



MATLAB: Tvorba grafických uživatelských rozhraní

Jaroslav Čmejla
Martin Vrátný
Miroslav Holada
ITE FM TUL

jaroslav.cmejla@tul.cz martin.vratny@tul.cz miroslav.holada@tul.cz

Tvorba grafických uživatelských rozhraní (GUI) v Matlabu

Postup programování

- Kdo bude uživatel?
- K čemu má GUI sloužit?
- Jak bude vypadat?
- Tvorba GUI v Matlabu
 - Ruční programování pomocí příkazu `uicontrol`
 - App Designer

Co je nová aplikace v Matlabu

- GUI je `figure`
- Aplikace čeká na akci uživatele
- Každý objekt (`figure`, tlačítka, radioboxy, ...) má své vlastnosti, které lze načíst a měnit
 - příkazy `set` a `get`
příklad.: `get(objekt, 'vlastnost')` a `set(objekt, 'vlastnost', 'zapnuto')`
 - případně přes tečku příklad.: `objekt.vlastnost = 'zapnuto'`
- Každý objekt má `handle`, což je obdoba ukazatele
- `Callback` je vlastnost, ve které se nastavuje ukazatel na funkci, která se zavolá při dané události (stisk tlačítka, průchod myši, ...)
- Program je řízený událostmi, které generuje Matlab (a uživatel).

"Ruční" programování

- Objekty vkládáme "ručně" v kódu (`uicontrol`), nastavujeme vlastnosti, callbacky
- Funkce volané callbacky mají pevnou strukturu vstupních parametrů
- Ukládáme do souborů `*.m` (funkce - callbacky) a `*.fig` (aplikace)
- Často používané příkazy: `gcf`, `gca`, `get`, `set`

"Ruční" programování

- Pozici objektů nastavujeme pomocí parametru `Position`
- Vlastnost `Units: normalized` řeší škálovatelnost velikosti prvku vzhledem k velikosti okna

```
uicontrol( ...
    'Style','pushbutton', ... % styl nastavuje typ objektu
    'Units','normalized', ...
    'Position',[0.80 0.94 0.16 0.05], ...
    'String','KONEC', ...
    'FontWeight','bold',...
    'ForegroundColor',[1 0 0],...
    'Callback',@(~,~) close(gcf));
```

"Ruční" programování

- Datový segment aplikace vytváříme přes:
 - nastavení vlastností `UserData`
 - používání sdíleného paměťového prostoru vnořených funkcí

"Ruční" programování - UserData

- Na ukládání vlastních dat aplikace používáme vlastnost objektů `UserData`, která může obsahovat jakýkoliv datový typ (čili všechno).
- Je na programátorovi, jestli bude ukládat všechna data do `UserData` jednoho objektu (např. `figure`), nebo bude používat více objektů.
- `UserData` tedy plní funkci datového segmentu aplikace

"Ruční" programování - Datový segment pomocí vnořených funkcí

```
fig = figure(1);  
tlacitko = uicontrol(fig, "Style","pushbutton", ...  
    "Units","normalized", ...  
    "Position",[0,0,1,1], ...  
    "String","Tlačítko", ...  
    "Callback",@stisk_tlacitka, ...  
    "UserData",0);  
  
function stisk_tlacitka(src,event)  
    src.UserData = src.UserData+1;  
    src.String = num2str(src.UserData);  
end
```


“Ruční” programování - Datový segment pomocí vnořených funkcí

- Vnořené funkce sdílejí paměťový prostor s rodičovskou funkcí. Této vlastnosti lze využít k vytvoření datového segmentu aplikace.

"Ruční" programování - Datový segment pomocí vnořených funkcí

```
function testgui()
    fig = figure(1);
    pocet_kliknuti = 0;
    tlacitko = uicontrol(fig, "Style","pushbutton", ...
                        "Units","normalized", ...
                        "Position",[0,0,1,1], ...
                        "String","Tlačítko", ...
                        "Callback",@stisk_tlacitka);

    function stisk_tlacitka(src,event)
        pocet_kliknuti = pocet_kliknuti+1;
        tlacitko.String = num2str(pocet_kliknuti);
    end
end
```

"Ruční" programování - Datový segment pomocí vnořených funkcí

```
function gui_ukazka()
    fig = figure(1);
    ax = axes(fig, "Units", "normalized", "OuterPosition",[0,0.1,1,0.9]);
    slider = uicontrol(fig, "Style","slider", "Units","normalized",...
        "OuterPosition",[0.25,0,0.75,0.1], ...
        "Callback", @klik_slider,...
        "Min", 0, "Max", 100);
    text = uicontrol(fig, "Style","text", "Units","normalized" ,...
        "Position",[0,0,0.25,0.1], "FontSize", 20,...
        "String","Amplitude");

    function klik_slider(source,event)
        value = source.Value;
        plot(ax, value*peaks);
        set(ax,"YLim",[-1000,1000]) %ax.Ylim = [-1000,1000];
    end
end
```

"Ruční" programování

- Událost můžeme odchytit samostatnou funkcí
- Je-li aplikace ovládána jedinou funkcí, odchytáváme událost většinou pomocí `switch`.
- příklad `makevase`

Struktura typické aplikace řízené jednou funkcí

```
function mojeaplikace(udalost)

if nargin==0, udalost='zacatek'; end

switch udalost
    case 'zacatek'
        fig=figure(...
            data.tlacidlo1=uicontrol('Style','push',...
                ...
                'Callback','mojeaplikace(''stisktlacitka1'')');
        ...
        set(fig,'UserData',data); % uložení vlastních dat

    case 'stisktlacitka1'
        data=get(gcf,'UserData');
        ...
        set(gcf,'UserData',data);
end % switch
```

App Designer (doporučeno)

- Objektově orientovaný způsob programování
- Aplikace je vytvářena jako nový (zděděný) objekt s novými vlastnostmi (datový segment) a metodami (obsluha událostí)



Děkuji za pozornost

Jaroslav Čmejla
Martin Vrátný
Miroslav Holada
ITE FM TUL

jaroslav.cmejla@tul.cz martin.vratny@tul.cz miroslav.holada@tul.cz