Eclipse チュートリアル・ガイド for Eclipse 3.3.2

更新履歴

- 2007/04/xx 橋山作成 for 3.2.x
- 2007/12/19 橋山改訂 for 3.3.x
- 2008/04/21 松澤改訂 for 3.3.2

目次

1. Eclipseとは？

1.1 Eclipseの概要

Eclipseとは，もともとIBMが自社用の開発環境として開発されていたものですが，2001年11月にオープンソースコミュニティにソースが寄付され，以後Eclipseプロジェクトによって開発され続けている統合開発環境（IDE）です．

1.2 Eclipseのメリット

- オープンソースであるため，無料で入手することが出来ます
- 市販のIDEに匹敵する優秀なコード生成（補完）機能，デバッグ機能，ビルド機能，テスト機能，リファクタリング機能などを兼ね備えています
- プラグインとしてさまざまな機能を組み込むことができ，プラグイン次第でJava以外の多様な言語への対応も可能です
- CPL(Common Public License)にしたがって配布されているため，商用アプリケーションを作成することも可能です

2. チュートリアル

2.1 プロジェクトの作成

Eclipse上で開発を行うときは，プロジェクトという単位でプログラムの管理を行います．今回は，最初の課題を行うプロジェクトとして，「example」プロジェクトを作成します．オブプロでは，課題1つ1つをプロジェクトとして扱います．毎回，課題の数だけプロジェクトを作成することになります．

1. メニューの中から「ファイル」→「新規」→「プロジェクト」の順に選択し，新規プロジェクト画面を開きます．その中から「Javaプロジェクト」を選択し，「次へ」ボタンを押します．
2. 新規Javaプロジェクト画面で，プロジェクト名に「example」を入力します．その他の項目は変更せずに，「次へ」ボタンを押します（下図参照）．
   ※プロジェクト名は，毎回指示された名前にして下さい．

図：プロジェクト作成画面

3. Javaビルド設定を行います．ここでは何も入力せず「終了」ボタンを押します．これで，「example」プロジェクトが作成されました．

   ※「この種類のプロジェクトはJavaパースペクティブに関連付けられます」というメッセージが出ることがありますが，その場合は「はい」ボタンを押して下さい．パースペクティブとはウィンドウ構成のことで，Java開発に最適なウィンドウ構成が設定されます．

2.2 Hello Worldを動かす

実際にEclipse上でプログラムを作成して，実行してみましょう．ここでは，「Hello World」という文字列をコンソールに表示するプログラムを作成します．

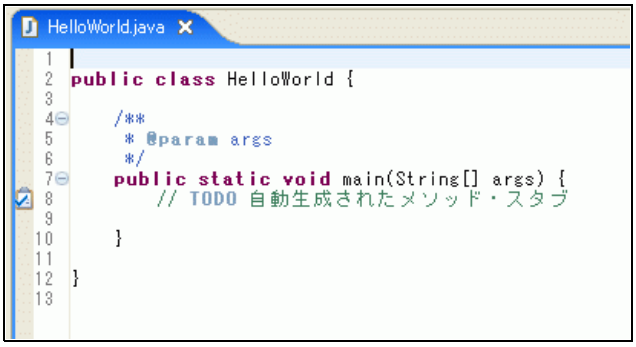
2.2.1 クラスを作成する

1. 2.1で作成した「example」プロジェクト内にある「src」フォルダを右クリックします．
2. 開いたメニューの中から「新規」→「クラス」の順に選択し，新規Javaクラス画面を開きます．
3. 新規Javaクラス画面で，クラス名に「HelloWorld」を入力します（命名規則に注意してください）．
4. 「作成するメソッド・スタブの選択」の「public static void main(String[] args)」にチェックを入れて，「終了」ボタンを押します（下図参照）．
   ※Javaでは，プログラムを実行するクラスには mainメソッドが必要ですが，Eclipseでは上記3の手順によって，この mainメソッドの雛形を自動生成してくれます．

   ※「デフォルトパッケージの使用は推奨されません」というメッセージは，ここでは気にしないで結構です．

図：新規Javaクラス設定画面

5. これで，HelloWorldクラスが作成されます（下図参照）．



図：クラス作成時

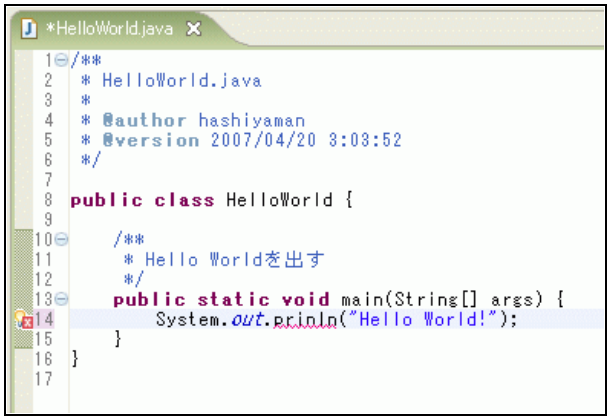
2.2.2 プログラムを書く

2.2.1で作成した「Hello World」クラスに，プログラムを記述します．今回は，コンソールに「Hello World!」という文字列を表示させるプログラムを記述します（下図参照）．

図：プログラム記述後

Eclipseではプログラムを書く際にコンパイルエラーがあると，該当箇所に赤い波線が，該当箇所の左にエラーマークが表示されます（下図参照）． エラーマークには2種類あり，修正候補（4.5で解説）が予測可能な場合には電球マークが，予測不可能な場合には×マークが出ます．

プログラムを実行する前に，すべてのエラーマークを取り除いてコンパイルが通る状態にしましょう．



図：修正候補が予測可能なコンパイルエラー（例：スペルミス）

図：修正候補が予測不可能なコンパイルエラー（例：全角スペースの混入）

2.2.3 プログラムを実行する

1. 2.2.2で書き直した「HelloWorld」クラスを右クリックして，メニューの中から「実行」→「Java アプリケーション」の順に選択します．
2. 画面右下にあるコンソールに，実行結果が表示されれば成功です（下図参照）．

図：Hello World実行結果

2.3 課題を提出する

オブプロでは，課題を提出する際にプロジェクトごと提出してもらいます．ここでは，2.1，2.2で作成した「example」プロジェクトを例に，課題の提出方法を説明します．

まず，課題を提出する前に，以下の点を確認してください．

- フォーマット（Ctrl + Shift + F）を行っている（4.4で解説）
- インポート編成（Ctrl + Shift + O）を行っている（4.4で解説）
- 問題ビューが空である（3.2で解説）
- 実行時エラーが起こらない
- コメント（タイトルコメント，ブロックコメント，行コメント）がついている

※以上の点が守られていないと，採点中止になります．

1. exampleプロジェクトを右クリックして，メニューを開きます
2. エクスポートを選択して，エクスポート画面を開きます．
3. エクスポート画面で「一般」→「アーカイブ」の順に選択して，「次へ」ボタンを押します．

図：エクスポート画面

4. エクスポートするプロジェクト（ここではexample）にチェックが入っていることを確認して，宛先（保存先とファイル名）を入力します（下図参照）．
   ※ファイル名は「ログイン名_プロジェクト名.zip」にして下さい（例：t03794mh_example.zip）

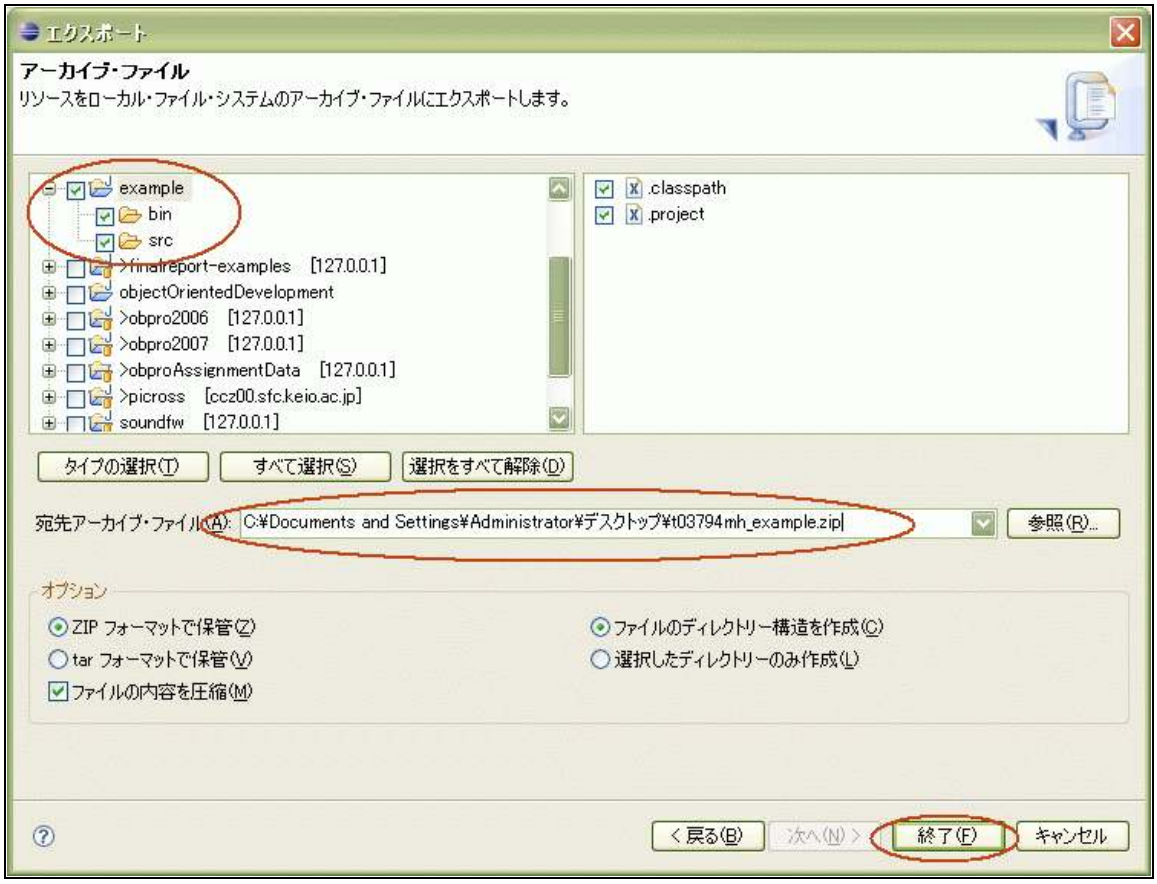


図:エクスポート設定画面

5. その他の設定は変更せずに「終了」ボタンを押すと，zipファイルが作成されます．このzipファイルをmoodleで提出して課題提出は完了です．おめでとう．

---

3. Eclipseの環境設定

ここでは，開発をする上で設定しておくと便利な設定について説明します．

※環境設定はすべて設定画面（メニューの中から「ウィンドウ」→「設定」の順に選択）で行います．

3.1 ソース保管時に自動でフォーマットを行う

この設定を行うと，ソースコードを保存時するときにソースコードのフォーマットが自動で行われます．これにより，フォーマットを忘れることがなくなり，常にきれいなソースコードを保つことができるようになり，第三者の可読性も向上します．手動でフォーマットを行う方法については，「4.4 ショートカットコマンド」を参考にして下さい．

1. 設定画面で「Java」→「エディター」→「保管アクション」の順に選択します．
2. 「保管時に選択したアクションを実行」→「ソース・コードのフォーマット」にチェックを入れて，「OK」ボタンを押して下さい（下図参照）．

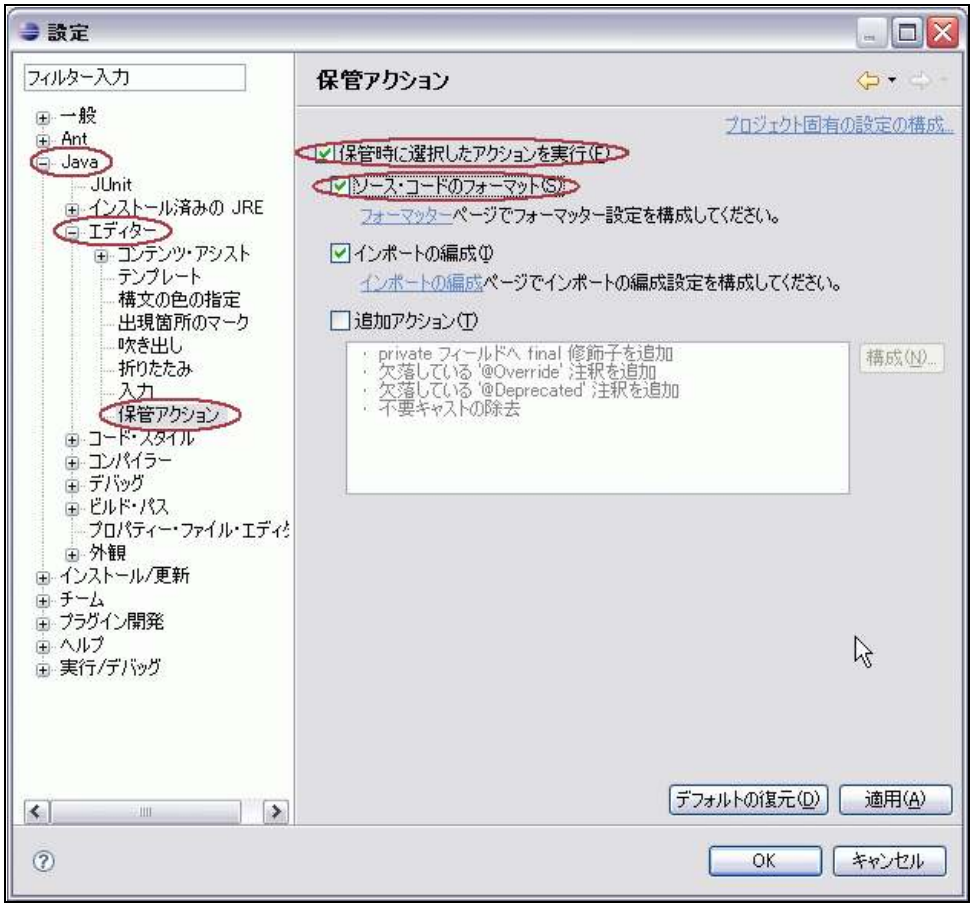
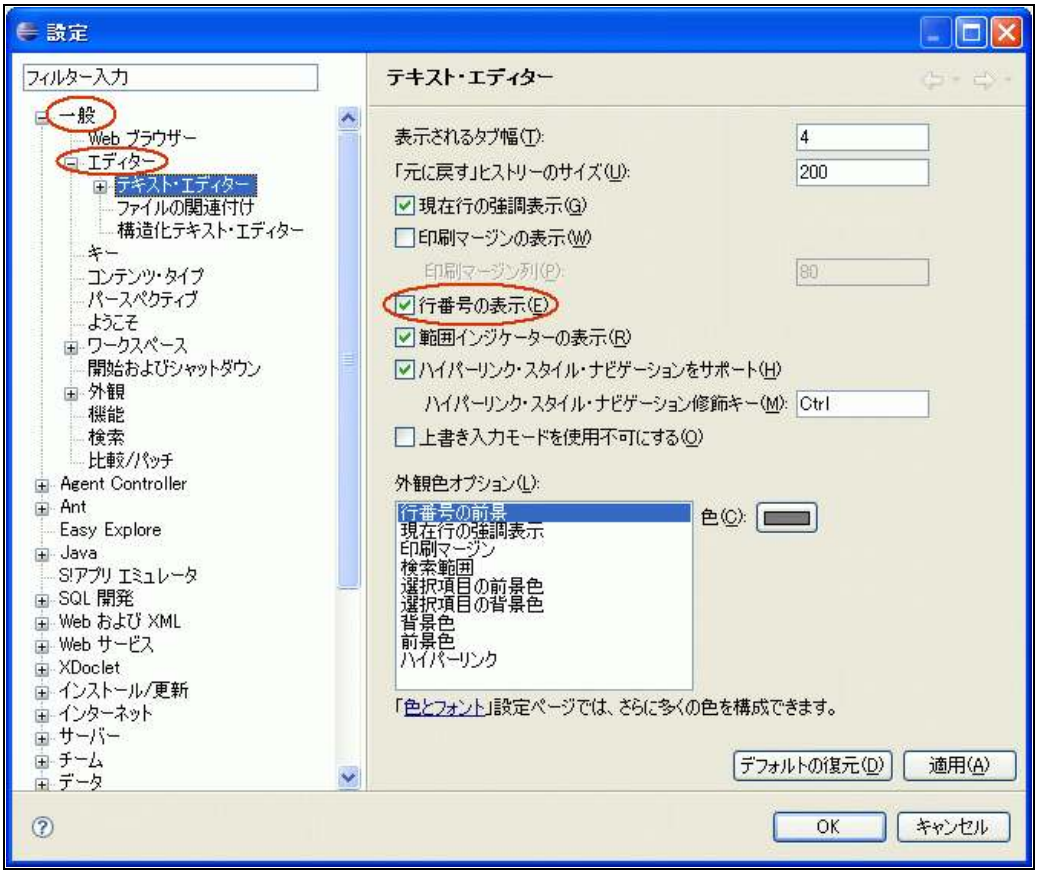


図: 保管アクション設定画面

3.2 行番号を表示する

コーディングを行う際に行番号が表示されていると便利ですが，Eclipseはデフォルトでは行番号を表示しません．以下の方法で行番号を表示させましょう．

1. 設定画面で「一般」→「エディター」→「テキストエディター」の順に選択します．
2. 「テキストエディター」設定画面で「行番号を表示する」にチェックを入れて，「OK」ボタンを押して下さい（下図参照）．



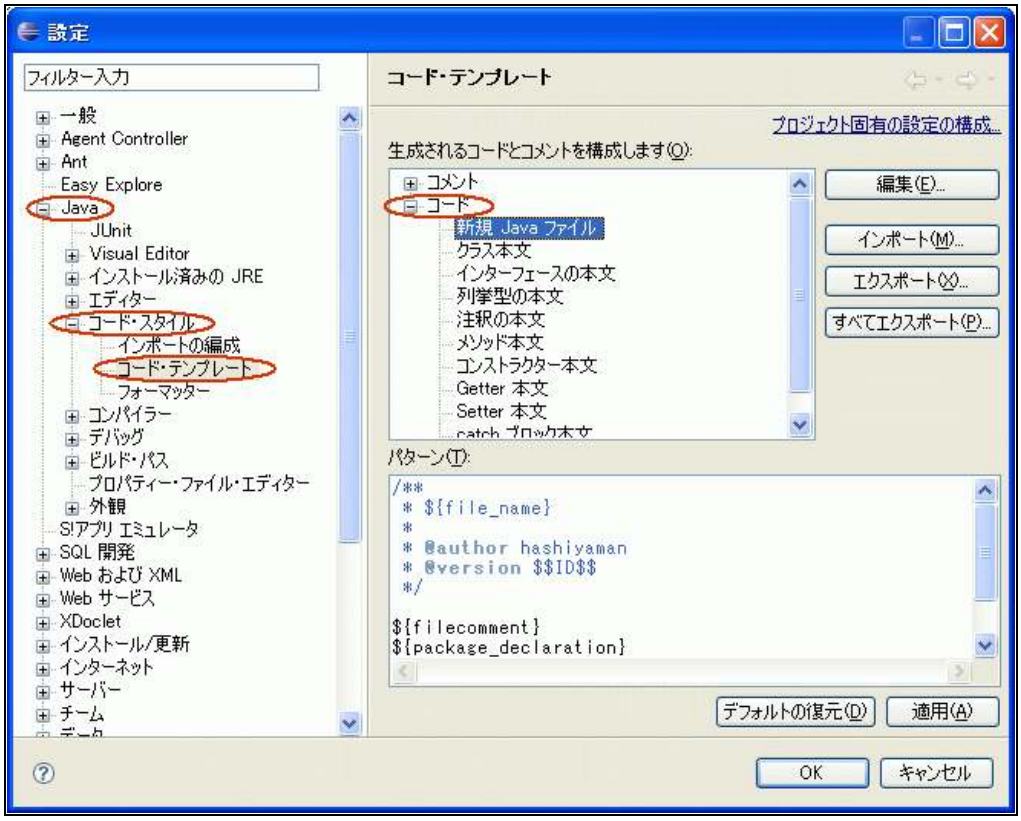
図：テキストエディター設定画面

3. これで，画面左に行番号が表示されるようになります．

3.3 タイトルコメントの自動生成

テンプレート機能を使うと，クラスを作成したり，メソッドを追加したときにテンプレートに従って自動的にコメントを付け加えることができます． 今回は，新しいクラスを作成したときに，自動的にタイトルコメント（著者，日付）を生成するテンプレートの設定方法を紹介します．

1. 設定画面で「Java」→「コード・スタイル」→「コード・テンプレート」の順に選択します．
2. 「コード・テンプレート」設定画面で「コード」→「新規Javaファイル」の順に選択し，「編集」ボタンを押して下さい（下図参照）．

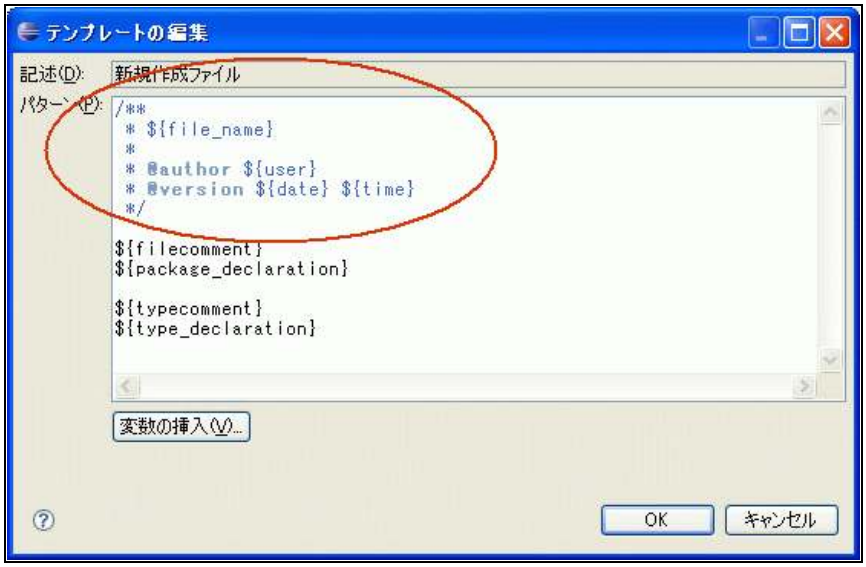


図：コード・テンプレート設定画面

3. 「テンプレートの編集」画面で，パターンに設定したいタイトルコメントを記入して，「OK」ボタンを押します（下図参照）．

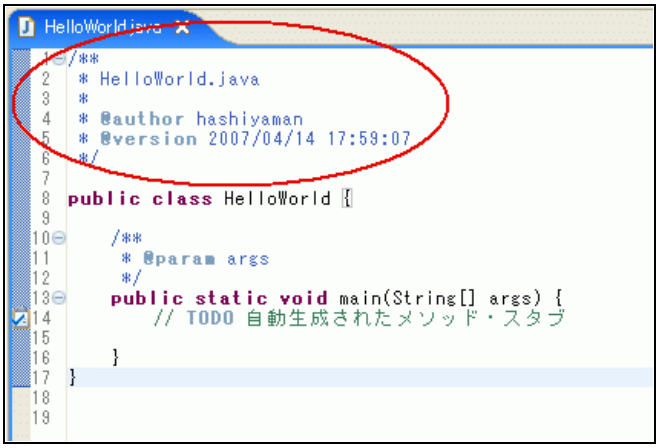
   以下に，タイトルコメントの例と利用できる変数の一部を紹介します．

   - ${file_name} ・・・作成するファイルの名前を取得します
   - ${user} ・・・現在パソコンにログインしているユーザーの名前を取得します
   - ${date} ・・・現在の日付を取得します
   - ${time} ・・・現在の時刻を取得します



図：コード・テンプレート編集画面

4. 「コード・テンプレート」設定画面に戻ったら，パターン画面に設定が反映されていることを確認して，「OK」ボタンを押します．
5. これで，テンプレートの設定は完了です．新しいクラスを作成するとタイトルコメントが自動生成されます（下図参照）．

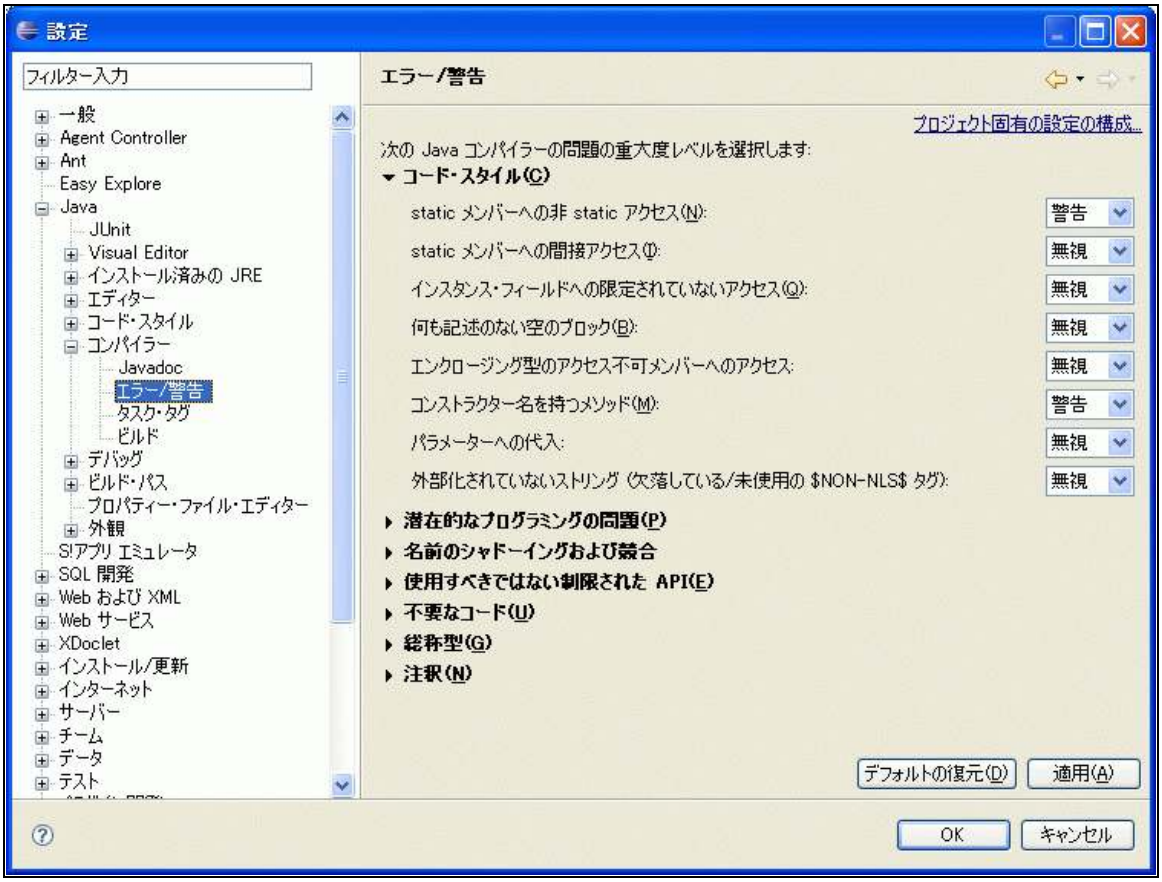


図：コード・テンプレート設定後

3.4 エラー・警告の設定

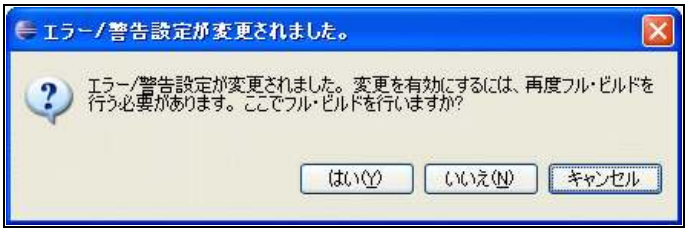
Eclipseでは，項目ごとに警告の重要度が設定されていて，それに応じて警告やエラーを出しますが，この設定は自由に変更することができます．

1. メニューの中から「ウィンドウ」→「設定」の順に選択し，設定画面を開きます．
2. 設定画面で，「Java」→「コンパイラー」→「エラー／警告」の順に選択します．
3. 「エラー／警告」設定画面で項目ごとに「エラー」「警告」「無視」の3段階を選択することができます（下図参照）．



図：エラー・警告の設定

4. 必要に応じて，各項目についてエラー・警告を設定します．
5. 設定を変更して「OK」ボタンを押すと，再ビルドを行うか聞いてくるので「はい」を選択してください（下図参照）．



図：再ビルドの確認

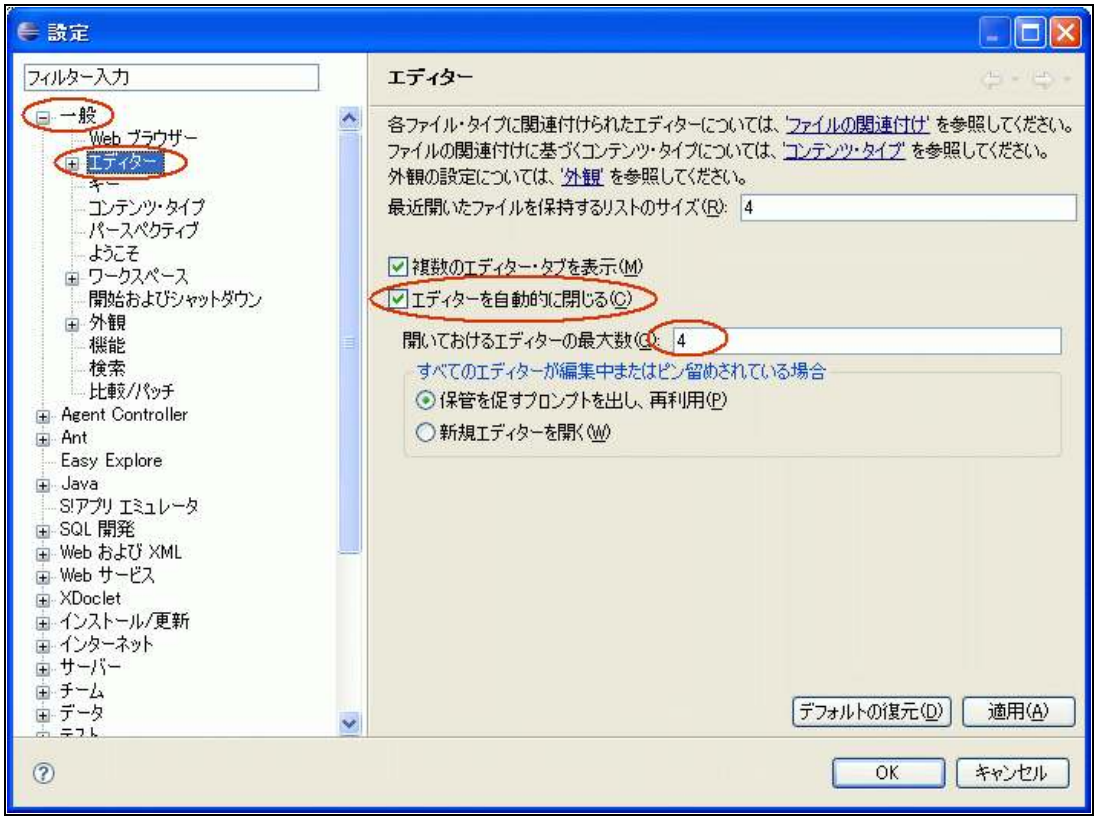
6. ビルドが終了すると，新しいエラー・警告の設定が反映されます．

3.5 エディタを自動的に閉じる

新しいファイルを開くと，画面上に新しいエディタ（タブ）が開きます．これを放置しているとエディタがどんどん増えて，目的のファイルを編集しているエディタを探すことが困難になります．そこで，一定数以上のエディタが開いたら自動的に古い順から閉じるようにする設定することを推奨します．

メニューの中から「ウィンドウ」→「設定」の順に選択し，設定画面を開きます．その中から「一般」→「エディター」の順に選択します．「エディター」設定画面で「エディターを自動的に閉じる」にチェックを入れて，「開いておけるエディターの最大数」に好きな数を入力してください（下図参照）．

＊「開いておけるエディターの最大数」は「4つ」がお勧めです



図：エディター設定画面

---

4. Eclipseを使いこなす

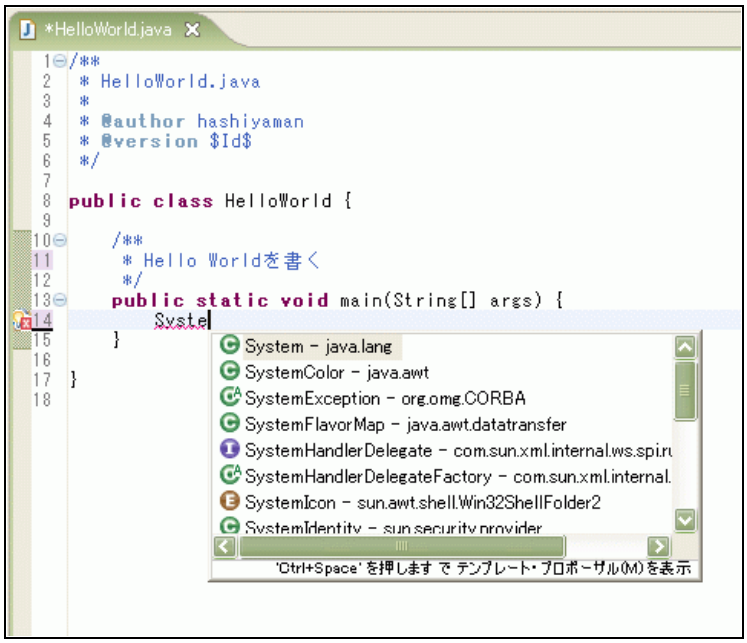
4.1 コンテンツ／コード・アシスト

Eclipseでは，コードの途中で「Ctrl + Space」を押すことで，残りのコードの予測して補完してくれたり，候補を出してくれたりします．この機能をコンテンツ／コード・アシストと呼びます．コンテンツ／コード・アシストで良く使う場面として，以下の3つを紹介します．

4.1.1 クラス名，メソッド名などを補完・予測する

クラス名，メソッド名の途中で「Ctrl + Space」を押すと，名前の補完，または修正候補をリストアップしてくれます（下図参照）．

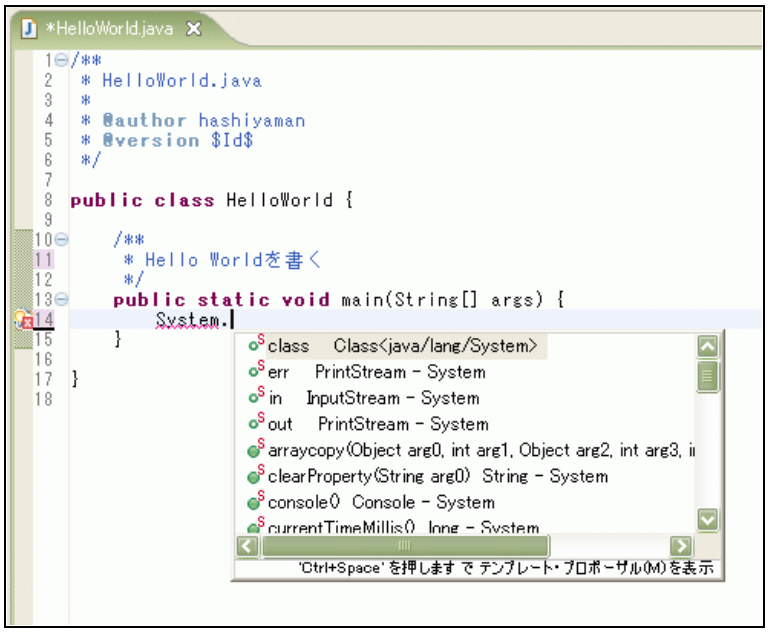
クラス名，メソッド名だけでなく，引数名やローカル変数名なども補完してくれるので，名前が長めの変数やスペルがうろ覚えなメソッドなどを入力する手間を省くことできます．



図：クラス名，メソッド名などを補完する

4.1.2 利用できるフィールド名やメソッド名を補完・予測する

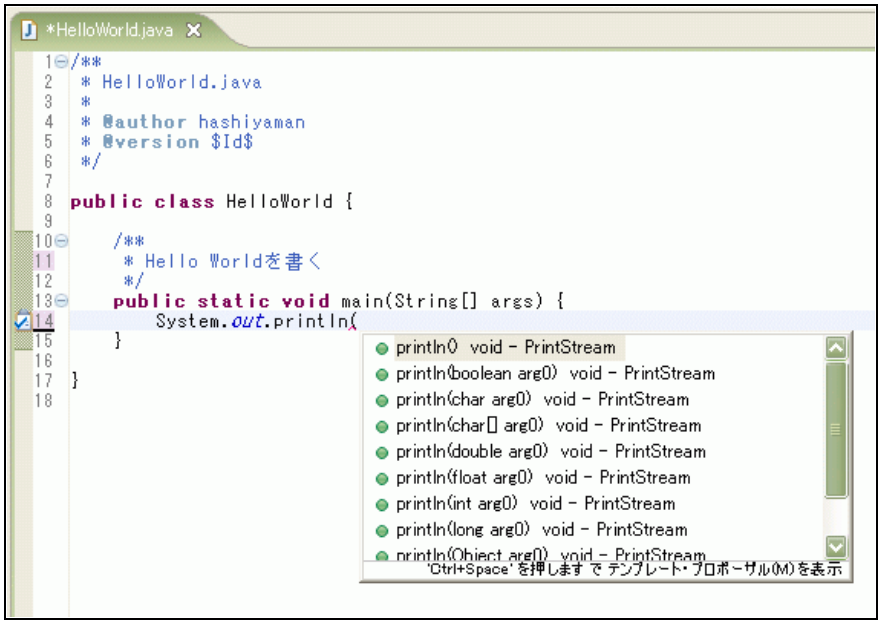
あるクラスの名前を入力したあとに「．（ピリオド）」を入力すると，そのクラスが持っているフィールド名やメソッド名などを補完してくれたり，候補を出してくれたりします（下図参照）．



図：利用できるフィールド名などを補完する

4.1.3 引数を予測する

あるメソッドやクラスを利用するときに，名前の後の開き括弧を入力したあとに「Ctrl + Space」を押すと，そのクラスやメソッドで必要な引数を教えてくれます（下図参照）．

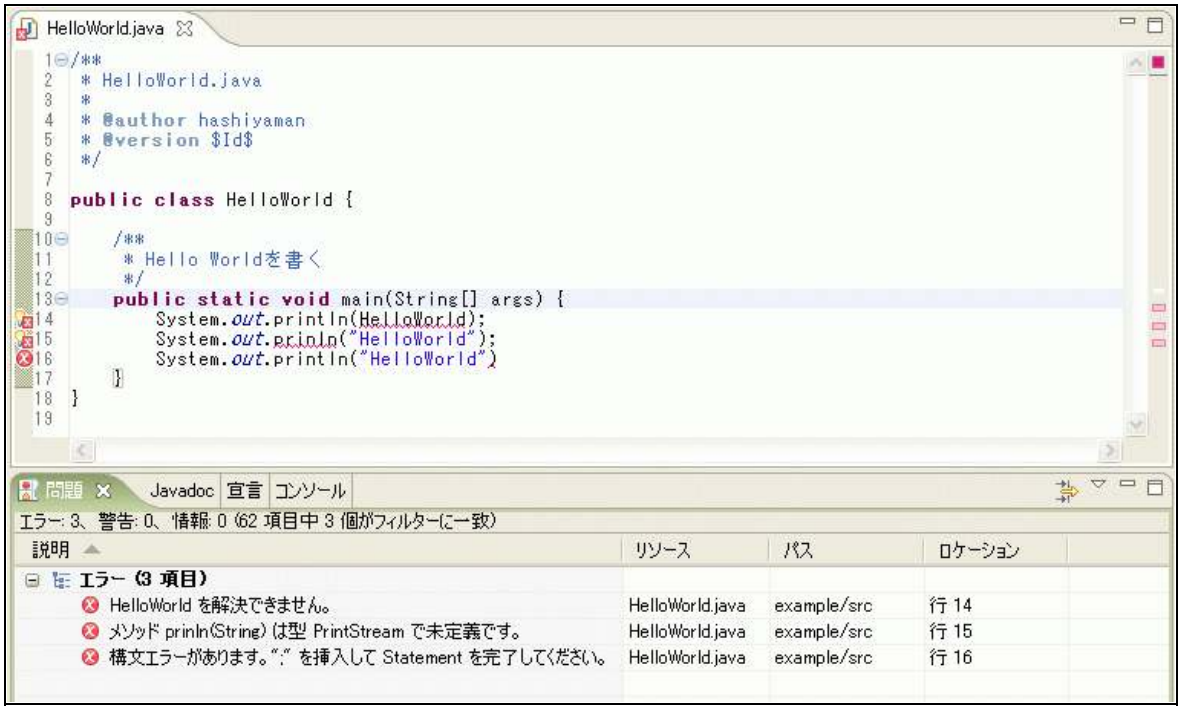


図：引数を予測する

4.2 問題ビュー

Eclipseでは，警告やコンパイルエラーが残ったままファイルを保存すると，画面下の問題ビューにコンパイルエラーの一覧が表示されます（下図参照）．警告やエラーの原因・発生箇所が一覧できる上，ビュー内の項目をクリックすると，問題の発生箇所にジャンプすることができます．

課題を提出する前には，この問題ビューを空にして下さい（警告・エラーを0にして下さい）．問題ビューは，メニューの中から「ウィンドウ」→「ビュー」→「問題」の順に選択することで開きます．



図：問題ビュー

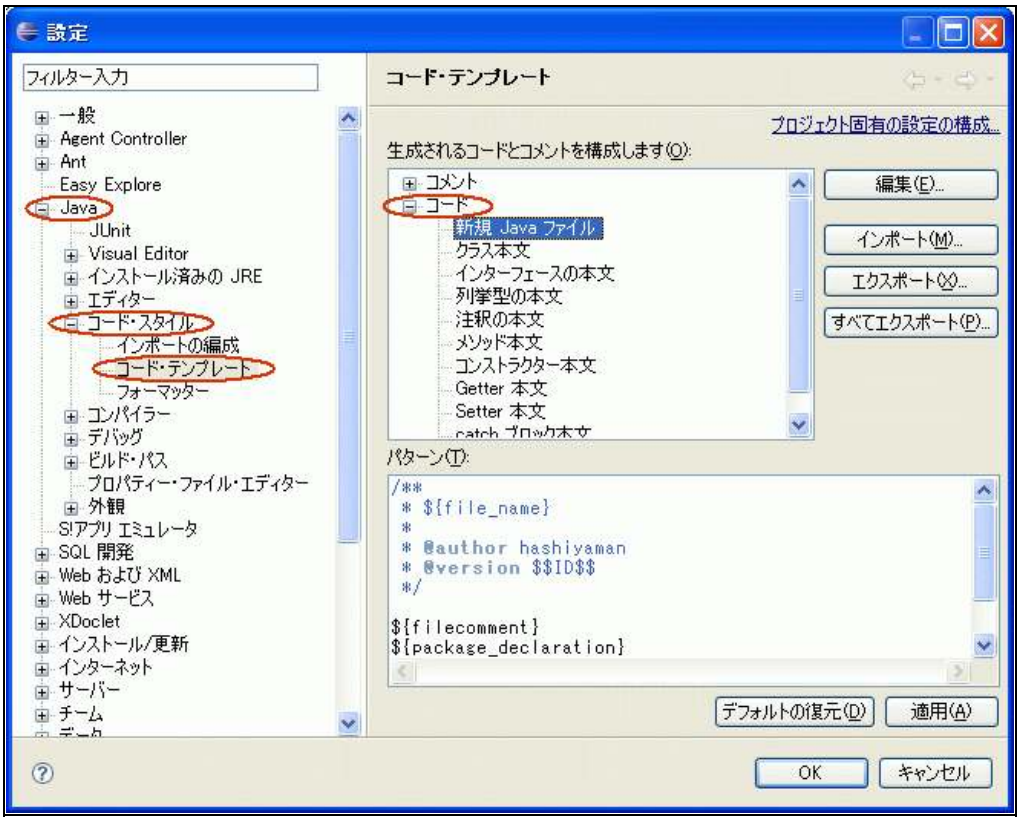
4.3 リファクタリング

リファクタリングとは，外から見たプログラムの振る舞いを変えることなく，その内部構造を改善するテクニックです．ここでは，良く使う代表的なリファクタリング機能を紹介します．

- **名前変更**

  クラス名やメソッド名，変数名などを変更する際に，その名前を参照しているすべてのエレメントを自動的に修正してくれます．

1. 名前を変更したいクラス名，変数名，メソッド名などを選択して，右クリックします．
2. メニューの中から「リファクタリング」→「名前変更」の順に選択します．
3. 名前変更画面で，新しい名前を入力して「OK」ボタンを押します（下図参照）．



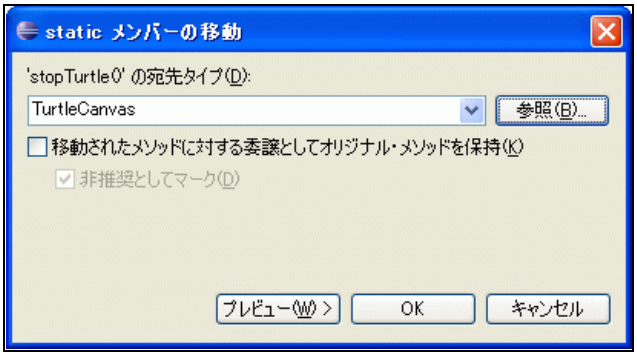
図：名前変更画面

4. これで，参照元も含めてすべての名前を一括で修正します．

- **移動**

変数，メソッドなどを別クラスに移動する際に，それらを参照しているすべてのエレメントを自動的に修正してくれます．また，あるクラスを別のパッケージに移動するときに，クラスにおけるパッケージの依存関係（import文など）を適切に修正してくれます．

1. 移動したいクラス，変数，メソッドなどを選択して，右クリックします．
2. メニューの中から「リファクタリング」→「移動」の順に選択します．
3. 移動先指定画面で，宛先タイプ（移動先のクラスなど）を指定して「OK」ボタンを押します（下図参照）．



図：移動先指定画面

4. これで，移動が完了し，すべての参照元を修正します．

4.4 ショートカット・コマンド

Eclipseには便利なショートカット・コマンドがたくさん存在します．そのうち，良く使うものを紹介し

ます．また，これらのショートカットについては，以下の方法で閲覧・編集することができます．

1. メニューの中から「ウィンドウ」→「設定」の順に選択し，設定画面を開きます．
2. 設定画面で，「一般」→「キー」の順に選択します．

- **フォーマット（Ctrl + Shift + F）**

ソースコード内の括弧の対応やスペース（空白），インデントなどを一度に整えてくれます．ソースコードの可読性が格段に上がるので，課題を提出する前には必ずフォーマットを行うクセをつけて下さい．フォーマットの設定を変更する場合は，以下のように行います．

1. メニューの中から「ウィンドウ」→「設定」の順に選択し，設定画面を開きます．
2. 設定画面で，「Java」→「コード・スタイル」→「フォーマッター」の順に選択します．
3. 「フォーマッター」設定画面でプロファイルを選択し，「表示」ボタンを押すと詳細な設定ができます．

※プロファイルは「Eclipse 2.1[ビルドイン] 」がお勧めです

- **インポートの編成（Ctrl + Shift + O）**

クラス内で利用していないimport文を削除したり，省略したimport文を具体的なものに置き換えてくれたりといったインポートの整理を行ってくれます．フォーマットと合わせて，逐一行うクセをつけましょう．

- **コメントの切り替え（Ctrl + /）**

選択範囲を一度にコメントアウト（コメント化）することができます．また，コメントアウトされている範囲が選択されていた場合，これらを元に戻すことができます．

- **1行削除（Ctrl + D）**

現在カーソルがある行を削除します．

- **宣言を開く（Ctrl + 左クリック or F3）**

メソッドや変数名にマウスカーソルを合わせた状態でCtrlキーを押すと，名前に下線がつきます．この状態で，マウスの左クリックを押すと，そのメソッドや変数の宣言にジャンプすることが出来ます．

4.5 注意すべき（上級者向け）機能

4.5.1 コンパイルの修正候補の表示

Eclipseでは，修正が予測できるコンパイルエラーがあると，該当箇所の左に電球マークが表示されます．これが表示されている時に電球マークをクリックするか「Ctrl + 1」を入力すると，修正候補一覧が表示されます（下図参照）．この状態で修正候補を選択すると，コンパイルエラーが選択した修正通りに修正されます．

※修正候補が意味している内容が分からずにこの機能を使うと，修正されたコードが理解できなくなる恐れがあります．そのため，この機能は修正候補の内容が正しく理解できる人以外は利用しないで下さい．

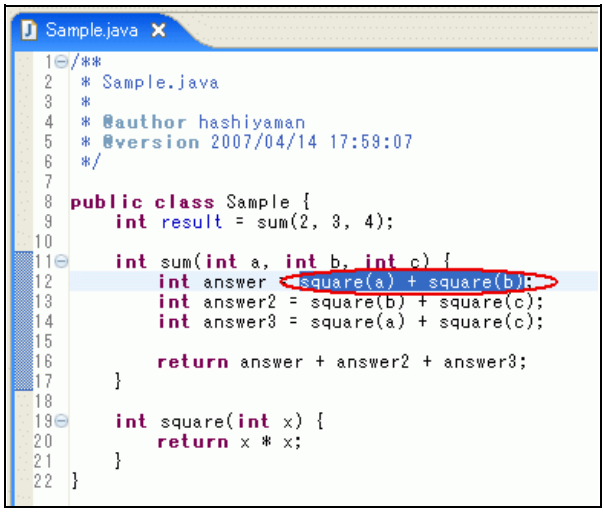
図：コンパイルエラー修正候補一覧

4.5.2 メソッドの抽出

選択範囲から式やステートメントを抜き出し，メソッドとして宣言するとともに，元の式またはステートメントは，抽出されたメソッドに置換してくれます．このとき，必要なパラメータやメソッドの型も自動的に抽出し，宣言してくれます．さらに，抽出した部分と同じコードを，抽出するメソッドで置換できます．

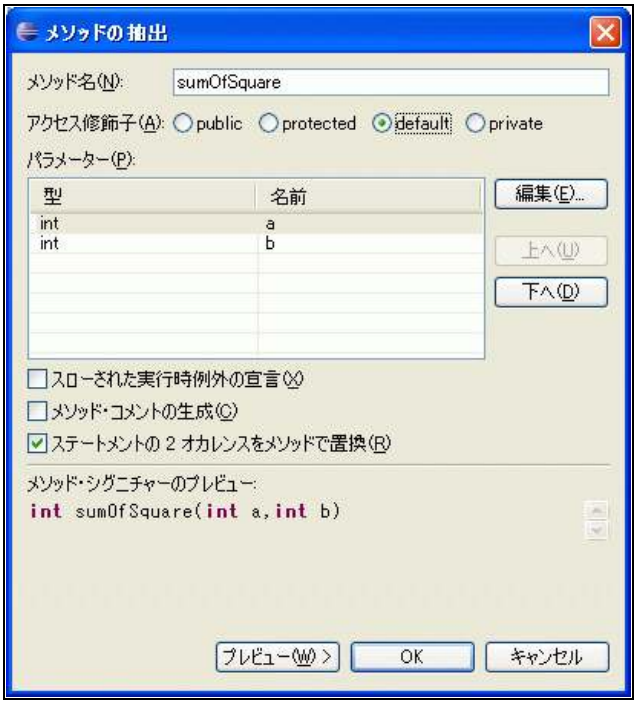
※下記の説明で何をやっているのか分からない人がこの機能を使うと，コードが複雑になり混乱する恐れがあります．メソッドの抽出がどういうことか正しく理解できる人以外は利用しないで下さい．

1. メソッドを抽出したい範囲を選択して，右クリックします（下図参照）．



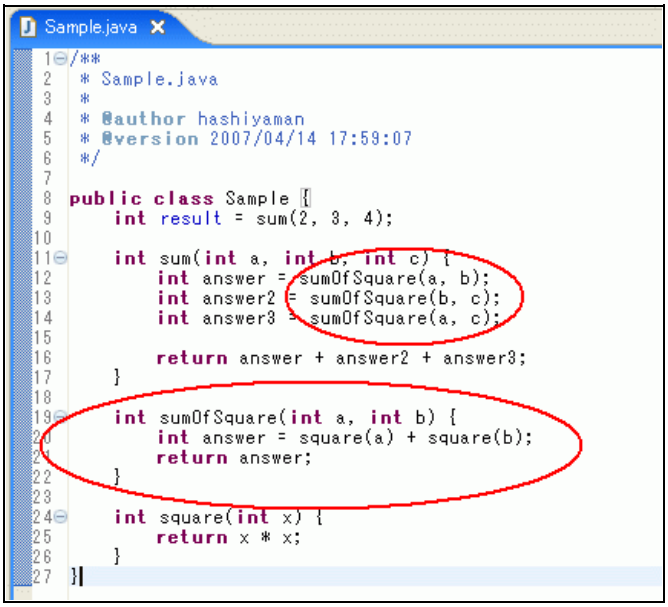
図：メソッド抽出範囲の選択画面

2. メニューの中から「リファクタリング」→「メソッドの抽出」の順に選択します．
3. メソッドの抽出画面で，メソッド名を入力して「OK」ボタンを押します（下図参照）．



図：メソッド名入力画面

4. これで，メソッドの抽出が完了し，重複したコードもすべて置換されます（下図参照）．



図：メソッド抽出後画面

---

5. その他の情報

5.1 リンク集

- Eclipse 本家
- Eclipse Wiki