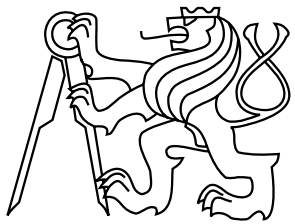
Programování grafických aplikací (BI-PGA), Přednáška č. 2

2D grafika - GIMP: Metodika programování pluginů

Jiří Chludil

Fakulta informačních technologií
České vysoké učení technické v Praze
https://courses.fit.cvut.cz/BI-PGA/



ZS 2022/2023

Lety, pády, kotrmelce při vývoji zásuvných modulů v GIMPu

- Mizerná dokumentace!
- Rozdíly mezi Python a C++ implementací!
- Špatná zpětná vazba a ladění!
- GIMP používá Python 2 (OMG)!!!!!
- Náročné napojení externích knihoven!
- POZOR!! Na data uložená v hlavní aplikaci GIMP
- Cvičící je občas zmatený (Grrrrr)

Co je třeba řešit u zásuvných modulů GIMPu

- **Vývojový jazyk a prostředí**
- Typ modulu
- Životní cyklus modulu
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Vývojový jazyk a prostředí (podpora OS)

- C
  - Linux - potřeba Cxx, knihovny jsou součástí instalace Gimpu
  - Windows - potřeba Cxx,MINGW, knihovny jsou součástí instalace Gimpu - velmi složité nastavení
- C++
  - nadstavba nad C API
  - Linux - potřeba G++, knihovny jsou součástí instalace Gimpu
  - Windows - potřeba G++ ,MINGW, knihovny jsou součástí instalace Gimpu - velmi složité nastavení
- Python (3. přednáška)
  - kompilátor součástí GIMPu
  - pomalejší jak C a C++
  - Linux - stačí nakopírovat do správného adresáře
  - Windows - stačí nakopírovat do správného adresáře

Co je třeba řešit u zásuvných modulů GIMPu

Vývoj pod C/C++

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- **Typ modulu**
- Životní cyklus modulu
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Typy modulů u Gimpu

- Filtry
- Import
- Export
- Formát
- Výběr
- Výběr Barvy
- Automatizace

1. Primárně jde o formální dělení
2. Je třeba koordinovat s umístěním v menu
3. Funkcionální API je nezávislé na typu modulu

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- Typ modulu
- **Životní cyklus modulu**
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Životní cyklus modulu

```
struct GimpPlugInInfo {
    GimpInitProc init_proc;
        // Volání při startu modulu před zobrazením GUI
    GimpQuitProc quit_proc;
        // Volání při ukončení modulu
    GimpQueryProc query_proc;
        //Instalace modulu a inicializace dle zdrojů
    GimpRunProc run_proc;
        // Vlastní provedení funkce modulu
};
```

1. Pokud nechceme volání použít, stačí dát $NULL$

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- Typ modulu
- Životní cyklus modulu
- **Nastavení prostředků (konfigurace)**
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Nastavení prostředků (konfigurace) C

```
void gimp_install_procedure(
    char* name, // Název (identifikace) modulu
    char* blurb, // Název v menu
    char* help, //Popis modulu
    char* author, //Jméno autora
    char* copyright, //Copyright
    char* menu_path, //Cesta v menu
    char* image_types, //Podporované formáty
    GimpPDBProcType type, //Typ procedury
    int nparams, //Počet vstupních parametrů modulu
    int nreturn_vals, //Počet výstupních parametrů
    GimpParamDef* params, //Pole vstupních parametrů
    GimpParamDef* return_vals //Pole výstup. parametrů
);
```

Nastavení prostředků - Podporované formáty

```
char* image_types, //Podporované formáty
```

- RGB
- RGBA
- RGB*
- GRAY
- GRAYA
- GRAY*
- INDEXED,
- INDEXEDA
- INDEXED*

Nastavení prostředků - Typ procedury

```
GimpPDBProcType type, //Typ procedury
```

Enum typ

- GIMP_INTERNAL, Interní procedura GIMP
- **GIMP_PLUGIN** Zásuvný modul GIMPu
- GIMP_EXTENSION Rozšíření GIMPu
- GIMP_TEMPORARY Dočasná (temporary) procedura

Nastavení prostředků - Parametry

```
    GimpParamDef* params, //Pole vstupních parametrů
    GimpParamDef* return_vals //Pole výstup. parametrů
```

```
void GimpParamDef struktura(
    GimpPDBArgType type; //Typ parametru
    gchar *name; //Jméno parametru
    gchar *description;//Popis parametru
);
```

GimpPDBArgType

- GIMP_PDB_INT32
- GIMP_PDB_INT16
- GIMP_PDB_INT8
- GIMP_PDB_FLOAT
- GIMP_PDB_STRING

Nastavení prostředků - Parametry

GimpPDBArgType (Pokr.)

- GIMP_PDB_INT32ARRAY
- GIMP_PDB_INT16ARRAY
- GIMP_PDB_INT8ARRAY
- GIMP_PDB_FLOATARRAY
- GIMP_PDB_STRINGARRAY
- GIMP_PDB_COLOR - barva
- GIMP_PDB_REGION - obdélníková oblast
- GIMP_PDB_DISPLAY - ???
- GIMP_PDB_IMAGE - obrázek
- GIMP_PDB_LAYER - vrstva
- GIMP_PDB_CHANNEL - kanál

Nastavení prostředků - Parametry

GimpPDBArgType (Pokr.)

- GIMP_PDB_DRAWABLE - výstup
- GIMP_PDB_SELECTION - Výběr
- GIMP_PDB_BOUNDARY - Hranice
- GIMP_PDB_PATH - Cesta
- GIMP_PDB_PARASITE - ???
- GIMP_PDB_STATUS
  - návratový status modulu např. GIMP_PDB_SUCCESS
- GIMP_PDB_END - ???

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- Typ modulu
- Životní cyklus modulu
- Nastavení prostředků (konfigurace)
- **Registrace**
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Registrace v menu

```
void gimp_plugin_menu_register(
    const gchar *procedure_name,
                        // Název (identifikace) modulu
    const gchar *menu_path //cesta v menu
);
```

1. Název modulu musí být shodný s jménem uvedeným v instalaci
2. Cesta musí existovat

Nastavení prostředků a registrace - Příklad

```
static GimpParamDef args[] = {
    {
        GIMP_PDB_INT32, "run-mode", "Run mode"
    },
    {
        GIMP_PDB_IMAGE, "image", "Input image"
    },
    {
        GIMP_PDB_DRAWABLE, "drawable", "Input drawable"
    }
};
```

Nastavení prostředků a registrace - Příklad

```
gimp_install_procedure (
    "add-color",
    "Posun barev ve všech kanálech",
    "Modul posune RGB barvy o hodnotu v parametru",
    "Jiří Chludil",
    "JCH",
    "2017",
    "Posun barev (bez náhledu)...",
    "RGB*, GRAY*",
    GIMP_PLUGIN,
    G_N_ELEMENTS (args), 0,
    args, NULL
);
```

Nastavení prostředků a registrace - Příklad

```
gimp_plugin_menu_register (
    "add-color",
    "<Image>/Filters/Misc"
);
```

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- Typ modulu
- Životní cyklus modulu
- Nastavení prostředků (konfigurace)
- Registrace
- **Uživatelské rozhraní modulu**
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Uživatelské rozhraní modulu C

Použito GimpWidgets s použitím GTK (API na GIMP Widget)

```
dialog = gimp_dialog_new ("
    Posun Barev",
    "addColor",
    NULL,
    0,
    gimp_standard_help_func,
    "add-color",
    GTK_STOCK_CANCEL,
    GTK_RESPONSE_CANCEL,
    GTK_STOCK_OK,
    GTK_RESPONSE_OK,
    NULL
);
```

Uživatelské rozhraní modulu C

```
spinbutton = gimp_spin_button_new (
    &spinbutton_adj, bvals.level, 1,255,1,1,1,5,0);
gtk_box_pack_start (
    GTK_BOX (main_hbox), spinbutton, FALSE, FALSE, 0);
gtk_widget_show (spinbutton);
```

```
g_signal_connect_swapped (
    spinbutton_adj,
    "value_changed",
    G_CALLBACK (gimp_preview_invalidate),
    preview
);
```

Uživatelské rozhraní modulu C

```
g_signal_connect (
    spinbutton_adj,
    "value_changed",
    G_CALLBACK (gimp_int_adjustment_update),
    &bvals.level
);
```

```
gtk_widget_show (dialog);
```

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- Typ modulu
- Životní cyklus modulu
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- **Použité matematické a grafické funkce**
- API rozšiřované aplikace

Použité matematické a grafické funkce

- GimpMath —Matematické definice a makra
- GimpMatrix — Funkce pro manipulaci s maticemi pro 2D
- GimpVector — Funkce pro manipulaci s 2D vektory
- GimpMD5 — MD5 algoritmus

GimpMath

- RINT
- ROUND
- SQR
- MAX255
- CLAMP0255
- gimp_deg_to_rad
- gimp_rad_to_deg

GimpMatrix

- gimp_matrix3_mult
- gimp_matrix3_translate
- gimp_matrix3_scale
- gimp_matrix3_rotate
- gimp_matrix3_xshear, gimp_matrix3_yshear
- gimp_matrix3_determinant
- gimp_matrix3_invert
- a mnoho dalších

GimpVector

- gimp_vector3_add
- gimp_vector3_sub
- gimp_vector3_cross_product
- gimp_vector3_rotate
- gimp_vector_2d_to_3d
- gimp_vector3_neg
- gimp_vector3_normalize
- a mnoho dalších

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- Typ modulu
- Životní cyklus modulu
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- **API rozšiřované aplikace**

API rozšiřované aplikace - GIMP Library

- Manipulace s obrázky
  - čtení a zápis po pixelu, řádku, sloupci
  - nástroje výběru
  - nástroje vrstev
- podpora výběrových nástrojů
- GUI

API rozšiřované aplikace - GIMP Base Library

- Stav aplikace GIMP
- Správa paměti
- Nastavení Gimpu (jednotky)
- Správa signálů

API rozšiřované aplikace - GIMP Color Library

- Manipulace s grafickou reprezentací pixelu
- Nástroje pro změnu osvětlení, teploty atd.
- Nástroje pro interpolaci barev
- Převodu mezi formáty
- Cairo

API rozšiřované aplikace - GIMP Config Library

- Serializace a deserializace objektů
- Správa Chyb
- Management nastavení GIMP
- Podpora GScaner
- Utility

API rozšiřované aplikace - GIMP Widgets

- Funkce pro GUI
  - Tlačítka
  - Posuvníky
  - Textová pole
  - Elementy pro práci s barvou
  - Výběry
  - atd.

API rozšiřované aplikace - GIMP Module, Thumbnail Library

- Funkce pro miniatury
- Funkce jádra
- Management nastavení GIMP
- Podpora GScaner
- Utility

Co je třeba řešit u zásuvných modulů GIMPu

Vývoj pod Pythonem

Co je třeba řešit u zásuvných modulů

- **Vývojový jazyk a prostředí**
- Typ modulu
- Životní cyklus modulu
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Zásuvný modul v Pythonu

+ Script bez potřeby kompilace
+ Triviální instalace
+ Jednoduché GUI pro jednoduché příklady
+ Dostupnější příklady
+ Dobrá dokumentace API
+ Neobjektový i objektový přístup
– Pomalejší než C a C++
– Složitější instalace, pokud jsou potřeba speciální knihovny
– Náročná realizace pokročilého GUI
– Náročná realizace náhledu

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- **Typ modulu**
- Životní cyklus modulu
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Typy modulů u Gimpu

- Filtry
- Import
- Export
- Formát
- Výběr
- Výběr Barvy
- Automatizace

1. Stejné jaku u C, C++
2. Primárně jde o formální dělení
3. Je třeba koordinovat s umístěním v menu
4. Funkcionální API je nezávislé na typu modulu

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- Typ modulu
- **Životní cyklus modulu**
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Životní cyklus modulu Python

```
metody v třídě
    def start(self):
        ...

    def init(self):
        pass

    def quit(self):
        pass

    def query(self):
        ...
```

1. Pokud nechceme volání použít, stačí dát "pass"

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- Typ modulu
- Životní cyklus modulu
- **Nastavení prostředků (konfigurace)**
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Nastavení prostředků (konfigurace)

```
gimp.install_procedure(
    "Jméno",
    "Popisek v menu",
    "Help",
    "Autor", "Copyright",
    "Datum","Lokace v menu",
    "Typ obrázku", "Typ modulu",
    [
        (PDB_INT32,"run_mode","Run Mode"),
        (PDB_IMAGE,"image","Input image"),
        (PDB_DRAWABLE,"drawable","Input drawable"),
        (PF_INT32, "my_param","Můj param."),
    ], []
)
```

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- Typ modulu
- Životní cyklus modulu
- Nastavení prostředků (konfigurace)
- **Registrace**
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Životní cyklus modulu Python

```
#!/usr/bin/python
register(
    "python_fu_resize",
    "Saves the image at a maximum width and height",
    "Saves the image at a maximum width and height",
    "Nathan A. Good", "Nathan A. Good",
    "2010",
    «Image>/Image/Resize to max...",
    "RGB*, GRAY*",
    [], [],
    plugin_main)
main()
```

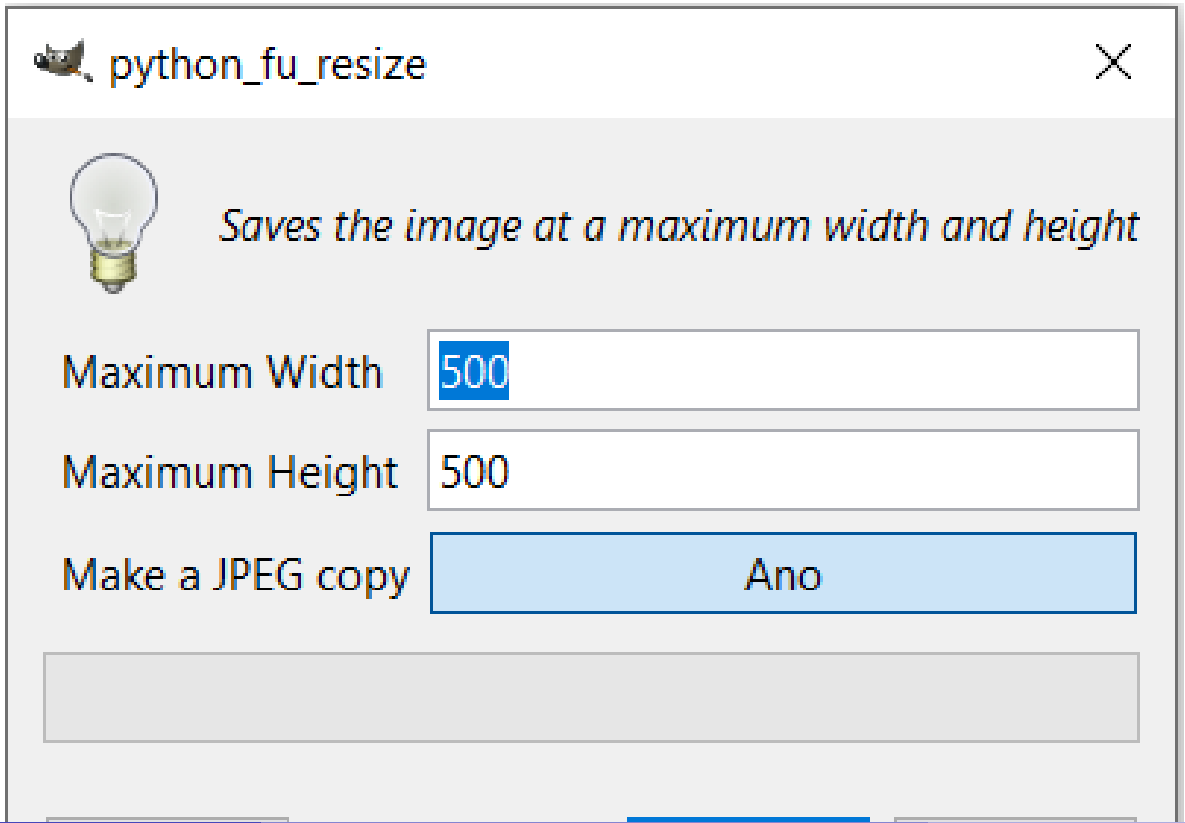
Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- Typ modulu
- Životní cyklus modulu
- Nastavení prostředků (konfigurace)
- Registrace
- **Uživatelské rozhraní modulu**
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Uživatelské rozhraní modulu I

- Automatické generování

```
(PF_INT, "max_width", "Maximum Width", 500),
(PF_INT, "max_height", "Maximum Height", 500),
(PF_BOOL, "copy", "Make a JPEG copy", TRUE),
```

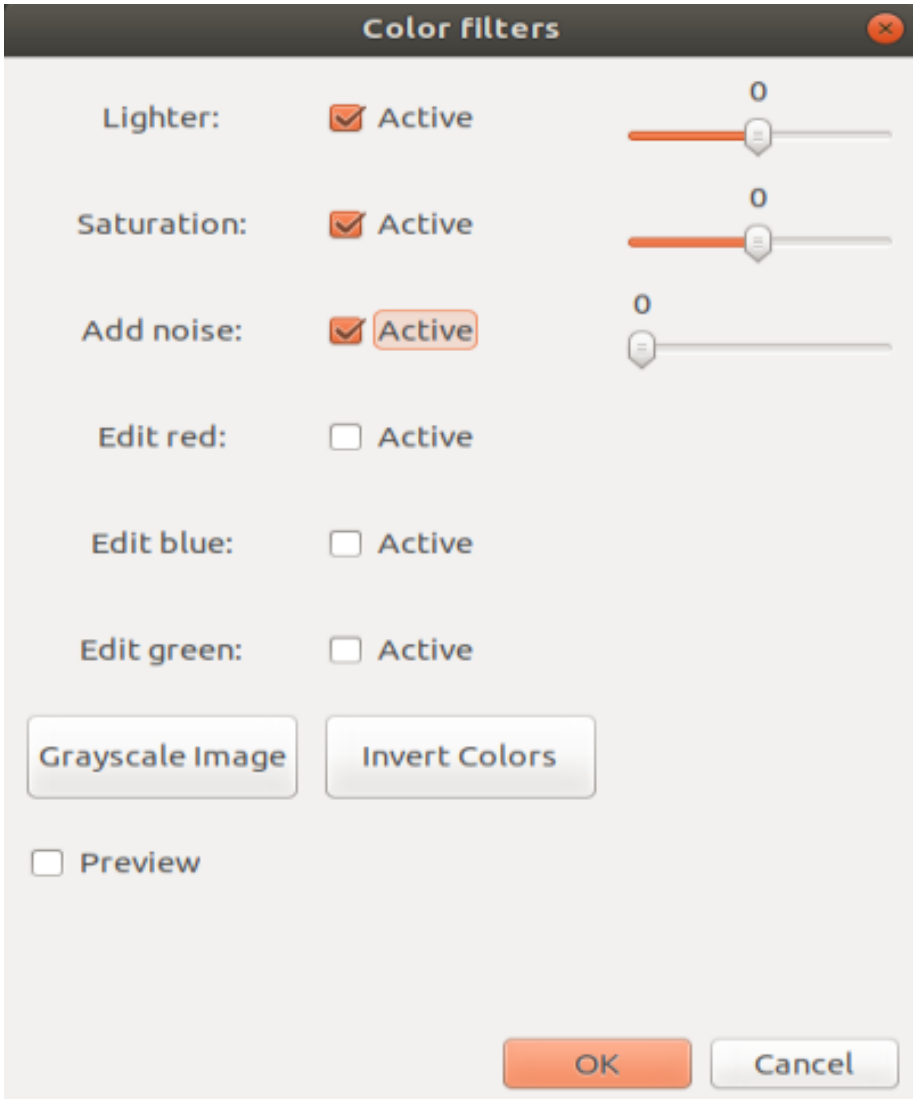


Uživatelské rozhraní modulu II

- GTK knihovna - import gtk, gimpui

1. Definice struktury UI - GTK
   - button
   - input
   - select
   - radio
   - checkbutton
2. napojení GUI na obslužné funkce
   - self.ok_button.connect("clicked", self.ok_clicked)

Uživatelské rozhraní modulu III



Uživatelské rozhraní modulu - náhled

- Náhled v nové vrstvě
- Problém s dobou trvání transformace
  - Snížení rozlišení
  - Zjednodušení algortimu

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- Typ modulu
- Životní cyklus modulu
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- **Použité matematické a grafické funkce**
- API rozšiřované aplikace

Použité matematické a grafické funkce

- matematická knihovna numpy
- opencv-python

Co je třeba řešit u zásuvných modulů

- Vývojový jazyk a prostředí
- Typ modulu
- Životní cyklus modulu
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- **API rozšiřované aplikace**

API rozšiřované aplikace

- https://www.gimp.org/docs/python/index.html
- GIMP2 -> Python Console -> Browse
- Ukázkové příklady

Ladění

- print a sys.stdout = open('cesta.txt', 'w')
- Logování do souboru

Ukázkové příklady

- Dokumentace
- Styl programování
- Memory management
- Náhled

Co bychom po dnešku měli znát

Témata probraná na dnešní přednášce:

- Základní principy psaní modulů pod C/C++
- Základní principy psaní modulů pod Python