

FreeMind プログラミングの支援機能

Support Function for FreeMind Programming

川原拓也[†] 武田利浩[†] 平中幸雄[†]

Takuya Kawahara Toshihiro Taketa Yukio Hiranaka

1. まえがき

FreeMind[1]のようなマインドマップツール[2]を使って、プログラム作成をすると、プログラミングが快適に行える。具体的には、

- ・詳細隠蔽機能により、全体構造が把握しやすくなる
- ・前後方向と詳細方向の2次元的な把握ができる
- ・階層構造表示により行頭インデントが不要になる
- ・マウス操作により編集が直感的に行える

しかし、マインドマップツールには開発環境としての機能は無く、作成したプログラムの実行では、マインドマップファイルデータを一旦テキストファイルに変換した上で、コンパイル、実行の手順を踏むことになる。そのため、コンパイルエラーや、実行時エラーの場所を特定するには、変換したテキストファイル上で、行番号を元に探す必要がある。せっかくマインドマップツールで快適にプログラミングが行えても、デバッグ作業では、旧来のテキストファイル上で行わないといけない。

そこで、コンパイルや実行までをマインドマップツールで行える工夫を行った。その方法と結果について報告する。

2. FreeMind を使ったプログラミングについて

マインドマップツールのひとつである FreeMind を使ったプログラミングでは、図 1 のように、マインドマップのノードにプログラム文を書き込み、上から下へとプログラムを書き進める。C や Perl、Java 等でプログラムブロックを表す { } で囲まれる部分は、階層下位のノードに記述する。FreeMind では、上位ノードをクリックするたびに、下位ノードの隠蔽もしくは表示が行える。この機能により、詳細を隠すことで全体構造を見やすしたり、必要な部分だけを展開表示し、編集することができる。

マインドマップツールの作成するファイルは一般に拡張子が mm で、内部は XML 記述となっている。この XML 形式ファイルを、研究室で作成した mm2pl と呼ぶプログラムでテキスト形式のプログラムファイルに変換して、コンパイラにかける。mm2pl は Perl プログラムを対象とした変換プログラムであるが、同じ構造の C や Java でも使用できる。この mm2pl では、マインドマップファイルの XML データから、ノードのテキスト部分を順に取り出すだけであるが、FreeMind の編集画面で表示されるツリー構造について、ノードが階層下位になると、{ のみの行を挿入した後、行頭段付け文字数を増やす。ノードの階層が 1 段戻ると、行頭段付けの文字数を元に戻し、} のみの行を挿入するという処理を行う。

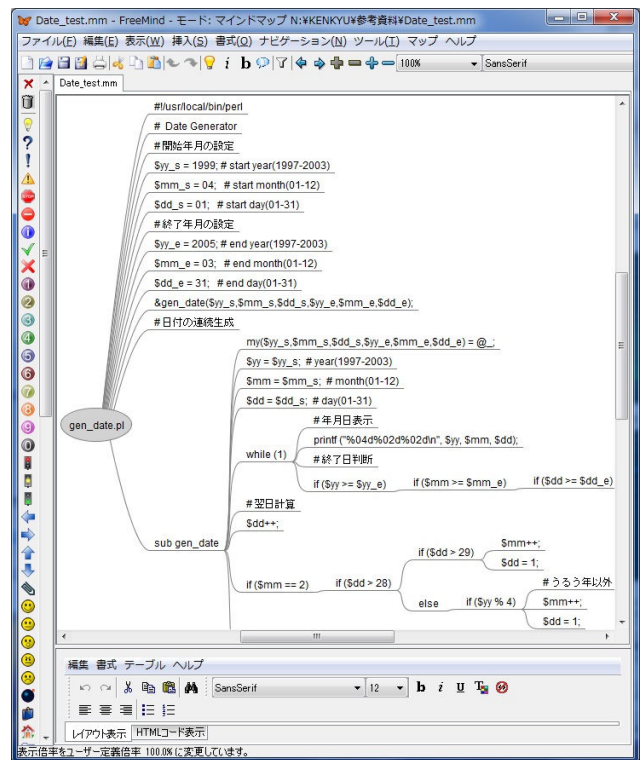


図 1 FreeMind で表示されるプログラム例

3. FreeMind の構成と修正点

本研究では、FreeMind 1.0.0_Beta_7 を利用した。大きく分けて 3 つのプログラム群に分けられる。

```
freemind.controller.Controller
```

```
freemind.controller.MenuBar
```

```
freemind.view.mindmapview.MapView
```

Controller では FreeMind の各種機能の記述が行われており、MenuBar ではメニューバーについての記述を、MapView では FreeMind でマインドマップを表示するための機能について記述されている。

今回は、主に Controller クラスを改変し、FreeMind にプログラム開発環境を付加した。具体的には、FreeMind のファイルメニューに「保存して実行」機能を追加した。これに合わせ MenuBar クラスに対してボタンの登録と、言語ファイルに表示する文章の登録を行った。その機能の動作概要は図 2 の通りである。

今回の「保存して実行」は Perl プログラムの実行を対象とした。Perl の場合はインタプリタ動作なので、エラーが発生した場合、解釈上のエラーと実行時のエラーが

[†] 山形大学 Yamagata University
Yamagata College of Industry and Technology

同様に、出力されることになる。そのエラーを、FreeMind 上に反映するため、エラーを一旦ファイルに書き出し、その内容からプログラム文を特定し、FreeMind のファイルデータ中で表示色を変えることでマークする。そのファイルを FreeMind が再読み込みすると、FreeMind の編集画面上に色付けされて、表示されることになる。

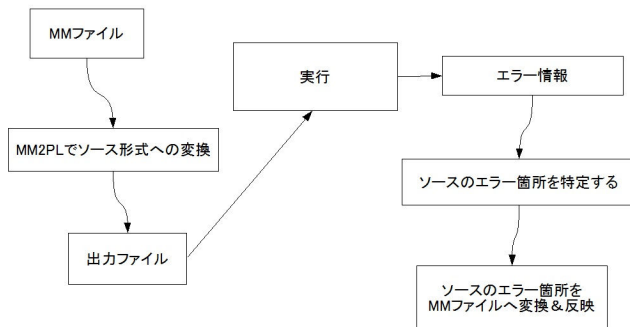


図2 「保存して実行」の動き

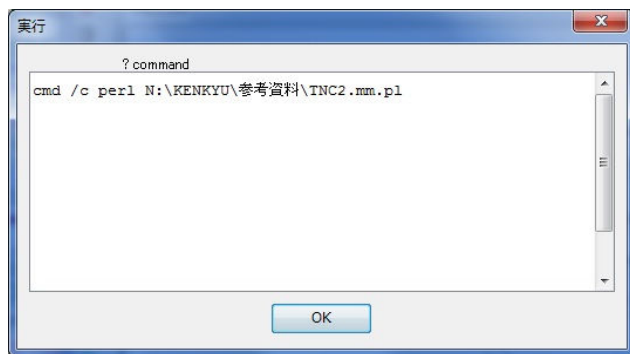


図3 「実行」コマンドの表示

ソースコードへの変換は、Controller クラスに変換を行うクラスを追加して実現した。ファイルを保存する動作については、FreeMind にあるほかのクラスと干渉を起こさないよう FreeMind に備わっている保存動作に、できる限り沿うようにして行うこととした。mm ファイルからソースファイルへの変換動作を行う部分には、ProcessBuilder クラスを利用し、mm2pl を呼び出すこととした。

図3は、「実行」動作時に表示される確認画面で、コマンドの修正やパラメータの追加が可能となる。またこのとき、java から起動した外部プロセスはそのままでは標準出力を行えないこと、出力が多い場合エラーとなることから出力部をスレッド化した。

4. エラーの反映

プログラムの実行段階でエラーがあった場合、

- 1) エラーメッセージの行数を抽出する
- 2) 行数からテキストファイルのプログラム文を抽出する
- 3) その文に対応する mm ファイルのノードを特定する
- 4) 特定したノードのテキスト色を XML データで変更する

以上の結果をmmファイルへの書き込むと、FreeMind がファイルの変更を検知し、再読み込みの確認ダイアログが表示される。これに「はい」で答えると、再読み込みの結果、エラー箇所を色指定したファイルデータに基づき、画面が更新表示される(図4)。エラー箇所が赤くなりユーザーがエラーの修正を行う際の目印となる。今回は、エラーの発見と反映までのみ実装したため、エラー修正の際にはユーザーの手によって赤く変更された部分の色を、黒色に変更する必要がある。

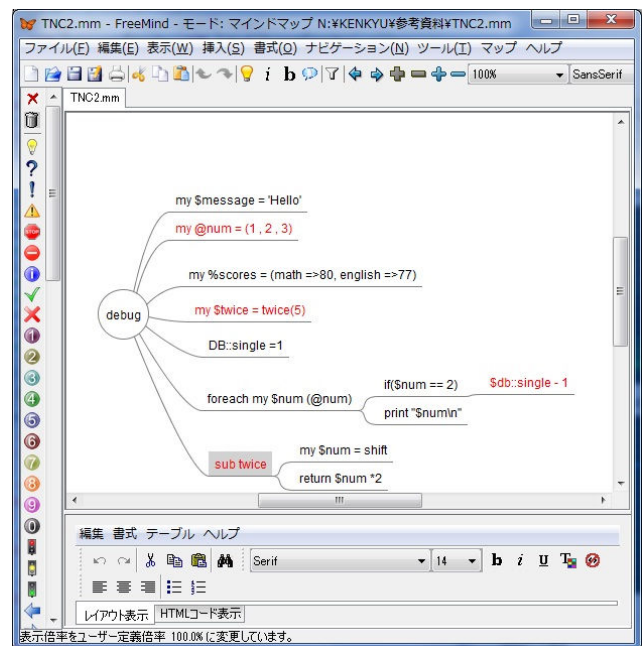


図4 エラーの表示例

5. まとめ

FreeMind にプログラム開発のための機能を付加し、マインドマップツールを利用したプログラム作成がより実用的に実現できるようになった。

今回の「実行」機能では、70 行ほどのプログラムを実行した場合、全体の処理が約1秒ほどかかった。これはエラーの探索で効率の悪い部分があるためで、改善の余地がある。また、今回は Perl に絞って機能実現したが、他の言語に対応するのも可能と考えている。

参考文献

- [1]FreeMind, <http://freemind.sourceforge.net>
- [2]mind map tools, https://en.wikipedia.org/wiki/List_of_mind_mapping_software
- [3]FreeMind Programming, <http://eatz.yz.yamagata-u.ac.jp/FreeMindProgramming/FreeMind%20Programmin%20g.htm>, ここに mm2pl が置いてある