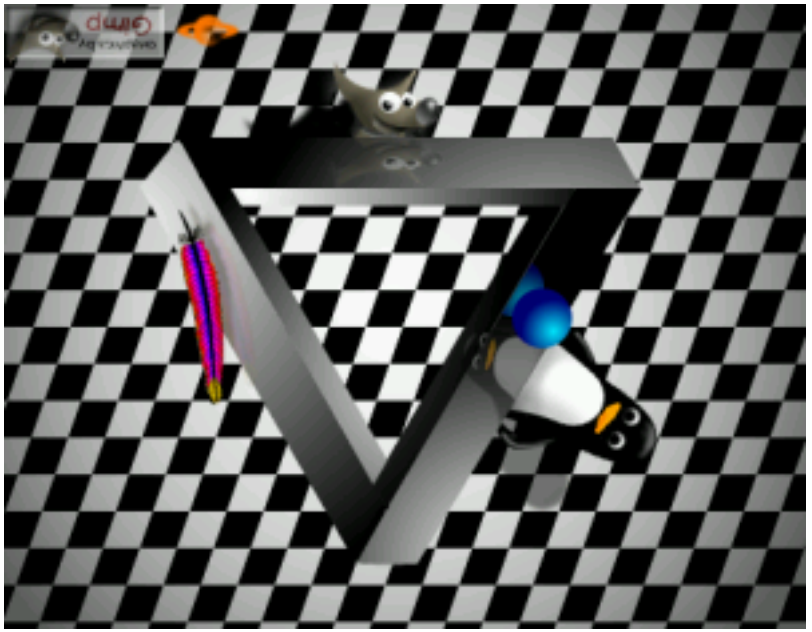
GIMPで いこう



藤田裕二 ←「日本の Linux 情報」製作スタッフ
yuji@wl.me.titech.ac.jp

GNU Image Manipulation Program

ぎんぷ

テーマ？

□**1.1.x** 乗り換えへの教唆扇動

□**GIMP** の布教

もくじ

- **GIMP** の歴史
- **ChangeLog**
- **GIMP** 解剖 前編
- **GIMP** 解剖 後編
- **script-fu** を書こう
- **GIMP** の未来
- 実演**(**時間があったらね**)**

GIMP の歴史

原作者：**Spencer Kimball と Peter Mattis**

- **0.54 (1996年2月)**
  - **motif, アルファチャネル, 任意回の undo, レイヤ無し**
- **0.6x (1996年7月)**
  - **Gtk, 基本的なレイヤを既に備えていた.**
- **1.0 (1998年6月5日)**
  - **Gtk+**
  - 多彩な描画ツール
  - **tile による, 独自のメモリ管理**
  - レイヤ
  - チャネル
  - **undo 無制限**
  - スクリプト **(scheme, perl)**
  - **XCF フォーマット (version 0, 1)**

GIMP の現在

相補的な2つの開発ブランチ

- **1.1.x - 1.2 (1998-2000)**
  - 基本的な部分の大規模な改変は抑えて, 機能の不足や不具合の修正.
- **hollywood branch (1998-)**
  - 機能の追加よりも, 基本的な作りの見直し. 特に画像データや描画関数を更に抽象化.

ChangeLog

ChangeLog - 1.2 の新しいところ

- □新しい描画ツールの追加
- □タブレットのサポート
- □パス
- □**Idle Renderer**
- □タイルキャッシュのスレッド化
- □国際化
- □クイックマスク
- □新しいプラグイン
- □パラサイト
- □セッションマネジメント
- □インターフェースの改善, 洗練.

全体の印象

1.0 に比べて

□幾つか本質的な変更がある
- ○パスの追加

□本質的な変化は多くない - **1.1** で出来ることの多くは **1.0** でも(凄く)頑張ればできる
- ○**1.0** から無理なく移行できる

□全面的に洗練された
- ○インターフェースの改善
- ○レスポンスの向上

結論.**1.1** を使うと **1.0** には戻れない.

追加された描画ツール

□インク　タブレット対応の新たなペイントブラシ
□覆い焼き ハイライトや陰影をつける
□指　絵の具風, 水彩ふう

全面的に指で描いた墨絵 (のつもり)

新しいプラグイン

- 水彩ふう色選択
- **newsprint**
- アンシャープマスク
- コンテキスト ヘルプブラウザ
- 選択範囲 → ベジェパス
- ダイナミックテキスト - フォント, 文面, レイアウトを憶えている
- **jpeg** プラグイン - プレヴューできる

これはプラグインじゃないけど...

- 三角形色選択

その他

- 様々な **drag & drop**
  - ファイルマネージャから **drag** でファイルを開く
  - レイヤ窓のあらゆる **drag & drop**
- 適切なデフォルト
- ユーザ設定項目
  - 単位系
  - 補完アルゴリズム - 変形ツール, 拡大 / 縮小
    - **nearest neighbour**
    - **linear**
    - **bicubic**

用語

用語

解剖コーナーに先だち, 関連する用語の紹介.

□チャネル

○画像データを構成する要素. 赤, 緑, 青 など

□マスク

○レイヤ窓の「チャネル」等, 選択範囲を保持するデータ

○レイヤ窓のアレは“「 」”で括って表記

□アルファチャネル

○透明度に関するチャネル

アルファチャネルを持つ文字

GIMP 解剖 (前篇)

GIMP 解剖(前編)

機能ではなく，**GIMP** の扱うデータに注目し, プログラムの全貌をつかむ.

□パス **(**バージョン **1.1** から**)**

□**drawable**

パス

bezier 曲線パス

種類
□開曲線/閉曲線
□連結/非連結

使い方
□選択範囲の生成
□描画ツール (ペイントブラシ, 指, ケシゴムなど)の適用

パスの中身

- **path** は点の線形リスト
- 点は **3**個の **float** からなる構造体
  - **2**つは座標, 残る一個は点の種類 **(**タイプ **1, 2, 3)**
- タイプ**1** アンカポイント
- タイプ**2** コントロールポイント
- タイプ**3** 非連結成分の先頭アンカポイント

- 開 / 閉曲線パス
  - 閉曲線 **“122”** の自然数回繰り返し
  - 開曲線 **“122”** の自然数回繰り返しの最後に **"12"**
- 非連結パス
  - リストにタイプ **3** の点が出現

パスのセーブ

- **XCF** ファイルでは **Image** 付きのプロパティ

- サポート外のプロパティは, 読み込みの際に無視されるので, **path** を持つ **XCF** ファイルも **1.0** で読み込み可能.

パスにある制限

- **undo** なし
- 閉曲線→開曲線
  - ハサミボタン無し！
- 拡大 / 縮小, 反転, 回転 なし
- 開曲線の連結成分は **1**つ
- 分岐もダメ **(**線形リストだから当然？**)**

次は

draw

drawables

GIMP において描画可能なデータ一般をさす

種類

- □ レイヤ**(1-4 channels)**
  - ○普通のレイヤ
  - ○フローティングセレクション
- □ マスク**(1 channel)**
  - ○「チャンネル」
  - ○クイックマスク
  - ○レイヤマスク
  - ○セレクションマスク

drawable の性質

- **8bit/channel**

- 操作可能な **drawable** は一つ **- active drawable**

クイックマスク

- **1.1** で新しく導入
- 画面左下の赤ボタンで生成
- 普通に描画可能
- レイヤ窓の「チャンネル」ページからもアクセス可能
- 画面左下の点線ボタンでセレクションマスクへ変換

結論

- **1.0** に比較して, 増えたデータは パス
- 移行のカギは
  - パス
  - 増えた描画ツール / プラグイン

GIMP 解剖 (後篇)

GIMP 解剖 後編

GIMP が独自に持っているメモリ管理の話

□タイル
□タイルキャッシュ
□**tile-cache-size**

GIMP の死活を制する設定に関連
1.1 で改善

タイルとは

drawable の画像データ本体を構成する単位

□サイズ **64*64 pixel**

全てはタイルへの操作
□確保
□描画
□開放
□**undo** スタックに積む **....**

GIMP のメモリ管理機構

- 管理の単位はタイル
- 使っていないタイルはディスク (**~/.gimp**以下の **swap** ファイル)に配置
- アプリケーションが作る仮想タイル空間

疑問

- 何故 **OS** にメモリ管理を任せず, アプリケーションが行っているのか？
- **OS** のメモリ管理だけで, 何か不都合があるのか？

メモリについて

OS にメモリ管理を任せた場合

- □利点 **(on unices)**
  - ○速い
  - ○信用できる
  - ○自分でやらなくていい
- □欠点
  - ○**OS** の用意したアドレス空間しか使えない

GIMP で必要なメモリのサイズは？

- □**A4 600dpi** 一枚で **140MBytes. 512M** 積んでもレイヤ **3**枚

タイルの効用

システムが用意するアドレス空間と独立に, 独自のアドレス空間を構築.

- 使わないタイルは **~/.gimp** 以下のスワップファイルに保持
- **GIMP**のアドレス空間 **=** ファイルサイズ **+ OS** の用意するアドレス空間
- **2GBytes = A4(600dpi) x 14**枚

データ処理の効率化

- タイル化により, 画像データの必要な部分だけをメインメモリにキャッシュして処理 **-**タイルのキャッシング

1.2 における改善

1.0.x - スワップ中は全ての処理が停止

タイルのスワップは別のスレッドに
□スワップ中でも描画関数の発行をはじめ, 各種のイベント処理が可能

アルゴリズムの改善により, キャッシュの効率が向上
□キャッシュがあふれてしまうまえに, 不要なタイルをスワップ

大きなデータが扱いやすくなった

tile-cache-size の選び方

OS から貰ったメモリ上に配置するタイルの合計
<ToolBox>/File/Preferences.../Environmentで設定

ダメな例
□小さすぎ
○**1.1.17** 以前のデフォルトは, 僅か**10MBytes**
○いくらメモリを増やしても, これを変更しないと増やしたメモリが使われない
□大きすぎ
○**OS** の **swap** から **~/.gimp** へ引っ越しはダメスギ
上の範囲内で, 自分で **GIMP** の優先度を評価して決める.

お勧め
□ユーザが自分だけ **-- free** を全部くれてやる

スクリプト ふー

script-fu

なんと, **scheme** で, しかも **GIMP**に画を描かせることが！
語源(？) は **kung-fu (カンフー)** → 達人は黒帯

□入門 **Mike Terry** の黒帯道場**(GUM**収録**)**を
`script-fu-register`という関数が紹介されているところまで読む.

以下, 黒帯道場を補完する情報.

いちいち car

関数の仕様に若干問題アリ

```
(set! img (gimp-image-new 64 64 0))
(gimp-display-new img)
```

このコードを評価すると **wrong type argument** でコケる.

script-fu では

```
(set! img (car(gimp-image-new 64 64 0)))
(gimp-display-new img)
```

こう書かないとダメ. 知っていてもつい間違える.

汎用 scheme 関数

`gimp-hoge-fuga`以外にどんな関数が使えるのか？

- **GUM に掲載. 付録のwrite_scriptfu_reference.html**
- **plug-ins/script-fu/interp_slib*(ソースコード) を見る.**

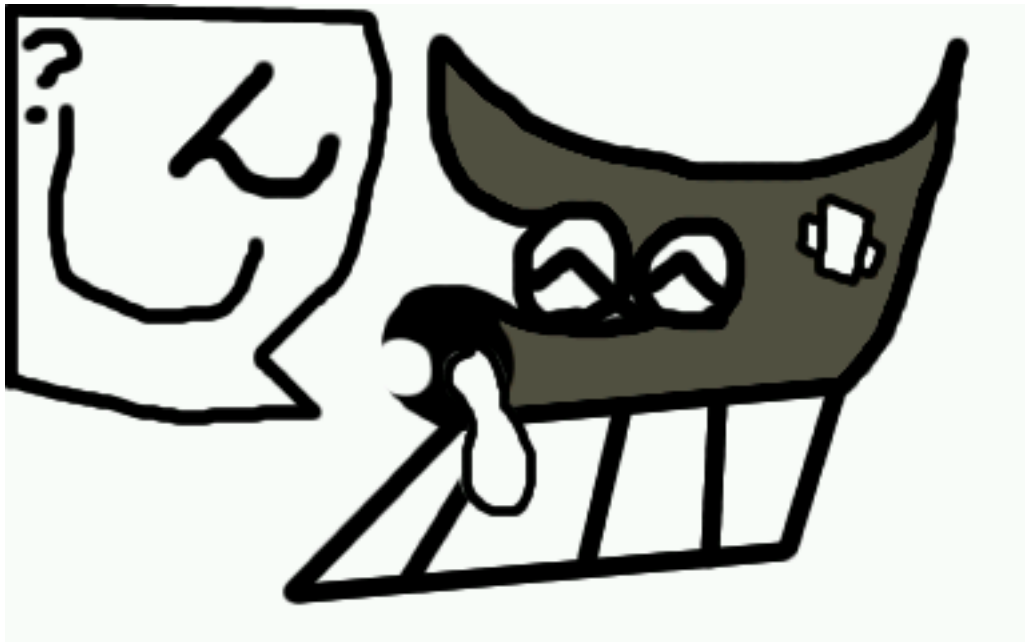
script-fu を書こう！

プラグイン **script-fu** が予期せず終了したら？

- **<ToolBox>/Xtns/Script-Fu/Console...**を出す
- そこで,**`(extension-script-fu)`**という式を入力すれば復活.

評価すると画の描ける **scheme** だって？ **-** 面白すぎ！

GIMP の未来



GIMPの未来

ユーザの開発

プロは自分で自分の使う道具を工夫したり創作するもの.それゆえに **GIMP** はプロフェッショナルの道具に相応しい.

- 本職：プログラミングができれば, 差が付けられる
  - **script-fu** で簡単に拡張できる **GIMP** は, 人気が出るはず. という妄想
- ハッカーも画を描こう
  - アイコン
  - デスクトップの背景
  - **Enlightenment** の **theme**

画の修行

- □とりあえず描く - 最初からうまくいくと思わない
- □優れた意匠 / 作品を見る / マネをする
  - ○技術 イタリア ルネサンス, マニエリスム
  - ○コンセプト バウハウスと インターナショナルスタイル
  - ○着想 ダダ, シュルレアリスム
  - ○和風の意匠 鳩居堂
  - ○その他 まんが, 工業製品, 極限の風景
- □形を憶え, イメージする
- □**photoshop** 本を立ち読みする
- □名人に聞く

参考リソース (順不同)

GIMP ソース

メーリングリスト

- **gimp@lists.hypercore.co.jp**
- **gimp-developer@scam.xcf.berkeley.edu**

ドキュメント

- **Gimp User’s Manual**

一般

- 本家 **http://www.gimp.org/**
- **Hollywood ブランチ http://film.gimp.org/**

日本語入力

□紅民さん **http://www.din.or.jp/~khoming/gimp/**

□榎田さん

http://www.geocities.co.jp/SiliconValley-SanJose/6944/

内部構造解説, **script-fu**

□楢崎さん **http://www.gimp.org/~narazaki/**

ありがとうございました！