MS Word a Makra – Makra nejsou jen viry!

Martin Fussek

Katedra měřicí a řídicí techniky, VŠB-Technická univerzita Ostrava, FEI, tř. 17. listopadu, 708 33 Ostrava – Poruba, Česká republika, e-mail: Martin.Fussek@vsb.cz, http://kat455.vsb.cz

Abstrakt

Příspěvek předkládá praktické zkušenosti z vytváření „programů v programu“ – maker - v balíku MS Office (se zaměřením na MS Word). Protože se jedná o málo známé (natož v praxi využívané) využití programu, které však velmi zjednodušuje každodenní práci běžným i pokročilým uživatelům, klade si příspěvek za cíl ukázat jak jednoduché a lehce zvládnutelné je programování vlastních maker v objektově orientovaném jazyce Visual Basic. Příspěvek zhodnocuje zkušenosti, popisuje konkrétní techniky tvorby maker a jejich zapojení do každodenního užívání.

1. Úvod

Chceme-li hovořit o makrech, jejich tvorbě a následném praktickém využití, musíme si nejprve objasnit pojem makro. Pro většinu uživatelů jakéhokoliv textového editoru je pojem **makro** pouze jednou z položek menu a mají pramalou představu jak této položky využít.

Makra nejsou v textových editorech (a obecně v jakémkoliv softwaru) ničím novým (podporu maker v sobě obsahoval již textový editor T602!). Tedy co je makro:

Makro je sled (posloupnost) akcí (příkazů), které se postupně vykonávají v rámci nějakého programu a které může uživatel spustit najednou.

Z této velmi jednoduché definice vyplývá jejich užití. Makra jsou vhodná tam, kde potřebujeme opakovat určitý sled akcí, tak abychom obdrželi požadovaný výsledek. Je jasné, že pokud se bude jednat o více než 5 akcí v rámci jedné série, bylo by vhodné si těchto pět akcí zapamatovat a posléze jejich vykonání spustit jediným kliknutím myši.

V současné době je však představa maker rozšířena z původně příkazově orientovaného makro jazyka do programovatelného jazyka maker, který v sobě zahrnuje možnost vytvářet struktury ovládacích prvků, pracovat se subrutinami, funkcemi a možnosti přístupu k Windows API.

Současné makro můžeme tedy nazvat *aktivním* prvkem, protože umožňuje vytvářet složité programové struktury a navíc může komunikovat s uživatelem prostřednictvím grafického uživatelského prostředí.

Demonstrujme si myšlenku maker na praktickém případu číslování vzorců. Každý kdo někdy vytvářel dokument a chtěl v něm číslovat vzorce a na tyto se dále v textu odkazovat mi určitě dá za pravdu, že je to mravenčí a velmi nepříjemná práce, zvláště pokud se rozhodneme změnit pořadí, či přidat některý vzorec. Takový vložený vzorec může vypadat například takto:

$$\int \frac{\sin x}{x} dx \qquad (1)$$

S využitím makra se však vytvoření i velmi dlouhého dokumentu obsahujícím desítky vzorců stává velmi jednoduchou záležitostí. Makro totiž umožní vložit číslo vzorce (aniž by se uživatel musel zajímat o hodnotu, program si ji vždy vypočte automaticky podle pozice vzorce v dokumentu), zarovnat jej doprava, vytvořit jeho jméno (aby bylo možno se na něj v textu odkazovat) a nakonec vložit samotný vzorec.

Odkaz na daný vzorec pak může vypadat následovně (1). O vložení správného odkazu se také postará makro, které otevře okno se seznamem všech vzorců, uživatel vybere ze seznamu požadovaný vzorec, potvrdí svou volbu a odkaz je automaticky vložen. Rychlé, jednoduché a velmi prosté.

Podobným způsobem můžeme vytvořit makro pro vkládání a popisování obrázků/grafů a odkazování se na ně.

Poznámka: Zmíněná a některá další makra jsou dostupná i s popisem a obrázky na http://kat455.vsb.cz/sablony

2. Makro jazyk

K psaní maker je využíváno tzv. makro jazyků. Makro jazyky prošly vývojem stejně jako jiné programovací jazyky a v současné době jsou to jazyky hybridní, tedy jak procedurálně, tak objektově orientované (přesněji objektově zaměřené). Makro jazyk využívaný v produktech firmy MS má svůj základ v jazyce Visual Basic a nazývá se Visual Basic for Application (VBA).

Současné jazyky pro tvorbu maker poskytují:

- Jazyk založený na funkcích a subrutinách s orientací na objekty;
- Používání standardních programovacích ovládacích struktur (If…Then…Else, Do…Loop, For…Next, Select Case, atd.);
- Možnost zachycování a ošetřování výjimek (On Error…);
- Ošetření událostí (OnClick, OnOpen atd.)
- Plný přístup k příkazům a funkcím hostitelské aplikace (v našem případě Word);
- Vestavěné editory pro psaní a tvorbu kódů (editor jazyka Visual Basic);
- Možnost ladění;
- Vytváření vlastního uživatelského rozhraní – GUI (dialogová okna, nabídky - menu, panely nástrojů, nápověda, atd.);

- Možnost využívat GUI hostitelské aplikace, případně jej měnit a doplňovat.
- Využívání programového rozhraní aplikace (API) Windows, vlastních DLL knihoven.

Z uvedeného seznamu je patrné, že makro jazyk poskytuje dostatečné nástroje, aby s jeho pomocí vznikly aplikace usnadňující každodenní používání programů. Připočteme li navíc možnost propojení všech produktů Office a přístup k datům v databázích, dostáváme do ruky velmi silný nástroj pro tvorbu komplexních kancelářských aplikací.

Ačkoliv aplikace založené na Office, nemohou nikdy v oblasti programování nahradit „plnohodnotné“ programovací jazyky (C++, SmallTalk, Delphi, Java) jsou tyto aplikace výhodné zejména z těchto důvodů:

- Umožňují vytvářet aplikace „na míru“ pouhým přizpůsobením již hotové hostitelské aplikace (Poznámka: Zkuste číslovat vzorce standardními nástroji Word pomocí titulků - zcela nepoužitelné. Na druhé straně výše zmíněné makro také využívá standardní funkce a je velmi dobře použitelné.)
- Pracují rychle, protože mohou využívat standardních funkcí hostitelské aplikace;
- Umožňují uživatelům zůstat ve známém pracovním prostředí;
- Umožňují vytvářet vlastní aplikace i zkušeným uživatelům s minimální znalostí programování.

Tento příspěvek však není zaměřen na tvorbu takových komplexních aplikací, ale na makra využitelná (a naprogramovatelná) běžným uživatelem. Zájemce o tvorbu složitějších aplikací odkazuji na knihu „Tvorba aplikací v Microsoft Office“ od Christine Solomon, která je česky a poskytuje dostatek základních informací.

3. Vytváříme vlastní makro

Chceme-li vytvořit vlastní makro, máme dvě možnosti:

1. Využijeme tzv. „Záznamu makra“
2. Napíšeme vlastní kód

3.1 Záznam makra

Záznam makra je nejjednodušší způsob, jak vytvořit makro. Jedná se o jakousi „kameru“, které sleduje práci uživatele (pohyby myší, práci s klávesnicí atp.) a převádí ji do makro jazyka. V programu MS Word 97 se ke kameře dostaneme prostřednictvím položky menu Nástroje/Makro/Záznam nového makra… Ve stejném menu pak můžeme námi zaznamenané makro spustit.

Tento způsob vytváření maker je zvláště vhodný pro začátečníky a jednoduchá makra. Ovšem záznamu makra využijí i ti, kteří se chtějí naučit makra vytvářet, protože než studovat dlouhý seznam parametrů určitého příkazu, je výhodnější si tento sestavit pomocí záznamu makra a pak zaznamenané makro otevřít v editoru a prostudovat jej.

Jako příklad takového makra nám může posloužit makro, které použijete při konečných úpravách dokumentu. Toto makro postupně v dokumentu vyhledá všechny typografické nesrovnalosti jako například „(_“, „._“ atp. a nahradí je správnou posloupností znaků, tedy „_)“ resp. „._“atp. Znak _ představuje mezeru.

Takové makro lze nejsnadněji vytvořit spuštěním záznamu makra a následným nahrazováním všech výskytů špatných posloupnosti znaků jejich správnou posloupností (menu Úpravy/Nahradit…). Po nalezení a nahrazení všech kombinací ukončíme záznam makra a makro je vytvořeno a můžeme ho kdykoliv znovu využít.

Znovu zdůrazňuji, že tento způsob vytváření maker je určen pouze pro vytvoření těch **nejjednodušších** maker. Slepením několika různých maker dohromady nevzniká aplikace, ale jakýsi na pohled sice funkční, ale zcela nečitelný, neupravitelný a neodladitelný slepenec, protože takovýto slepenec neobsahuje žádné proměnné, žádné programové struktury ani komentáře.

3.2 Psaní kódu

Rozhodneme-li se pro tento způsob tvorby maker, otvírají se před námi netušené možnosti. Nejdůležitější je, že nemusíme **vůbec NIC** instalovat ani dokupovat, protože vše je součástí standardní instalace a to včetně vývojového prostředí podobnému „standardním“ vývojovým prostředím typu MS Developer Studio, Delphi, Java Workshop. V této souvislosti značně překvapí, že makra nejsou skoro vůbec využívána, když je vlastně každý uživatel má na svém počítači a může je kdykoliv využívat. (Ovšem na druhé straně uvědomíme-li si, kolik uživatelů nepochopilo ani význam tak banální věci jako jsou styly odstavců, není vůbec překvapivé, že nevyužívají ani maker.)

Nadto je samotný makro jazyk Visual Basic poměrně jednoduchý, v určitém pohledu výkonný, a hlavně velmi rychle zvládnutelný jazyk, a to i bez speciálních příruček pouze za pomoci kontextově sensitivního „helpu“, který je vyvolatelný z vývojového prostředí pomocí klávesy F1. Nápověda je velmi slušně zpracována (daleko lépe a kvalitněji, než nápověda k samotnému textovému editoru!) a je obohacena o velké množství, ve většině případů funkčních, příkladů. Upozorňuji na skutečnost, že nápovědu je nutno si nainstalovat pomocí instalátoru Office, protože není součástí standardní instalace (což ovšem není popřením předchozího tvrzení, že nic není třeba instalovat, protože soubor s nápovědou není pro tvorbu maker nutný!).

Shrneme-li dohromady dostupnost, vývojové prostředí, jazyk a nápovědu, zjistíme, že se nám do rukou dostává opravdu velmi silný nástroj pro tvorbu vlastních maker (o čemž ostatně svědčí i velké množství virů vytvořených právě v makro jazyce programů skupiny Office), který nám umožní vytvářet skutečné „aplikace v aplikacích“.

4. Vlastní makra

Nejjednodušší situací, která při tvorbě makra může nastat, je vytvořit makro, které zobrazí dialogové okno (nazývané Formulář) vyžadující zadání různých informací od uživatele a jejich následné vložení do aktivního dokumentu. Takovéto jednoduché

makro můžeme s výhodou využít například ve spojení se šablonou hlavičkového papíru firmy, kdy po vytvoření nového dokumentu je automaticky spuštěno makro, které zobrazí okno, počká až uživatel vyplní vyžadované informace, tyto vloží do dokumentu a nakonec přesune kurzor na místo, kde začíná text.

Poznámka: Praktickým příkladem, jak využít šablony a makra je psaní příspěvků na konference. Každoročně dostáváme od pořadatelů Tvorby softwaru „vodítko pro psaní textu“. Každý kdo napíše příspěvek jej pak poměrně složitě podle uvedené předlohy formátuje (pokud nemá schovány příspěvky z předchozích ročníků, které využije jako šablony) a odesílá ho (ve formátu Microsoft Word– jeden z požadavků) pořadatelům. Pořadatel obdrží desítky dokumentů, které nemají žádné sjednocující prvky, kromě toho že **vypadají** shodně. Přitom k tomu, aby všechny příspěvky **byly** (ne jenom vypadaly) shodné, by stačilo, aby pořadatel rozeslal všem šablonu, která by obsahovala požadované formátování a makra, která by zajistila, aby popisy obrázků, vzorce, úvodní hlavička atd. byly přesně tam kde mají být a měly formát jaký mají mít.

Tuto situaci považuji za nejjednodušší, protože součástí vývojového prostředí je editor dialogových oken a veškeré programování spočívá v napsání kódu procedur ošetřující různé události okna.

Z příkladu (obr 1.) je vidět, že pro vytvoření jednoduchého makra stačí znát několik málo příkazů a vytvoření funkčního makra je pro zkušeného programátora otázkou několika málo minut.

Dokonce i pro koncového uživatele je nasazení tohoto makra v praktických podmínkách nenáročné a ve svém důsledku i příjemné, protože od něj nevyžaduje žádné neobvyklé akce, nic se nemusí učit ani přizpůsobovat. Jako obvykle vytvoří nový dokument a jediná změna pro něj nastane v tom, že již nemusí vyhledávat místa v dokumentu, do kterých vepisoval požadované údaje, ale vše vyplní v jediném okně a makro vykoná zbytek za něj.

4.1 Programujeme

Základním prostředím pro programování vlastních maker je editor jazyka Visual Basic. Editor se spouští z menu Nástroje/Makro/Editor jazyka Visual Basic, nebo zkratkou Alt + F11. Po spuštění editoru musíme do našeho projektu vložit buď formulář (vytvářející okna GUI a nevíc obsahující kód ošetřující události okna), Modul (obsahující programový kód bez GUI) nebo Modul třídy (pro vytváření vlastních tříd).

Chceme-li vložit modul, vložíme jej z menu Vložit/Modul. Po volbě se vytvořil Modul1 do kterého můžeme začít psát vlastní subrutiny a funkce. V tuto chvíli doporučuji spustit nápovědu (F1) a začít studovat Nápovědu, která je nejlepším startovním bodem pro vytváření vlastních maker.

5. Závěr

Závěrem shrnu některé vlastní zkušenosti z vytváření aplikací v prostředí Office (Word) a jejich zavádění do praktického života.

S makry v programech Office jsem se setkal již v době Word verze 2, tedy v době, kdy byly na vrcholu slávy textové editory ESO, T602 a AmiPro. V té době jsem byl jeden z mála, kdo používal tento editor pro psaní textů a navíc v něm programoval. V té době nebyla dostupná žádná literatura o tom, jak makra v WordBasic (předchůdce VBA) vytvářet, přesto nebyl problém se s makry naučit pracovat. Jedno z prvních maker vůbec bylo právě číslování vzorců… Postupem času se mi podařilo na Word přeorientovat rozsáhlou skupinu spolužáků a to právě díky makrům, které poskytovaly to, co nebylo v žádném z editorů dostupné.

Touto krátkou exkurzí do minulosti jsem chtěl ukázat, že naučit se vytvářet makra je poměrně jednoduché a vyžaduje jediné: umět ovládat program jako celek. Tedy být zkušeným uživatelem, vědět, čeho chci dosáhnout a jaké možnosti mi program poskytuje - toho lze docílit jedině aktivním používáním programu. Zbytek je jenom záležitostí studia Nápovědy a zkušeností získaných vytvářením maker.

Na příkladu spolužáků je vidět, že přešli k Word jenom proto, že jim poskytoval *něco* navíc. A to něco nebylo dáno tím, že by se v té době Word výrazně lišil od konkurentů, ale tím, že v sobě (díky makrům) obsahoval to, co jsme jako studenti potřebovali pro svou každodenní práci při vytváření nejrůznějších protokolů – automatizované vytváření hlavičky, číslování vzorců, popisy obrázků/grafů, jednotné formátování atp.

5.1 Zkušenosti s makry

Klady:

1. Dostupnost: možnost vytvářet a používat makra nevyžaduje speciální součásti, vše je již ve standardní instalaci.
2. Naučit se vytvářet makra je jednoduché a vyžaduje pouze minimální znalost programování.
3. Po proniknutí do problematiky vytváření maker a uživatelských aplikací je možné vytvářet komplexní kancelářské aplikace netušených možností.
4. Makra je možno „uzamknout“ a tím znemožnit jejich narušení.
5. Málokdy dochází k nekorektnímu chování maker. Nekorektní chování je většinou způsobeno poškozením instalace hostitelského programu jiným programem. V nejhorším případě je nutná reinstalace Windows.
6. Uživatelské aplikace založené na makrech jsou dostatečně rychlé.
7. Nechuť uživatelů používat makra velmi rychle opadá, když zjistí, že jim výrazně ulehčují práci. Další makra většinou vítají.
8. Uživatelé, kteří začnou využívat šablony s makry začnou využívat i další funkce programu, které dosud nevyužívali.
9. Možnost využívat makra lokálně s vazbou na určitou šablonu, nebo globálně a rozšířit tak možnosti celé aplikace.
10. Pro dosažení požadovaného cíle není nutno dokupovat další software a nutit uživatele pracovat s dalšími programy.

Zápory:

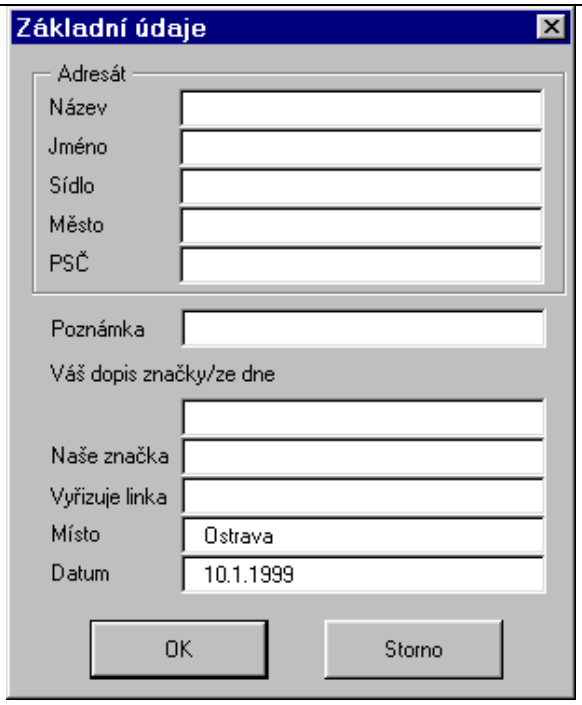
1. Soubory s makry rychle přibývají na velikosti. Velikost ovšem nemá viditelný vliv na rychlost zpracování. Jde spíš o záležitost vnitřní komprese, protože soubor zkomprimovaný pomocí ZIP se zmenší až 10x.
2. Velmi málo uživatelů ví o možnosti využívat makra.
3. Nedořešené ovládání editoru dialogových oken (specielně ve Word).
4. Někdy velmi podivné a nestandardní chování GUI (z pohledu programátora).
5. Nechci vyvolat dojem, že makra nemají žádné zápory a práce s nimi je vždy jenom příjemná, ale záporných zkušeností mám opravdu velmi málo a většinou se týkají zrušení, či přejmenování některých funkcí dostupných v předchozích verzích Word a nedostupných ve verzi současné a problémy s GUI.

6. Příklad

Následující příklad ukazuje konkrétní příklad makra využitého v šabloně hlavičkového papíru. Makro vyžaduje, aby v dokumentu byly přítomny záložky „Nazev“, „Jmeno“ (další názvy viz kód) na místech kam má být vložen příslušný text z textového pole formuláře. Formulář musí obsahovat prvky textových polí s názvy „TextBox_Název“, „TextBox_Jméno“ (další názvy viz kód) a dvě tlačítka s názvy „Button_OK“ a „Button_Cancel“.

```
REM Kód ošetřující události okna
REM Deklarace proměnných je explicitně vyžadována
Option Explicit

REM Ošetření události tlačítka Storno
Private Sub Button_Cancel_Click()
REM Skrytí okna
   Me.hide
REM Zobrazení hlášení uživateli
   MsgBox "Vyplňování přerušeno, nyní musíš vše vyplnit sám.", vbInformation,
"Přerušení"
REM Nastavení vlastnosti dokumentu na „nezměněn“, to umožní uživateli zavřít
REM dokument bez hlášení „Dokument není uložen. Uložit?“
   ActiveDocument.Saved = True
End Sub

REM Ošetření dvojkliku na formuláři
Private Sub UserForm_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
REM Volání další subrutiny
   Call Ošabloně
End Sub

REM Ošetření události Inicializace formuláře
Private Sub UserForm_Initialize()
REM Předvyplnění textového pole datum formuláře
   Me.TextBox_Datum.Value = Date
REM Text ve stavovém řádku aplikace
```

```
   StatusBar = "Doplň požadované informace, které se zobrazí v hlavičce hlavičkového papíru tak jak je uvedeno"
End SubPrivate

REM Ošetření události tlačítka OK
Sub Button_OK_Click()
REM Skrytí okna
   Me.hide
REM Přechod na záložku v dokumentu a vložení textu z textového pole okna
   Selection.GoTo What:=wdGoToBookmark, Name:="Nazev"
   Selection.TypeText Text:=Trim(Me.TextBox_Název)
   Selection.GoTo What:=wdGoToBookmark, Name:="Jmeno"
   Selection.TypeText Text:=Trim(Me.TextBox_Jméno)
   Selection.GoTo What:=wdGoToBookmark, Name:="Poznamka"
   Selection.TypeText Text:=Trim(Me.TextBox_Poznámka)
   Selection.GoTo What:=wdGoToBookmark, Name:="Sidlo"
   Selection.TypeText Text:=Trim(Me.TextBox_Sídlo)
   Selection.GoTo What:=wdGoToBookmark, Name:="PSC"
   Selection.TypeText Text:=Trim(Me.TextBox_PSČ)
   Selection.GoTo What:=wdGoToBookmark, Name:="Mesto"
   Selection.TypeText Text:=Trim(Me.TextBox_Město)
   Selection.GoTo What:=wdGoToBookmark, Name:="VaseZnacka"
   Selection.TypeText Text:=Trim(Me.TextBox_VášDopis)
   Selection.GoTo What:=wdGoToBookmark, Name:="NaseZnacka"
   Selection.TypeText Text:=Trim(Me.TextBox_NašeZnačka)
   Selection.GoTo What:=wdGoToBookmark, Name:="Vyrizuje"
   Selection.TypeText Text:=Trim(Me.TextBox_VyřizujeLinka)
   Selection.GoTo What:=wdGoToBookmark, Name:="MistoOdeslani"
   Selection.TypeText Text:=Trim(Me.TextBox_MístoOdeslání)
   Selection.GoTo What:=wdGoToBookmark, Name:="Datum"
   Selection.TypeText Text:=Trim(Me.TextBox_Datum)
REM Přechod na záložku začátku dokumentu
   Selection.GoTo What:=wdGoToBookmark, Name:="ZačátekDokumentu"
End Sub
```



Obrázek 1. : Dialogové okno hlavičkového papíru