+         maxHeight + "px");
84+ }
85+
```

```
86 protected class ScrollableToolBarMenuBar extends ToolbarMenuBar
87     implements HasSelectionHandlers<MenuItem>
88 {
89     public ScrollableToolBarMenuBar(boolean vertical)
90     {
91         super(vertical);
92     }
93
94     public HandlerRegistration addSelectionHandler(Handler<MenuItem> handler)
```

...e/client/widget/ScrollableToolBarPopupMenu.java

After Elements

* extracted code

CodeFragment[]

Add Location

invocation

MethodInvocation

extracted method

MethodDeclaration

target method

MethodDeclaration

element name

ElementType

☐ multiple

Add Element

Annotating Extract Method

Before:

- extracted code
- target method

After:

- extracted code
- invocation
- extracted method
- target method

Save

Discard

Extract Method

Extract Method protected setMaxHeight(maxHeight int) : void extracted from protected wrapMenuBar(menuBar ToolbarMenuBar) : Widget in class org.rstudio.core.client.widget.ScrollableToolBarPopupMenu

Before Elements

* extracted code

CodeFragment[]

src/gwt/src/org/rstudio/c

71 : 7 ~ 72 : 78

Add Location

target method

MethodDeclaration

wrapMenuBar

src/gwt/src/org/rstudio/c

64 : 4 ~ 74 : 4

element name

ElementType

☐ multiple

Add Element

Lock

59 public int getSelectedIndex()
60 {
61 | return menuBar_.getSelectedIndex();
62 }
63
64 @Override
65 protected Widget wrapMenuBar(ToolBarMenuBar menuBar)
66 {
67 | scrollPanel_ = new ScrollPanel(menuBar);
68 | scrollPanel_.addStyleName(ThemeStyles.INSTANCE.scrollPanelStyle);
69 | scrollPanel_.getElement().getStyle().setOverflowY(Overflow.SCROLL);
70 | scrollPanel_.getElement().getStyle().setOverflowX(Overflow.SCROLL);
71 | scrollPanel_.getElement().getStyle().setProperty("maxHeight", getMaxHeight());
72 | return scrollPanel_;
73 }
74
75
76 protected int getMaxHeight()
77 {
78 | return 300;
79 }
80

...e/client/widget/ScrollableToolBarPopupMenu.java

Lock

59 public int getSelectedIndex()
60 {
61 | return menuBar_.getSelectedIndex();
62 }
63
64 @Override
65 protected Widget wrapMenuBar(ToolBarMenuBar menuBar)
66 {
67 | scrollPanel_ = new ScrollPanel(menuBar);
68 | scrollPanel_.addStyleName(ThemeStyles.INSTANCE.scrollPanelStyle);
69 | scrollPanel_.getElement().getStyle().setOverflowY(Overflow.SCROLL);
70 | scrollPanel_.getElement().getStyle().setOverflowX(Overflow.SCROLL);
71+ | scrollPanel_.getElement().getStyle().setProperty("maxHeight", getMaxHeight());
72 | return scrollPanel_;
73 }
74
75 protected int getMaxHeight()
76 {
77 | return 300;
78 }
79
80
80+ protected void setMaxHeight(int maxHeight)
81+ {
82+ | scrollPanel_.getElement().getStyle().setProperty("maxHeight", maxHeight + "px");
83+ |
84+ }
85+
86 protected class ScrollableToolBarMenuBar extends ToolbarMenuBar
87 | implements HasSelectionHandlers<MenuItem>
88 {
89 | public ScrollableToolBarMenuBar(boolean vertical)
90 | {
91 | | super(vertical);
92 | }
93
94 | public HandlerRegistration addSelectionHandler(Handler<MenuItem> handler)
95 | {
96 | | addHandler(handler, this);
97 | }
98 }
99

...e/client/widget/ScrollableToolBarPopupMenu.java

After Elements

* extracted code

CodeFragment[]

src/gwt/src/org/rstudio/c

82 : 7 ~ 83 : 30

Add Location

invocation

MethodInvocation

extracted method

MethodDeclaration

setMaxHeight

src/gwt/src/org/rstudio/c

80 : 4 ~ 84 : 4

target method

MethodDeclaration

element name

ElementType

☐ multiple

Add Element

RefactorHub:

A Commit Annotator for Refactoring



Fork me on GitHub



github.com/
salab/**RefactorHub**



Ryo
Kuramoto



Motoshi
Saeki



Shinpei
Hayashi

Tokyo Tech.
Japan

