Solution 1

- On the console execute the following commands:

- ```
  val p = org.opalj.br.analyses.Project(new java.io.File("/Library/
  Java/JavaVirtualMachines/jdk1.8.0_77.jdk/Contents/Home/jre/lib/
  rt.jar"))
  ```

- ```
  import org.opalj.br.instructions._
  def hasUselessIf(body : org.opalj.br.Code): Boolean={
    body.collectWithIndex{
      case (pc, i : SimpleConditionalBranchInstruction) if
        i.branchoffset == i.length => pc
    }.nonEmpty
  }
  ```

- ```
  println(p.allProjectClassFiles.flatMap(_.methods.filter(m =>
  m.body.isDefined && hasUselessIf(m.body.get)  )).map(m =>
  m.toJava(p.classFile(m))).mkString("\n"))
  ```
  ```

Results

- javax.swing.JInternalFrame{ public void setClosed(boolean) }
- javax.swing.ToolTipManager{ public void mouseExited(java.awt.event.MouseEvent) }
- sun.awt.im.ExecutableInputMethodManager{ void setInputContext(sun.awt.im.InputContext) }
- sun.security.pkcs.PKCS8Key{ public void decode(java.io.InputStream) }
- sun.security.provider.AuthPolicyFile{ private void initPolicyFile() }
- sun.nio.fs.UnixChannelFactory{ protected java.io.FileDescriptor open(int,sun.nio.fs.UnixPath,java.lang.String,sun.nio.fs.UnixChannelFactory$Flags,int) }
- sun.font.CMap$CMapFormat4{ char getGlyph(int) }
- javax.swing.text.html.parser.DTD{ public void read(java.io.DataInputStream) }
- com.sun.xml.internal.ws.model.SOAPSEIModel{ protected void populateMaps() }
- com.sun.xml.internal.ws.model.JavaMethodImpl{ private void setWsaActions(com.sun.xml.internal.ws.api.databinding.MetadataReader) }
- com.sun.xml.internal.ws.model.ExternalMetadataReader$Util{ public com.oracle.xmlns.internal.webservices.jaxws_databinding.JavaWsdlMappingType read(javax.xml.transform.Source,boolean,boolean) }
- com.sun.xml.internal.ws.handler.ServerSOAPHandlerTube{ public void <init>(com.sun.xml.internal.ws.api.WSBinding,com.sun.xml.internal.ws.api.model.wsdl.WSDLPort,com.sun.xml.internal.ws.api.pipe.Tube) }
- com.sun.xml.internal.ws.handler.ClientSOAPHandlerTube{ public void <init>(com.sun.xml.internal.ws.api.WSBinding,com.sun.xml.internal.ws.api.model.wsdl.WSDLPort,com.sun.xml.internal.ws.api.pipe.Tube) }
- com.sun.xml.internal.ws.api.message.Packet{ public com.oracle.webservices.internal.api.message.ContentType getContentType() }
- com.sun.xml.internal.ws.addressing.EndpointReferenceUtil{ private javax.xml.ws.wsaddressing.W3CEndpointReference toW3CEpr(com.sun.xml.internal.ws.developer.MemberSubmissionEndpointReference) }
- com.sun.xml.internal.messaging.saaj.soap.MessageImpl{ protected void <init>(javax.xml.soap.SOAPMessage) }
- com.sun.xml.internal.messaging.saaj.soap.SOAPPartImpl{ protected com.sun.xml.internal.messaging.saaj.util.XMLDeclarationParser lookForXmlDecl() }
- com.sun.xml.internal.messaging.saaj.packaging.mime.internet.BMMimeMultipart{ public boolean find(java.io.InputStream,byte[],long[],com.sun.xml.internal.messaging.saaj.util.ByteOutputStream,com.sun.xml.internal.messaging.saaj.packaging.mime.internet.SharedInputStream) }
- com.sun.rowset.JoinRowSetImpl{ private void initJOIN(javax.sql.rowset.CachedRowSet) }
- com.sun.rowset.CachedRowSetImpl{ private java.lang.String buildTableName(java.lang.String) }
- com.sun.org.apache.xml.internal.serializer.WriterToUTF8Buffered{ public void write(java.lang.String) }
- com.sun.org.apache.xml.internal.dtm.ref.dom2dtm.DOM2DTM{ protected int addNode(org.w3c.dom.Node,int,int,int) }
- com.sun.org.apache.xml.internal.dtm.ref.dom2dtm.DOM2DTM{ protected boolean nextNode() }
- com.sun.org.apache.xml.internal.dtm.ref.DTMTreeWalker{ protected void startNode(int) }
- com.sun.org.apache.xml.internal.dtm.ref.DTMDocumentImpl{ public void appendChild(int,boolean,boolean) }
- com.sun.org.apache.xerces.internal.parsers.XIncludeParserConfiguration{ public void setProperty(java.lang.String,java.lang.Object) }
- com.sun.org.apache.xerces.internal.impl.xs.traversers.XSDHandler{ private void addGlobalAttributeDecls(com.sun.org.apache.xerces.internal.impl.xs.SchemaGrammar,com.sun.org.apache.xerces.internal.impl.xs.SchemaGrammar) }
- com.sun.org.apache.xerces.internal.impl.xs.traversers.XSDHandler{ private void addGlobalAttributeGroupDecls(com.sun.org.apache.xerces.internal.impl.xs.SchemaGrammar,com.sun.org.apache.xerces.internal.impl.xs.SchemaGrammar) }
- com.sun.org.apache.xerces.internal.impl.xs.traversers.XSDHandler{ private void

Results

Search

Stack Trace Search | Eclipse | IntelliJ | Contact | FAQ | grepcod

JDK / jdk / openjdk / 8u40-b25 / **javax.swing.ToolTipManager**

G+1 Find Usages Diff Raw Download HTML Widget

Outline | Files | Hierarchy | Comments

Profile: Standard JRE

- javax.swing
  - ToolTipManager
    - ToolTipManager() : void
    - checkForTipChange(MouseEvent) : void
    - createFocusChangeListener() : FocusListener
    - frameForComponent(Component) : Frame
    - getDismissDelay() : int
    - getDrawingGC(Point) : GraphicsConfiguration
    - getHeightAdjust(Rectangle, Rectangle) : int
    - getInitialDelay() : int
    - getPopupFitHeight(Rectangle, Component) : i
    - getPopupFitWidth(Rectangle, Component) : in
    - getReshowDelay() : int
    - getWidthAdjust(Rectangle, Rectangle) : int
    - hide(JComponent) : void
    - hideTipWindow() : void
    - initiateToolTip(MouseEvent) : void
    - isEnabled() : boolean
    - isLightWeightPopupEnabled() : boolean
    - mouseClicked(MouseEvent) : void
    - mouseDragged(MouseEvent) : void
    - mouseEntered(MouseEvent) : void
    - mouseExited(MouseEvent) : void
    - mouseMoved(MouseEvent) : void

```
490       // implements java.awt.event.MouseListener
491
```

Called when the mouse exits the region of a component. Any tool tip showing should be hidden.

Parameters:

`event` the event in question

```
496
497       public void mouseExited(MouseEvent event) {
498           boolean shouldHide = true;
499           if (insideComponent == null) {
500               // Drag exit
501           }
502           if (window != null && event.getSource() == window && insideComponent != null) {
503             // if we get an exit and have a heavy window
504             // we need to check if it if overlapping the inside component
505               Container insideComponentWindow = insideComponent.getTopLevelAncestor();
506               // insideComponent may be removed after tooltip is made visible
507               if (insideComponentWindow != null) {
508                   Point location = event.getPoint();
509                   SwingUtilities.convertPointToScreen(location, window);
510
511                   location.x -= insideComponentWindow.getX();
512                   location.y -= insideComponentWindow.getY();
513
514                   location = SwingUtilities.convertPoint(null, location, insideComponent);
515                   if (location.x >= 0 && location.x < insideComponent.getWidth() &&
516                           location.y >= 0 && location.y < insideComponent.getHeight()) {
517                       shouldHide = false;
518                   } else {
```

Results

Search | Stack Trace Search | Eclipse | IntelliJ | Contact | FAQ | grepcode™

JDK / jdk / openjdk / 8u40-b25 / **javax.swing.JInternalFrame**

Find Usages | Diff | Raw | Download | HTML Widget

requestFocus() : void
requestFocus(boolean) : boolean
requestFocusInWindow() : boolean
requestFocusInWindow(boolean) : boolean
resetKeyboardActions() : void
reshape(int, int, int, int) : void
~~resize(Dimension) : void~~
~~resize(int, int) : void~~
restoreSubcomponentFocus() : void
revalidate() : void
S safelyGetGraphics(Component) : Graphics
S safelyGetGraphics(Component, Component
scrollRectToVisible(Rectangle) : void
F setActionMap(ActionMap) : void
setAlignmentX(float) : void
setAlignmentY(float) : void
setAutoscrolls(boolean) : void
setBackground(Color) : void
setBorder(Border) : void
setBounds(Rectangle) : void
setBounds(int, int, int, int) : void
setClosable(boolean) : void
setClosed(boolean) : void
setComponentOrientation(ComponentOrie
setComponentPopupMenu(JPopupMenu) : v
setComponentZOrder(Component, int) : vo
setContentPane(Container) : void
setCreatedDoubleBuffer(boolean) : void
setCursor(Cursor) : void
setDebugGraphicsOptions(int) : void
setDefaultCloseOperation(int) : void
S setDefaultLocale(Locale) : void
setDesktopIcon(JInternalFrame.JDesktopIco
setDoubleBuffered(boolean) : void
setDropLocation(TransferHandler.DropLoca
setDropTarget(DropTarget) : void

dispose()
javax.swing.event.InternalFrameEvent.INTERNAL_FRAME_CLOSING

Beaninfo:
bound: true constrained: true description: Indicates whether this internal frame has been closed.

```
814
815         public void setClosed(boolean b) throws PropertyVetoException {
816             if (isClosed == b) {
817                 return;
818             }
819
820             Boolean oldValue = isClosed ? Boolean.TRUE : Boolean.FALSE;
821             Boolean newValue = b ? Boolean.TRUE : Boolean.FALSE;
822             if (b) {
823               fireInternalFrameEvent(InternalFrameEvent.INTERNAL_FRAME_CLOSING);
824             }
825             fireVetoableChange(IS_CLOSED_PROPERTY, oldValue, newValue);
826             isClosed = b;
827             if (isClosed) {
828               setVisible(false);
829             }
830             firePropertyChange(IS_CLOSED_PROPERTY, oldValue, newValue);
831             if (isClosed) {
832               dispose();
833             } else if (!opened) {
834               /* this bogus -- we haven't defined what
835                  setClosed(false) means. */
836               //        fireInternalFrameEvent(InternalFrameEvent.INTERNAL_FRAME_OPENED);
837               //            opened = true;
838             }
839         }
```

Sets whether the `JInternalFrame` can be resized by some user action.

Parameters:
`b` a boolean, where `true` means this internal frame can be resized

Beaninfo:
preferred: true bound: true description: Determines whether this internal frame can be resized by the user.