JFace – Content Assist – Three Traces
Use Concept Trace Slicing

Supporting Applications : [JSP Editor, Example Java Editor, HTML Editor]

```
import org.eclipse.jface.text.contentassist.IContentAssistProcessor;
import org.eclipse.jface.text.IDocument;
import org.eclipse.jface.text.contentassist.ContentAssistant;
import org.eclipse.jface.text.source.IAnnotationHover;
import org.eclipse.jface.text.source.SourceViewerConfiguration;
import org.eclipse.swt.graphics.Color;
import org.eclipse.jface.text.ITextViewer;
import org.eclipse.jface.text.DocumentCommand;
import org.eclipse.jface.text.contentassist.ICompletionProposal;
import org.eclipse.jface.text.TextAttribute;
import org.eclipse.jface.text.ITextHover;
import org.eclipse.jface.text.source.ISourceViewer;
import org.eclipse.jface.text.IAutoEditStrategy;

public class AppContentAssistProcessor
implements IContentAssistProcessor  {

      public char[] getCompletionProposalAutoActivationCharacters() {
            CompletionProposal completionProposal = new CompletionProposal(String,int,int,int)||
                                                                  (String,int,int,int,Image,String,IContextInformation,String);
      }

      public String getErrorMessage() {
      }

      public char[] getContextInformationAutoActivationCharacters() {
      }

      public ICompletionProposal[] computeCompletionProposals(ITextViewer,int) {
      }
}

public class AppAutoEditStrategy
implements IAutoEditStrategy  {

      public void customizeDocumentCommand(IDocument,DocumentCommand) {
      }
}

public class AppTextHover
implements ITextHover  {
}

public class AppAnnotationHover
implements IAnnotationHover  {
}

public class SomeClass {

      public void someMethod() {
            AppAnnotationHover appAnnotationHover = new AppAnnotationHover();
            SourceViewerConfiguration sourceViewerConfiguration = new SourceViewerConfiguration();
            int app_int = IDocument.getLineOffset(int);      // MAY REPEAT!
            TextAttribute textAttribute = new TextAttribute(Color)||(Color,Color,int);    // MAY REPEAT!
            AppTextHover appTextHover = new AppTextHover();
            String string = sourceViewerConfiguration.getConfiguredDocumentPartitioning(ISourceViewer);
            ContentAssistant contentAssistant = new ContentAssistant();
            contentAssistant.setDocumentPartitioning(string);
            contentAssistant.enableAutoActivation(boolean);
            contentAssistant.setAutoActivationDelay(int);
            contentAssistant.setProposalPopupOrientation(int);
            contentAssistant.setContextInformationPopupOrientation(int);
            AppContentAssistProcessor appContentAssistProcessor = new AppContentAssistProcessor();
            contentAssistant.setContentAssistProcessor(appContentAssistProcessor);
      }
}
```

Description of False Negatives:

We have the following three false negatives. The first two are because of not implementing the following class since the third sample application did not implement it:

```
public class AppSourceViewerConfiguration
extends SourceViewerConfiguration {

      public IContentAssistant getContentAssistant(ISourceViewer) {
      }

}
```

The third false negative is because of not calling the following instruction since it is in org.eclipse.ui, not in JFace:
`TextEditor.setSourceViewerConfiguration(appSourceViewerConfiguration);`