GIMP でいこう

藤田裕二[1]

1 はじめに

このプログラムでは, 現在 バージョン 1.1.x として開発中の, 次期 GIMP であるバージョン 1.2 活用のためのノウハウを, リリースに先行する形で紹介します.

2 ChangeLog

1.0 から比較して, 1.2 で新しくなったところをざっと紹介します.

3 GIMP 解剖 前編

プログラムの全貌を掴むには, 個々の機能を数え上げるよりも, どのようなデータを扱っているのか, ということに注目した方がわかりやすいことが多いようです.

3.1 パス

ここでは, まず バージョン 1.2 で新しく加わったデータであるパスについて, プログラム内部での取扱などに必要に応じて踏み込みつつ, 詳しく紹介します. パスの取扱を掴むことは, 1.0 からの移行の重要なポイントの一つとなるでしょう.

3.2 drawables

drawable は, GIMP において描画可能なデータ一般を指す用語です. drawable は大きくレイヤとマスクに分類されます. 1.0 から比較して, drawable 自体には大きな変化はありませんから, 1.0 からの移行は無理なく可能であると思われます. 1.2 に特化した項目はありませんが, GIMP の能力を引き出す上で drawable の把握は本質的に重要です. GIMP のパワーの源は, あらゆる描画関数があらゆる drawable に適用可能であるということに由来するからです.

[1] 「日本の Linux 情報」(www.linux.or.jp) 製作スタッフ. 東京工業大学 博士課程在学

4 GIMP 解剖 後編

ここでは, GIMP の死活を制する設定である tile-cache-size を理解するために, GIMP が独自にアプリケーションレベルで実装している, 画像処理に特化したメモリ管理について紹介します.

GIMP は画像データを線形の一次元 / 二次元配列ではなく, タイルという単位で扱います. この機構はプログラムのもっとも下層の基礎を構成しているのですが, その動作を決定する上で本質的なパラメータを, ユーザが直接設定するという仕様となっています. こう言ってはなんですが, かなりアレゲですね. この設定が不適切な場合は, 極めて切ないパフォーマンスしか得られないので, tile-cache-size の設定は, GIMP を活用するうえで避けては通れません.

タイルの管理はバージョン 1.2 ではスレッド化されました. 少ないメモリで大きなデータを扱う際のレスポンスが非常に改善されています. これを経験したら, もう 1.0 には戻れないでしょう.

5 script-fu

GIMP では LISP の方言のひとつである scheme によって, プログラミングが可能です. GIMP のパワーを全開にまで持って行くには, script-fu の習得は必須であると思います. scheme 自体, 書いていて非常に楽しいプログラミング言語であり, しかも実行の結果が直ちに GIMP による描画となって現れるとあっては, もうやるしかないでしょう.

ここでは, そんな楽しい script-fu を実際に書く上で必要なノウハウを幾つか紹介します.

6 GIMP の未来

最後に GIMP の未来として, マーケティング (?) と技術面を少し考えます.