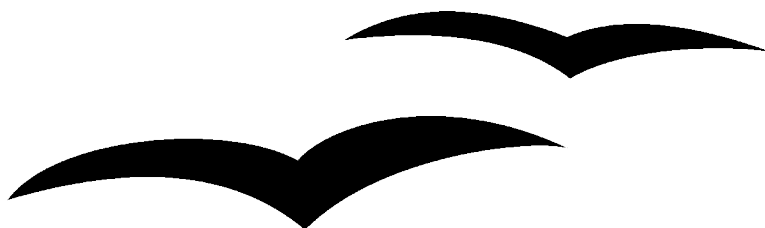


Makrá v OpenOffice.org

Dialógy pod drobnohl'adom



Obsah

Viacstránkové dialógy.....	3
Definícia dialógu.....	3
Rozdelenie dialógu na viac stránok.....	4
Priradenie makier k tlačidlám.....	5
Ďalšie možnosti dialógov.....	7
Ozdobné prvky v dialógoch.....	7
Vstupné prvky v dialógoch.....	8
Textové vstupy.....	8
Číselné vstupy.....	9
Textový zoznam záznamov.....	10
Rozbaľovací zoznam záznamov.....	14
Dátum a čas.....	15
Výber mena súboru.....	16
Dialóg pre formátovanie dokumentu.....	17
Čo vlastne chceme formátovať?.....	17
Nezalomitelné medzery.....	18
Odstraňovanie nadbytočných medzier a pod.....	18
Vkládanie medzier.....	19
Tlačidlá.....	19
Systémové cesty závislé od používateľa.....	19
Nastavenie používateľského profilu.....	22
Dialóg pre používateľské profily.....	22
Makrá pre prácu so zoznamom používateľských profilov.....	23
Makrá pre prácu s vlastnými používateľskými profilmi.....	28
Makrá pre formátovanie dokumentu.....	34
Nezalomitelné medzery počas písania textu.....	34
Funkcie pre výmenu a hľadanie.....	39
Vymazávanie viacnásobných medzier.....	42
Ostatné formátovacie funkcie.....	45
Záverečné funkcie.....	48
Niekoľko slov na záver.....	50

Predchádzajúcu – štvrtú časť seriálu o programovaní makier v OpenOffice.org sme venovali základom práce s dialógmi. Dialógy v OpenOffice.org však môžu byť omnoho väčšie a komplikovanejšie a preto sa nimi budeme naďalej podrobnejšie zaoberať.

V doterajších dieloch sme si ukázali jednoduché dialógy, kde sme zatiaľ použili tri možnosti zadávania vstupných údajov – zaškrŕavacie políčka, prepínače a tlačidlá. Toto nám však mnohokrát nevystačuje a navyše, niekedy jedno dialógové okno svojimi rozmermi nemusí stačiť na zadanie všetkých parametrov, ktoré postačujeme a preto musíme naprogramovať celú postupnosť okien s tým, že sa budeme môcť medzi nimi presúvať.

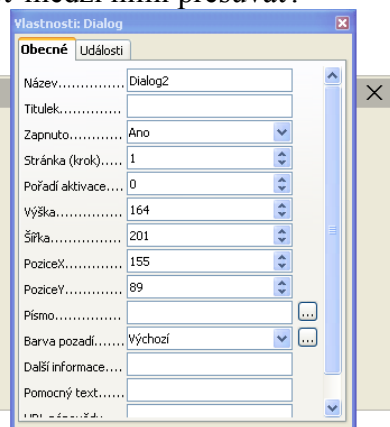
Týmto všetkým možnostiam sa budeme venovať v nasledujúcej piatej časti seriálu o programovaní makier v OpenOffice.org. Pravdaže, nadviažeme na tie časti, ktoré sme si už predstavili s týmto sa nebudeme už venovať, ale predpokladáme, že cez prázdny mal každý dost času si nejaké jednoduché dialógy odskúšať.

Viacstránkové dialógy

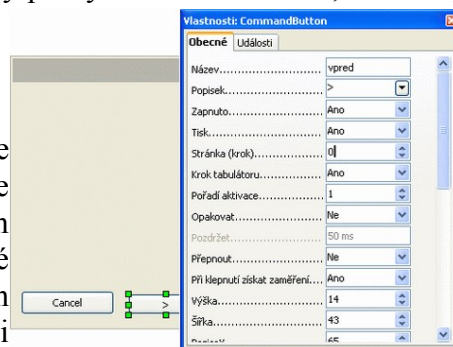
Ako prvé, čomu sa budeme venovať, budú dialógy, ktoré budú zaberat viacero listov – strán. Za týmto účelom je v zozname vlastností jednotlivých prvkov, ktoré vkladáme do vlastného dialógu položka „Stránka (krok)...“. Jej štandardná hodnota je nula, čo znamená, že práve zadávaný prvok sa bude nachádzať na úplne všetkých listoch navrhovaného dialógu – a naopak, na liste č. nula sa budú zobrazovať úplne všetky prvky bez ohľadu na to, na ktorej strane sme ich definovali.

Definícia dialógu

Podme však po poriadku a pozrime sa, ako máme vlastne začať. Hneď pri začiatku definície dialógu je vhodné označiť prvé pracovné okno, do ktorého ešte len ideme vkladat prvky ako stránku č. 1. Teraz je vhodné vložit tie prvky, ktoré budeme mať určite na všetkých stránkach – obvykle to budú tlačidlá pre prechod medzi nimi a prípadne tlačidlá „OK“ a „Cancel“. Pri vkladaní týchto tlačidiel nesmieme zabudnúť, že sa nám bude štandardne ponúkať stránka č. 1 a preto

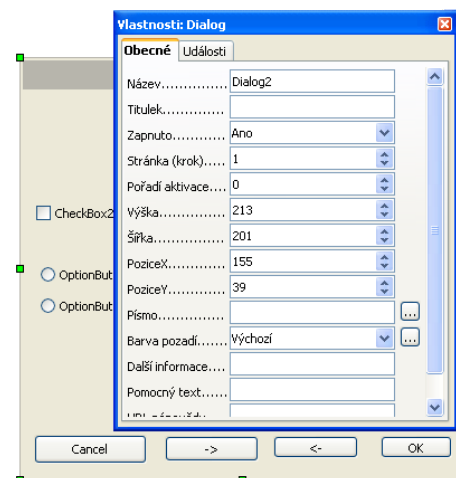


Obrázok 1: Zobrazenie parametrov dialógu



Obrázok 2: Zmena stránky na „0“

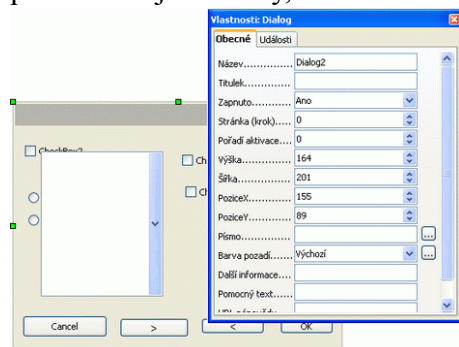
túto musíme opraviť na nulu.



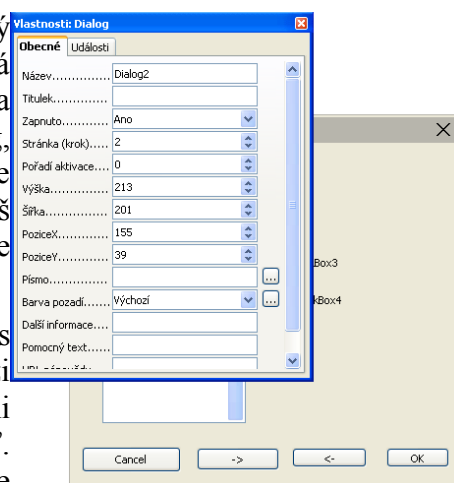
Obrázok 3: Definícia prvkov na stránke „1“

Vkladanie ďalších prvkov je jednoduché. Podobne, ako sme si to už ukázali postupne vkladáme do pracovného rozhrania všetky potrebné prvky. Pokiaľ chceme vložit novú stránku, klepneme na hlavné okno navrhovaného dialógu a zmeníme číslo strany. Tým sa nám zobrazí príslušná strana, kde môžeme pokračovať v definícii. Ku predchádzajúcim stranám sa dostaneme podobne – stačí, ak po označení celého dialógového okna napíšeme príslušné číslo strany. Pozor, ak toto urobíme na niektorom prvku, tento sa automaticky presunie na zadanú stranu a z práve definovanej „zmizne“. Na tomto mieste ešte musíme upozorniť, že OpenOffice.org rozoznáva aj záporné čísla stránok.

Pretože sa pri takejto definícii jedná o jeden a ten istý dialóg, ktorý je akurát rozdelený do viacerých listov, každá zmena jeho celkovej veľkosti sa automaticky prejaví na každej strane a preto nie je možné, aby sme navrhli dialóg, ktorý bude mať rôzne veľkosti strán. Na toto musíme pamätať najmä vtedy, ak sa nám nejaká stránka zdá príliš prázdna a radi by sme ju zmenšili.



Pravdaže, teraz nás zaujíma, ako sa medzi jednotlivými stránkami dokážeme presúvať. O tom si však budeme hovoriť až nabadúce, pričom, ako inak, sa



Obrázok 4: Definícia prvkov na stránke „2“

Obrázok 5: Zobrazenie všetkých znovuzvratíme k problematike nezalomiteľných medzier za prvkov naraz je možné pomocou stránky jednoznakové predložky a spojky – ved' prečo by sme nemali zmeniť práve tento dialóg na viacstránkový.

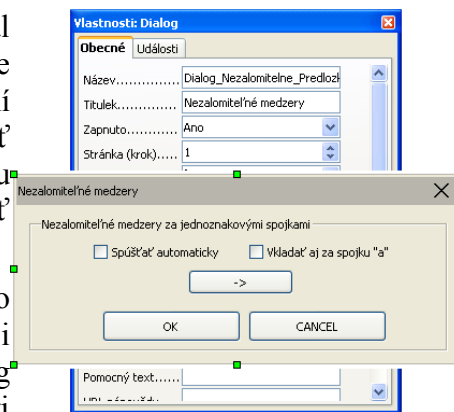
Rozdelenie dialógu na viac stránok

V dnešnom dieli seriálu o programovaní makier v OpenOffice.org si ukážeme praktickú ukážku dvojstránkového dialógu pre nastavovanie parametrov pri vkladaní nezalomiteľných medzier za jednoznakové predložky a spojky.

Ako sme si už naznačili v minulom dieli, dnes si ukážeme, ako sa prakticky programujú podprogramy pre prepínanie medzi jednotlivými oknami viacstránkových dialógov. Aby sme nemuseli vymýšľať nový príklad, rozdelíme si dialóg, ktorý sme si vytvorili pre nastavovanie parametrov pri vkladaní nezalomiteľných medzier za jednoznakové predložky a spojky.

Ako si určite dobre pamätáte, tento dialóg sa skladal z dvoch zaškrťovacích okienok (či sa má vkladať nezalomiteľných medzier spúšťať automaticky pri spustení OpenOffice.org a či sa má nezalomiteľná medzera vkladat' aj za spojku „a“) a dvoch prepínacích políček, pomocou ktorých sme určovali, aký znak sa má za spojky vkladat' („Non Break Space“ alebo „Word Joiner“).

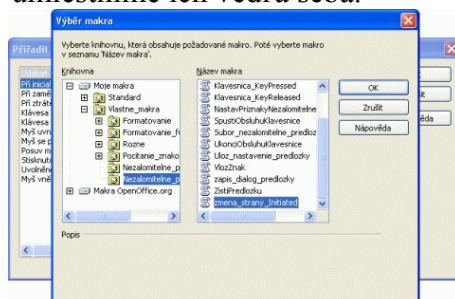
Teraz si ukážeme postup, ako tento dialóg rozdelíme do dvoch stránok. Ako prvé, čo urobíme, bude to, že si v návrhovej časti dialógov označíme celý dialóg a zmeníme jeho číslo stránky (krok) na 1. Teraz vo vnútri dialógu označíme postupne zaškrťavacie okienka a tak isto zmeníme túto položku u nich na číslo stránky na hodnotu jedna.



Obrázok 6: Presunutie prvkov na prvú stránku

V druhom kroku znovu označíme celý dialóg a zmeníme číslo stránky na 2. Teraz vo vnútri dialógu označíme postupne prepínacie políčka a zmeníme aj ich čísla stránok na hodnotu dva. Keď teraz budeme prepínať medzi jednotlivými stránkami uvidíme, že nám tam zostali iba príslušné políčka a tlačidlá „OK“ a „Cancel“. Aby sme však mohli medzi nimi prepínať aj po spustení dialógu, musíme si pridať aspoň jedno tlačidlo, pomocou ktorého to budeme realizovať. Nazveme ho „zmena_strany“, musíme mu priradiť číslo strany 0 a ako zobrazovaný text zadáme šípku „->“. Nakoniec si ešte upravíme vzhľad jednotlivých stránok

tak, aby nám viac vyhovoval – napr. tým, že namiesto toho, aby sme mali políčka pod sebou, umiestnime ich vedľa seba.



Obrázok 7: Výber makra pre tlačidlo

V tomto kroku sme dospeli k najdôležitejšej časti – tlačidlu „zmena_strany“ musíme priradiť makro, pomocou ktorého vlastne budeme jednotlivé stránky prepínať. Pretože máme iba dve stránky, bude nám stačiť pomerne jednoduchá procedúra, kde stačí tieto stránky vymieňať. Zároveň však bude vhodné, ak budeme meniť aj zobrazovanú šípku (text) v tlačidle „zmena_strany“ z „->“ na „<-“ a naopak. Príslušná procedúra môže potom vyzeráť napr. takto:

rem Prepínanie medzi dvoma stranami dialógu

sub zmena_strany_Initiated

if dlg.model.step=1 then

' Ak sme na strane 1, prepneme sa na stranu 2

dlg.model.step=2 ' týmto priradením sa automaticky zobrazila 2 strana dialógu

dlg.model.zmena_strany.enabled=true ' povolíme platnosť tlačidla „zmena_strany“

dlg.model.zmena_strany.label="<-“ ' a zmeníme jeho popis

else

' Ak sme na strane 2, prepneme sa na stranu 1

dlg.model.step=1

dlg.model.zmena_strany.enabled=true

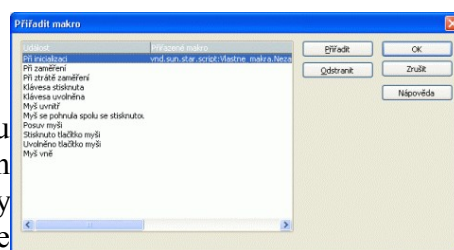
dlg.model.zmena_strany.label="->"

end if

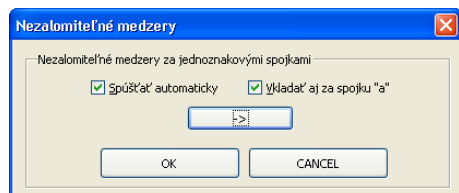
end sub

Priradenie makier k tlačidlám

Teraz túto procedúru priradíme ku tlačidlu „zmena_strany“. V návrhu dialógu sa v zobrazených vlastnostiach tohto tlačidla prepneme do druhej záložky „Udalosti“, kde stlačením prvého tlačidla „...“ vyberieme položku „Pri inicializácii“. Tam následne pomocou tlačidla „Priradiť“ vyhľadáme a priradíme túto procedúru k označenému tlačidlu.



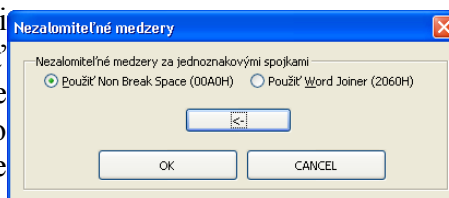
Obrázok 8: Vo vlastnostiach tlačidla vidíme priradené makro



Obrázok 9: Prvky dialógu na prvej stránke

Teraz nám ešte zostáva trochu upraviť ostatné funkcie, ktoré sme používali pri práci s týmto dialógom. Ako prvé je nutné, aby sme premennú dlg definovali ako globálnu, pretože inak ju nebude pred chvíľou definovaná procedúra poznať a pri priradovaní makier k tlačidlám nemáme možnosť definovať parametre funkcie.

Okrem toho je potrebné, aby sme pri inicializácii dialógu nastavili aj stranu dialógu, ktorá sa má zobrazit' ako prvá a pretože nevieme, aký sme mali posledne zobrazovaný text tlačidla „zmena_strany“, tak aj túto položku. Ostatné časti sa nemusia menit', pretože umiestnením nejakej položky na inú stranu dialógu sa nemenia jej ostatné vlastnosti. Vlastné makrá potom môžu vyzerat' nasledovne (uvádzame iba zmenené procedúry):



Obrázok 10: Prvky dialógu na druhej stránke

global dlg as object

sub init_dialog_predlozky(oDialog as object)

Citaj_nastavenie_predlozky

oDialog.model.Automaticky.State=Priznaky_nezalomitelne_medzery(1)

oDialog.model.Za_spojku_a.State=Priznaky_nezalomitelne_medzery(2)

oDialog.model.TabIndex=3

oDialog.model.TabIndex=4

select case Priznaky_nezalomitelne_medzery(3)

case 2

oDialog.model.Volba1.State=0

oDialog.model.Volba2.State=1

case else

oDialog.model.Volba1.State=1

oDialog.model.Volba2.State=0

end select

oDialog.model.step=1 ' Ako prvú zobrazíme stranu 1

oDialog.model.zmena_strany.enabled=true ' Povolíme prepínanie na stranu 2

oDialog.model.zmena_strany.label="->" ' A zobrazíme text – šípku – naznačujúcu smer dopredu na ďalšiu stranu

end sub

sub NastavPriznakyNezalomitelnePredlozky

DialogLibraries.LoadLibrary("Vlastne_makra")

dlg=CreateUnoDialog(DialogLibraries.Vlastne_makra.Dialog_Nezalomitelne_Predlozky)

init_dialog_predlozky(dlg)

```

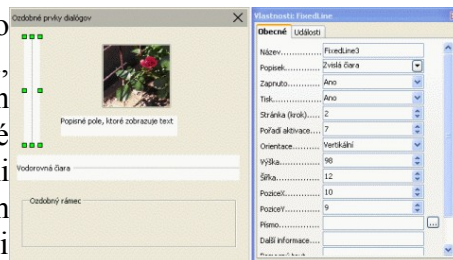
if dlg.Execute()=1 then
    zapis_dialog_predlozky(dlg)
endif
dlg.dispose()
end sub

```

Ďalšie možnosti dialógov

Dnes sa zoznámime s ďalšími – tentoraz ozdobnými – prvkami, ktoré môžeme vkladať do našich dialógov, keď doteraz sme sa zoznámili iba so štyrmi druhmi z vyše dvadsiatich.

V doterajších dieloch o dialógoch sme si ukázali, ako sa pracuje s tlačidlami, zaškrŕavacími okienkami, prepínacími políčkami a rámcami, pomocou ktorých môžeme rozdeľovať okno dialógu na rôzne prehľadné skupiny. Pretože už vieme pracovať aj s viacstránkovými dialógmi (hoci sme si ukázali iba prácu s dvojstránkovým dialógom, určite si dokážete predstaviť prechod aj medzi viacerými stránkami – to ponechávam prípadným záujemcom ako domácu úlohu), nezostáva nám nič iné, ako si postupne ukázať ostatné prvky, ktoré v dialógoch môžeme používať.

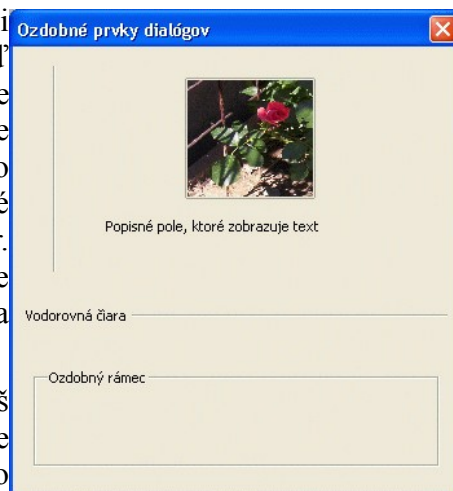


Obrázok 11: Vkladanie ozdobných

Ozdobné prvky v dialógoch

Ako prvé si ukážeme „ozdobné“ prvky, t.j. také, ktoré sa svojim charakterom podobajú už predstaveným rámcom. Obvykle ich nepoužívame na žiadne operácie (aj keď i to sa dá priradením makra k nejakej akcii, napr. pri posune myši dovnútra vloženého obrázku), ale pomocou nich skrášľujeme a sprehladňujeme naše dialógy, najmä ak sú rozsiahlejšieho charakteru.

Pretože sme už spomínali a používali rámce, teraz si ako prvé pridáme oddeľujúce čiary. Môžu byť buď vodorovné alebo zvislé s tým, že ku každej z nich môžeme pridať aj textový popis. V prípade, že čiaru zadefinujeme ako vodorovnú, tento popis sa zobrazí na jej začiatku. Ako ďalší „ozdobný“ prvok si môžeme pridať textové popisné pole, t.j. ľubovoľný jednoriadkový text, ktorým napr. označíme či popíšeme niektorú časť dialógového okna. Je to podobný prvok ako rámec s tým rozdielom, že sa nezobrazujú žiadne čiary.



Obrázok 12: Dialóg s ozdobnými

Posledný prvok, pomocou ktorého môžeme ozdobiť náš dialóg je obrázok. Na tomto mieste musíme upozorniť, že obrázok sa do vlastného dialógu nevkladá (pri jeho vkladaní nemáme možnosť zmeniť parameter „odkaz“) a preto pri jeho prípadnom premiestnení sa zobrazí iba prázdny rámik. Pokiaľ si vytvárame dialóg iba sami pre seba, nie je to až tak na závalu, ale v prípade, že chceme náš dialóg uložiť napr. do súboru, resp. nainštalovať ho aj iným používateľom, môže nám to spôsobiť určité komplikácie.

Vstupné prvky v dialógoch

V dnešnej, už štvrtej časti programovania makier v OpenOffice.org, ktorú zameriavame na prácu s vlastnými dialógovými oknami sa zameriame na jednoduché formátované a neformátované textové vstupy.

Textové vstupy

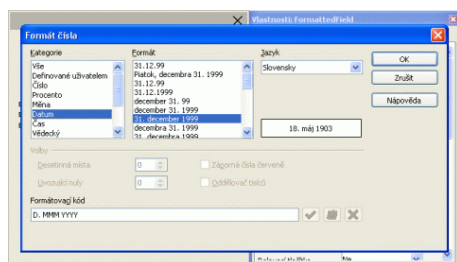
Z hľadiska jednoduchých textových vstupov máme k dispozícii dva rôzne dialógové prvky. Jedná sa o jednoduché neformátované textové pole a formátované textové pole. Teraz si v jednoduchosti popíšeme ich vlastnosti a možnosti.

Neformátované textové pole je určené na interaktívne zadanie ľubovoľného textu s tým, že môžeme preddefinovať jeho vstupnú hodnotu. Predpokladajme, že máme definovaný dialóg „dlg“, do ktorého vložíme textové pole „Textove_pole“. Vlastný text potom nájdeme v premennej:

`dlg.model.Textove_pole.text`

Pokiaľ chceme, môžeme pre textové pole v jeho vlastnostiach povoliť viacriadkový vstup. Jednotlivé riadky sú potom oddelené znakom „Carriage return“ (hexadecimálny kód 0D), takže inicializácia dvoch riadkov vstupnej hodnoty textu vyzerá napríklad takto:

`dlg.model.Textove_pole.text="prvý riadok"+chr(13)+"druhý riadok"`

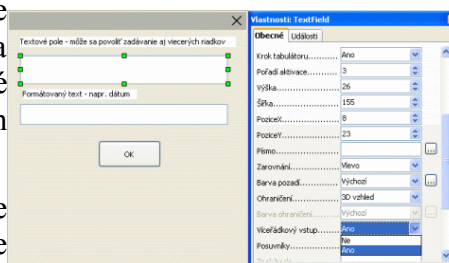


Obrázok 14: Formátované textové pole

Pravdaže, OpenOffice.org nás v tomto smere nesklame a nech do takto zadefinovaného políčka zadáme čokoľvek, prevedie to do príslušného formátu a naozaj vám doporučujem zadať napr. do políčka zadefinovaného ako dátum napríklad svoje krstné meno – po kliknutí na iné políčko program automaticky túto hodnotu zmení napr. na minimálnu hodnotu, ktorú definujeme vo vlastnostiach tohto typu poľa.

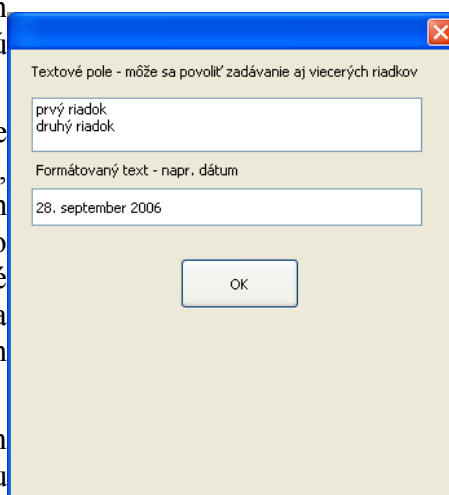
Prístup k formátovanému textovému poľu je presne ten istý, ako pri neformátovanom, t.j. napr. ak sme do dialógu „dlg“ vložili formátované textové pole „Formatovacie_pole“, inicializácia môže vyzeráť takto:

`dlg.model.Formatovacie_pole.text="28.september 2006"`



Obrázok 13: Textové vstupné pole

Formátované textové pole sa veľmi podobá bunkám v tabuľkovom procesore Calc. Je to tak preto, že môžeme zadať jeho formát, pričom sa používajú presne tie isté formátovacie prvky, ako pri bunkách v tomto tabuľkovom procesore. Pretože vlastné zadávané hodnoty môžu mať rôzne vlastnosti (text, číslo, dátum, logická hodnota a pod.) a preto sa inicializujú a získavajú ako text a je iba na nás, ako z nich získame prípadnú „netextovú“ hodnotu.



Obrázok 15: Neformátované textové pole môže mať aj viacero riadkov

Pravdaže, ako sme už spomínali, z premennej `dlg.model.Textove_pole.text` získame zadaný vstup po zavretí nášho dialógového okna vo formáte textu a v prípade potreby si ho musíme sami previesť na hodnotu iného typu.

Číselné vstupy

V dnešnom pokračovaní seriálu o programovaní makier v OpenOffice.org sa zameriame na číselné vstupy, ktoré môžeme použiť v našich dialógoch.

V minulom dieli sme si predstavili dva jednoduché textové vstupy, ktoré sme rozdelili na neformátované a formátované. Podobná situácia je aj v oblasti číselných vstupov, kde máme tak isto dve základné možnosti – obyčajný číselný vstup a číselný vstup pre zadanie korunovej (menovej) hodnoty.

Neformátovaný číselný vstup je určený na interaktívne zadanie ľubovoľného čísla. Podobne, ako pri ostatných vstupoch, aj tu môžeme preddefinovať jeho hodnotu. Špecialitou číselných vstupov je možnosť definovania minimálnej a maximálnej hodnoty spolu s počtom desatinných miest (t.j. desatinnou presnosťou). Pri vlastnom vstupe síce môžeme zadať aj viac desatinných miest, číslo však bude automaticky zaokrúhlené.

Ak budeme predpokladať, že máme definovaný dialóg „dlg“, do ktorého vložíme číselný vstup „Ciselný_vstup“, vlastnú hodnotu potom nájdeme v premennej:

`dlg.model.Ciselný_vstup.value`

a inicializáciu prevedieme napríklad takto (pozor, desatinné miesta sa oddeľujú bodkou – tu neplatia gramatické, ale programovacie pravidlá):

`dlg.model.Ciselný_vstup.value=123.57`

Číselný vstup pre zadanie meny je určený na interaktívne zadanie čísla s tým, že vo vstupnom okienku sa automaticky vypíše aj názov menovej jednotky (Sk a pod.). Tento názov môžeme ľubovoľne meniť a podobne, ako pri neformátovanom číselnom vstupe môžeme zadať počet desatinných miest, čím si môžeme vytvoriť napr. kurzový lístok aj s prípadným prepočtom.

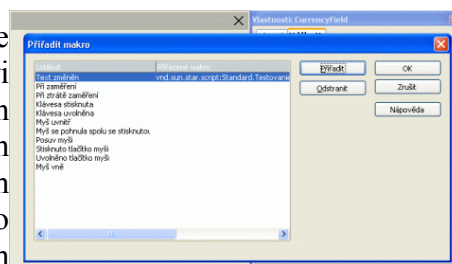
Ak budeme predpokladať, že máme definovaný dialóg „dlg“, do ktorého vložíme číselný menový vstup „Menový_vstup“, vlastnú hodnotu potom nájdeme

Obrázok 16: Nastavenie parametrov v premennej:
pre číselný vstup

`dlg.model.Menový_vstup.value`

A aby sme nezostali iba v teoretickej rovine, urobme si jednoduchý príklad na práve spomínanú možnosť kurzového lístka spojeného aj s prepočtom nákupu alebo predaja povedzme medzi českou a slovenskou korunou.

Navrhovaný dialóg môže mať jednoduchý tvar – dve obyčajné okienka pre zadanie kurzov (nákup a predaj) a tri menové okienka – v prvom zadáme, koľko českých korún chceme nakúpiť alebo prediť a vo zvyšných dvoch sa nám zobrazia príslušné údaje v slovenských korunách. Okrem toho nám do dialógu stačí pridať tlačidlo „OK“ pre jeho ukončenie a niekoľko popisných textov, aby sme sa v ňom jednoducho dokázali orientovať.



Obrázok 17: Priradenie makra
k vstupnému číselnému poličku

Okrem toho musíme k vstupnému menovému poli priradiť funkciu, pomocou ktorej budeme pri zadávaní čísla prepočítavať príslušné výstupné hodnoty. Toto dosiahneme tým, že v časti „Udalosti“ priradíme príslušné makro (nazvime si ho „Prepocet_meny“) k udalosti „Zmenený text“. O priradovaní funkcií k jednotlivým udalostiam sme sa už venovali v časti o viacokienkových dialógoch a preto to teraz spomíname už iba okrajovo.

Vlastné makrá môžu potom vyzerat' nasledovne:

```
sub Prepocet_meny
```

```
    dlg.model.Menove_pole1.value=dlg.model.Cisalne_pole1.value*dlg.model.Menove_pole1.value
```

```
    dlg.model.Menove_pole2.value=dlg.model.Cisalne_pole2.value*dlg.model.Menove_pole1.value
```

```
end sub
```

```
sub Dialog_kurzovy_listok
```

```
    DialogLibraries.LoadLibrary("Standard")
```

```
    dlg=CreateUnoDialog(DialogLibraries.Standard.Dialog2)
```

```
    REM Kurzy sa dajú zmeniť priamo aj v dialógu
```

```
    dlg.model.Cisalne_pole1.value=1.323
```

```
    dlg.model.Cisalne_pole2.value=1.299
```

```
    dlg.model.Menove_pole1.value=0
```

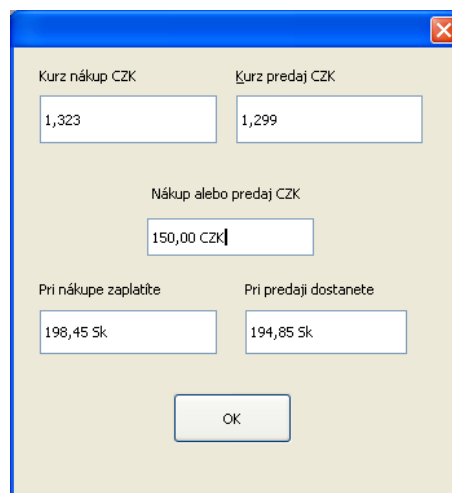
```
    dlg.model.Menove_pole1.value=0
```

```
    dlg.model.Menove_pole2.value=0
```

```
    dlg.Execute()
```

```
    dlg.dispose()
```

```
end sub
```



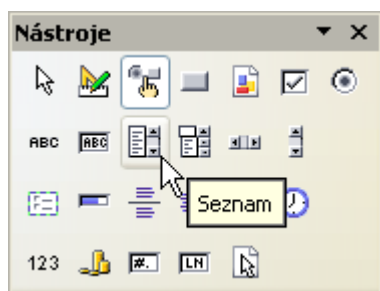
Obrázok 18: Prepočet kurzu v praxi

Textový zoznam záznamov

Pri praktickej práci niekedy požadujeme, aby sme si mohli vyberať hodnoty z určitého preddefinovaného zoznamu. A práve na takéto typy makier sa zameriame v dnešnom pokračovaní nášho seriálu.

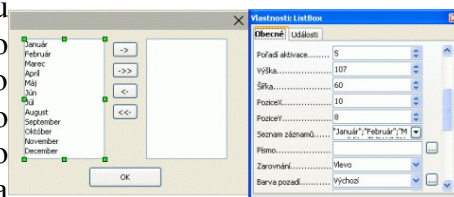
Určite sa už mnohí stretli s problematikou, keď si potrebovali vybrať určitú množinu údajov (napr. kalendárne mesiace), podľa ktorých chceli následne vykonať určité operácie. Preto si v dnešnom dieli ukážeme, ako si dokážeme vytvoriť jednoduchý dialóg so základnými funkciami pre presúvanie údajov z preddefinovaného zoznamu do druhého. Pravdaže, následné spracovanie vybraných údajov už záleží iba od konkrétnych požiadavok toho-ktorého používateľa a nebudeme sa tým teraz zaoberať.

Ako prvé, čo urobíme, je pochopiteľne vlastný návrh príslušného dialógu (povedzme si ho nazveme „Dialog2“). Ten sa bude skladať z dvoch zoznamov, pričom do prvého z nich si zároveň zadefinujeme vstupné hodnoty. Tieto sa zadávajú v položke „Zoznam záznamov“, pričom musíme upozorniť na to, že pri ich zadávaní sa musí na ich oddelenie používať kombinácia kláves „Shift“+„Enter“. Ukončenie zadávania následne vykonáme zadáním obvyčajnej klávesy „Enter“.



Obrázok 19: Výber zoznamu spustením vlastného dialógu.
záznamov

Okrem definície záznamov si musíme do dialógu zadefinovať aj tlačidlá pre presun údajov medzi jednotlivými okienkami. Aby sme ukázali všetky možnosti presunu údajov, zadefinujeme si štyri tlačidlá, ktoré budú postupne reprezentovať možnosť presunu jedného vybraného prvku alebo všetkých prvkov zo zdrojového zoznamu do výberu a prípadné spätné vrátenie jedného prvku alebo všetkých prvkov nazad do pôvodného zdrojového zoznamu. Spätné vrátenie je určené vlastne na opravy a preto nebudeme v dialógu definovať klávesu „Cancel“.



Obrázok 20: Definícia zoznamu záznamov v dialógu

Pri jednotlivých tlačidlách musíme, pochopiteľne, definovať aj inicializačné procedúry, ktoré sa vykonajú po ich stlačení.

sub PresunVybrany_Initiated

REM Procedúra pre presun vybraného prvku zo zdrojového zoznamu

dim vstupne_pole, vystupne_pole as object

' Priradíme si riadenie zoznamov k dvom pomocným premenným

vstupne_pole=dlg.getControl("Zoznam_1") ' Zdrojový zoznam

vystupne_pole=dlg.getControl("Zoznam_2") ' Vytváraný zoznam

if vstupne_pole.SelectedItem>0 then

' Ak sme vybrali nejaký prvok v zdrojovom zozname, tak ho pridáme na začiatok vytváraného zoznamu

vystupne_pole.AddItem(vstupne_pole.SelectedItem, 0)

' Vybraný prvok následne vymažeme zo zdrojového zoznamu

vstupne_pole.removeItem(vstupne_pole.SelectedItemPos, 1)

end if

end sub

sub PresunVsetko_Initiated

REM Procedúra pre presun všetkých prvkov zo zdrojového zoznamu

```

dim i as integer
dim vstupne_pole as object

vstupne_pole=dlg.getControl("Zoznam_1")

' Cyklus pre všetky prvky pôvodného zoznamu
for i=lbound(dlg.model.Zoznam_1.StringItemList()) to
ubound(dlg.model.Zoznam_1.StringItemList())
' Vyberieme vždy prvý prvok (má index 0), pretože následným vymazaním sa zoznam
posúva
vstupne_pole.SelectItem(dlg.model.Zoznam_1.StringItemList(0), 1)
' Prevedieme vlastný presun pomocou už naprogramovanej procedúry pre presun jedného
vybraného prvku
PresunVybrany_Initiated
next i
end sub

sub VratVybrany_Initiated
REM Procedúra pre vrátenie vybraného prvku do zdrojového zoznamu
dim vstupne_pole, vystupne_pole as object

vstupne_pole=dlg.getControl("Zoznam_1")
vystupne_pole=dlg.getControl("Zoznam_2")

if vystupne_pole.SelectedItem>0 then
vstupne_pole.AddItem(vystupne_pole.SelectedItem, 0)
vystupne_pole.removeItem(vystupne_pole.SelectedItemPos, 1)
end if
end sub

sub VratVsetko_Initiated
REM Procedúra pre vrátenie všetkých prvkov do zdrojového zoznamu
dim i as integer
dim vystupne_pole as object

vystupne_pole=dlg.getControl("Zoznam_2")

```

```

    for i=lbound(dlg.model.Zoznam_2.StringItemList()) to
ubound(dlg.model.Zoznam_2.StringItemList())
        vystupne_pole.SelectItem(dlg.model.Zoznam_2.StringItemList(0), 1)
        VratVybrany_Initiated
    next i
end sub

```

Teraz si trochu vysvetlíme použité metódy, aby sme si v prípade potreby mohli naprogramovať aj makrá s inými vlastnosťami.

Metóda **AddItem**(vkladaný prvok, pozícia) je určená na vkladanie prvkov a má dva parametre – prvý je text, ktorý vkladáme do príslušného zoznamu a druhý je pozícia, na ktorú ho vkladáme. Pozor, pri textových poliach každá Indexácia začína od 0. Pomocou tejto metódy by sme mohli inicializovať zoznam mesiacov napr. takto:

```

sub Init_mesiac
    dim vstupne_pole as object

    vstupne_pole=dlg.getControl("Zoznam_1")

    ' Prvky budeme vkladat' na začiatok, preto ich musíme zadať v poradí od posledného ku
    'prvému
    vstupne_pole.AddItem("December", 0)
    vstupne_pole.AddItem("November", 0)
    vstupne_pole.AddItem("Október", 0)
    vstupne_pole.AddItem("September", 0)
    vstupne_pole.AddItem("August", 0)
    vstupne_pole.AddItem("Júl", 0)
    vstupne_pole.AddItem("Jún", 0)
    vstupne_pole.AddItem("Máj", 0)
    vstupne_pole.AddItem("Apríl", 0)
    vstupne_pole.AddItem("Marec", 0)
    vstupne_pole.AddItem("Február", 0)
    vstupne_pole.AddItem("Január", 0)
end sub

```

Metóda **RemoveItems**(pozícia prvku, počet) je určená na mazanie prvkov a má dva parametre – prvý je pozícia prvého vymazávaného prvku a druhá počet vymazávaných prvkov.

Ako ste si mohli všimnúť, zoznam všetkých prvkov nájdeme v položke StringItemList, t.j. v našom prípade `dlg.model.Zoznam_1.StringItemList` alebo `dlg.model.Zoznam_2.StringItemList`.

Nakoniec si ešte uvedieme príklad makra pre vlastnú inicializáciu tohto dialógu:

```
sub makro_zoznam
```

```
DialogLibraries.LoadLibrary("Standard")
```

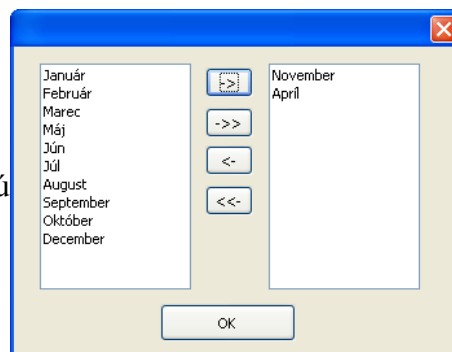
```
dlg=CreateUnoDialog(DialogLibraries.Standard.Dialog2)
```

```
dlg.Execute()
```

```
REM tu si musí každý vložiť svoje príkazy pre spracovanie vybraných prvkov zoznamu
```

```
dlg.dispose()
```

```
end sub
```

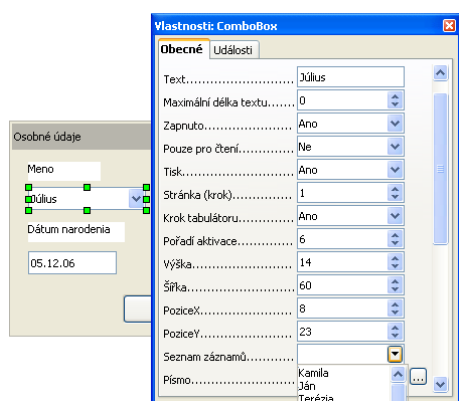


Obrázok 21: Dialóg pre výber mesiacov v praxi

Rozbaľovací zoznam záznamov

V minulom dieli nášho seriálu sme si ukázali, ako dokážeme vyberať prvky z jedného zoznamu a v prípade potreby ich prenášať do druhého a naspäť. Dnes budeme pokračovať v tejto problematike a ukážeme si výber hodnoty z rozbaľovacieho zoznamu.

Možnosti, kde sa dajú používať preddefinované zoznamy určite nemusíme pripomínať, pretože tí, ktorí sa so zoznamami stretávajú už určite vidia ich praktické použitie. Z praktického hľadiska sa nám niekedy namiesto obyčajného zoznamu hodí tzv. kombinovaný zoznam, alebo lepšie povedané kombinovaný rozbaľovací zoznam (známy aj pod názvom „Combo box“). Tento sa líši oproti bežnému zoznamu nielen tým, že v zabalenom tvare je zobrazený iba jeden záznam, ale najmä tým, že na výstupe nám vráti vždy práve jednu hodnotu, a to aj vtedy, ak si žiadnu nevyberieme – vtedy vráti základnú hodnotu, ktorú máme definovanú vlastne mimo tento zoznam – preto hovoríme o tzv. kombinovanom zozname.



Obrázok 22: Definícia rozbaľovacieho zoznamu

Pozrime sa však na to, ako sa takýto zoznam definuje. Je to veľmi podobné, ako keď sme minule definovali obyčajný zoznam – jednotlivé záznamy zadávame v jeho vlastnostiach do položky „Zoznam záznamov“ s tým, že ich musíme oddeľovať kombináciou kláves „Shift“+„Enter“, pretože zadávanie sa ukončuje klávesou „Enter“. Okrem toho, pravdaže, môžeme tento záznam definovať aj programovo pomocou metódy „AddItem“, ktorú sme si popisovali už minule, napr.:

```
dim vstupne_pole as object
```

REM Predpokladajme, že máme definovaný rozbaľovací zoznam s názvom „Zoznam“

```
vstupne_pole=dlg.getControl("Zoznam")
```

```
vstupne_pole.AddItem("naš text", 0)
```

Ako sme spomínali, v rozbaľovacom zozname máme aj jednu štandardnú hodnotu, ktorú sa nám vráti v prípade, že nevyberieme žiadnu inú položku zo zoznamu. Túto hodnotu definujeme vo vlastnostiach zoznamu hneď za jeho názvom v položke „Text“. Výstupnú hodnotu získame nakoniec takisto cez hodnotu text:

```
dlg.model.Zoznam.Text
```

Vlastnú inicializáciu dialógu vrátane následného získania vybranej hodnoty pre ďalšie spracovanie môžeme nakoniec urobiť napr. pomocou takejto jednoduchšej funkcie:

```
function Daj_položku_zoznamu
```

```
DialogLibraries.LoadLibrary("Standard")
```

```
dim co as string
```

```
dlg=CreateUnoDialog(DialogLibraries.Standard.Dialog2)
```

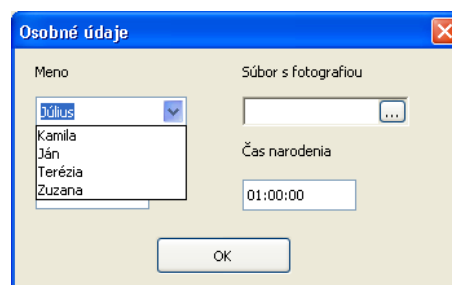
```
dlg.Execute()
```

```
co=dlg.model.Zoznam.Text
```

```
dlg.dispose()
```

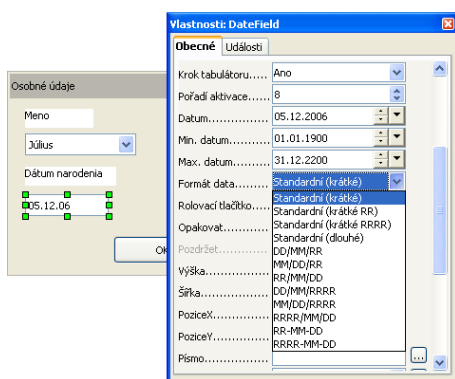
```
Daj_položku_zoznamu=co
```

```
end function
```



Obrázok 23: Rozbaľovací zoznam v praxi

Dátum a čas



Obrázok 24: Definovanie dátumového vstupu

Doteraz sme si už ukázali veľa možností vstupov v dialógoch – od zaškrtnutých či prepínacích políчков, cez rôzne číselné a textové vstupy až po zoznamy. Nakoniec nám ešte zostali možnosti vstupu dátumu, času alebo výberu mena súboru.

Pravdaže, v doterajších možnostiach sme sa už stretli aj možnosťou zadávania dátumov alebo časov pomocou formátovaného textového poľa. To nám však niekedy nemusí vyhovovať, pretože chceme dostať dátum alebo čas priamo v číselnej podobe. A na to nám slúžia iné – špeciálne vstupné položky.

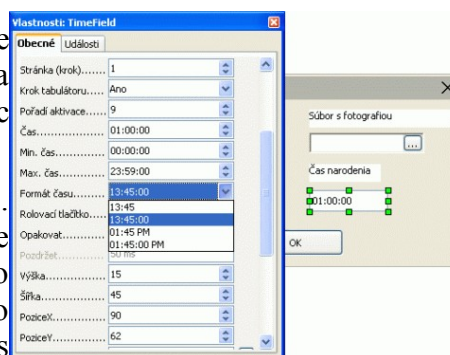
Prvá z nich je dátum. Pri jeho definícii nájdeme viacero parametrov, ktoré nás musia zaujímať. Medzi takéto patria ohraničenie rozsahu dátumu spodná horná hranica a formát, v akom chceme zadávať dátum. Okrem toho môžeme preddefinovať jeho štandardnú hodnotu. Výstupnú hodnotu následne získame pomocou

metódy „Date“, ktorá vracia číslo typu Long. Ak teda máme napr. definovanú vstupnú položku dátum s názvom „Datum“, tak pomocou štruktúry:

dlg.model.Datum.Date

získame dátumové číslo, ktoré je vždy v tvare RRRRMMDD, kde štyri cifry RRRR označujú rok, za nimi nasledujú dve cifry MM, ktoré označujú mesiac a nakoniec sú dve cifry DD, ktoré označujú deň.

Dátumovému vstupu je veľmi podobný aj časový vstup. Aj tu nájdeme parametre, pomocou ktorých môžeme obmedziť jeho rozsah. Kým pri dátumoch toto obmedzenie nemusí mať až taký význam, pri čase to môžeme veľmi prakticky využiť napr. iba na pracovný čas a pod. Pravdaže, aj tu môžeme zadať preddefinovanú hodnotu a môžeme si vybrať formát, v akom chceme čas zadávať. Výstupnú hodnotu získame pomocou metódy „Time“, ktorá vracia číslo typu Long. Pri predpoklade, že máme definovanú vstupnú položku čas s názvom „Cas“, výstupnú hodnotu získame pomocou štruktúry:



Obrázok 25: Definovanie časového vstupu

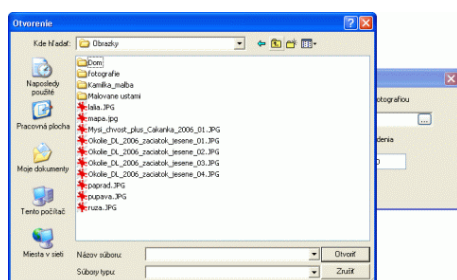
dlg.model.Cas.Time

Výstupné číslo je vždy vo formáte HHMMSSTT, kde dve cifry HH označujú hodinu (od 00 do 23, t.j. v 24 hodinovom formáte), za nimi dve cifry MM označujú minútu, následné dve cifry SS označujú sekundy a posledné dve cifry TT označujú stotiny sekúnd.

Výber mena súboru

Posledný typ vstupu, ktorý si preberieme v rámci dialógov je zadanie mena súboru pomocou jeho výberu z už existujúcich súborov. Už v minulých dieloch sme ukazovali, ako si môžeme uložiť nastavené parametre do nejakého súboru na disku. Vtedy sme však meno súboru zadávali priamo v programe. Mnohokrát však potrebujeme vybrať na spracovanie už existujúci súbor na disku. A práve na toto je určený posledný vstup, ktorý si preberáme.

Po jeho výbere nemusíme nastavovať okrem názvu (ako ho vôbec chceme zmeniť) vôbec nič, pokiaľ nepožadujeme, aby sa nám vracala nejaká preddefinovaná hodnota. Pokiaľ ju chceme zadať, musíme ju zadefinovať, podobne ako pri iných textových vstupoch, do položky „Text“. Vlastný výber súboru prevádzame stlačením tlačidla s tromi bodkami, ktoré sa nachádza v pravej časti tohto vstupného prvku. Toto tlačidlo vyvolá systémovú funkciu pre výber názvu ľubovoľného súboru. Výstupnú hodnotu získame pomocou metódy „Text“. Pokiaľ budeme predpokladať, že máme definovanú vstupnú súborovú položku „Subor“, tak výstupnú hodnotu získame



Obrázok 26: Výber súboru

pomocou nasledujúcej štruktúry:

dlg.model.Subor.Text

Príslušný dialóg môžeme následne inicializovať a výstupné hodnoty získať napríklad takouto procedúrou:

dim meno, fotografia as string

dim datum_narodenia, cas_narodenia as long

```
sub Informacie_o_osobe
```

```
DialogLibraries.LoadLibrary("Standard")
```

```
dlg=CreateUnoDialog(DialogLibraries.Standard.Dialog2)
```

```
dlg.Execute()
```

```
meno=dlg.model.Zoznam.Text
```

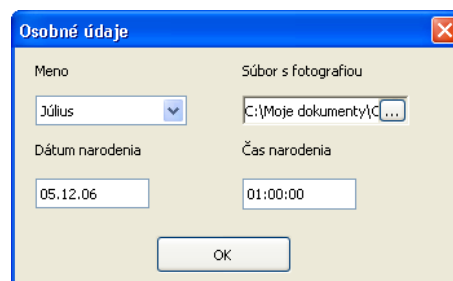
```
fotografia=dlg.model.Subor.Text
```

```
datum_narodenia=dlg.model.Datum.Date
```

```
cas_narodenia=dlg.model.Cas.Time
```

```
dlg.dispose()
```

```
end sub
```



Obrázok 27: Príklad vstupu dátumu, času a mena súboru

Dialóg pre formátovanie dokumentu

Po dlhšom čase sa znovu vrátíme k formátovaniu dokumentu s tým, že si všetky podmienky pre jeho formátovanie nastavíme v jednom dialógovom okne. Pravdaže, podľa toho následne upravíme aj príslušné funkcie a procedúry.

V minulom dieli nášho seriálu sme ukončili možnosti vstupov v dialógoch. Teraz si na niekoľkých praktických príkladoch ukážeme, ako tieto dialógy využijeme v praxi. Ako prvý príklad si zoberieme formátovanie dokumentov, ktoré sme tu už mali veľmi často, aby sme uzavreli aj túto časť nášho seriálu.

Čo vlastne chceme formátovať?

Najprv si znovu spomeňme, čo všetko sme vo formátovaní dokázali zatiaľ naprogramovať. V prvom rade to bolo odstraňovanie viacnásobných medzier (a to nielen v texte, ale aj na začiatku či na konci odstavcov). Ďalej sme si ukázali odstraňovanie nadbytočných medzier pri interpunkčných znamienkach (a naopak aj ich vkladaniu, ak chýbali) a zátvorkách. Nie malú časť formátovania dokumentu sme venovali nahrádzaniu zalomiteľných medzier za nezalomiteľné medzery za jednoznakovými predložkami a spojkami (so špecifickým problémom okolo spojky „a“), za a pred akademickými titulmi, v číslach a pod.

Venovali sme sa aj problematike automatického nahrádzania zalomiteľných medzier za nezalomiteľné priamo počas písania jednoznakových predložiek a spojok a vkladaniu tzv. pružnej nezalomiteľnej medzery, čo sa nám podarilo pomocou znaku „Word Joiner“. Nakoniec sme si ukázali jednoduchú formu ukladania nášho nastavenia do konfiguračných súborov (tejto časti sa budeme venovať ešte podrobnejšie).

Ako vidíme, problematika formátovania dokumentov, ktorej sme sa doteraz venovali je naozaj široká a preto si aj záverečné práce rozdelíme do viacerých etáp. Dnes sa budeme venovať návrhu dialógu „Formátovanie dokumentov“. Pravdaže, nebudeme sa teraz venovať vlastnému vkladaniu jednotlivých prvkov, ale si ich v krátkosti iba vymenujeme, aby sme sa nabudúce mohli na ne odvolávať.

Nezalomiteľné medzery

V rámci nezalomiteľných medzier si tieto rozdelíme na tri skupiny. Prvá sa bude venovať jednoznakovým spojкам a predložkám. Tu si zadefinujeme dve zaškrŕavacie políčka „Za_predlozkami“ a „Za_spojku_a“ a dve dvojice prepínacích políčk. Prvou dvojicou „NBSP1“ a „WJ1“ budeme určovať typ nezalomiteľnej medzery (NBSP – „Non Break Space“, WJ – „Word Joiner“) a druhou dvojicou „On_line“ a „Off_line“ budeme určovať, či sa bude nezalomiteľná medzera vkladať priamo počas písania textu, alebo iba vtedy, keď spustíme celé formátovanie dokumentu.

Ďalej si v rámci skupiny nezalomiteľných medzier zadefinujeme zaškrŕavacie políčko „Za_titulmi“, ktorým budeme určovať, či sa majú zameniť zalomiteľné medzery za a pred akademickými titulmi. K tomuto tlačidlu ďalej zadefinujeme dvojicu prepínacích políčk „NBSP2“ a „WJ2“, ktorej význam je taký istý, ako pri spojkách a predložkách, akurát sa týka akademických titulov.

Posledné – tretie zaškrŕavacie políčko v rámci skupiny nezalomiteľných medzier bude „Za_cislami“ bude určovať, či budeme vkladať nezalomiteľné medzery za jednociferné čísla, medzi číslice v rámci telefónnych čísiel, medzi čísla a merné jednotky a medzi čísla v dátumoch. K tomu zadefinujeme poslednú dvojicu prepínacích políčk „NBSP3“ a „WJ3“ pre určenie typu nezalomiteľnej predložky. Pravdaže, kto chce, môže si túto časť rozdeliť podrobnejšie – podľa jednotlivých typov (jednociferné čísla, čísla a merné jednotky...), ale to už necháme na každého používateľa podľa toho, ak to potrebuje osobitne riešiť.

Odstraňovanie nadbytočných medzier a pod.

Druhá veľká skupina, ktorú si nastavíme bude sada zaškrŕavacích políčk, ktorými určíme, čo chceme v dokumente opraviť pri jeho formátovaní. Budú to tieto tlačidlá:

„Zrus_medzery“ – pomocou tohto zaškrŕavacieho okienka určíme, že sa budú vymazávať viacnásobné medzery vo vnútri textu.

„Zrus_okraje“ – pomocou tohto zaškrŕavacieho okienka určíme, že sa budú vymazávať viacnásobné medzery na začiatku a na konci odstavcov. Možno sa teraz čudujete, prečo túto možnosť uvádzam ako osobitnú – veď na začiatku odstavca by sa medzery nemali používať. Je to síce pravda, ale nie vždy ich vymazanie je vyhovujúce... napríklad ak si skopírujete zdrojový text programu, ktorý máte napísaný prehľadne tak, že príkazy, ktoré sú vnorené do vnútra iného príkazu máte posunuté vpravo, tak máte na začiatku odstavca práve medzery – a vtedy ich tam aj potrebujete ponechať (osobne túto možnosť musím používať aj pri písaní článkov o programovaní makier v OpenOffice.org). Z podobných dôvodov sa tu nachádzajú aj ďalšie nastavenia, ktoré môžu využiť iní používatelia.

„Zrus_zatvorky“ – pomocou tohto zaškrŕavacieho okienka určíme, že sa budú vymazávať medzery za ľavými a pred pravými zátvorkami.

„Zrus_uvodzovky“ – pomocou tohto zaškrŕavacieho okienka určíme, že sa budú vymazávať medzery za dolnými a pred hornými úvodzovkami.

„Zrus_interpunkcie“ – pomocou tohto zaškrŕavacieho okienka určíme, že sa budú vymazávať medzery medzi textom a interpunkčným znamienkom (čiarka, bodka,...).

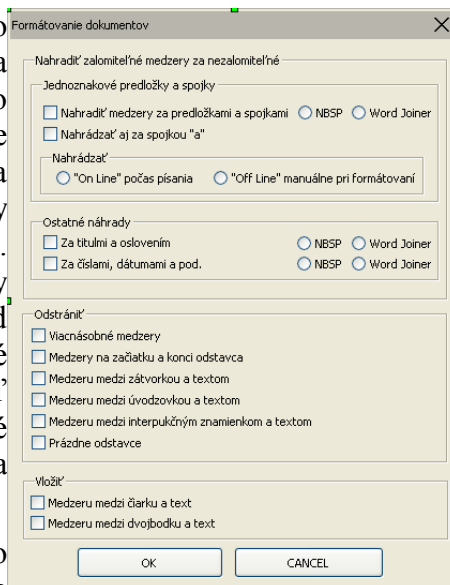
„Zrus_odstavce“ – posledným zaškrŕavacím okienkom z tejto skupiny nastavíme odstránenie všetkých prázdnych odstavcov. Tejto voľbe sme sa venovali iba v rámci popisu regulárnych výrazov, teraz si ju však aj naprogramujeme.

Vkladanie medzier

Posledná skupina sa venuje vkladaniu chýbajúcich medzier za interpunkčné znamienka. Na toto miesto si vložíme dve zaškrŕavacie tlačidlá:

„Vloz_bodku“ – pomocou tohto zaškrŕavacieho okienka určíme, či sa budú vkladat' chýbajúce medzery za znak bodky. Aj tento prípad je špecifický a to preto, lebo niekedy nám toto nevyhovuje – konkrétne v prípade, že v texte používame názov „OpenOffice.org“, kde za znakom bodky medzera nie je. Pretože sa jedná o oficiálny názov, nemôžeme ho meniť podľa pravidiel typografie. Pravdaže, dalo by sa to vyriešiť aj tak, že medzery vložíme všade a potom pri tomto texte ich nazad vymažeme – je to však nepohodlné a navyše nesystémové riešenie, pretože takýchto výnimiek môže byť viac. Zatiaľ preto necháme riešenie iba v tomto tvare a prípadné vhodnejšie riešenie si necháme niekedy nabudúce, resp. na domácu úlohu.

„Vloz_interpunkciu“ – pomocou tohto zaškrŕavacieho okienka určíme, či sa budú vkladat' chýbajúce medzery za ostatné interpunkčné znamienka.



Obrázok 28: Dialóg pre nastavenie parametrov formátovania dokumentu

Tlačidlá

Na koniec nezabudneme vložiť aj dve tlačidlá – tlačidlo pre potvrdenie zadaných údajov „OK“ a tlačidlo pre zrušenie zadaných zmien „CANCEL“.

Systémové cesty závislé od používateľa

V dnešnom dieli seriálu si ukážeme, ako môžeme automaticky ukladať rôzne inicializačné nastavenia do súborov v adresároch, ktoré sú závislé od používateľa počítača.

Už v štvrtom dieli tretej časti tohto seriálu o programovaní makier v OpenOffice.org sme si ukázali, ako si môžeme ukladať nastavenia do súboru. Vtedy sme, pre jednoduchosť, použili iba priamu cestu do konkrétneho adresára:

Subor_nezalomitelne_predlozky="file:///c:/Program files/OpenOffice.org 2.0/Nezalomitelne_Predlozky.dat"

Toto riešenie má, pravdaže, niekoľko nedostatkov. Konkrétne uvedený príklad je hádam ten najhorší, aký sa dá pri programovaní zvoliť (a práve preto je dobré, že sa zvolil, aby sa ukázali jeho všetky nedostatky).

Ako základ je to, že pokiaľ si nainštalujeme novú verziu OpenOffice.org, tak sa zmení aj jeho kmeňový adresár „OpenOffice.org 2.0“ - v súčasnosti je to pre najnovšiu verziu „OpenOffice.org 2.1“. Po odinštalovaní pôvodnej verzie, pravdaže, môžeme pôvodný adresár s príslušnými súbormi ponechať, vedie to však k zníženiu prehľadnosti.

Iná možnosť je zmeniť vlastné makro a presunúť inicializačný definičný súbor na nové miesto. Toto by sme však museli pracne robiť pri každej novej inštalácii, takže to tiež nie je práve vyhovujúce riešenie.

Pravdaže, ako miesto pre ukladanie inicializačných definičných súborov si môžeme zvoliť aj iný adresár, než je práve adresár, kde sa nachádza program OpenOffice.org, napríklad:

Subor_nezalomitelne_predlozky="file:///c:/Program files/OpenOffice.org/init_subory/Nezalomitelne_Predlozky.dat"

Toto riešenie je nezávislé od inštalácie OpenOffice.org. Má však ešte stále jednu nevýhodu – pokiaľ sa pri počítači strieda viacej používateľov, nemôže si každý nastaviť svoje vlastné podmienky, čo môže byť niekedy veľmi nevyhovujúce. Preto je najlepšie také riešenie, kde si každý používateľ uloží svoje nastavenia do osobitného adresára, ktorý mu poskytuje OpenOffice.org v rámci príslušného operačného systému.

Na tento účel máme v rámci používateľského rozhrania jazyka StarOffice Basic k dispozícii objekt „com.sun.star.util.PathSettings“, ktorý nám poskytne všetky potrebné cesty a ktorý môžeme inicializovať napríklad nasledovným spôsobom:

dim cesty as object

cesty=CreateUnoService("com.sun.star.util.PathSettings")

Objektová premenná „cesty“ obsahuje teraz nasledovné údaje:

Backup – Cesta ku kópiám dokumentov, ktoré boli automaticky zálohované.

Basic – Cesta k makrám a dialógom, ktoré sú napísané v jazyku Basic. Táto hodnota môže obsahovať viacej ciest, ktoré sú oddelené bodkočiarkou.

Favorite – Cesta k obľúbeným položkám.

Gallery – Cesta ku galérievej databáze a multimediálnym súborom. Táto hodnota môže obsahovať viacej ciest, ktoré sú oddelené bodkočiarkou.

Graphic – Táto cesta sa zobrazuje v dialógu pre otvorenie alebo ukladanie grafických objektov.

Help – Cesta k nápovedným súborom OpenOffice.org.

Module – Cesta k jednotlivým modulom OpenOffice.org.

Storage – Cesta k informáciám o pošte, súborom správ, FTP serveri a pod.

Temp – Cesta k dočasným súborom.

Template – Cesta k šablónam, ktoré máme zaradené do OpenOffice.org. Táto hodnota môže obsahovať viacej ciest, ktoré sú oddelené bodkočiarkou.

UserConfig – Cesta k adresáru pre používateľské nastavenia.

Work – Cesta k používateľskej pracovnej zložke. Používa sa v dialógu pri otváraní a zatváraní súborov a môže byť používateľom aj zmenená.

Pre zistenie všetkých ciest si môžeme napísať takéto jednoduché makro:

sub zoznam_ciest

dim cesty as object

dim zoznam, CR as string

CR=chr\$(10)

```

cesty=CreateUnoService("com.sun.star.util.PathSettings")
zoznam="Backup: "+CR+cesty.Backup+CR+CR
zoznam=zoznam+"Basic: "+CR+cesty.Basic+CR+CR
zoznam=zoznam+"Favorite: "+CR+cesty.Favorite+CR+CR
zoznam=zoznam+"Gallery: "+CR+cesty.Gallery+CR+CR
zoznam=zoznam+"Graphic: "+CR+cesty.Graphic+CR+CR
zoznam=zoznam+"Help: "+CR+cesty.Help+CR+CR
zoznam=zoznam+"Module: "+CR+cesty.Module+CR+CR
zoznam=zoznam+"Storage: "+CR+cesty.Storage+CR+CR
zoznam=zoznam+"Temp: "+CR+cesty.Temp+CR+CR
zoznam=zoznam+"Template:
"+CR+cesty.Template+CR+CR
zoznam=zoznam+"UserConfig:
"+CR+cesty.UserConfig+CR+CR
zoznam=zoznam+"Work: "+CR+cesty.Work
msgbox zoznam
end sub

```

Pokiaľ chcete zmeniť adresár, ktorý nájdeme v adresári „Work“, môžeme si na to naprogramovať takúto jednoduchú procedúru:

```

sub zmen_work (novy_work as string)
REM Predpokladáme, že cesta nie je zadaná vo
všeobecnom URL formáte
dim cesty as object
cesty=CreateUnoService("com.sun.star.util.PathSettings")
cesty.work=ConvertToURL(novy_work)
end sub

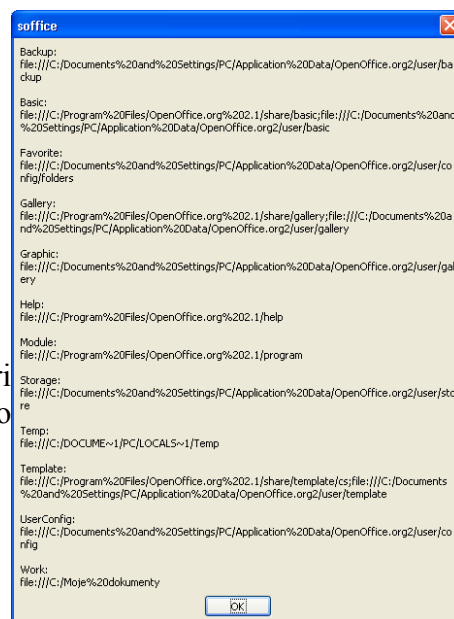
```

Alebo takúto procedúru:

```

sub zmen_work (novy_work as string)
REM Predpokladáme, že cesta je uložená vo všeobecnom URL formáte
dim cesty as object
cesty=CreateUnoService("com.sun.star.util.PathSettings")
cesty.work=novy_work

```



Obrázok 29: Zoznam všetkých ciest

end sub

Pre ukladanie našich nastavení sa javí, že bude najvhodnejšie používať adresár, kde sú uložené aj ostatné používateľské nastavenia (UserConfig). Funkciu pre zistenie názvu inicializačného súboru môžeme potom naprogramovať napríklad takto:

```
function inicializacny_subor (nazov_suboru as string) as string
    REM Funkcia vracia názov inicializačného súboru v „UserConfig“ adresári aktuálneho
    používateľa
    dim cesty as object
    cesty=CreateUnoService("com.sun.star.util.PathSettings")
    inicializacny_subor=cesty.UserConfig+"/"+nazov_suboru
end function
```

Nastavenie používateľského profilu

V minulom dieli sme si ukázali, ako je možné ukladať súbory do adresárov, ktoré sú závislé od používateľa počítača. Dnes tieto služby využijeme naplno a rozšírime ich ešte o možnosť vytvárania vlastných používateľských profilov.

Dialóg pre používateľské profily

Na tento účel si zadefinujeme ďalší dialóg, ktorý nazveme povedzme „Dialog_Profily“. Vložíme do neho nasledovné prvky:

kombinované rozbalovacie textové pole „**Zoznam_Profilov**“, v ktorom budeme mať následne zobrazené všetky profily;

vstupné textové pole „**Nový_Profil**“ pre zadávanie názvov nových profilov;

tlačidlo „**Pridaj**“ pre potvrdenie nového profilu;

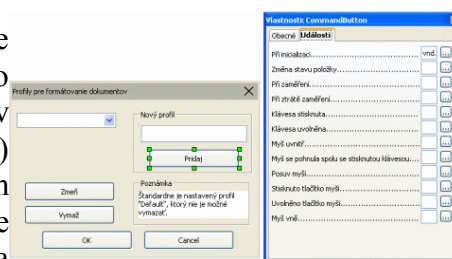
tlačidlo „**Zmeň**“ pre editovanie vybraného profilu;

tlačidlo „**Vymaž**“ pre vymazanie vybraného profilu;

tlačidlá „**OK**“ a „**Cancel**“ pre ukončenie činnosti.

Okrem toho si môžeme pridať informačný text, že „Štandardne je nastavený profil "Default", ktorý nie je možné vymazať“. Pokiaľ chceme, aby sa tento text zobrazoval na viacero riadkov, je potrebné na príslušných miestach vložiť kombináciu kláves „Shift“+„Enter“, ktorá to umožňuje.

Teraz sa pozrime na vlastné makrá, ktoré budeme pre prácu potrebovať. Vzhľadom na to, že sú úzko previazané aj s dialógom pre nastavovanie parametrov formátovania dokumentu (ktorý sme si už uvádzali) a s vlastnými funkciami pre toto formátovanie, je ich značné množstvo a preto ich neuvedíme naraz, ale postupne v niekoľkých dieloch tohto seriálu. Dnes sa budeme venovať iba tým, ktoré priamo súvisia s dialógom „Dialog_Profily“.



Obrázok 30: Dialógové okno pre nastavovanie používateľských profilov

Makrá pre prácu so zoznamom používateľských profilov

Vytvorme si v zozname makier nový modul „Profily_funkcie“, kde vložíme (zatiaľ) nasledovné definície premenných a procedúry (budeme ich priebežne komentovať):

```
dim dlg as object
dim zoznam_profilov(50) as string
dim pocet_profilov as integer
global priznaky_formatovania(16) as integer
global aktualny_profil as string
dim oVstupy as object
```

Objektová premenná „dlg“ je určená na vlastné priradenie dialógu „Dialog_Profilov“. Musí byť definovaná externe preto, lebo sa na ňu odvolávame vo funkciách, ktoré sú priradené k tlačidlám „Pridaj“, „Zmeň“ a „Vymaž“.

V premennej „zoznam_profilov“ budeme mať uložené názvy profilov. Jej rozsah sme si dali 50 profilov, čo je určite viac než dostatočný počet pre jedného používateľa. Pokiaľ nie, stačí jednoduchá zmena tohto parametra. Táto premenná je určená na testovanie názvov profilov pri ich pridávaní, aby sme nemali dva rovnaké profily. Počet profilov máme uložený v nasledujúcej premennej „pocet_profilov“.

S globálnou premennou „priznaky_formatovania“ sa dnes ešte nestretneme, lebo je určená pre dialóg „Dialog_formatovanie“, ktorý sme si už navrhovali, ale jeho makrá budeme popisovať až neskôr. Táto premenná je preto globálna, lebo sa podľa nej riadi celá množina funkcií pre formátovanie dokumentu, ktoré si tak isto budeme uvádzať až neskôr.

Do globálnej premennej „aktualny_profil“ uložíme nakoniec vybraný profil. Podobne, ako premenná „priznaky_formatovania“ je určená pre neustálu informáciu o tom, aký profil práve chceme používať pre formátovanie dokumentu.

Pomocná premenná „oVstupy“ je určená pre pridávanie a odoberanie položiek do zoznamu profilov vo vlastnom dialógu.

A teraz sa pozrime na vlastné funkcie. Pretože makrá sú pomerne dobre okomentované, nebudeme bližšie rozoberať to, ako sú naprogramované.

```
function inicializacny_subor (nazov_suboru as string) as string
' Funkcia vracia názov inicializačného súboru v „UserConfig“ adresári aktuálneho používateľa
' K názvu súboru pridávame automaticky príponu „.dat“
dim cesty as object
cesty=CreateUnoService("com.sun.star.util.PathSettings")
inicializacny_subor=cesty.UserConfig+"/"+nazov_suboru+".dat"
end function

sub zapis_nazvy_profilov
```

```

' Procedúra je určená na zápis názov profilov do definičného súboru
dim i, subor as integer

subor = Freefile
open inicializacny_subor("Zoznam_Profilov_Formatovania_Dokumentov") for Output as
#subor
for i=0 to pocet_profilov
print #subor, zoznam_profilov(i)
next i
close #subor
end sub

sub init_dialog_profilov
' Inicializácia zoznamu profilov z definičného súboru
' Vždy máme minimálne profil „Default“
dim i, subor as integer
dim riadok as string

zoznam_profilov(0)="Default"
pocet_profilov=0
oVstupy=dlg.getControl("Zoznam_Profilov")
oVstupy.AddItem("Default", 0)

' Ak nastavovací súbor neexistuje, vytvoríme ho
if not fileexists(inicializacny_subor("Zoznam_Profilov_Formatovania_Dokumentov")) the
n
    zapis_nazvy_profilov
endif
' Teraz súbor určite existuje - načítame ho
subor = Freefile
open inicializacny_subor("Zoznam_Profilov_Formatovania_Dokumentov") for Input as #s
ubor
' Inicializácia pomocného počítadla počtu profilov
i=0
While not eof(subor)
Line Input #subor, riadok
if i>0 then

```

```

' Profil Default už máme zapísaný v zozname, zaujímajú nás iba tie ostatné
pocet_profilov=pocet_profilov+1 ' aktuálny počet profilov
oVstupy.AddItem(riadok, pocet_profilov) ' pridanie profilu do dialógu
zoznam_profilov(pocet_profilov)=riadok ' pridanie profilu do zoznamu pre testovanie
end if
i=i+1
wend
close #subor

dlg.model.Zoznam_Profilov.Text="Default" ' Aby sa nám zobrazil v zozname prvý profil

end sub

sub Vloz_profil_Initiated
' Túto procedúru musíme priradiť k tlačidlu „Pridaj“!

pom_text=dlg.model.Novy_profil.Text
if pom_text<>"" then
' Názov musí byť zadaný
existuje=false
for i=0 to pocet_profilov
' Otestujeme či profil s daným názvom existuje bez ohľadu na veľkosť písmen
existuje=existuje or (ucase(pom_text)=ucase(zoznam_profilov(i)))
next i
if not existuje then
' Ak profil neexistuje, pridáme ho do zoznamu
oVstupy=dlg.getControl("Zoznam_Profilov")
pocet_profilov=pocet_profilov+1
zoznam_profilov(pocet_profilov)=pom_text
oVstupy.AddItem(pom_text, pocet_profilov)
else
' Oznam o tom, že nie je možné pridať už existujúci profil
msgbox "Profil "+pom_text+" už existuje."
end if
else
' Oznam o tom, že názov musí byť zadaný

```

```

    msgbox "Názov musí byť zadaný"
end if
end sub

sub Vymaz_profil_Initiated
' Túto procedúru musíme priradiť k tlačidlu „Vymaž“!

oVstupy=dlg.getControl("Zoznam_Profilov")
' Priradenie názvu vybraného profilu do pomocnej premennej
pom_text=dlg.model.Zoznam_Profilov.Text
if pom_text<>"Default" then
' Mažeme iba iné profily ako je „Default“
for i=1 to pocet_profilov
' Vyhľadáme pozíciu vybraného profilu
if pom_text=zoznam_profilov(i) then
' Vymažeme profil z dialógu
oVstupy.removeItem(i, 1)
for j=i to pocet_profilov-1
' Vymažeme profil z pomocného zoznamu a posunieme položky v zozname
zoznam_profilov(j)=zoznam_profilov(j+1)
next j
end if
next i
' Upravíme počet záznamov
pocet_profilov=pocet_profilov-1
' Nastavíme zobrazovaný text v rozbaľovacom poli, lebo predtým zobrazený sme práve vy-
mazali
dlg.model.Zoznam_Profilov.Text="Default"
else
' Oznam o tom, že nemôžeme vymazať profil „Default“
msgbox "Profil Deafault nie je možné vymazať!"
end if

end sub

sub Zmen_profil_Initiated

```

' Túto procedúru musíme priradiť k tlačidlu „Zmeň“!

' Uloženie názvu vybraného profilu do pomocnej premennej

pom_text=dlg.model.Zoznam_Profilov.Text

' Nastavenie parametrov pre formátovanie

Spusti_dialog_formatovania(pom_text)

end sub

sub Nastav_Profil_Formatovania

' Vlastná funkcia pre spustenie dialógu „Dialog_profil“

DialogLibraries.LoadLibrary("Vlastne_makra")

dlg=CreateUnoDialog(DialogLibraries.Vlastne_makra.Dialog_Profil)

init_dialog_profil

Citaj_definiciju_profilu("Default")

if dlg.Execute()=1 then

zapis_nazvy_profilov

Aktualny_profil=dlg.model.Zoznam_profilov.text

end if

' Ak nie je náhodou vybraný žiadny profil, nastaví sa profil „Default“

if Aktualny_profil="" then

Aktualny_profil="Default"

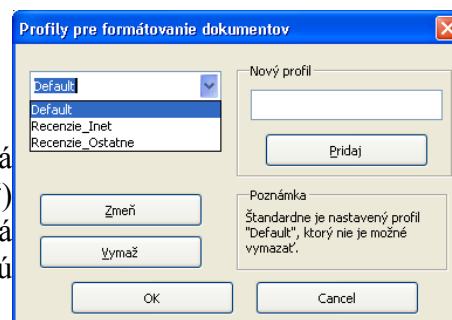
end if

Formatovanie.Nastav_formatovanie

dlg.dispose()

end sub

Procedúry „Spusti_dialog_formatovanie“ (bola volaná v procedúre „Zmen_profil_Initiated“) a „Formatovanie.Nastav_formatovanie“ (bola volaná v procedúre „Nastav_Profil_Formatovania“) budú uvedené v ďalších dieloch seriálu.



Obrázok 31: Dialóg pre nastavenie používateľského profilu

Makrá pre prácu s vlastnými používateľskými profilmi

V dnešnom pokračovaní seriálu o makrách v OpenOffice.org si uvedieme procedúry a funkcie pre nastavovanie parametrov formátovania dokumentov osobitne podľa zvoleného používateľského profilu.

V predchádzajúcom dieli seriálu sme si ukázali prácu s používateľskými profilmi s tým, že vzhľadom na značný rozsah procedúr a funkcií, ktoré s tým následne súvisia, tieto rozdelíme do viacerých skupín a uvedieme ich postupne. Dnes sa budeme venovať skupine procedúr pre nastavovanie parametrov formátovania dokumentov (dialóg „Dialog_Formatovanie“).

Podobne, ako minule, nebudeme sa vlastným procedúram venovať príliš osobitne, pretože sú pomerne dobre okomentované a uvedieme iba ich zdrojové texty. Premenné dialógu „Dialog_Formatovanie“ sme podrobne rozoberali pri jeho definícií.

Vlastné volanie dialógu „Dialog_Formatovanie“ je zabezpečené cez dialóg „Dialog_Profil“ (tlačidlo „Zmeň“) a preto nie je potrebné, aby sa volalo osobitne. Ešte spomeňme, že nasledujúce procedúry budeme ukladať do rovnakého modulu pre makrá ako minule, t.j. do modulu „Profily_funkcie“.

```
sub Zapis_definiciu_profilu(profil as string)
```

```
' Procedúra zapíše príznaky pre formátovanie do súboru pre daný profil.
```

```
' Názov súboru je totožný s názvom profilu.
```

```
dim i, subor as integer
```

```
subor = Freefile
```

```
open inicializacny_subor(profil) for Output as #subor
```

```
for i=1 to 16
```

```
print #subor, str(priznaky_formatovania(i))
```

```
next i
```

```
close #subor
```

```
end sub
```

```
sub Citaj_definiciu_profilu(profil as string)
```

```
' Procedúra načíta príznaky zo súboru pre daný profil.
```

```
' Ak súbor neexistuje, alebo je poškodený (má menej údajov ako je potrebné), bude oprave  
ný.
```

```
dim i, subor as integer
```

```
dim riadok as string
```

```
' Inicializácia na štandardné štandardné hodnoty - všetky políčka sú zaškrtnuté.
```

```
' Ako pevné medzery sa vkladajú NBSP.
```

```
' Za jednoznakovými predložkami vrátane "a" sa vkladajú NBSP „off-line“.
```

```

for i=1 to 16
    Priznaky_formatovania(i)=1
next i
' Nastavenie „off-line“ vkladania medzier za jednoznakové predložky
Priznaky_formatovania(4)=2

' Ak nastavovací súbor neexistuje, vytvoríme ho a uložíme do neho štandardné hodnoty
if not fileexists(inicializacny_subor(profil)) then
    Zapis_definiciu_profilu(profil)
endif

' Teraz súbor určite existuje - načítame ho
subor = Freefile
open inicializacny_subor(profil) for Input as #subor
i=0
While not eof(subor)
    i=i+1
    Line Input #subor, riadok
    Priznaky_formatovania(i)=val(riadok)
wend
close #subor

' Ak v súbore chýbal nejaký údaj, tak to teraz opravíme
if i<16 then
    Zapis_definiciu_profilu(profil)
endif

end sub

sub Init_dialog_formatovania(oDialog as object, profil as string)
' Inicializácia dialógu „Dialog_Formatovanie“ podľa definičného súboru vybraného profilu
.

Citaj_definiciu_profilu(profil)

```

oDialog.model.Za_predlozkami.State=Priznaky_formatovania(1)

oDialog.model.TabIndex=4

oDialog.model.TabIndex=5

select case Priznaky_formatovania(2)

case 2

oDialog.model.NBSP1.State=0

oDialog.model.WJ1.State=1

case else

oDialog.model.NBSP1.State=1

oDialog.model.WJ1.State=0

end select

oDialog.model.Za_spojku_a.State=Priznaky_formatovania(3)

oDialog.model.TabIndex=8

oDialog.model.TabIndex=9

select case Priznaky_formatovania(4)

case 2

oDialog.model.On_line.State=0

oDialog.model.Off_line.State=1

case else

oDialog.model.On_line.State=1

oDialog.model.Off_line.State=0

end select

oDialog.model.Za_titulmi.State=Priznaky_formatovania(5)

oDialog.model.TabIndex=13

oDialog.model.TabIndex=14

select case Priznaky_formatovania(6)

case 2

oDialog.model.NBSP2.State=0

oDialog.model.WJ2.State=1

case else

oDialog.model.NBSP2.State=1

```
oDialog.model.WJ2.State=0
```

```
end select
```

```
oDialog.model.Za_cislami.State=Priznaky_formatovania(7)
```

```
oDialog.model.TabIndex=16
```

```
oDialog.model.TabIndex=17
```

```
select case Priznaky_formatovania(8)
```

```
case 2
```

```
oDialog.model.NBSP3.State=0
```

```
oDialog.model.WJ3.State=1
```

```
case else
```

```
oDialog.model.NBSP3.State=1
```

```
oDialog.model.WJ3.State=0
```

```
end select
```

```
oDialog.model.Zrus_medzery.State=Priznaky_formatovania(9)
```

```
oDialog.model.Zrus_okraje.State=Priznaky_formatovania(10)
```

```
oDialog.model.Zrus_zatvorky.State=Priznaky_formatovania(11)
```

```
oDialog.model.Zrus_uvodzovky.State=Priznaky_formatovania(12)
```

```
oDialog.model.Zrus_interpunkcie.State=Priznaky_formatovania(13)
```

```
oDialog.model.Zrus_odstavce.State=Priznaky_formatovania(14)
```

```
oDialog.model.Vloz_bodku.State=Priznaky_formatovania(15)
```

```
oDialog.model.Vloz_interpunkciu.State=Priznaky_formatovania(16)
```

```
end sub
```

```
sub Spusti_dialog_formatovania (profil as string)
```

```
' Vlastná práca s dialógom „Dialog_Formatovanie“.
```

```
' Volanie tohto dialógu je zabezpečené v dialógu „Dialog_Profily“ cez tlačidlo „Zmeň“.
```

```
dim oDialog as object
```

```
' Ak nie je zadany názov profilu, budeme robiť s profilom „Default“.
```

```
if profil="" then profil="Default"
```

```
oDialog=CreateUnoDialog(DialogLibraries.Vlastne_makra.Dialog_Formatovanie)  
Init_dialog_formatovania(oDialog, profil)
```

```
if oDialog.Execute()=1 then
```

```
' Nastavenie premenných pre formátovanie podľa príznakov v dialógu.
```

```
Priznaky_formatovania(1)=oDialog.model.Za_predlozkami.State
```

```
if oDialog.model.NBSP1.State=1 then
```

```
Priznaky_formatovania(2)=1
```

```
else
```

```
Priznaky_formatovania(2)=2
```

```
endif
```

```
Priznaky_formatovania(3)=oDialog.model.Za_spojkou_a.State
```

```
if oDialog.model.On_line.State=1 then
```

```
Priznaky_formatovania(4)=1
```

```
else
```

```
Priznaky_formatovania(4)=2
```

```
endif
```

```
Priznaky_formatovania(5)=oDialog.model.Za_titulmi.State
```

```
if oDialog.model.NBSP2.State=1 then
```

```
Priznaky_formatovania(6)=1
```

```
else
```

```
Priznaky_formatovania(6)=2
```

```
endif
```

```
Priznaky_formatovania(7)=oDialog.model.Za_cislami.State
```

```
if oDialog.model.NBSP3.State=1 then
```

```
Priznaky_formatovania(8)=1
```

```
else
```

```
Priznaky_formatovania(8)=2
```

endif

Priznaky_formatovania(9)=oDialog.model.Zrus_medzery.State

Priznaky_formatovania(10)=oDialog.model.Zrus_okraje.State

Priznaky_formatovania(11)=oDialog.model.Zrus_zatvorky.State

Priznaky_formatovania(12)=oDialog.model.Zrus_uvodzovky.State

Priznaky_formatovania(13)=oDialog.model.Zrus_interpunkcie.State

Priznaky_formatovania(14)=oDialog.model.Zrus_odstavce.State

Priznaky_formatovania(15)=oDialog.model.Vloz_bodku.State

Priznaky_formatovania(16)=oDialog.model.Vloz_interpunkciu.State

Zapis_definiciu_profilu(profil)

endif

oDialog.dispose()

end sub

Pokiaľ sa predsa len nájdú používatelia, ktorí nechcú využívať výhody viacerých profilov, môžu si upraviť začiatok procedúry „Spusti_dialog_formatovania“ (ktorú v tomto prípade musia volať osobitne) napríklad takto:

sub Spusti_dialog_formatovania

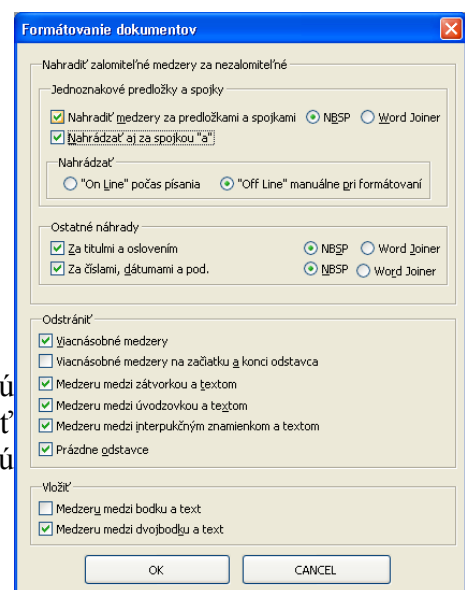
' Vlastná práca s dialógom „Dialog_Formatovanie“.

dim oDialog as object

dim profil as string

profil="Default"

Pravdaže, ako názov súboru pre uloženie parametrov (ktorý je totožný s názvom profilu a v tomto prípade je to „Default“) si môžu zvoliť ľubovoľný iný názov.



Obrázok 32: Dialóg pre nastavenie parametrov formátovania dokumentu

Makrá pre formátovanie dokumentu

V nasledujúcej časti si postupne uvedieme všetky potrebné makrá pre formátovanie dokumentu s využitím nastaviteľných parametrov v rôznych používateľských profiloch.

Nezalomiteľné medzery počas písania textu

V rámci formátovania dokumentu má osobitné postavenie vkladanie nezalomiteľných medzier za jednoznakové predložky a spojky, pretože pri týchto máme možnosť ich vkladania priamo počas písania textu, a, pravdaže, aj počas celkového formátovania dokumentu.

Čiastočne sa teda musíme vrátiť k štvrtej časti tohto seriálu, kde sme sa zmieňovanou problematikou zaoberali. Pravdaže, nemá zmysel, aby sme znovu rozoberali všetky podrobnosti, ktoré sme vtedy uviedli a preto sa hneď budeme venovať príslušným makrám.

V zozname modulov si zadefinujeme modul – „Nezalomitelna_predlozky_funkcie“, kde si naprogramujeme príslušné funkcie pre vkladanie nezalomiteľných medzier priamo počas písania textu. Pravdaže, v príslušných funkciách už budeme využívať nastavenia, ktoré sme si nastavili a vybrali podľa nami definovaných používateľských profilov.

Aby sme sa v nových nastaveniach (príznakoch) lepšie orientovali, ešte raz si v krátkosti zopakujeme ich význam aj s príslušnými hodnotami. Príznamy sú definované v globálnej premennej „Priznaky_formatovania“, ktorú sme zadefinovali v module „Profily_funkcie“:

Priznaky_formatovania(1) – príznak (=1) vkladania nezalomiteľných medzier za jednoznakovými predložkami a spojkami.

Priznaky_formatovania(2) – znak, ktorý sa vkladá za jednoznakové predložky a spojky (NBSP=1, WJ=2).

Priznaky_formatovania(3) – príznak, či sa má vkladať nezalomiteľná medzera aj za predložku „a“ (=1).

Priznaky_formatovania(4) – príznak, či sa majú nezalomiteľné medzery vkladať za jednoznakové spojky (ak je to povolené) priamo (ON-line) počas písania textu (=1).

Priznaky_formatovania(5) – príznak (=1) vkladania nezalomiteľných medzier za akademické tituly.

Priznaky_formatovania(6) – znak, ktorý sa vkladá za akademické tituly (NBSP=1, WJ=2).

Priznaky_formatovania(7) – príznak (=1) vkladania nezalomiteľných medzier za číslce a dátumy.

Priznaky_formatovania(8) – znak, ktorý sa vkladá za číslce a dátumy (NBSP=1, WJ=2).

Priznaky_formatovania(9) – príznak (=1), či sa majú odstrániť viacnásobné medzery.

Priznaky_formatovania(10) – príznak (=1), či sa majú odstrániť viacnásobné medzery na začiatku a konci odstavcov.

Priznaky_formatovania(11) – príznak (=1), či sa majú odstrániť medzery medzi zátvorkami a textom.

Priznaky_formatovania(12) – príznak (=1), či sa majú odstrániť medzery medzi úvodzovkami a textom.

Priznaky_formatovania(13) – príznak (=1), či sa majú odstrániť medzery medzi interpunkčnými znamienkami a textom.

Priznaky_formatovania(14) – príznak (=1), či sa majú odstrániť prázdne odstavce.

Priznaky_formatovania(15) – príznak (=1), či sa má vložiť medzera medzi bodku a text.

Priznaky_formatovania(16) – príznak (=1), či sa má vložiť medzera medzi dvojbodku a text.

Ale teraz sa už venujme vlastným makrám pre vkladanie nezalomiteľných medzier za jednoznakové spojky a predložky priamo počas ich písania:

```
global ObsluhaDokumentu, ObsluhaKlavesnice as object
```

```
global PosledneSlovo, OddelovaceSlov as string
```

```
global Obsluhovat_Klavesnicu as boolean
```

```
function JednoznakovéPredložky as string
```

```
' Pretože počas písania môžeme zmeniť nastavenie profilu,
```

```
' definícia zoznamu predložiek je definovaná ako funkcia
```

```
JednoznakovéPredložky="AiIkKoOsSuUvVzZ"
```

```
if Profily_funkcie.Priznaky_formatovania(3)=1 then
```

```
    JednoznakovéPredložky="a"+JednoznakovéPredložky
```

```
endif
```

```
end function
```

```
sub SpustiObsluhuKlavesnice
```

```
on error goto next
```

```
' Obsluhu klávesnice spustíme iba vtedy, ak vôbec povolili zámenu predložiek
```

```
' a zároveň chceme túto zámenu robiť ON-line
```

```
Obsluhovat_Klavesnicu=(Profily_funkcie.Priznaky_formatovania(1)=1) and (Profily_funkcie.Priznaky_formatovania(4)=1)
```

```
PosledneSlovo = ""
```

```
' medzera, nezalomiteľná medzera, tabelátor, CR, LF
```

```
OddelovaceSlov=""&chr(&HA0)&chr(&H09)&chr(&H0A)&chr(&H0D)
```

```
' pomlčka, lomítko, bodka, čiarka, výkričník, otáznik, bodkočiarka, trojbodka
```

```
OddelovaceSlov=OddelovaceSlov+"-./,!?;..."
```

```
' zátvorky
```

```
OddelovaceSlov=OddelovaceSlov+"()[]{}"
```

```
if ThisComponent.supportsService("com.sun.star.text.TextDocument") then
```

```
' Sme v textovom dokumente, kde chceme nahrádzať medzery
```

```
ObsluhaDokumentu = ThisComponent.getCurrentController
```

```
ObsluhaKlavesnice = createUnoListener("Klavesnica_", "com.sun.star.awt.XKeyListener")
```

```

    ObsluhaDokumentu.addHandler(ObsluhaKlavesnice)
endif
end sub

sub UkonciObsluhuKlavesnice
on error goto next
if ThisComponent.supportsService("com.sun.star.text.TextDocument") then
' Sme v textovom dokumente a chceme ukončiť obsluhu klávesnice
    ObsluhaDokumentu.removeKeyHandler(ObsluhaKlavesnice)
    PosledneSlovo = ""
endif
end sub

function VlozZnak(Retazec as string)
' Na aktuálnu pozíciu v dokumente vložíme reťazec
dim ViditelnyKurzor, AktualnyText, AktualnyKurzor as object

ViditelnyKurzor = ThisComponent.getCurrentController().getViewCursor()
AktualnyText = ThisComponent.getText()
AktualnyKurzor = AktualnyText.createTextCursorByRange(ViditelnyKurzor.getStart())
AktualnyText.insertString(AktualnyKurzor.getStart(), Retazec, true)
end function

function Klavesnica_KeyPressed(StlacenaKlavesa) as boolean
' Obsluha stlačenej klávesy
dim Obsluzena as boolean
'Predpokláame, že nebudeme vkladat' neزالomitel'nú medzeru
Obsluzena=False
if instr(OddelovaceSlov, StlacenaKlavesa.KeyChar)>0 then
' Došli sme na koniec slova
if Obsluhovat_Klavesnicu then
' Ak máme vôbec meniť medzery za neزالomitel'né
if ZistiPredlozku and (StlacenaKlavesa.KeyChar=" ") then
' Je to predložka, ktorá končí zalomitel'nou medzerou - túto nahradíme
if Profily_funkcie.Priznaky_formatovania(2)=1 then
VlozZnak(chr$(&HA0)) ' No Break Space (NBSP)

```

```

' Sami sme vložili medzeru - obslúžili sme stlačenú klávesu
Obsluzena=true
else
VlozZnak(chr$(&H2060)) ' Word Joiner (WJ)
' V tomto prípade sa musí ešte vložiť aj zadaná medzera a preto sme neobslúžili stlačen
ú klávesu
endif
endif
endif
PosledneSlovo=""
else
' Ešte musíme otestovať riadiace znaky, ktoré nebudeme sledovať v zadávanom slove
select case StlacenaKlavesa.KeyChar
case chr$(&H00) to chr$(&H07)
PosledneSlovo=""
case chr$(&H08) ' BS
if len(PosledneSlovo)>0 then
PosledneSlovo=left(PosledneSlovo, len(PosledneSlovo)-1)
else
PosledneSlovo=""
endif
case chr$(&H09) to chr$(&H1F)
PosledneSlovo=""
case chr$(&H20) to chr$(&H7E)
PosledneSlovo=PosledneSlovo+StlacenaKlavesa.KeyChar
case chr$(&H7F) ' CTRL BS
PosledneSlovo=""
case else
PosledneSlovo=PosledneSlovo+StlacenaKlavesa.KeyChar
end select
endif
Klavesnica_KeyPressed=Obsluzena
end function

function Klavesnica_KeyReleased(StlacenaKlavesa) as boolean
' Obsluha pustenej klávesy

```

```
Klavesnica_KeyReleased = False
end function
```

```
function ZistiPredlozku as boolean
dim JePredlozka as boolean
JePredlozka=false
if len(PosledneSlovo)=1 then
' Ak je to jednoznakové slovo, tak to môže byť predložka
JePredlozka=instr(1,JednoznakovePredlozky, PosledneSlovo, 0)
endif
ZistiPredlozku=JePredlozka
end function
```

Pravdaže, nesmieme zabudnúť ani na funkcie, ktoré nám zjednodušujú prístup k uvedeným makrám. Na tieto funkcie si zadefinujeme nový modul „Nezalomitelne_predlozky“:

```
sub Automaticky_Obsluhuj_Klavesnicu
' Procedúra pre automatické spustenie vkladania nezalomiteľných medzier
' ktorú môžeme volať počas otvárania súboru. Spustenie závisí od nastavených
' parametrov v používateľskom profile.
Formatovanie.Nastav_formatovanie
' Podľa aktuálneho profilu sa zistia príznaky
if (Profily_funkcie.Priznaky_formatovania(1)=1) and (Profily_funkcie.Priznaky_formatov
ania(4)=1) then
Nezalomitelne_predlozky_funkcie.SpustiObsluhuKlavesnice
endif
end sub
```

```
sub Obsluhuj_Klavesnicu
' Spustenie automatického vkladania bez ohľadu na nastavené parametre v používateľsko
m profile.
Nezalomitelne_predlozky_funkcie.SpustiObsluhuKlavesnice
Nezalomitelne_predlozky_funkcie.Obsluhovat_Klavesnicu=True
end sub
```

```
sub Neobsluhuj_Klavesnicu
```

' Zastavenie automatického vkladania bez ohľadu na nastavené parametre v používateľskom profile.

Nezalomitelne_predlozky_funkcie.Obsluhovat_Klavesnicu=False

Nezalomitelne_predlozky_funkcie.UkonciObsluhuKlavesnice

end sub

sub Povol_Klavesnicu

' Povolenie pozastaveného automatického vkladania bez ohľadu na nastavené parametre v používateľskom profile.

Nezalomitelne_predlozky_funkcie.Obsluhovat_Klavesnicu=True

end sub

sub Pozastav_Klavesnicu

' Pozastavenie automatického vkladania bez ohľadu na nastavené parametre v používateľskom profile.

Nezalomitelne_predlozky_funkcie.Obsluhovat_Klavesnicu=False

end sub

Funkcie pre výmenu a hľadanie

V minulom dieli sme si uviedli funkcie pre automatické vkladanie nezalomiteľnej medzery za jednoznakové predložky a spojky priamo počas písania textu. Teraz nám už nezostáva nič iné, ako si uviesť makrá pre ostatné formátovanie dokumentu, ktoré sme si v našom seriáli už viackrát spomínali. Teraz ich upravíme na nastavenia podľa parametrov, ktoré si môžeme definovať v používateľských profiloch a nebudeme sa už venovať iným verziám.

Vlastné formátovacie funkcie si uložíme do modulu „Formatovanie_funkcie“ a ponecháme ich v obecnom tvare, aby sa dali použiť aj bez obmedzení, ktoré sú nastavené v používateľských príznakoch pre formátovanie dokumentu. Tieto príznaky budeme následne akceptovať až vo funkciách celkového formátovania dokumentu (tieto si uložíme do modulu „Formatovanie“), čím dosiahneme obcejšie možnosti ich použitia.

Hoci sa makrá oproti naposledy uverejneným verziám dost' pomenili, sú dostatočne okomentované na to, aby sme sa im osobitne venovali a preto znovu uvedieme iba ich zdrojové texty. Vzhľadom na veľký rozsah týchto zdrojových textov dnes si znovu uvedieme iba makrá pre vyhľadávanie a nahrádzanie textu.

REM Definície spoločných premenných sú úplne pred definíciou podprogramov

dim Dokument, Vymena, Hladaj as object

dim NM, TAB, WJ as string

REM Inicializácia spoločných premenných

sub Init

Dokument=ThisComponent

```

Vymena=Dokument.createReplaceDescriptor()
Hladaj=Dokument.createSearchDescriptor()
NM=chr$(&HA0) ' Nezalomitel'ná medzera – hexadecimálny kód A0
TAB=chr$(&H09) ' Tabelátor – hexadecimálny kód 09
WJ=chr$(&H2060) ' Word Joiner
end sub

```

```

REM Výmena reťazca Co za reťazec Za
function Vymen(Co, Za as string, Regularne, Cele_slova, Velke_pismena as Boolean) as long
Vymena.SearchString=Co
Vymena.ReplaceString=Za
Vymena.SearchRegularExpression=Regularne
Vymena.SearchWords=Cele_slova
Vymena.SearchCaseSensitive=Velke_pismena
Vymen=Dokument.replaceAll(Vymena)
end function

```

```

REM Vlastná výmena reťazca Co za reťazec Za vo vnútri reťazca V_Com
function Zamen_Vo_Vnutri (V_Com, Co, Za as string) as string
dim pom as string
dim i, j as long
pom=V_Com
i=instr(V_Com, Co)
j=len(V_Com)-i-len(Co)+1
if i>0 then
pom=left(V_Com, i-1)+Za
if j>0 then
pom=pom+right(V_Com, j)
end if
end if
Zamen_Vo_Vnutri=pom
end function

```

```

REM Výmena vnútorného reťazca Co za reťazec Za hľadáním reťazca V_Com
function Vymen_hladanim(V_Com, Co, Za as string, Regularne, Cele_slova, Velke_pismena as Boolean) as long

```

```

dim nasiel as object
dim kolko as long
Hladaj.searchString=V_Com
Hladaj.SearchRegularExpression=Regularne
Hladaj.SearchWords=Cele_slova
Hladaj.SearchCaseSensitive=Velke_pismena
kolko=0
nasiel = Dokument.findFirst(Hladaj)
Do While NOT isNull(nasiel)
  if len(nasiel.String)>0 then
    nasiel.String=Zamen_Vo_Vnutri(nasiel.String, Co, Za)
    kolko=kolko+1
  end if
  nasiel = Dokument.findNext(nasiel.End, Hladaj)
Loop
Vymen_hladanim=kolko
end function

```

```

REM Vyhľadanie reťazca V_Com a na pozíciu Kde do neho vloží reťazec Co
function Vloz_hladanim(V_Com, Co as string, Kde as long, Regularne, Cele_slova, Velke_
pismena as Boolean) as long
  dim nasiel as object
  dim kolko, i, j as long
  dim pom as string
  Hladaj.searchString=V_Com
  Hladaj.SearchRegularExpression=Regularne
  Hladaj.SearchWords=Cele_slova
  Hladaj.SearchCaseSensitive=Velke_pismena
  kolko=0
  i=kde+1
  nasiel = Dokument.findFirst(Hladaj)
  Do While NOT isNull(nasiel)
    if len(nasiel.String)>0 then
      j=len(nasiel.String)-i+1
      pom=left(nasiel.String, i-1)+Co
      if j>0 then

```

```

    pom=pom+right(nasiel.String, j)
end if
nasiel.String=pom
kolko=kolko+1
end if
nasiel = Dokument.findNext(nasiel.End, Hladaj)
Loop
Vloz_hladanim=kolko
end function

```

Vymazávanie viacnásobných medzier

Naposledy sme si uviedli všeobecné makrá pre vyhľadávanie a nahrádzanie textu. Dnes budeme pokračovať v už konkrétnych formátovacích makrách a začneme makrami pre mazanie viacnásobných medzier. Myslíme si, že vzhľadom na rozsah iba makra pre odstraňovanie medzier je toho viac než dosť, pretože toto makro prekonalo veľmi veľké zmeny a vlastne spojilo do seba viacero drobných podprogramov.

REM Makro pre odstraňovanie viacnásobných medzier

function f_Viacnasobna_Medzera (viacnasobne, odstavce, uvodzovky, zatvorky, znamienka as integer) as long

```

' Funkcia odstráni medzery podľa parametrov (ak sa =1, tak platia):
' viacnasobne - odstránenie viacnásobných medzier v strede dokumentu
' odstavce - odstránenie viacnásobných medzier na začiatku a konci odstavca
' zatvorky - odstránenie medzery medzi zátvorkou a textom
' uvodzovky - odstránenie medzery medzi úvodzovkou a textom
' znamienka - odstránenie medzery medzi znamienkami a textom

```

```

dim kolko as Long
dim hladany_retazec as string
kolko=0

```

```

if viacnasobne=1 then
    kolko=kolko+Vymen(" +", " ", TRUE, false, false)
    kolko=kolko+Vymen(" "+NM+" |"+NM+" | "+NM, NM, TRUE, false, false)
    kolko=kolko+Vymen(" \t | \t | \t", "\t", TRUE, false, false)
endif

```

```

if odstavce=1 then

```

```

kolko=kolko+Vymen("^[: space:]*[: space:]*$", "", TRUE, false, false)
kolko=kolko+Vymen("^\t*\t*$", "", TRUE, false, false)
endif

' Makro naprogramujeme tak, aby čo najmenejkrát prechádzalo dokumentom
' T.j. najprv si pripravíme kompletný regulárny výraz a až potom budeme prevádzať výmen
y
hladany_retazec=""
if uvodzovky=1 then
' Nastavíme si regulárny výraz pre ľavé úvodzovky
hladany_retaz="(<|<<|"+chr$(&H22)
endif
if zatvorky=1 then
' Nastavíme si regulárny výraz pre ľavé zátvorky
if hladany_retazec<>"" then
' Ak už máme úvodzovky, musíme pridať znak "alebo"
hladany_retazec=hladany_retazec+"|"
else
' Ešte nemáme nič, musíme zadať začiatok regulárneho výrazu
hladany_retazec="("
endif
hladany_retazec=hladany_retazec+"(\[|\[{"
endif
if hladany_retazec<>"" then
' Ešte ukončíme regulárny výraz
hladany_retazec=hladany_retazec+")"
' Ak máme nastavený výraz pre zátvorky alebo úvodzovky, tak vymažeme nadbytočné me
dzery
' Za nadbytočné medzery považujeme obyčajné medzery, NBSP a tabelátory
kolko=kolko+Vymen_hladanim(hladany_retazec+" ", " ", "", true, false, false)
kolko=kolko+Vymen_hladanim(hladany_retazec+NM, NM, "", true, false, false)
kolko=kolko+Vymen_hladanim(hladany_retazec+TAB, TAB, "", true, false, false)
endif

hladany_retazec=""
if uvodzovky=1 then
' Nastavíme si regulárny výraz pre pravé úvodzovky

```

```

    hladany_retaz="(">|"+chr$(&H22)
endif
if zatvorky=1 then
    ' Nastavíme si regulárny výraz pre pravé zátvorky
    if hladany_retazec<>"" then
        ' Ak už máme úvodzovky, musíme pridať znak "alebo"
        hladany_retazec=hladany_retazec+"|"
    else
        ' Ešte nemáme nič, musíme zadať začiatok regulárneho výrazu
        hladany_retazec="("
    endif
    hladany_retazec=hladany_retazec+")\|}"
endif
if znamienka=1 then
    ' Nastavíme si regulárny výraz pre interpunkčné znamienka
    if hladany_retazec<>"" then
        ' Ak už máme úvodzovky alebo zátvorky, musíme pridať znak "alebo"
        hladany_retazec=hladany_retazec+"|"
    else
        ' Ešte nemáme nič, musíme zadať začiatok regulárneho výrazu
        hladany_retazec="("
    endif
    hladany_retazec=hladany_retazec+"\\.|!|\\?|;|:|..."
endif
if hladany_retazec<>"" then
    ' Ešte ukončíme regulárny výraz
    hladany_retazec=hladany_retazec+")"
    ' Ak máme nastavený výraz pre zátvorky alebo úvodzovky, tak vymažeme nadbytočné me
    dzery
    ' Za nadbytočné medzery považujeme obyčajné medzery, NBSP a tabelátory
    kolko=kolko+Vymen_hladanim(" "+hladany_retazec," ",true,false,false)
    kolko=kolko+Vymen_hladanim(NM+hladany_retazec,NM,true,false,false)
    kolko=kolko+Vymen_hladanim(TAB+hladany_retazec,TAB,true,false,false)
endif
f_Viacnasobna_Medzera=kolko
end function

```

Ostatné formátovacie funkcie

V ďalšom dieli seriálu o programovaní makier si doplníme všetky ostatné funkcie pre vlastné formátovanie dokumentu, ktoré sme doteraz v seriáli spomínali.

Minule sme si ukázali veľmi dlhé makro pre vymazávanie viacnásobných medzier. Dnes k tomu pridáme zvyšné formátovacie funkcie, s ktorými sme sa v tomto seriáli už postupne stretali – odstraňovanie prázdnych odstavcov, nahrádzanie nezalomiteľných medzier za akademickými titulmi, číslami a pod. Pravdaže, nezabudneme ani vložiť chýbajúce medzery medzi interpunkčné znamienka a text.

REM Makro pre odstraňovanie prázdnych odstavcov

```
function f_Prazdny_odstavec as long
kolko=Vymen("^$", "", TRUE, false, false)
end function
```

REM Makro pre vkladanie nezalomiteľných medzier za predložky a spojky

```
function f_Nezalomitelne_Spojky(spojka_a, NBSP as integer) as long
dim spojky as string
dim znak as string
```

' Definícia znaku, za ktorý budeme meniť zalomiteľnú medzeru

```
if NBSP=1 then
```

```
    znak=NM
```

```
else
```

```
    znak=WJ+" "
```

```
endif
```

' Definícia regulárneho výrazu pre spojky

```
if spojka_a=1 then ' Yes
```

```
    spojky="\<(A|a|I|i|K|k|O|o|S|s|U|u|V|v|Z|z) "
```

```
else ' No
```

```
    spojky="\<(A|I|i|K|k|O|o|S|s|U|u|V|v|Z|z) "
```

```
endif
```

```
f_Nezalomitelne_Spojky=Vymen_hladanim(spojky," ", znak, true, false, true)
```

```
end function
```

```

REM Makro pre vkladanie nezalomitel'nych medzier pred a za akademické tituly
function f_Nezalomitelne_Tituly (NBSP as integer) as long
dim Tituly_pred(), Tituly_pred_spec(), Tituly_za() as string
dim i, kolko as long
dim znak as string

' Definícia znaku, za ktorý budeme meniť zalomitel'nú medzeru
if NBSP=1 then
    znak=NM
else
    znak=WJ+" "
endif

    Tituly_pred()=array("Bc", "Dr", "Ing", "JUDr", "Mgr", "MUDr", "MVDr", "PaedDr", "Pha
rmDr", "PhDr", "PhMr", "RNDr", "ThDr")
    Tituly_pred_spec()=array("akad", "Akad", "arch", "Arch", "doc", "Doc", "prof", "Prof")
    Tituly_za()=array("CSc.", "DrSc.")

' Špeciálne vloženie nezalomitel'nej medzery medzi skratku h. c. a následne pred ňu.
kolko=Vymen("h. c.", "h."+znak+"c.", false, false, false)
kolko=kolko+Vymen(" h."+znak+"c.", znak+"h."+znak+"c.", false, false, false)

' Pre všetky tituly pred menom, kde sa opravuje aj veľkosť znakov
for i=lbound(Tituly_pred()) to ubound(Tituly_pred())
    kolko=kolko+Vymen("\<"&Tituly_pred(i)+"\.", Tituly_pred(i)+"."+znak, true, false, false)
next i

' Pre všetky tituly pred menom, kde sa neopravuje aj veľkosť znakov
for i=lbound(Tituly_pred_spec()) to ubound(Tituly_pred_spec())
    kolko=kolko+Vymen("\<"&Tituly_pred_spec(i)+"\.", Tituly_pred_spec(i)+"."+znak, true,
false, true)
next i

' Pre všetky tituly za menom
for i=lbound(Tituly_za()) to ubound(Tituly_za())
    kolko=kolko+Vymen(" "+Tituly_za(i), znak&Tituly_za(i), false, false, false)
next i

```

```

f_Nezalomitelne_Tituly=kolko
end function

REM Makro pre vkladanie nezalomitel'nych medzier medzi čísla a text
function f_Nezalomitelne_Cisla (NBSP as integer) as long
dim merne_jednotky as string
dim kolko as long
dim znak as string

' Definícia znaku, za ktorý budeme meniť zalomitel'nú medzeru
if NBSP=1 then
    znak=NM
else
    znak=WJ+" "
endif

' Definícia regulárneho výrazu pre merné jednotky
merne_jednotky="(m|km|mm|cm)"

' Nezalomitel'né medzery do dátumov
kolko=Vymen_hladanim("[0-9]\. [0-9]" , " ", znak, true, false, false)
kolko=kolko+Vymen_hladanim("[0-9]\. [0-9]" , " ", znak, true, false, false)

' Jednociferné čísla
kolko=kolko+Vymen_hladanim("\<[0-9]" , " ", znak, true, false, false)

' Telefónne čísla
kolko=kolko+Vymen_hladanim("[0-9] [0-9]" , " ", znak, true, false, false)

' Merné jednotky
kolko=kolko+Vymen_hladanim("[0-9] "+merne_jednotky," ", znak, true, false, false)

f_Nezalomitelne_Cisla=kolko
end function

```

```

REM Makro pre vkladanie chýbajúcich medzier pri interpunkčných znamienkach
function f_Vloz_Interpunkcia_Medzera (bodka, ostatne as integer) as long
dim kolko as long

kolko=0
if bodka=1 then
' Vloženie chýbajúcej medzery medzi znak bodky a text
kolko=Vloz_hladanim("[^0-9]\.[: alnum:]{1}"," ", 2,true, false, false)
endif
if ostatne=1 then
' Vloženie chýbajúcej medzery medzi ostatné znamienka a text
kolko=kolko+Vloz_hladanim("(\\?!|;|...)[: alnum:]{1}"," ", 1,true, false, false)
kolko=kolko+Vloz_hladanim("[^0-9](\\,|:)[: alnum:]{1}"," ", 2,true, false, false)
endif
f_Vloz_Interpunkcia_Medzera=kolko
end function

```

Záverečné funkcie

V (takmer) poslednom dieli o formátovaní dokumentu si ukážeme, ako vlastne dokážeme všetky naprogramované funkcie a dialógy využiť pomocou niekoľkých jednoduchých funkcií a procedúr.

Doteraz sme si uviedli všetky funkcie, ktoré chceme používať pre formátovanie nášho dokumentu. Zároveň sme si naprogramovali aj dialógový modul, pomocou ktorého dokážeme nastaviť viacero používateľských profilov. Ako to však všetko zjednotiť, aby sme v tom mali prehľad? Najlepšie tak, že si vytvoríme (ako sme už spomínali v minulých dieloch) modul „Formatovanie“, kde si naprogramujeme tieto makrá:

```

REM Makro pre načítanie nastavených formátovacích parametrov aktuálneho profilu
sub Nastav_formatovanie
' ON-line/Off-line nahrádzanie sa riadi manuálne alebo automaticky pomocou funkcií,
' ktoré sú definované v module "Nezalomitelne_predlozky" a tu sa tým nezaoberať
if Profily_funkcie.aktualny_profil="" then
' Ak nie je vybraný iný profil, nastaví sa defaultný profil
Profily_funkcie.aktualny_profil="Default"
endif
' Načítame parametre
Profily_funkcie.Citaj_definiciju_profilu(Profily_funkcie.aktualny_profil)
end sub

```

```

REM Makro pre celkové formátovanie dokumentu
sub Formatuj_dokument
dim kolko as long
Nastav_formatovanie
Formatovanie_funkcie.Init
kolko=Formatovanie_funkcie.f_Viacnasobna_Medzera(_
  Profily_funkcie.Priznaky_formatovania(9),_
  Profily_funkcie.Priznaky_formatovania(10),_
  Profily_funkcie.Priznaky_formatovania(12),_
  Profily_funkcie.Priznaky_formatovania(11),_
  Profily_funkcie.Priznaky_formatovania(13))
if Profily_funkcie.Priznaky_formatovania(14)=1 then
  kolko=kolko+f_Prazdny_odstavec
endif
if Profily_funkcie.Priznaky_formatovania(1)=1 then
  kolko=kolko+f_Nezalomitelne_Spojky(Profily_funkcie.Priznaky_formatovania(3),_
  Profily_funkcie.Priznaky_formatovania(2))
endif
if Profily_funkcie.Priznaky_formatovania(5)=1 then
  kolko=kolko+f_Nezalomitelne_Tituly(Profily_funkcie.Priznaky_formatovania(6))
endif
if Profily_funkcie.Priznaky_formatovania(7)=1 then
  kolko=kolko+f_Nezalomitelne_Cisla(Profily_funkcie.Priznaky_formatovania(8))
endif
kolko=kolko+f_Vloz_Interpunkcia_Medzera(Profily_funkcie.Priznaky_formatovania(15),
—
  Profily_funkcie.Priznaky_formatovania(16))
msgbox("Prevedných "+kolko+" náhrad.", 0,"Formátovanie textu")
end sub

```

REM Niekoľko makier ako ukážky aj iných volaní, než pomocou celkového formátovania dokumentu

```

REM Makro pre vymazanie prázdnych odstavcov
sub Prazdny_odstavec
Formatovanie_funkcie.Init

```

```
msgbox("Vymazanie "+Formatovanie_funkcie.f_Prazdny_odstavce()+" prázdných odstavc  
ov.", 0, "Prázdné odstavce")  
end sub
```

```
REM Makro pre vkladanie NBSP za predložky a spojky  
sub Nezalomitelne_Spojky  
Formatovanie_funkcie.Init  
msgbox("Vložených "+Formatovanie_funkcie.f_Nezalomitelne_Spojky(1,1)+" nezalomite  
lných medzier.", 0, "Nezalomitelné spojky a predložky")  
end sub
```

Niekoľko slov na záver

Pre vlastnú prácu nám teraz stačia dve makrá – prvé je makro „Nastav_Profil_Formatovania“, ktoré nájdeme v module „Profily_funkcie“ a druhé je makro „Formatuj_dokument“, ktoré sme si práve naprogramovali do modulu „Formatovanie“. Ako ste si mohli všimnúť, v makre „Formatuj dokument“ sa nachádza na konci riadkov znak podčiarknutia. Je to regulárny zápis v jazyku StarOffice Basic v prípade, že riadok je príliš dlhý a chceme ho takto rozdeliť na viacero riadkov, aby sme nestratili potrebný prehľad.

Pravdaže, ako sme si uviedli v predchádzajúcich zdrojových textoch, môžeme si vytvoriť viacero pomocných funkcií, ktoré budú volať iba niektoré z naprogramovaných funkcií so špecifickým nastavením parametrov. To všetko však už záleží od každého používateľa a jeho požiadavok.

Končíme piatu, veľmi veľkú časť o programovaní makier v OpenOffice.org. V šiestej časti budeme ešte stále pokračovať v oblasti práce s textovým procesorom Writer a uvedieme si niekoľko zaujímavých a praktických makier, ako je napríklad vzájomná záměna znakov Word Joiner za Non Break Space, záměna troch bodiek za znak „trojbodky“ a pod.

V prílohe PDF dokumentu je uložený súbor Makra5.odt, ktorý obsahuje v sebe všetky posledné makrá vrátane definície dialógov. Zdrojové texty makier, ktoré sme uvádzali na začiatku sa už v prílohe nenachádzajú, pretože boli použité iba ako prechodné a prípadní záujemci si ich musia prepísať sami.