Code Completion on Steroids

tw: @MarcelBruch
g+: +Marcel Bruch
The Pragmatic Programmer: From Journeyman to Master [Paperback]
Andrew Hunt (Author), David Thomas (Author)

Customer Reviews

List Price: $49.99
Price: $38.99 & this item ships for FREE with Super Saver Shipping. Details
You Save: $11.00 (22%)

In Stock.
Ships from and sold by Amazon.com. Gift-wrap available.

Customers Who Bought This Item Also Bought

Code Complete: A Practical Handbook of Software by Steve McConnell
$31.49

Clean Code: A Handbook of Agile Software Craftsman by Robert C. Martin
$35.99

Refactoring: Improving the Design of Existing Code by Martin Fowler
$50.69

The Mythical Man-Month: Essays on Software by Frederick P. Brooks
$32.99

The Passionate Programmer: Creating a Remarkable Career by Chad Fowler
$16.29

What Do Customers Ultimately Buy After Viewing This Item?

75% buy the item featured on this page:
The Pragmatic Programmer: From Journeyman to Master ★★★★★ (150) $38.99

9% buy
Code Complete: A Practical Handbook of Software Construction ★★★★★ (119) $31.49

6% buy
Design Patterns: Elements of Reusable Object-Oriented Software ★★★★★ (271) $35.57

5% buy
Clean Code: A Handbook of Agile Software Craftsmanship ★★★★★ (67) $35.99
Developers Who Called "new Text()" Method Also Called

<table>
<thead>
<tr>
<th>Method</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>setText()</td>
<td>89%</td>
</tr>
<tr>
<td>addListener()</td>
<td>53%</td>
</tr>
<tr>
<td>setLayout()</td>
<td>99%</td>
</tr>
<tr>
<td>setFont()</td>
<td>35%</td>
</tr>
</tbody>
</table>

What Do Developers Ultimately Override After Extending This Class?

100% overwrote
PreferencePage.createContents()

96% overwrote
PreferencePage.performOk()

5% overwrote
PreferencePage.performCancel()
“Other developers frequently bought used the following methods...”

INTELLIGENT CODE COMPLETION
What does the developer need, i.e., which methods should the code completion present to the user?
All 164 Methods of Text?
JButton? 381 Methods.
@Override
protected Control createDialogArea(final Composite parent) {
    Composite container = (Composite) super.createDialogArea(parent);

   .swtTextWidget = new Text(container, SWT.BORDER);
    swtTextWidget.
    return container;
}

What does the developer need, i.e., which methods should the code completion present to the user?
Intelligent Code Completion

... or just the three missing ones?
“Welcome to the jungle…”

CALL-CHAIN COMPLETION
How to obtain an instance of...

```java
public class MyView extends ViewPart {

    public void updateMessage(final String newMessage) {
        // How do I get an instance of IStatusLineManager?

        final IStatusLineManager manager = |
        manager.setMessage(newMessage);
    }
}
```

There is no trivial solution anymore. Again, what should code completion return?
What we need...

```java
public void updateMessage(final String newMessage) {
    // How do I get an instance of
    final IStatusLineManager manager =
       this.getViewSite()
       .getActionBars()
       .getStatusLineManager();
    manager.setMessage(newMessage);
}
```
“Developers who bought overwrote this method typically also overwrote...”

**USAGE-DRIVEN JAVADOC**
In good documentation we trust...

```java
/**
 * Rollsback the transaction if any and clears different lists to
 * start with an empty resource again.
 * Note that the super.doUnload is not called because that clears
 * the list resulting in all kinds of undesirable inverseremoves.
 */
@Override
protected void doUnload() {
```
In good documentation we trust...

```java
/**
 * Rollsback the transaction if any and clears different lists to
 * start with an empty resource again.
 * Note that the super.doUnload is not called because that clears
 * the list resulting in all kinds of undesirable inverseremoves.
 */
@override
protected void doUnload() {
    super.doUnload();
}
```
In good documentation we trust...

```java
/**
 * Rollsback the transaction if any and clears different lists to
 * start with an empty resource again.
 * Note that the `super.doUnload` is not called because that clears
 * the list resulting in all kinds of undesirable inverseremoves.
 */

@Override
protected void doUnload() {
  super.doUnload();
}
```

No? Are you sure?
In good documentation we trust...

```java
/**
 * Rollsback the transaction if any and clears different lists to
 * start with an empty resource again.
 * Note that the super.doUnload is not called because that clears
 * the list resulting in all kinds of undesirable inverseremoves.
 */
@Override
protected void doUnload() {
    super.doUnload();
}
```

No? Are you sure?
Generating high-quality API documentation
/**
 * The number of questions.
 */
private int numberOfQuestions;

/**
 * Sets the number of questions.
 * @param numberOfQuestions the number of questions
 * @throws IllegalArgumentException the illegal argument exception
 */
public void setNumberOfQuestions(int numberOfQuestions) throws IllegalArgumentException {
    if (numberOfQuestions < 0) {
        throw new IllegalArgumentException("numberOfQuestions < 0");
    }
    this.numberOfQuestions = numberOfQuestions;
}
/**
 * The number of questions.
 */
private int numberOfQuestions;

/**
 * Sets the number of questions.
 *
 * @param numberOfQuestions the number of questions
 * @throws IllegalArgumentException the illegal argument exception
 */
public void setNumberOfQuestions(int numberOfQuestions)
    throws IllegalArgumentException {
    if (numberOfQuestions < 0) {
      throw new IllegalArgumentException("numberOfQuestions < 0");
    }
    this.numberOfQuestions = numberOfQuestions;
  }
private int numberOfQuestions;

/**
 * The number of questions.
 */

public void setNumberOfQuestions(int numberOfQuestions) throws IllegalArgumentException {
    if (numberOfQuestions < 0) {
        throw new IllegalArgumentException("numberOfQuestions < 0");
    }
    this.numberOfQuestions = numberOfQuestions;
}
private int numberOfQuestions;

/**
 * The number of questions.
 */

public void setNumberOfQuestions(int numberOfQuestions)
throws IllegalArgumentException {
    if (numberOfQuestions < 0) {
        throw new IllegalArgumentException("numberOfQuestions < 0");
    }
    this.numberOfQuestions = numberOfQuestions;
}
/**
 * The number of questions.
 */
private int numberOfQuestions;

/**
 * Sets the number of questions.
 *
 * @param numberOfQuestions the number of questions
 * @throws IllegalArgumentException the illegal argument exception
 */
public void setNumberOfQuestions(int numberOfQuestions) throws IllegalArgumentException {
    if (numberOfQuestions < 0) {
        throw new IllegalArgumentException("numberOfQuestions < 0");
    }
    this.numberOfQuestions = numberOfQuestions;
}
/**
 * The number of questions.
 */
private int numberOfQuestions;

/**
 * Sets the number of questions.
 *
 * @param numberOfQuestions the number of questions
 * @throws IllegalArgumentException the illegal argument exception
 */
public void setNumberOfQuestions(int numberOfQuestions)
   throws IllegalArgumentException {
    if (numberOfQuestions < 0) {
        throw new IllegalArgumentException("numberOfQuestions < 0");
    }
    this.numberOfQuestions = numberOfQuestions;
}
private int numberOfQuestions;

/**
 * Sets the number of questions.
 *
 * @param numberOfQuestions the number of questions
 *
 * @throws IllegalArgumentException the illegal argument exception
 */
 public void setNumberOfQuestions(int numberOfQuestions) throws IllegalArgumentException {
     if (numberOfQuestions < 0) {
         throw new IllegalArgumentException("numberOfQuestions < 0");
     }
     this.numberOfQuestions = numberOfQuestions;
 }
private int numberOfQuestions;

public void setNumberOfQuestions(int numberOfQuestions) throws IllegalArgumentException {
    if (numberOfQuestions < 0) {
        throw new IllegalArgumentException("numberOfQuestions < 0");
    }
    this.numberOfQuestions = numberOfQuestions;
}
```java
/**
 * The number of questions.
 */
private int numberOfQuestions;

/**
 * Sets the number of questions.
 * @param numberOfQuestions the number of questions
 * @throws IllegalArgumentException the illegal argument exception
 */
public void setNumberOfQuestions(int numberOfQuestions) throws IllegalArgumentException {
    if (numberOfQuestions < 0) {
        throw new IllegalArgumentException("numberOfQuestions < 0");
    }
    this.numberOfQuestions = numberOfQuestions;
}
```
What people say about JAutodoc...
What people say about JAutodoc...

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Wow exactly what I needed!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>----------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Wow exactly what I needed!</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Thank you... this plugin rocks!</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
What people say about JAutodoc...

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Wow exactly what I needed!</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Thank you... this plugin rocks!</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Works perfectly. Smarter than I expected!</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
What people say about JAutodoc...

<table>
<thead>
<tr>
<th>User</th>
<th>Rating</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anonymous</td>
<td>9</td>
<td>2009-08-02 11:32:37</td>
</tr>
<tr>
<td>Wow exactly what I needed!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Anonymous</td>
<td>9</td>
<td>2009-02-13 19:58:32</td>
</tr>
<tr>
<td>Thank you... this plugin rocks!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Anonymous</td>
<td>9</td>
<td>2009-02-13 19:58:32</td>
</tr>
<tr>
<td>Works perfectly. Smarter than I expected!</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

WTH? What did you expect?!
How Recommenders is coming to rescue...
What documentation would you expect when subclassing Dialog?

```java
package org.eclipse.recommenders.examples.demo;

import org.eclipse.jface.dialogs.Dialog;
import org.eclipse.jface.window.IShellProvider;

public class MyDialog extends Dialog {

  protected MyDialog(final IShellProvider parentShell) {
    super(parentShell);
  }
}
```
What Javadoc gives you...

org.eclipse.jface.dialogs.Dialog
What Javadoc gives you...

```
org.eclipse.jface.dialogs.Dialog

A dialog is a specialized window used for narrow-focused communication with the user.
```
What Javadoc gives you...

`org.eclipse.jface.dialogs.Dialog`

A dialog is a specialized window used for narrow-focused communication with the user.

Dialogs are usually modal. Consequently, it is generally bad practice to open a dialog without a parent. A modal dialog without a parent is not prevented from disappearing behind the application's other windows, making it very confusing for the user.

If there is more than one modal dialog is open the second one should be parented off of the shell of the first one otherwise it is possible that the OS will cuss to the first dialog potentially blocking the UI.
What Javadoc gives you...

org.eclipse.jface.dialogs.Dialog

A dialog is a specialized window used for narrow-focused communication with the user.

Dialogs are usually modal. Consequently, it is generally bad practice to open a dialog without a parent. A modal dialog without a parent is not prevented from disappearing behind the application's other windows, making it very confusing for the user.

If there is more than one modal dialog is open the second one should be parented off of the shell of the first one otherwise it is possible that the OS will cus to the first dialog potentially blocking the UI.

But what are the hot-spots of Dialog?
What code completion offers...
What code completion offers...

Which of the 56 methods should we override?
public class MyDialog extends Dialog {

    // Method overrides
    @Override
    protected void createDialogArea(Composite composite) {
        // Implementation
    }
    ...
}

---

Subclassing Directives Provider

Based on 636 direct subclasses of Dialog we created the following statistics.
Subclassers may consider to override the following methods.

- **should** override createDialogArea(Composite) (600 times - 94%)
- **may** override configureShell(Shell) (397 times - 62%)
- **may** override okPressed() (292 times - 46%)
- **may** override createButtonsForButtonBar(Composite) (212 times - 33%)
- **rarely** override buttonPressed(int) (148 times - 23%)
- **rarely** override createButtonBar(Composite) (76 times - 12%)
- **rarely** override createContents(Composite) (69 times - 11%)
- **rarely** override close() (55 times - 9%)
- **rarely** override cancelPressed() (54 times - 8%)
- **rarely** override getInitialSize() (53 times - 8%)
- **rarely** override create() (46 times - 7%)
public class MyWizard extends Wizard {

    @Override
    public void addPages() {
    }
};

---

**Subclassing Directives Provider**

Based on 488 implementations of addPages we created the following statistics. Implementors may consider to call the following methods.

- **must** call addPage(IWizardPage) (466 times - 95%)
- **rarely** call addPages() (105 times - 22%)

Show / Add Comments (0)
public class MyViewerSorter extends ViewerComparator {

Mining for Subclassing patterns

By analyzing 175 subclasses that override at least one method, the following subclassing patterns have been identified.

Pattern #1 – covers approximately 81% of the examined subclasses (143 subclasses).
  - override ViewerComparator.compare(Viewer, Object, Object) - 100%

Pattern #2 – covers approximately 18% of the examined subclasses (32 subclasses).
  - override ViewerComparator.category(Object) - 94%
  - override ViewerComparator.sort(Viewer, Object[]) - 31%

Show / Add Comments (0)
public class MyDialog extends Dialog {

@Override
protected Control createDialogArea(Composite parent) {

Example #1:
@Override
protected Control createDialogArea(final Composite parent) {
    final Composite container = (Composite) super.createDialogArea(parent);
    //...
    return container;
}

Example #2:
@Override
protected Control createDialogArea(final Composite parent) {
    final Composite container = createContainer(parent);
    final Text text = new Text(container, SWT.SINGLE | SWT.LEAD | SWT.BORDER);
    text.setLayoutData(new GridData(SWT.FILL, SWT.CENTER, true, false));
    CheckboxTableViewer v = new CheckboxTableViewer(null);
    return container;
}
Extended Documentation

@ Javadoc Provider

Sets the contents of the receiver to the given string. If the receiver has style SINGLE and the argument contains multiple lines of text, the result of this operation is undefined and may vary from platform to platform.

Show / Add Comments (0)

Code Examples Provider

Example #1:

```java
final Text text = new Text(composite, SWT.NONE);
text.setLayoutData(new GridData());
text.setText(s);
text.setFont(font);
```

Method Calls Provider

People who use `Text` Widget usually also call the following methods:

- `Text`
- `Text` = new `Text`(Composite, int)
- `Text` = new `Text`(Composite, int)
- `Text` = new `Text`(Object)
- `Text` = addModifyListener(ModifyListener)
- `Text` = setEnabled(boolean)
- `Text` = setFont(Font)
- `Text` = setBackground(Color)

Social Bookmarks Provider

Social bookmarks are collections of web resources for specific Java elements, shared by all users. You can

Click here to add a new bookmark.
“Templates meet Social - reloaded!”

SNIPMATCH

Doug Wightman, Zi Ye
**Eclipse Java Editor Templates**

<table>
<thead>
<tr>
<th>Java Keywords</th>
<th>SWT Keywords</th>
<th>Java Statements</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>@author</code></td>
<td><code>SWT statements</code></td>
<td><code>new Scale</code></td>
</tr>
<tr>
<td><code>&lt;b&gt;</code></td>
<td><code>new Image</code></td>
<td><code>new ScrolledComposite</code></td>
</tr>
<tr>
<td><code>&lt;code&gt;</code></td>
<td><code>new Label</code></td>
<td><code>new TabFolder</code></td>
</tr>
<tr>
<td><code>&lt;i&gt;</code></td>
<td></td>
<td><code>new TabItem</code></td>
</tr>
<tr>
<td><code>&lt;pre&gt;</code></td>
<td></td>
<td><code>new TableColumn for a Table</code></td>
</tr>
<tr>
<td><code>addListener</code></td>
<td></td>
<td><code>new TableItem for a Table</code></td>
</tr>
<tr>
<td><code>arrayAdd</code></td>
<td></td>
<td><code>new TreeColumn for a Tree</code></td>
</tr>
<tr>
<td><code>arrayMerge</code></td>
<td></td>
<td><code>new TreeItem for a Tree</code></td>
</tr>
<tr>
<td><code>Button</code></td>
<td><code>switch</code></td>
<td><code>JUnit 4</code></td>
</tr>
<tr>
<td><code>cast</code></td>
<td><code>case</code></td>
<td><code>try catch block</code></td>
</tr>
<tr>
<td><code>catch</code></td>
<td><code>catch</code></td>
<td><code>try catch block</code></td>
</tr>
<tr>
<td><code>Combo</code></td>
<td><code>catch</code></td>
<td><code>try iterate with condition</code></td>
</tr>
<tr>
<td><code>Composite</code></td>
<td><code>catch</code></td>
<td><code>while iterate with enumeration</code></td>
</tr>
<tr>
<td><code>Date/Time</code></td>
<td><code>catch</code></td>
<td><code>while iterate with iterator</code></td>
</tr>
<tr>
<td><code>do</code></td>
<td><code>else</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>else</code></td>
<td><code>else if</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>execute</code></td>
<td><code>for</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>ExpandItem</code></td>
<td><code>for loop</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>fall-through</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>false</code></td>
<td><code>forEach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>for</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>for each</code></td>
<td><code>forEach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>forEach</code></td>
<td><code>forEach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>gridData</code></td>
<td><code>forEach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>GridLayout</code></td>
<td><code>forEach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Group</code></td>
<td><code>forEach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>if</code></td>
<td><code>forEach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>if else</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>instanceOf</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Label</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Lazy</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Link</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>List</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Main</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Mainloop</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>new</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Nls</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>null</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>privateMethod</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>protectedMethod</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>PublicMethod</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Runnable</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>SashForm</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Scale</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>ScrollComposite</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Shell</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Spinner</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>StaticFinal</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>StyleSheet</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>StyleRange</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Switch</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Synchronized</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>SysErr</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>SysOut</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>SysTrace</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Text</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>Test</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>TestMethod</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>TestMethodJUnit4</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>toArray</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>ToolBar</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>toolItem</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>tree</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>treeColumn</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>treeItem</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>true</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>try</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>while</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>while</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>while</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>while</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
<tr>
<td><code>while</code></td>
<td><code>foreach</code></td>
<td><code>while</code></td>
</tr>
</tbody>
</table>
Eclipse Java Editor Templates

<table>
<thead>
<tr>
<th>@author</th>
<th>javadoc</th>
<th>author name</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;b&gt;</td>
<td>javadoc</td>
<td>&lt;b&gt; &lt;/b&gt;</td>
</tr>
<tr>
<td>&lt;code&gt;</td>
<td>javadoc</td>
<td>&lt;/code&gt; &lt;/code&gt;</td>
</tr>
<tr>
<td>&lt;i&gt;</td>
<td>javadoc</td>
<td>&lt;i&gt; &lt;/i&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>addlistener</th>
<th>ifelse</th>
<th>SWT statements</th>
</tr>
</thead>
<tbody>
<tr>
<td>arrayadd</td>
<td>Image</td>
<td>new Image</td>
</tr>
<tr>
<td>arraymerge</td>
<td>instanceof</td>
<td>Java statements</td>
</tr>
<tr>
<td>Browser</td>
<td>Label</td>
<td>dynamic type test and cast</td>
</tr>
<tr>
<td>Button</td>
<td>lazy</td>
<td>SWT statements</td>
</tr>
<tr>
<td>cast</td>
<td>Link</td>
<td>new Label</td>
</tr>
<tr>
<td>catch</td>
<td>List</td>
<td></td>
</tr>
<tr>
<td>Combo</td>
<td>main</td>
<td></td>
</tr>
<tr>
<td>Composite</td>
<td>new</td>
<td></td>
</tr>
<tr>
<td>DateTime</td>
<td>nls</td>
<td></td>
</tr>
<tr>
<td>do</td>
<td>null</td>
<td></td>
</tr>
<tr>
<td>else</td>
<td>private_method</td>
<td></td>
</tr>
<tr>
<td>elseif</td>
<td>protected_method</td>
<td></td>
</tr>
<tr>
<td>execute</td>
<td>public_method</td>
<td></td>
</tr>
<tr>
<td>ExpandBar</td>
<td>runnable</td>
<td></td>
</tr>
<tr>
<td>ExpandItem</td>
<td>SashForm</td>
<td></td>
</tr>
<tr>
<td>fall-through</td>
<td></td>
<td></td>
</tr>
<tr>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Scale       | SWT statements | new Scale |
| ScrolledComposite | SWT statements | new ScrolledComposite |
| Shell       | SWT statements | new Shell |
| Spinner     | SWT statements | new Spinner |
| static_final | SWT statements | static final field |
| StyledText  | SWT statements | new StyledText |
| StyleRange  | SWT statements | new StyleRange for a StyledText |
| switch      | Java statements | switch case statement |
| synchronized | Java statements | synchronized block |
| syserr      | Java statements | print to standard error |
| systout     | Java statements | print to standard out |
| systrace    | Java statements | print current method to standard out |
| TabFolder   | SWT statements | new TabFolder |
| Editor      | SWT statements |                |

| while       | Java statements | iterate with enumeration |
| while       | Java statements | iterate with iterator |
| while       | Java statements | while loop with condition |
Eclipse Java Editor Templates

42x Java Templates.
35x SWT Templates.
Eclipse Java Editor Templates

42x Java Templates.
35x SWT Templates.
7 Years.
The rise of code snippet repositories

and many more...
Searching snippets in your IDE...

```java
import java.io.File;

public class Test {
    public static void main(String[] args) {
        String settingsPath = "settings.txt";
        File file = new File(settingsPath);
        read file set| read fileObject from settingsPath
        read list of files in folder settingsPath
        read file <fileObject> into a string
        read fileObject from <filePathString>
        read lines in <fileObject> into an array
        read list of files in folder <pathString>
        create a thread
    }
}
```
“IDE 2.0? Leveraging the wisdom of the (Eclipse) community...”

**ECLIPSE CODE RECOMMENDERS**
From IDE 1.0 ...

interact with

search & browse

WWW

document

interact with
...to IDE 2.0

interact with Knowledge Base
...to IDE 2.0

interact with

Knowledge Base

collect usage data

document

collect usage data

interact with
...to IDE 2.0

Knowledge Base

interact with

interact with

collect usage data

collect usage data

document
...to IDE 2.0

Knowledge Base

interact with

mine data

collect usage data

document

interact with
...to IDE 2.0

interact with

suggest & notify

mine data

Knowledge Base

suggest & notify

interact with

collect usage data

search, query, vote

collect usage data

document
Today we leverage...
but in future we leverage...
but in future we leverage...
but in future we leverage...
but in future we leverage...
but in future we leverage...

context

object usage

project environment

selected proposals
but in future we leverage...
but in future we leverage...
Crowd-sourcing...
Code Recommenders is...

Intelligent Code Completion

Usage-Driven Javadocs

Smart Bug Detection

Stacktrace Search Engine

All 164 Methods of Text?

But what are the hot-spots of Dialog?

What have I missed?

What caused that stacktrace?

How do I get an instance of…?
Contributors so far...

Jan Kassens, Peter Schroeder, Daniel Glöckner, Maik Görtz, Johannes Lerch, Johannes Born, Mohsen Parisay, Andreas Sewe, Sebastian Ahlfeld, Sebastian Kasten, Daniel Staesche, David Kalnischkies, Sebastian Wörner, Boyan Yurukov, Jan Stolzenburg, Nico Wombacher, Dirk Kröhan, Florianrian Jakob, Julius Rücker, Steffen Remus, Christopher Mann, Stefan Henss, Andreas Kaluza, Nikolay Shindov, Michael Novotny, Kristin Arand, Sinem Emeröz, Michael Kutschke, Sebastian Proksch, Tomasz Kalbarczyk, Marko Martin, Sheip Dargutev, David Schuld, Jens Krause, Kristijan Madunic, Daniel Brandtner, Roman Getto, Christian Kilb, Johannes Kastl, Dennis Sänger, Annie Liu, Markus Migenda, Tjark Vandommele, Sebastian Denel, Florian Nöll, Gary Fritz, Dennis Siebert, Peter Sinzig, Laura Altmüller, Paul Schatygin, Jan-Michael Heller, Minh Hoang Nguyen, Sascha Nordquist, Paul Emmanuel Faidherbe, ...
Q & A
http://eclipse.org/recommenders/

Follow me on twitter: @MarcelBruch
google+: +Marcel Bruch
“Yours is ours and mine is mine...”

**BUT HOW ABOUT PRIVACY?**
Usage data sharing example

```java
import com.mycompany.*;
import org.eclipse.*;

public class MyCompanyDialog extends Dialog implements ICompanyService {
    MyCompanyController controller = new MyCompanyController();

    @Override
    protected Control createDialogArea(Composite parent) {
        Composite container = (Composite) super.createDialogArea(parent);
        Button sync = new Button(container, SWT.CHECK);
        sync.addSelectionListener(new SelectionAdapter() {
            @Override
            public void widgetSelected(final SelectionEvent e) {
                if (sync.getSelection()) {
                    controller.synchronize();
                    // lot of more things done with controller
                }
            }
        });
        return container;
    }

    @Override
    public void myCompanyServiceMethod() {
        controller.setup();
    }
}
```
import com.mycompany.*;
import org.eclipse.*;

public class MyCompanyDialog extends Dialog implements ICompanyService {

    MyCompanyController controller = new MyCompanyController();

    @Override
    protected Control createDialogArea(Composite parent) {
        Composite container = (Composite) super.createDialogArea(parent);
        Button sync = new Button(container, SWT.CHECK);
        sync.addSelectionListener(new SelectionAdapter() {
            @Override
            public void widgetSelected(final SelectionEvent e) {
                if (sync.getSelection()) {
                    controller.synchronize();
                    // lot of more things done with controller
                }
            }
        });
        return container;
    }

    @Override
    public void myCompanyServiceMethod() {
        controller.setup();
    }
}
import com.mycompany.*;
import org.eclipse.*;

public class MyCompanyDialog extends Dialog implements ICompanyService {

    MyCompanyController controller = new MyCompanyController();

    @Override
    protected Control createDialogArea(Composite parent) {
        Composite container = (Composite) super.createDialogArea(parent);
        Button sync = new Button(container, SWT.CHECK);
        sync.addSelectionListener(new SelectionAdapter() {
            @Override
            public void widgetSelected(final SelectionEvent e) {
                if (sync.getSelection()) {
                    controller.synchronize();
                    // lot of more things done with controller
                }
            }
        });
        return container;
    }

    @Override
    public void myCompanyServiceMethod() {
        controller.setup();
    }
}
import org.eclipse.*;

public class MyCompanyDialog extends Dialog implements ICompanyService {

    MyCompanyController controller = new MyCompanyController();
    @Override
    protected Control createDialogArea(Composite parent) {
        Composite container = (Composite) super.createDialogArea(parent);
        Button sync = new Button(container, SWT.CHECK);
        sync.addSelectionListener(new SelectionAdapter() {
            @Override
            public void widgetSelected(final SelectionEvent e) {
                if (sync.getSelection()) {
                    controller.synchronize();
                }
            }
        });
        return container;
    }

    @Override
    public void myCompanyServiceMethod() {
        controller.setup();
    }
}
Effective usage data shared

```java
import org.eclipse.*;

public class MyCompanyDialog extends Dialog implements ICompanyService {

    MyCompanyController controller = new MyCompanyController();

    @Override
    protected Control createDialogArea(Composite parent) {
        Composite container = (Composite) super.createDialogArea(parent);
        Button sync = new Button(container, SWT.CHECK);
        sync.addSelectionListener(new SelectionAdapter() {
            @Override
            public void widgetSelected(final SelectionEvent e) {
                if (sync.getSelection()) {
                    controller.synchronize();
                }
            }
        });

        return container;
    }

    @Override
    public void myCompanyServiceMethod() {
        controller.setup();
    }
}
```

Only the information how you used Eclipse is shared.
Still scared about crowd-sourcing?
Still scared about crowd-sourcing? How about a team server?
Still scared about crowd-sourcing?
How about a team server?
Code Recommenders is...

Intelligent Code Completion

Usage-Driven Javadocs

Stacktrace Search Engine

Smart Bug Detection

All 164 Methods of Text?

But what are the hot-spots of Dialog?

What have I missed?

What caused that stacktrace?

How do I get an instance of…?
Contributors so far...

Jan Kassens, Peter Schroeder, Daniel Glöckner, Maik Görtz, Johannes Lerch, Johannes Born, Mohsen Parisay, Andreas Sewe, Sebastian Ahlfeld, Sebastian Kasten, Daniel Staesche, David Kalnischkies, Sebastian Wörner, Boyan Yurukov, Jan Stolzenburg, Nico Wombacher, Dirk Kröhan, Florianrian Jakob, Julius Rücker, Steffen Remus, Christopher Mann, Stefan Henss, Andreas Kaluza, Nikolay Shindov, Michael Novotny, Kristin Arand, Sinem Emeröz, Michael Kutschke, Sebastian Proksch, Tomasz Kalbarczyk, Marko Martin, Sheip Dargutev, David Schuld, Jens Krause, Kristijan Madunic, Daniel Brandtner, Roman Getto, Christian Kilb, Johannes Kastl, Dennis Sänger, Annie Liu, Markus Migenda, Tjark Vandommele, Sebastian Denel, Florian Nöll, Gary Fritz, Dennis Siebert, Peter Sinzig, Laura Altmüller, Paul Schatygin, Jan-Michael Heller, Minh Hoang Nguyen, Sascha Nordquist, Paul Emmanuel Faidherbe, ...
Q & A