Efektivní využití maker v administrativní praxi

Gabriel Bílý

Bakalářská práce
2017

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2016/2017

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Gabriel Bílý**
Osobní číslo:	**A14222**
Studijní program:	**B3902 Inženýrská informatika**
Studijní obor:	**Informační technologie v administrativě**
Forma studia:	**prezenční**
Téma práce:	**Efektivní využití maker v administrativní praxi**
Téma anglicky:	**The Effective Use of Macros in Administration**

Zásady pro vypracování:

1. **Vypracujte literární rešerši na dané téma.**
2. **Definujte opakující se tabulkové operace v administrativní praxi.**
3. **Navrhněte automatizované řešení opakujících se tabulkových operací pomocí maker.**
4. **Vyčíslete úsporu času pracovníka při využití maker.**
5. **Ověřte využitelnost Vašeho řešení v praxi.**

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **ALEXANDER, Michael T. 101 ready-to-use excel macros (mr. spreadsheet). Indianapolis, IN: Wiley Pub., 2012. ISBN 1118281217.**
2. **BARILLA, Jiří, Pavel SIMR a Květuše SÝKOROVÁ. Microsoft Excel 2016: podrobná uživatelská příručka. Brno: Computer Press, 2016, 456 s. Podrobná uživatelská příručka. ISBN 978-80-251-4838-9.**
3. **BENÁČANOVÁ, Helena. Tvorba aplikací v Microsoft Office Excel. V Praze: Oeconomica, 2009. ISBN 978-80-245-1602-8.**
4. **BENÁČANOVÁ, Helena. Tvorba aplikací v MS Excel. Praha: Oeconomica, 2007. ISBN 978-80-245-1271-6.**
5. **ČERNÝ, Jaroslav. Excel 2000-2007: záznam, úprava a programování maker. 2., aktualiz. vyd. Praha: Grada Publishing, 2008, 183 s. Průvodce. ISBN 978-80-247-2305-1.**
6. **LAURENČÍK, Marek a Michal BUREŠ. Programování v Excelu 2010 & 2013: záznam, úprava a programování maker. Praha: Grada, 2013, 198 s. Průvodce. ISBN 978-80-247-5033-0.**
7. **LIEW VOON KIONG. Excel VBA made easy: a concise guide for beginners. Lexington, KY: Liew Voon Kiong, 2009. ISBN 1449959628.**

Vedoucí bakalářské práce: **Ing. Michal Pleva**
Ústav počítačových a komunikačních systémů

Datum zadání bakalářské práce: **3. února 2017**

Termín odevzdání bakalářské práce: **30. května 2017**

Ve Zlíně dne 3. února 2017

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
L.S.

doc. Mgr. Milan Adámek, Ph.D.
děkan

Ing. Miroslav Matýsek, Ph.D.
ředitel ústavu

Jméno, příjmení: Gabriel Bílý

Název bakalářské práce: Efektivní využití maker v administrativní praxi

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl jsem seznámen s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 24.5.2017

.....
podpis diplomanta

ABSTRAKT

Bakalářská práce se zabývá tvorbou maker a jejich použití. Srovnává možnosti maker v Microsoft Excel a volně dostupného Apache OpenOffice pro zefektivnění běžné administrativní činnosti. Cílem je zautomatizovat opakující se kroky využívané v administrativní praxi a navrhnout nejefektivnější řešení pro konkrétní příklady. Makra jsou vytvořena pomocí VBA v MS Excel, následně ověřena v praxi a na základě zjištění je vyčíslena možná úspora času při jejich využití.

Klíčová slova: makra, zautomatizování, administrativní činnost, Microsoft Excel, VBA, Apache OpenOffice, Basic

ABSTRACT

The bachelor thesis deals with the creation of the macros and their use. It compares macro options in Microsoft Excel with freely available Apache OpenOffice for the efficiency of routine administrative tasks. The aim of the work is to automate repetitive steps used in administrative practice and to propose the most effective solution for specific examples. Macros are created by using VBA in MS Excel, then they are verified in a practice and based on the findings, possible time saving is evaluated when using macros.

Keywords: macros, automation, administrative tasks, Microsoft Excel, VBA, Apache OpenOffice, Basic

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu mé bakalářské práce Ing. Michalu Plevovi za cenné rady, odborné vedení, připomínky a vstřícnost při zpracování práce.

Dále bych chtěl poděkovat svým přátelům a celé rodině za podporu a trpělivost; a především své sestře, která výsledek mé práce aplikovala ve svém zaměstnání.

OBSAH

ÚVOD

Současná doba je stále více náročná na časovou vytíženost v pracovním prostředí a je vyvíjen neustálý tlak na zvyšování produktivity. Proto je důležité práci zrychlovat, automatizovat a usnadňovat, aby lidé nemuseli opakující se procesy vykonávat manuálně.

Bakalářská práce je zaměřena na zautomatizování administrativních činností pomocí maker. Pro tvorbu těchto maker je zvoleno prostředí aplikace MS Excel. A to z toho důvodu, že je to jeden z nejrozšířenějších programů, který je využíván ve velkých i malých firmách, v nejrůznějších organizacích a veřejných institucích, ale i pro soukromou potřebu. Jeho znalost je vyžadována při téměř všech výběrových řízeních pro ekonomické a technické profese a zkrátka patří k základní počítačové gramotnosti. Ovšem v praxi si zvládá jen malá část uživatelů své každodenní úkony zjednodušit a zautomatizovat. Uživatelům plnících dílčí administrativní úkony v prostředí MS Excel by zefektivnilo a zrychlilo jejich práci, kdyby získávali lepší povědomí o rozšířených možnostech této aplikace. K tomu patří například možnosti vestavěného makrojazyka. Mezi těmito makrojazyky významné místo zaujímají jazyky vycházející z Visual Basicu. Je tomu tak u Microsoft Office a Apache OpenOffice.

Teoretická část této práce je rozdělena na možnosti maker v MS Excel a poté v open-source programu Apache OpenOffice. První kapitola vysvětluje, kdy se makra dají použít, jak se vytvoří pomocí záznamu a programovacího jazyka Visual Basic. Dále je popsán způsob uložení a spouštění maker. Studie řeší rovněž možnosti přerušení běhu makra a jeho odladění. Práce se věnuje také možnostem zabezpečení a záležitostem s tím souvisejících, které je nutno znát. Ve druhé části je představen kancelářský balík Apache OpenOffice. Pozornost je soustředěna na zjištění, zda makra vytvořená v odlišných aplikacích jsou kompatibilní a proběhne srovnání jednotlivých programů.

V praktické části jsou vypracovány příklady maker na často se opakující činnosti při administrativní práci v MS Excel. Jedná se o úkony spojené s formátováním buněk, manipulací se sešity a listy, práci s daty, odeslání dat e-mailem, či možnost spolupráce s dalšími aplikacemi balíčku Microsoft Office. V celé kapitole jsou ukázkové kódy napsané v jazyce VBA spolu s popsanými kroky. Cílem práce je návrh konkrétního automatizovaného řešení a ověření jeho využitelnosti v praxi. V závěru proběhne vyhodnocení časové úspory při používání maker a na základě grafů bude ukázáno, zda se pro potřeby administrativních činností vyplatí makro vytvářet či nikoliv.

I. TEORETICKÁ ČÁST

1 MAKRA V MS EXCEL

Tabulkové procesory představují jednu z velmi významných oblastí současných kancelářských systémů. V tomto prostředí lze vytvářet velmi zajímavé uživatelské aplikace z oblasti statistiky, financí, účetnictví, či informatiky. Pro tyto činnosti lze vhodně navrhnout automatizované řešení pomocí maker. [1] Makra v MS Excel slouží k zaznamenání skupiny příkazů, aby se poté mohla kdykoli spustit vyvoláním určitého makra nebo pomocí klávesové zkratky. Umožňuje to často opakované činnosti zautomatizovat, tím zrychlit práci v Excelu a snížit počet případných chyb při zpracování dat.

Všechny kroky makra se zaznamenávají v kódu Visual Basic for Applications (dále VBA). VBA je objektově orientovaný programovací jazyk, který je součástí Excelu. Díky VBA lze práci v Excelu plně zautomatizovat, ale na rozdíl od zaznamenávání maker je značně složitější a vyžaduje jisté programátorské schopnosti. [2]

1.1 Použití maker

Případů, kdy je vhodné využívat makra, je mnoho. Mezi ty nejdůležitější patří:

1. **Automatizace často vykonávaných úkonů** – jsou to ty nejjednodušší, ale také nejčastěji konané operace, jako např.:
   - Otevření sešitu, vytištění jeho části a uzavření. Otevření více sešitů zároveň, sloučení jejich obsahu do nového a uložení sešitu.
   - Zpracování více různých sešitů totožným způsobem.
   - Opakované vkládání delších textů (adresy, seznam telefonních čísel).
   - Vytváření grafů.
   - Shodné formátování odlišných oblastí buněk.
   - Tisk různých, přímo daných částí sešitu.
2. **Vytvoření vlastní funkce či příkazu** – vlastní funkcí se dají zjednodušit některé vzorce a vlastním příkazem zkombinovat několik příkazů.
3. **Vytvoření celé nové aplikace**. [3]

1.2 Vytváření maker

Excel umožňuje vytvářet makra dvojím postupem:

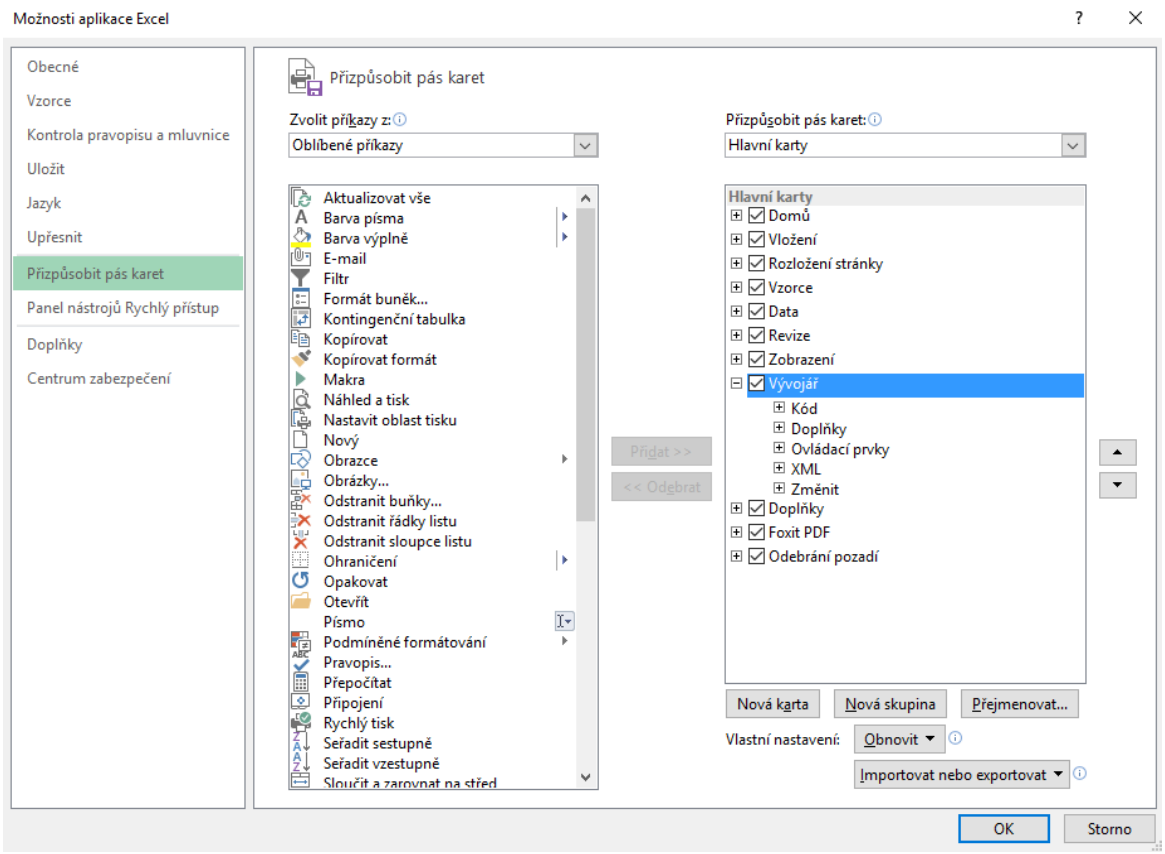
- **Uživatelsky** – makra jsou vytvořena pomocí záznamníku, nahráním posloupností příkazů, které provedeme.
- **Programátorsky** – makra se ručně zapisují v editoru jazyka VBA. Je nutné znát uvedený programovací jazyk.

Lze kombinovat i oba způsoby – makro zaznamenat uživatelsky a upravit jej přímým zápisem do kódu. Tento způsob je vhodný především pro začátečníky, kteří se učí orientovat v příkazech VBA. [4]

1.2.1 Záznam makra

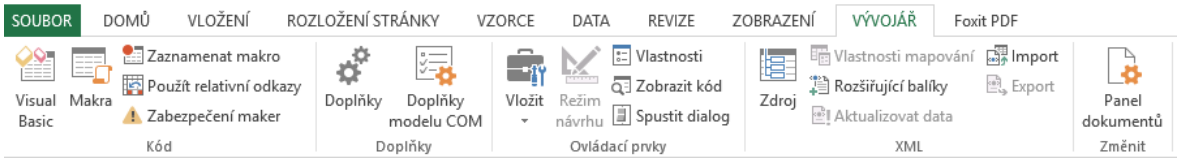
Při záznamu makra je důležité vědět, že se zachytí všechny akce, které provedeme. Takže pokud se provede nějaká chyba v posloupnosti (např. klikneme na tlačítko, na které jsme nechtěli), zaznamená se. Řešením je celý postup opakovat nebo upravit samotný VBA kód. Proto je nejlepší používat nahrávání makra v procesech, které jsou nám dobře známy a vždy si předem naplánovat kroky a příkazy, pak teprve začít se záznamem. Čím jednodušeji bude makro zaznamenáno, tím efektivnější bude, když se spustí. [5]

Pro úpravu maker a používání ovládacích prvků je nutné nejdříve zobrazit na pásu karet další kartu zvanou **Vývojář**. Postup, jak toho docílit, je kliknout na kartu **Soubor**, vpravo dole zvolit příkaz **Možnosti**, v levé části okna klepnout na příkaz **Přizpůsobit pás karet**, v pravém seznamu označite volbu **Vývojář** a potvrdit **OK**. Dialogové okno pro přízpůsobení pásu karet s povolenou kartou **Vývojáře** je zobrazeno na obr. 1.



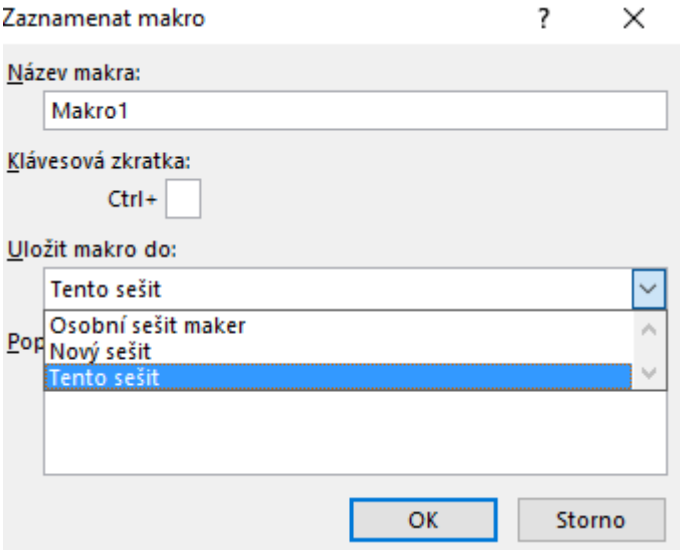
Obrázek 1: Přizpůsobení pásů karet v MS Excel 2013 *(zdroj: Vlastní tvorba)*

Pro práci s makry se využívájí tlačítka v části **Kód** na této kartě **Vývojář**, která je vyobrazena na následujícím obr. 2. Na kartě **Zobrazení** je k dispozici také tlačítko **Makra** umožňující záznam a spuštění makra, nicméně práce s kartou **Vývojář** je přehlednější a obsahuje více možností.



Obrázek 2: Karta Vývojář v MS Excel *(zdroj: Vlastní tvorba)*

Nahrávání makra se spustí tlačítkem **Zaznamenat makro** v nabídce na kartě **Vývojář** nebo ve stavovém řádku. Zobrazí se dialogové okno (viz obr. 3), ve kterém vyplníme **Název** makra. Systém automaticky nabízí názvy maker Makro1, Makro2, atd. Toto pojmenování se nedoporučuje, protože při větším množství maker je pak orientace v nich velmí obtížná. Je vhodné volit název tak, aby co nejlépe vypovídal o obsahu makra. Ke tvorbě názvu lze používat pouze písmena, číslice nebo podtržítko. Jiné znaky (mezera, čárka, pomlčka) nejsou přípustné. [6]



Obrázek 3: Dialogové okno Záznam makra v MS Excel *(zdroj: Vlastní tvorba)*

Pole **Klávesová zkratka** se využívá velmi málo (např. pro jedno makro, které spustí základní menu aplikace – při početnějším množství maker by nebylo možné si všechny zkratky pamatovat). Pole **Uložit makro do** definuje tzv. projekt a umožňuje uložit makro do:

- sešitu vlastní aplikace (Tento sešit) – makro se ukládá do stejného sešitu, ve kterém byly zaznamenávané akce provedeny;
- jiného nového sešitu – založí se nový prázdný sešit, do kterého se makro uloží;
- osobního sešitu maker – založí se speciální sešit s názvem Personal.xlsb, do kterého se vytvářené makro uloží, je umístěn ve složce XLStart a při každém dalším spuštění Excelu se automaticky otevře.

Do pole **Popis** je vhodné napsat poznámky týkající se obsahu makra (tento text je součástí komentářové části makra a lze jej poté ve VBA upravovat).

Potvrzením tlačítka **OK** se spustí záznam makra a lze provádět jednotlivé kroky, které má makro obsahovat. Současně se změní obě tlačítka **Zaznamenat makro** (na kartě Vývojář i ve stavovém řádku) na **Zastavit záznam,** čímž se záznamový režim ukončí. Při nahrávání makra se používají **absolutní odkazy**. Když v rámci nahrávání makra nastavíme kurzor do buňky A1, při spuštění makra se skutečně vybere buňka A1. [6]

Jenže někdy je třeba, aby makro vybralo buňky relativně od výchozí polohy aktivní buňky. Příkladem jsou makra pracující s dynamickými tabulkami (tabulky, jejichž rozsah se v průběhu aplikace měni – přidávají se nové řádky). Toho docílíme stistknutím tlačítka

Použít relativní odkazy buď na začátku záznamu nebo v jeho průběhu. Znamená to, že nebude adresovat posun kurzoru pomocí konkrétních adres, ale pomocí pohybu od aktuální pozice kurzoru o příslušný počet buněk v daném směru. Když bude kurzor před použitím relativních odkazů v buňce A1 a nahrajeme posun do A3, v kódu se zaznamená absolutní adresa A3, ale pohyb o 2 řádky směrem dolů. [6]

Výhody nahrávaných maker

- Je to nejrychlejší způsob vytvoření makra.
- Nahrávání je nezastupitelné během všech fází poznávání dílčích objektů Excelu.

Nevýhody nahrávaných maker

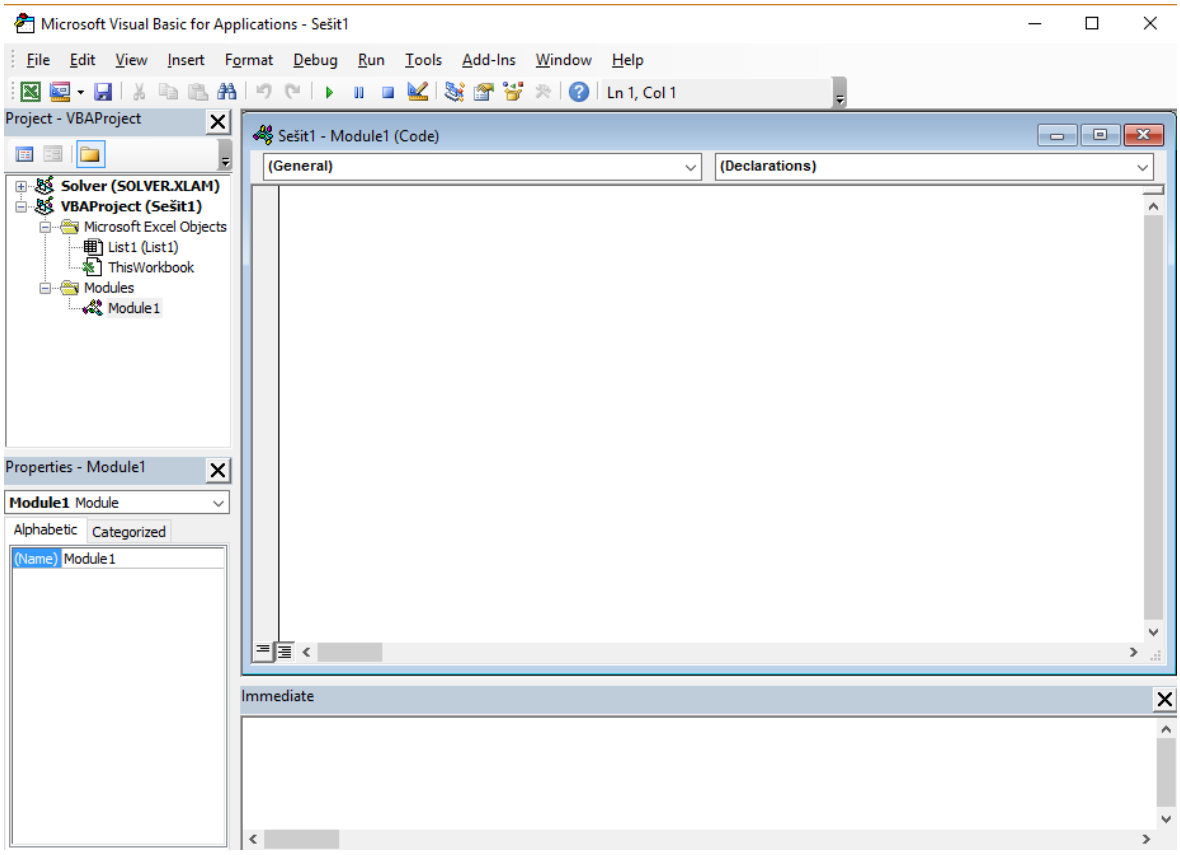
- Záznamník vždy jednotlivé objekty (buňky, listy, grafy) vybírá, v nahraném makru tedy vždy najdeme příkazy pro výběr – pokud si ovšem požadovaný objekt nevybereme sami ještě před spuštěním záznamníku. Výběr objektů však ve skutečnosti není nutný a prodlužuje dobu provádění makra.
- Výběr objektů má ještě následující důsledky: buď si příslušný objekt (buňky, graf) označíme ještě před začátkem nahrávání makra – v takovém případě pak bude makro vždy zpracovávat aktuální výběr – nebo budeme jednotlivé objekty vybírat až při vlastním nahrávání – pak ovšem bude nahrané makro pracovat vždy a právě jen s těmito objekty.
- Nahrané makro vždy zaznamená jen přesně daný sled činností, které jsme prováděli při jeho nahrazování. Není možné nahrát například výběr akce na základě nějaké podmínky, není možné žádnou akci na základě určité podmínky vynechat.[3]

1.2.2 Vytváření v editoru VBA

Veškerá práce s kódem VBA probíhá v editoru jazyka Visual Basic, který je zobrazen na obr. 4. Je to samostatná aplikace těsně provázaná s Excelem. Editor jazyka Visual Basic vyvoláme nejrychleji pomocí <ALT+F11> nebo tlačítkem **Visual Basic** na kartě **Vývojář**. Okno editoru je rozděleno na šest základních částí:

- **Panel nabídek** – najdeme na něm příkazy pro práci s různými součástmi VBE, které jsou k dispozici.
- **Panely nástrojů** – je umístěn přímo pod pruhem nabídek a můžeme si jej dle sebe přizpůsobit.

- **Okno průzkumníka projektu (Project Explorer)** - zobrazuje stromový diagram se všemi otevřenými sešity. Sešit je označen jako „Project“. Pokud tohle okno nevidíme, aktivuje se buď v nabídce **View** > **Project Explorer** nebo stiskem klávesových zkratek Ctrl+R.
- **Okno kódu (Code)** - obsahuje samotný kód VBA. Každá položka ve stromu projektu má přiřazeno vlastní okno kódu. Zobrazit lze v nabídce **View** > **Code** nebo stiskem F7.
- **Okno vlastností (Properties Window)** – obsahuje seznam všech vlastností daného objektu. Vlastnosti v něm můžeme také měnit. Opět jej můžeme aktivovat ve volbě **View** > **Properties Window** nebo stiskem F4.
- **Okno Immediate** – využívá se k přímému provádění příkazů VBA, jejich testování a také upravení napsaného kódu. Okno opět najdeme ve volbě **View** > **Immediate Window** nebo má přiřazenou klávesou zkratku Ctrl+G. [2]

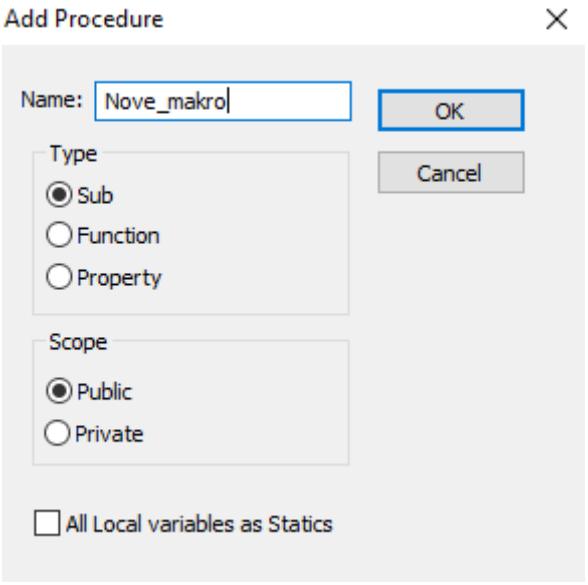


Obrázek 4: Okno editoru VBA *(zdroj: Vlastní tvorba)*

Každý otevřený sešit je v editoru považován za projekt. Jestli již obsahuje nějaké moduly, lze je vidět ve výpisu projektu s názvem **Modules**.V tomto okně se mezi projekty a moduly přepíná, a tím se mění i okno s vlastnostmi. Ve vlastnostech je možno poté projekt nebo modul přejmenovat. [2]

Nový modul se přidá zvolením **Insert > Module**. Klepneme do podokna **Properties** a vlastnost *Name* si přepíšeme na vhodné pojmenování. V podokně modulu se dá rovnou psát kód nebo vytvořit makro příkazem **Insert** > **Procedure**.V zobrazeném okně zůstává volba **Sub**, vyplní se název makra (pro název platí stejná pravidla jako u zaznamenávání makra) a potvrdit **OK** (viz obr. 5). [4] Tím se vytvoří základní struktura kódu:

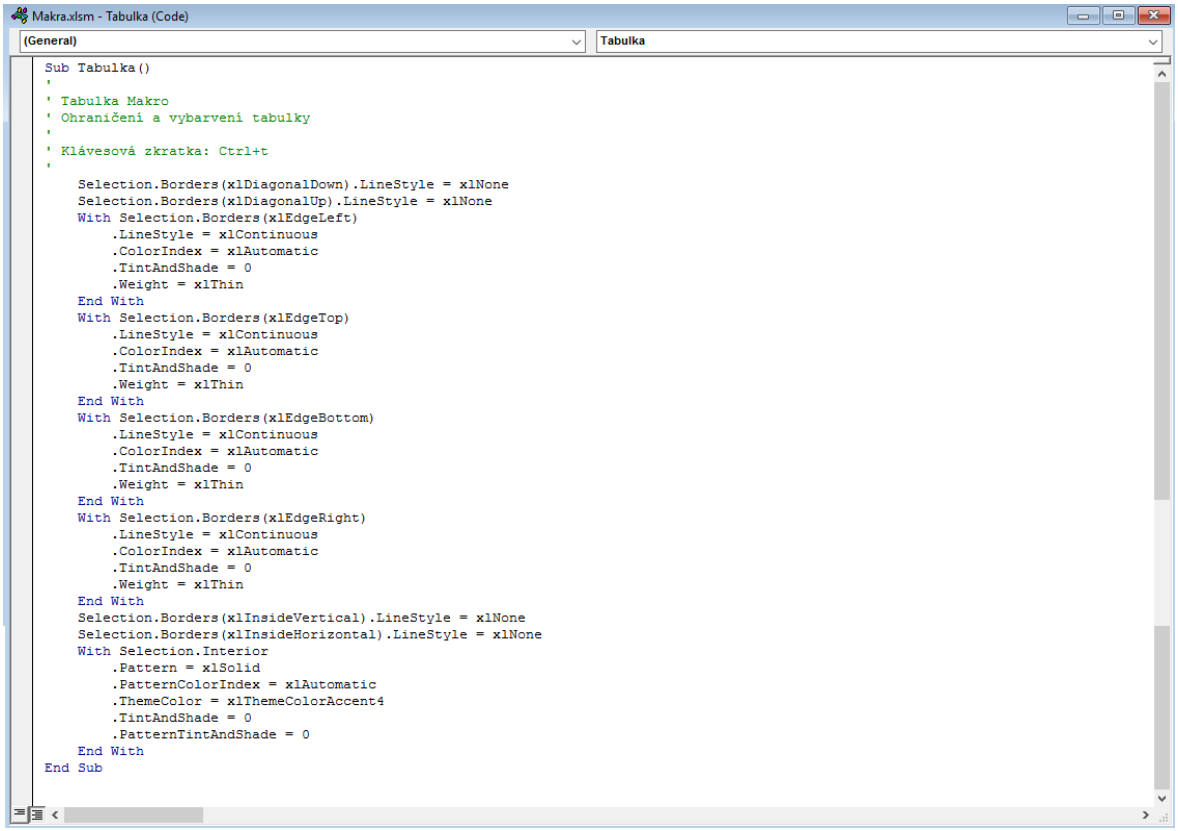
```
Public Sub nazev()
End Sub
```



Obrázek 5: Dialog pro vytvoření makra ve VBE *(zdroj: Vlastní tvorba)*

Každé makro má několik částí. Začíná hlavičkou – slovem Sub a názvem makra, za kterým následují závorky (v nich může být v případě potřeby uveden parametr, se kterým bude procedura pracovat). Další část je identifikační (komentářová), je uvozena apostrofem a navíc se zobrazuje světle zelenou barvou. Umožňuje lepší orientaci ve struktuře makra a může také sloužit k momentálně nevyužívané části kódu. Komentář (poznámku) se dá vložit do kódu i uvedením slova Rem na začátku řádku (zejména kvůli kompatibilitě starších zdrojových kódu), nebo uvedením apostrofu uprostřed řádku (od této pozice je pak text považován za poznámku, nevykonává se a překladačem je ignorován).

Další poznámkové řádky obsahují název makra a příp. klávesovou zkratku pro jeho spuštění. Vlastní tělo makra obsahuje kód, který je v černé barvě bez apostrofů a končí řádkem End Sub. Pokud se zapisujeme text makra přímo ve VBA a vloží se hlavička se jménem makra, VBA automaticky vloží koncový řádek procedury End Sub. [6]



Obrázek 6: Ukázka procedury ve VBE *(zdroj: Vlastní tvorba)*

Vlastní kód je rozdělen do jednotlivých řádků, které obsahují příkazy. Může obsahovat:

- klíčová slova (jsou zobrazena modrou barvou např. If, Then, End If, apod.),
- proměnné (pojmenovaná místa v paměti, která mohou obsahovat data, která se při běhu programu mohou měnit) – např. i, jmeno, klic apod.
- konstanty (např. 10, “ZISK“ apod.)
- operátory (+, -, *, atd.)
- výrazy (např. a=6).

Příkazy se dají rozdělit na

- příkazy deklarace, pomocí nichž se pojmenují a definují proměnné, konstanty, atd.
- vlastní proveditelné příkazy.

Řádky kódu mohou být fyzické nebo logické. Fyzický řádek se ukončuje klávesou <ENTER>, logický řádek (vlastně pokračování řádku) vznikne spojením několika fyzickým řádků pomocí znaku mezera a podtržítko (_). [6]

Příkazy je vhodné zapisovat vždy na nový řádek kvůli přehlednosti. Zapisuje-li se několik kratších příkazů na stejný řádek, musí se používat oddělovač dvojtečka (:). Makro je

vhodné graficky upravit odsazováním nebo předsazováním textu tabulátorem. Přehlednější makro se lépe upravuje a udržuje.

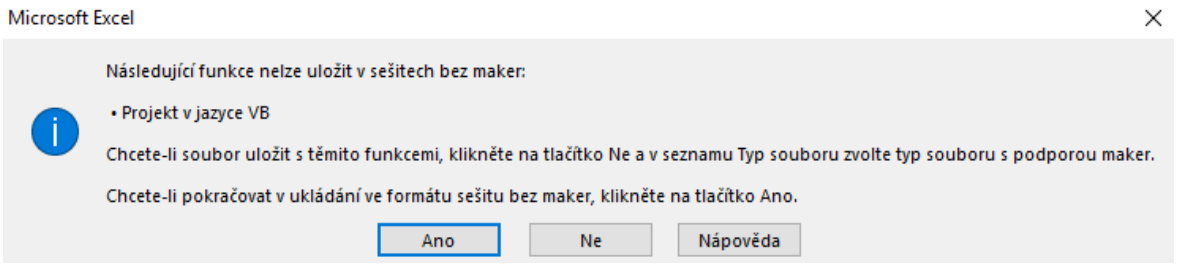
Při zápisu vlastního makra ve VBA dochází po přechodu na další řadek ihned ke **kontrole syntaxe** (správnost zapsaného kódu). Pokud je nalezena chyba, zobrazí se okno chybové hlášky a řádek je červeně označen. Chyba se musí odstranit, aby se dalo pokračovat v dalším zápisu kódu. Je-li syntaxe správná, kód se ihned překládá do vnitřní spustitelné formy. [6]

1.3 Uložení maker

Sešit, který obsahuje makra, se musí uložit s povolenými makry. Soubor dostane příponu XLSM. Tento typ souboru není výchozí formát, a proto se musí ukládaná přípona změnit manuálně. Postup je následující:

1. Na kartě **Soubor** klepnout na položku **Uložit jako**.
2. V dialogu **Uložit jako** otevřít rozevírací seznam **Uložit jako typ**.
3. V rozevíracím seznamu **Uložit jako typ** zvolit **Sešit Excelu s podporou maker**.

Pokud by se sešit ukládal ve výchozím formátu XLSX, vyskočí varování upozorňující na to, že funkce obsahující sešit nelze uložit v sešitech bez maker, jako je to zobrazeno na obr. 7.



Obrázek 7: Dialog MS Excel s varováním *(zdroj: Vlastní tvorba)*

Volbou **Ano** Excel všechny makra ze sešitu smaže, ale zůstávají v kopii, se kterou se pracuje. [2]

1.4 Spuštění makra

Makra lze spouštět několika způsoby, z nichž některé jsou využitelné spíše při vývoji aplikace.

- Spuštění makra klávesovou zkratkou.

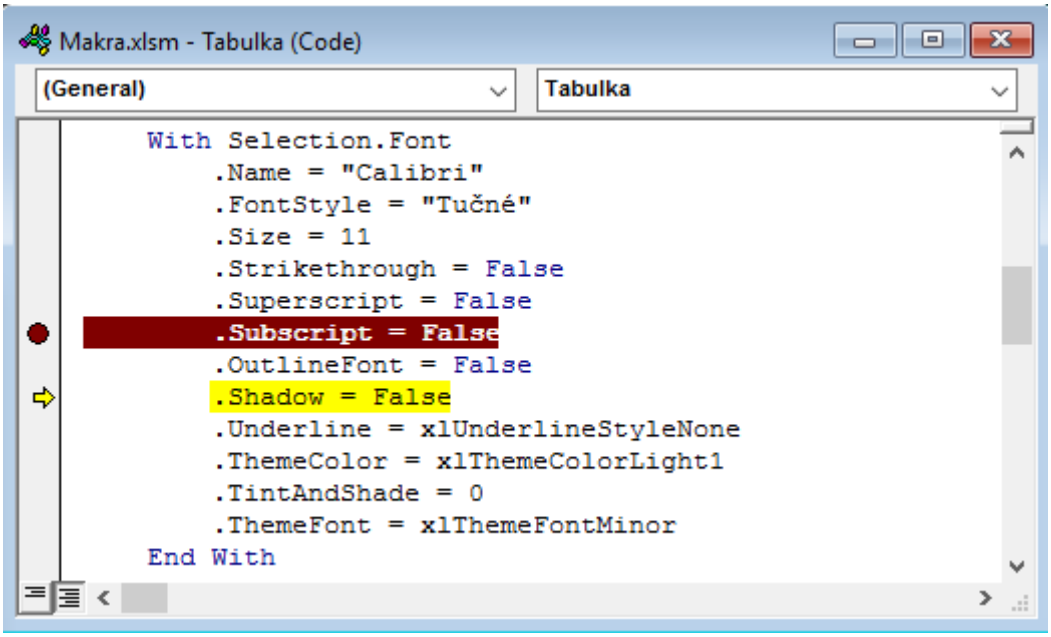
- Lze využít tlačítka **Makra** z karty **Vývojář** (nebo tlačítko **Makra** z karty **Zobrazení** a volbu Zobrazit makra), které zobrazí dialogové okno se seznamem dostupných maker a po výběru názvu makra pomocí tlačítka **Spustit**. Tento způsob je vhodný pouze při tvorbě ověřování správnosti fungování maker, v hotových aplikacích se už dále nepoužívá.
- Podobně lze spustit makro i z editoru jazyka VBA:
  - V nabídce **TOOLS** pomocí příkazu ***Macros*** tlačítkem ***Run*** nebo
  - Z nabídky **RUN** příkazem ***Run Sub/Userform***,
  - Pomocí funkční klávesy **<F5>** (spustí se makro, ve kterém stojí kurzor),
  - Využitím tlačítka ***Run Sub/Userform*** z panelu nástrojů.
- Makro lze přiřadit i nakreslenému objektu nebo ke grafickému ovládacímu prvku na daném listu. Makro se pak spustí automaticky po klepnutí na nakreslený objekt (příp. vložený obrázek, automatický tvar, apod.) nebo změnou ovládacího prvku – například zaškrtnutím políčka nebo klepnutím na položku seznamu. Grafické objekty lze seskupovat do dialogových oken, a vytvářet tak přehledná menu. Tento způsob je optimální.
- Lze vytvářet i vlastní pásy karet či vlastní tlačítka. [6]

1.5 Přerušení běhu makra, ladění

K přerušení běhu makra může dojít z několika důvodů:

- Při běhu programu došlo k chybě.
- Makro přerušil uživatel (klávesou Ctrl+Break nebo příkazem **Run** > **Break**). Toto řešení je nutné v případech, že se makro zacyklí (pořád pracuje a bez našeho zásahu by svou činnosti nikdy neukončilo)
- Je nastaveno **krokování** (volba **Debug** > **Step into** nebo klávesa F8), jedná se o sledování průběhu makra po jednotlivých příkazech. Pro další příkaz se pokračuje opět stejnou volbou. Sledovaný příkaz je v kódu zvýrazněn žlutě.
- Vložena byla **zarážka** nebo napsán příkaz `Stop` do zdrojového kódu, který makro zastaví na aktuální pozici. Zarážka se nastaví pomocí volby **Debug** > **Toggle Breakpoin**, klávesou F9 nebo klepnutím myší do šedého pruhu levé části okna kódu a řádek se označí červeně.

Na následujících obr. 8 je v kódu vyobrazeno krokování (žlutý řádek) a vložení zarážky (červený řádek).



Obrázek 8: Přerušení krokováním a zarážkou ve VBA *(zdroj: Vlastní tvorba)*

Nejčastější příčinou přerušení bude zřejmě chyba při běhu programu. Zobrazí se chybové hlášení s identifikací chyby a nabídkou tlačítek pro řešení. Kliknutím na tlačítko **Debug** se vyvolá editor VBA s oknem kódu a chybný řádek bude viditelně označen žlutou barvou. Opraví-li se chyba, je možno tlačítkem **Continue** (F5) na panelu nástrojů makro dokončit. Ne vždy je však možné pokračovat, protože nebyly splněny výchozí podmínky pro běh makra a je lepší makro ukončit. Lze to udělat tlačítkem **End** v okně chybového hlášení, nebo tlačítkem **Reset** v panelu nástrojů VBE. Pokud by se makro neukončilo, není možné provádět opravy. [6]

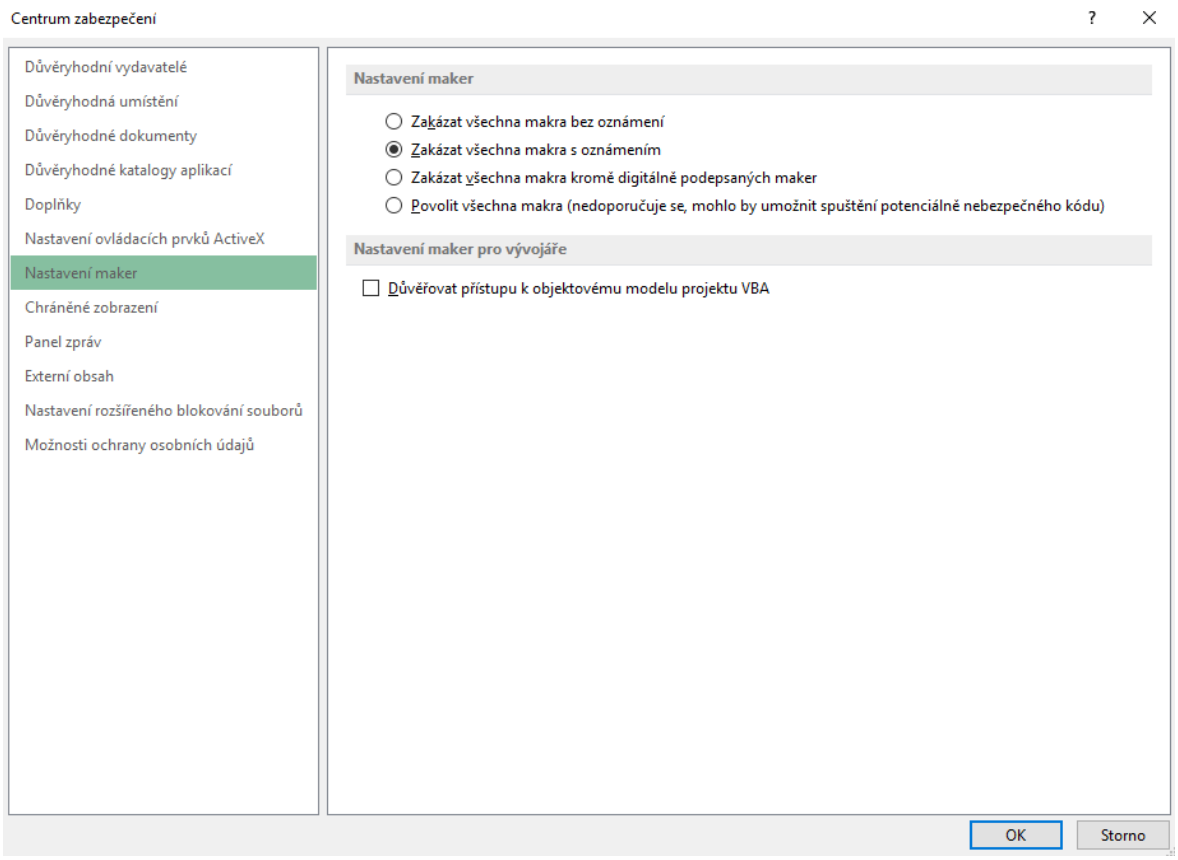
1.6 Bezpečnost maker

Makra jsou schopna poškodit náš počítač, proto byly vyvinuty bezpečnostní vlastnosti maker, aby se dalo problémům předcházet.Výchozím nastavení Excelu je zabezpečení **Zakázat všechna makra s oznámením.** Pokud se otevře sešit obsahující makra, Excel zobrazí nad řádkem vzorců upozornění **Makra jsou zakázána**. Pro možnost pracovat s makry se musí klepnout na tlačítko **Povolit obsah**.

Tohle nastavení zabezpečení lze změnit následujícím způsobem:

1. Na kartě **Soubor** kliknout na **Možnosti**.
2. V dialogu **Možnosti aplikace Excel** zvolit **Centrum zabezpečení** a klepnout na tlačítko **Nastavení centra zabezpečení**.

3. Zde v **Nastavení maker** vybrat požadované zabezpečení (viz obr. 9). [2]



Obrázek 9: Dialog Centrum zabezpečení v MS Excel *(zdroj: Vlastní tvorba)*

Lze vybrat z možností:

- **Zakázat všechna makra bez oznámení** – tato možnost, zakáže všechna makra a všechny výstrahy zabezpečení týkající se maker. Spouštět se budou pouze makra uložená v důvěryhodném umístění.
- **Zakázat všechna makra s oznámením** – jak již bylo zmíněno výše, je to výchozí nastavení MS Excel. Makra budou zakázána, ale při jejich výskytu uživatel dostane výstrahu a sám si zvolí, zda makro povolí. Je to ideální volba zabezpečení.
- **Zakázat všechna makra kromě digitálně podepsaných maker** – je to podobné nastavení jako možnost **Zakázat všechna makra s upozorněním** s rozdílem, kdy je makro digitálně podepsáno důvěryhodným vydavatelem. Pokud je vydavatel zařazen mezi důvěryhodnými, makro se spustí. Jestli v důvěryhodných zařazen není, ukáže se upozornění, zda se má povolit. Jakákoliv nepodepsaná makra jsou zakázána bez upozornění.

- **Povolit všechna makra (nedoporučuje se, mohlo by umožnit spuštění potenciálně nebezpečného kódu)** – povolení všech maker ze všech zdrojů. Jak je již v názvu volby, projde každý potenciálně škodlivý kód a počítač se stává zranitelným.
- **Důvěřovat přístupu k objektovému modelu projektu VBA** – nastavení sloužící vývojářům. Způsobuje, aby běžící kód mohl přistoupit k objektovému modelu a vytvořit kód nový. To je ale potřeba jen zřídka. [5]

1.6.1 Makroviry

Makra mohou také představovat určitou hrozbu pro náš počítač. Mohou v sobě obsahovat škodlivé kódy, které například způsobí stahování škodlivého softwaru. Tzv. makrovirus může z počítače vykrádat důvěrné informace, pracovat se soubory nebo spouštět aplikace. Opatrný uživatel sice může omezit množství spustitelných souborů, které si kopíruje na počítač, ale výměně elektronických dokumentů se nevyhne.

Makrovirusy se objevily s příchodem Windows 95 spolu s kancelářským balíkem Office. Již v této verzi byla umožněna schopnost vkládat do dokumentů makra se silným jazykem Visual Basic. Makra mohla skoro cokoliv, např. formátovat disky nebo rozesílat e-maily. První makrovirus přišel v prosinci v roce 1994, měl název DMV (Document Macro Virus) a šlo pouze o demonstrační kódy pro Word a Excel, které měly sloužit jako varování. V roce 1995 následoval už skutečný makrovirus pro Microsoft Word s názvem Concept. Jeho autor měl neuvěřitelnou znalost prostředí maker a spekuluje se, že šlo o některého ze zaměstnanců Microsoftu, který se podílel přímo na vývoji. Jelikož na příchod makrovirů nebyli připravení uživatelé ani antivirová ochrana, stal se z něj jeden z historicky nejrozšířenějších virů. [11] Nebyl však nijak destruktivní. V makru se nacházela pouze následující zpráva:

```
Sub MAIN
     REM That's enough to prove my point
End SUB
```
[12]

Dalším známým makrovirem pro MS Word je Melissa, též známý jako Simpsons a Kwyjibo, který se začal šířit roku 1999. Virus se velice rychle šířil, a proto si ho oblíbila média, která z něj učinila velice nebezpečný virus. Nezpůsoboval ale nijak závratné škody. Vir je zahrnutý v makru a šíří se pomocí e-mailu, v předmětu zprávy je "Important Message From <emailová adresa účtu, z kterého byl virus odeslán>". Odesílatel bude skutečná adresa, odkud pochází a obsah zprávy je "Here is that document you asked for ...

don't show anyone else ;-)". V příloze se poté nachází nakažený soubor pojmenovaný LIST.DOC, který obsahuje seznam adres 80 pornografických stránek. Virus se následně rozešle na prvních 50 v adresáři uložených e-mailových adres. Infikuje také šablonu NORMAL.DOT a všechny nově uložené dokumenty jsou automaticky nakaženy. Odhadovaná škoda přesáhla miliardu dolarů především ve formě ušlého zisku, kdy firmy musely vypnout servery, aby se zabránilo šíření. [12]

Obliba těchto makrovirů vzrůstala až do roku 2001 díky různým novým principům pro jejich tvorbu a šíření. Ke smrtícímu úderu na makroviry došlo s uvedením MS Office 2007, kdy jsou již všechna makra ve výchozím nastavení zakázána a spuštění každého makra se musí povolit. Ovšem i v nynějších verzích se stále objevují. Tvůrci těchto virů využívají své kreativity k tomu, aby uživatele donutili makra povolit. Principem je vnutit uživateli myšlenku, že při povolení maker získají přístup k dalšímu či podrobnějšímu obsahu. Typickým příkladem je dokument obsahující rozmazaný obsah. Rozostření je zvoleno tak, aby uživatele možný obsah zaujal, a že k čitelnému obsahu se dostane po povolení maker. V dokumentech se také objevuje velká šipka, která ukazuje, kde makra povolit. [13]

2 MAKRA V AOO CALC

Apache OpenOffice (zkratka AOO) je volně dostupný kancelářský balík pod licencí Apache. Předchůdcem byl StarOffice, který se vyvíjel od roku 1994. AOO obsahuje textový editor Writer, tabulkový procesor Calc, prezenční program Impress, nástroj pro vektorovou grafiku Draw, databázi Base a editor matematických vzorců Math. Dokáže číst a zapisovat většinu formátů používaných v Microsoft Office. Vhodný je především při využívání většího počtu počítačů (např. ve školách a firmách).

Hlavní výhody AOO:

- možnost legálně používat a sdílet software s minimálními nebo žádnými náklady,
- formátem AOO je XML, který je široce přijímán a dokumenty budou čitelné i v budoucnosti,
- jako open source software bude mít neustálou podporu a údržbu, která není závislá na existenci společnosti,
- provozován může být na operačních systémech Microsoft Windows, macOS, Linux a Solaris

Nevýhodou AOO je absence Groupware (ekvivalent k MS Outlook) a náklady na konverzi existujících souborů Microsoft Office, hlavně pokud obsahují makra. [7]

Zabývat se budu makry v OpenOffice Calc, což je tabulkový kalkulátor a zároveň alternativou k programu MS Excel. Soubor se ukládá v mezinárodním standardu pro dokumenty – Open Document Format (ODF). Je založen na XML a může být přečten, i pokud uživatel nemá AOO.

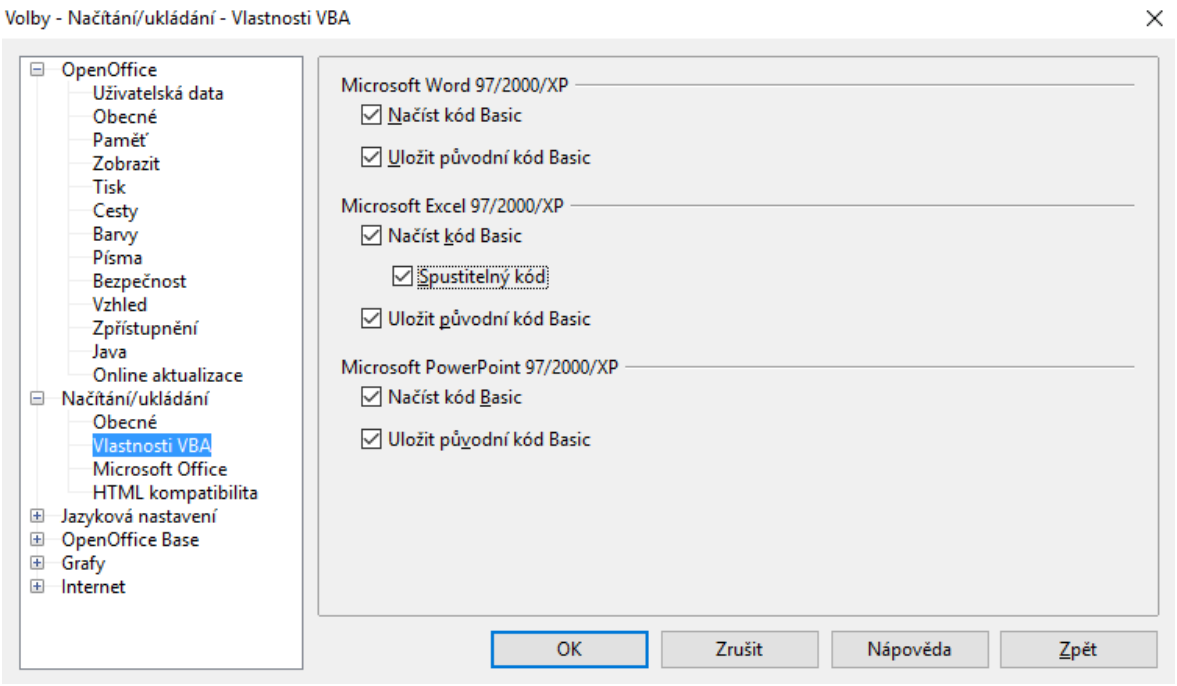
Základní rozdíl mezi makry již začíná v programovacím prostředí, kde se vytváří či upravují. Zatímco MS Excel používá VBA, makra v AOO Calc jsou naprogramovaná v jazyku OpenOffice Basic (také znám jako StarBasic), který je velmi podobný. Oba využívají stejný makrojazyk Basic, ale AOO může být také doplněn jazyky JavaScript, BeanShell a Python. Důvodem, proč VBA nefunguje v Calcu, i když je Calc schopen číst Excelovský sešit, jsou rozdíly v objektech a metodách. [8]

2.1 Použití maker z MS Office

Samotný OpenOffice umí načíst makra uložená v souborech Microsoft Office, které lze dodatečně upravit v OO Basic. Pokud chceme v OpenOffice odstranit nebo zachovat mak-

ra vytvořená ve VBA, otevřeme dokument MS Office a změníme obsah dle potřeby (text, buňky, obrázky), ale do maker se nezasahuje. Uložíme soubor ve formátu MS Office a makra zůstanou zachovaná.

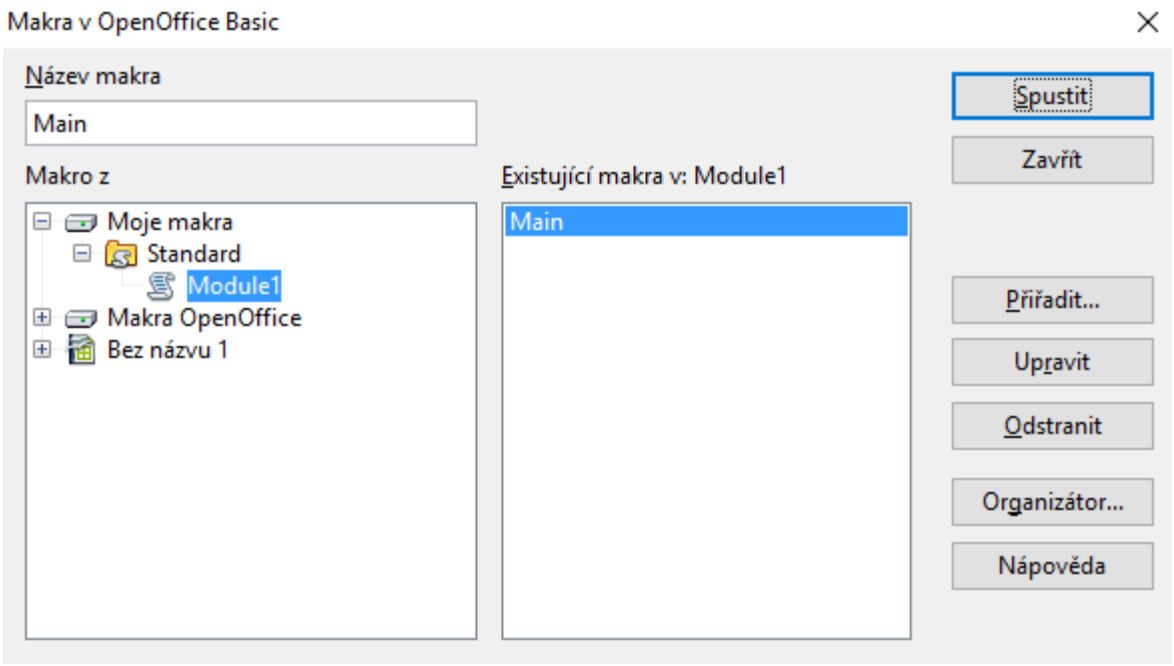
Chceme-li editovat makra vytvořená ve VBA, musíme upravit nastavení vztahující se k otevírání souborů MS Office. Volbu nastavení najdeme v **Nástroje** > **Volby** > **Načítání/Ukládání** > **Vlastnosti VBA**. Je nutné povolit možnost **Spustitelný kód** v části Microsoft Excel 97/2000/XP, která původně není zatržená (viz obr. 10). Kód VBA pak bude načten tak, že je připraven ke spuštění. Kdyby pole nebylo zaškrtnuto, kód VBA bude zakomentován a lze jej pouze prohlížet, nikoliv spouštět. [15]



Obrázek 10: Vlastnosti VBAv OpenOffice *(zdroj: Vlastní tvorba)*

2.2 Záznam makra

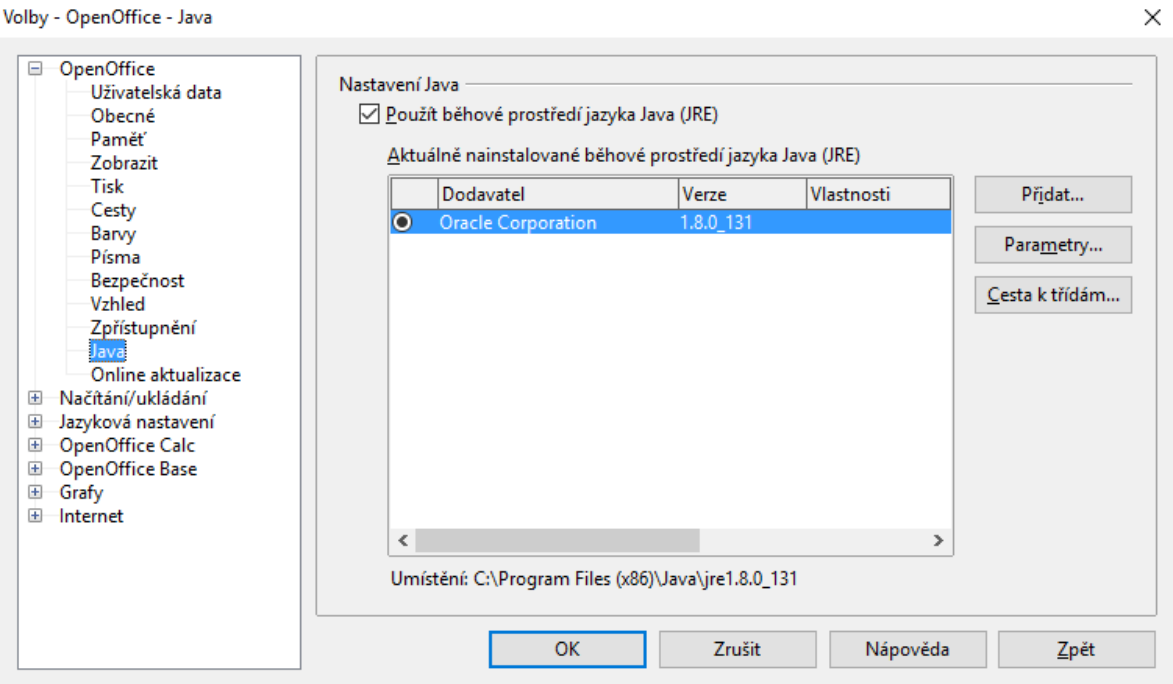
Zaznamenání makra v AOO Calc probíhá podobně jako u MS Excel. Hlavním rozdílem je, že v Calc se určuje název a místo uložení makra až při ukončení nahrávání. Zatímco u Excelu jsou tyto možnosti k vybrání před samotným nahráváním. V AOO jsou moduly uloženy v knihovnách, které se automaticky pojmenovávají Standard. Organizovat je lze přes volbu Nástroje > Makra > Správce maker > OpenOffice Basic a otevře se dialogové okno s knihovnami a moduly (viz obr. 11). Naproti tomu moduly v Excelu jsou obsaženy přímo v sešitu.



Obrázek 11: Dialogové okno Makra v OpenOffice Basic *(zdroj: Vlastní tvorba)*

Makra se ukládají buď do složky Moje makra, Makra OpenOffice nebo přímo do dokumentu. Pro vlastní makra se používají pouze složky Moje makra nebo Dokumentová makra nikoliv složka Makra OpenOffice.

Pro záznam makra musí být nainstalovaný programovací jazyk JAVA (JRE), protože jinak by se některé příkazy neprováděly správně. Povolení se provede přes možnost **Nástroje** > **Volby** > **Java**. Je potřeba zaškrtnout možnost **Použít běhové prostředí jazyka Java** (viz obr. 12).



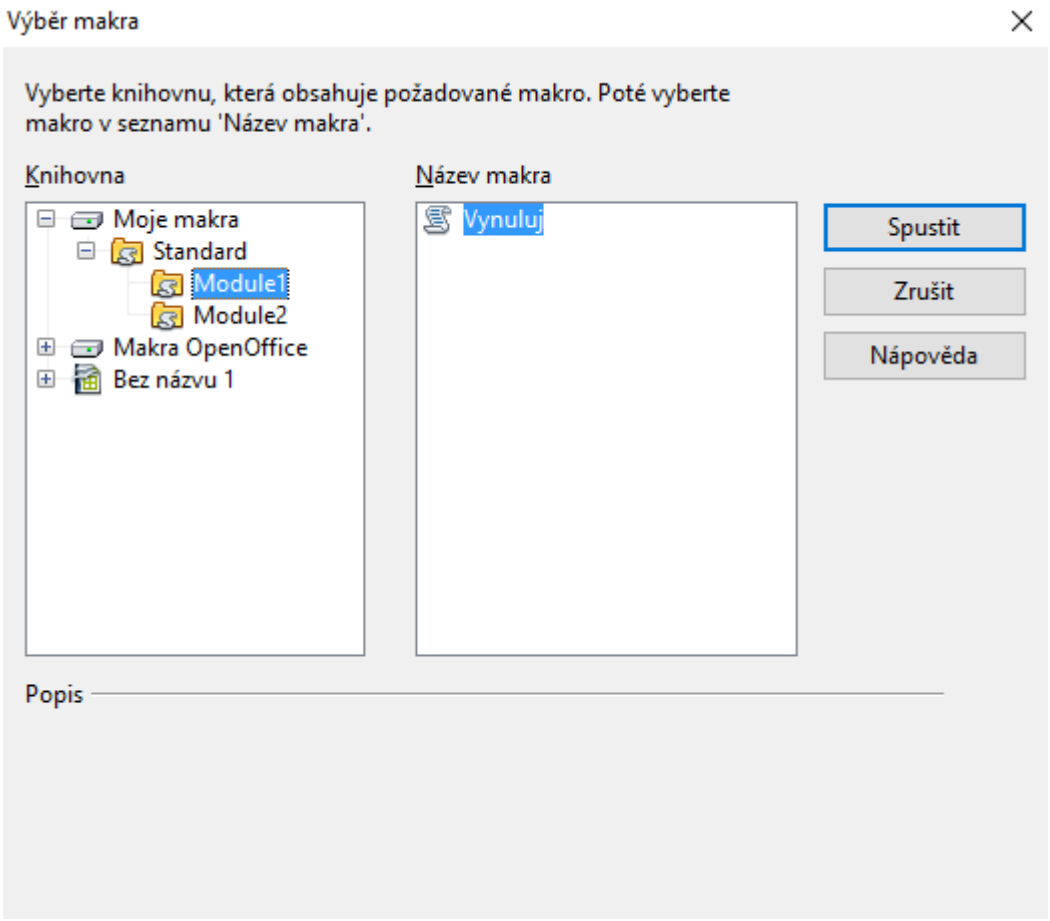
Obrázek 12: Povolení prostředí jazyka Java v OpenOffice *(zdroj: Vlastní tvorba)*

Samotné nahrávání makra se spustí možností **Nástroje** > **Makra** > **Zaznamenat makro**. Podle malého dialogového okna, které vyskočí, lze poznat, že se již makro nahrává. Po provedení všech činností se makro zastaví klikem na **Zastavit nahrávání**. Poté automaticky vyskočí okno Makra v OpenOffice Basic (obr. 11). Chceme-li, aby se makro spouštělo pouze v určitém dokumentu, musí se zvolit jeho umístění do knihovny se jménem, jakým je dokument uložen. Zde si vytvoříme nový modul (standardně pojmenovaný Module1). Pokud naopak chceme, aby makro fungovalo ve všech dokumentech, vybereme knihovnu „Moje makra“, v ní modul „Standard“. Nakonec je možno makro pojmenovat a uložit.

Po vytvoření makra se ve stejném dialogovém okně editují a upravují knihovny, moduly a dialogy přes volbu **Organizátor**. Je možno je **Odstranit**, tlačítkem **Upravit** se otevře editor Basic a možností **Přiřadit** lze spojit makro s určitou událostí. [9]

2.3 Spuštění makra

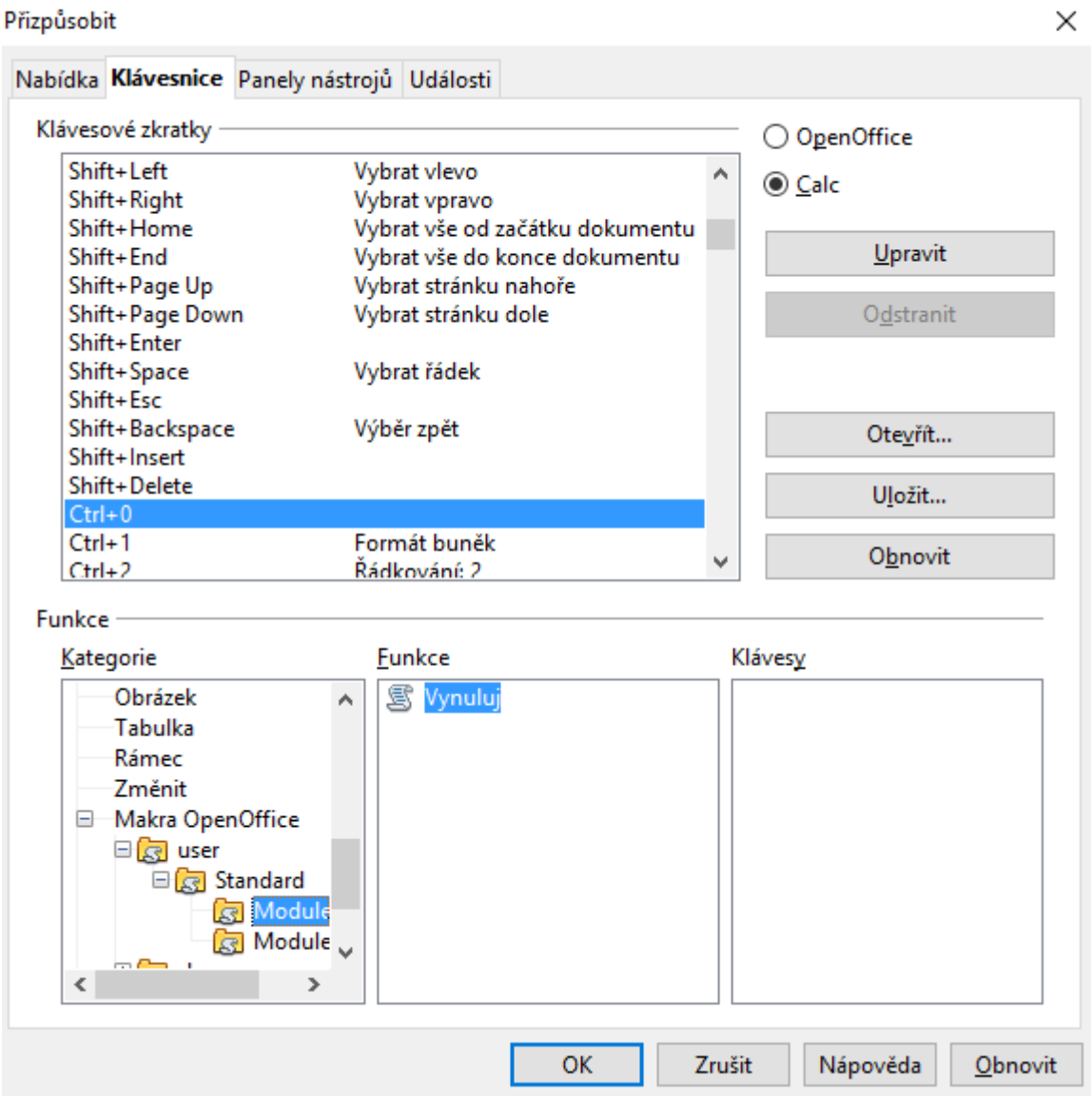
Stejně jako v MS Excelu i zde je několik způsobů, jak vytvořená makra spustit. Tou základní a nejméně pohodlnou metodou je spouštění z menu **Nástroje** > **Makra** > **Spustit makro** a vybere se makro, které je zrovna potřeba (viz obr. 13).



Obrázek 13: Výběr makra pro spuštění v OpenOffice *(zdroj: Vlastní tvorba)*

Rychlejší volbou je možnost přiřadit často používané makro přes nabídku **Nástroje** > **Přizpůsobit**. Na výběr je zde z jednotlivých záložek - **Nabídka**, **Klávesnice**, **Panely nástrojů** a **Události**.

Karta **Klávesnice** slouží k přiřazení klávesové zkratky ke spuštění makra. V Excelu je tato možnost na výběr přímo při vytváření samotného makra, ale v OpenOffice zvlášť spolu v nastavení všech dalších klávesových zkratek programu. V horní části dialogového okna je seznam klávesových zkratek a jim přiřazené akce. Makro je možné přiřadit pouze těm, které nejsou šedé a vhodné není ani přepisovat zkratky obsazené systémem. V dolní části v okně **Kategorie** uživatel musí najít umístění, kde se makro nachází a v okně **Funkce** již zvolí samotné makro. Přiřazení dokončí kliknutím na **Změnit** a potvrzením **OK**. [9] Možnost přiřazení klávesové zkratky je vyobrazena pro lepší orientaci v textu na obr. 14.

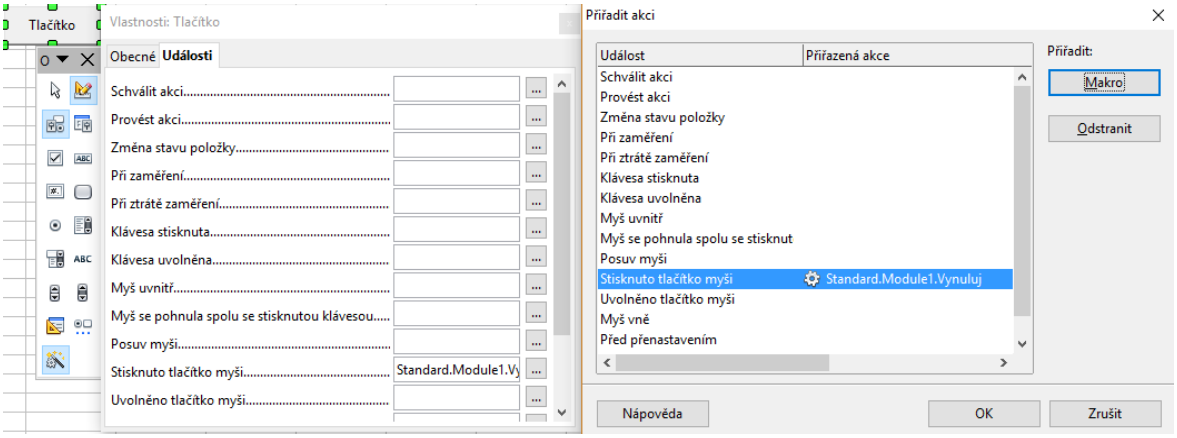


Obrázek 14: Přiřazení klávesové zkratky v OpenOffice *(zdroj: Vlastní tvorba)*

Spojení makra s určitou událostí se provádí na záložce **Události**. Lze si určit, zda se bude makro spouštět vždy pro vybraný dokument, nebo pro všechny dokumenty OpenOffice. [9]

Další možností je spuštění makra přiřazeným tlačítkem. K tomu je nejdříve nutné si zobrazit panel pro ovládací prvky formuláře, to se provádí přes nabídku **Zobrazit** > **Panely nástrojů** > **Ovládací prvky formuláře**. Režim návrhu formuláře musí být pro jeho editaci zapnutý (ikona tužky a pravítka v prvním řádku panelu), ale používat prvky formuláře lze až po vypnutí režimu návrhu. V zapnutém režimu návrhu formuláře klikneme na ikonu **Tlačítko** (ve 4. řádku napravo) a nyní přímo na listu Calcu nakreslíme tlačítko dle své potřeby. Kliknutím pravým tlačítkem myši na „Tlačítko“ vybereme v kontextovém menu položku **Ovládací prvek**. Vyskočí dialogové okno, kde v **Obecné** lze změnit popisek tlačítka, na záložce **Události** je možnost **Stisknuto tlačítko myši.** Klepnutím na „**…**“se ote-

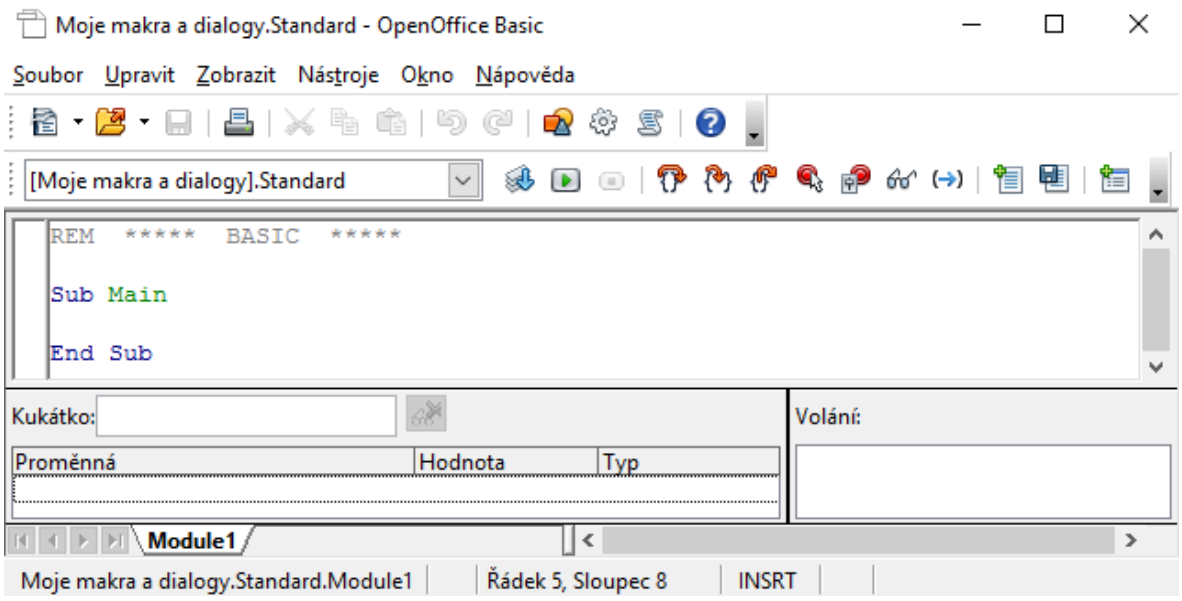
vře dialog **Přiřadit akci** a v pravé části je nutno vybrat **Makro**. Ukáže se další dialog, ve kterém již lze vybrat své makro. Vše se potvrdí tlačítkem **OK** a vypnutím režimu návrhu formuláře je tlačítko připraveno k použití. [14] Na následujícím obrázku 15 je zleva možno vidět „Tlačítko“ a pod ním panel nástrojů, dialog pro vlastnost tlačítka a přiřazení makra k akci.



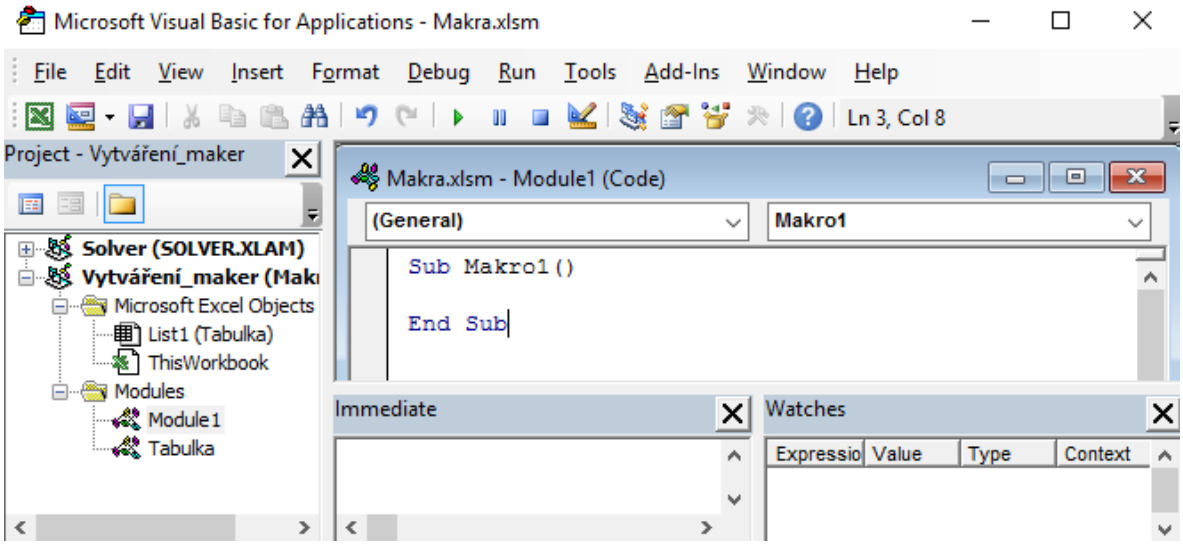
Obrázek 15: Panel nástrojů, vlastnosti tlačítka a přiřazení akce v Calcu *(zdroj: Vlastní tvorba)*

2.4 Editor OpenOffice Basic

Editor OpenOffice Basic se na první pohled od prostředí VBA odlišuje umístěním nástrojů, panelů a ikon. Srovnání je vylíčeno na obrázku 16, kde lze vidět editor OO Basic a obrázku 17, na kterém je editor VBA. V OpenOffice Basic je umístění modulu v podobě záložky ve spodní části pod oknem s proměnnou. Funguje formou překlikávání mezi moduly. Nový modul lze jednoduše přidat kliknutím pravého tlačítka myši do lišty modulů a zvolit **Vložit** > **Modul BASIC**. Naproti tomu v editoru VBA jsou moduly umístěny ve stromové struktuře v levém okně projektu.



Obrázek 16: Editor OpenOffice Basic *(zdroj: Vlastní tvorba)*



Obrázek 17: Editor VBA *(zdroj: Vlastní tvorba)*

Dalším viditelným rozdílem je jazykový překlad editorů. Zatímco celé prostředí OpenOffice Basic nabízí kompletní češtinu i s nápovědou, editor VBA je pouze v angličtině.

V OO Basic je celkově menší korektura než v editoru VBA. Nepodporuje automatické dokončování, jako je upravení počátečních písmen na velká, neupravuje velká a malá písmena v názvech proměnných a pří psaní procedury se hlavička sama neukončí. Pokud nedokončíme příkaz a klikneme do jiného prostoru, neobjeví se dialogové okno, ani nezčervená příkaz. Když chceme upravit makro v OO Basic, spustí se nám první makro v modulu, zatímco ve VBA se spustí makro, ve kterém je umístěn kurzor.

2.5 Odlišnosti zdrojového kódu Basic

Struktura kódu je blíže rozepsána v kapitole 1.2.2. Tato část se zabývá pouze jejich rozdíly. Excel a Calc pracují s odlišně pojmenovanými objekty, metody a vlastnosti a jejich chování je někdy jiné. Excel nepoužívá všechny rysy, které vytvářejí objektově zaměřené programové prostředí. Objektový model Excelu bývá označován za „jako-objektový“, protože nepodporuje koncept převzetí. To dovoluje, aby definice a realizace jednoho objektu byla založena na definici a realizaci jiného objektu.[8]

Přístup k buňkám je u Calcu o něco složitější, než je k tomu u Excelu. Rozdíl ve zdrojovém kódu je zobrazen na tomto příkladu:

Vložení číslo 10 do buňky na 2. řádku a v 5. sloupci na aktuálním sešitu a stránce:

VBA

```
Cells(2,5).Value=10
```

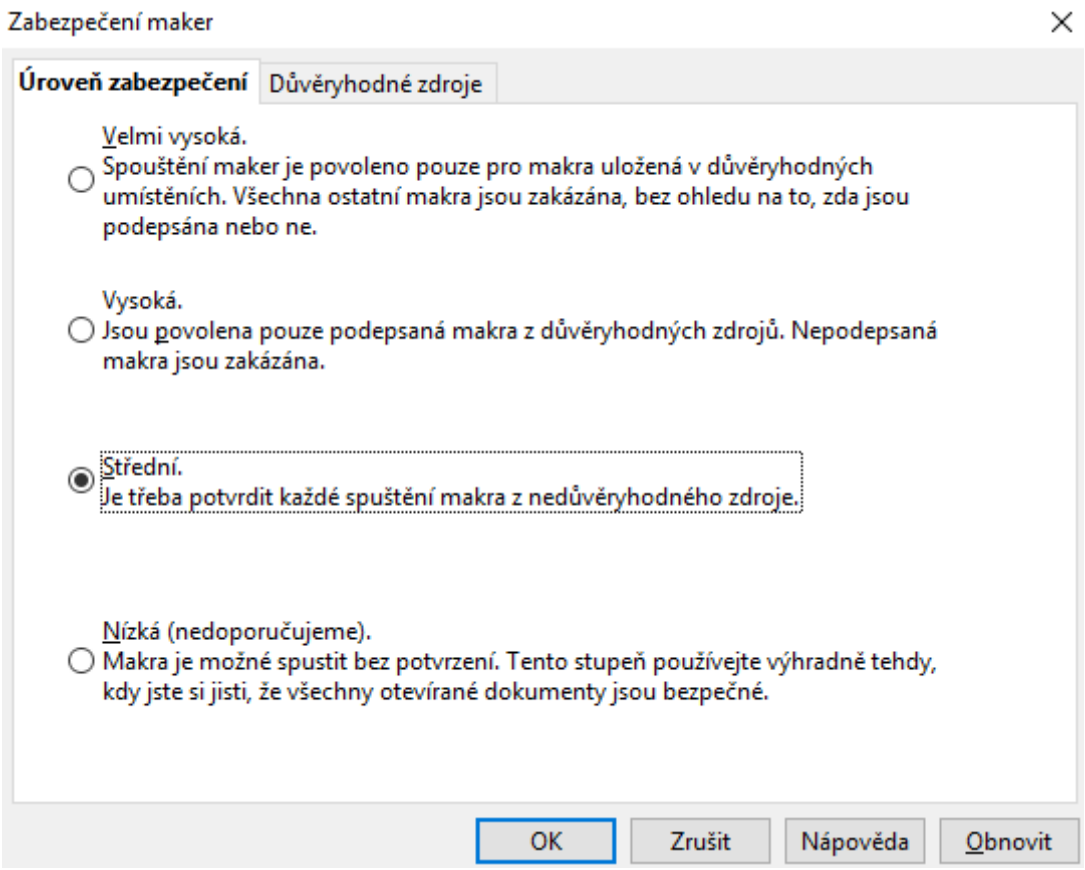
OO Basic

```
thisComponent.Sheets(0).getCellByPostition(4,1).Value=10
```

Při tom `thisComponent`je aktuální sešit, `Sheets(0)` je první list tohoto sešitu (lze použít i `ActiveSheet` jako aktivní sešit) a `getCellByPosition(4,1)` je odkaz na buňku v 5. sloupci a 2. řádku. Číslo sloupce a řádku se zadává v opačném pořadí, než v Excelu a navíc jsou číslovány od nuly. [14] Účinným pomocníkem pro převod zdrojového kódu je stránka http://www.business-spreadsheets.com/vba2oo.asp. Ta slouží k převodu kódu z VBA do OO Basic. Převody však nejsou úplně správné a je potřeba je ručně upravit.

2.6 Zabezpečení maker

Stejně jako v MS Excel, i v AOO Calc je nastaveno zabezpečení maker, protože mohou obsahovat viry a poškodit náš počítač (jak je již popsáno v kapitole 1.6.1). Chceme-li tedy makra používat, musíme je povolit. Výchozím nastavením programu je vysoká úroveň zabezpečení, a to dovoluje spouštět pouze podepsaná makra z důvěrných zdrojů. Toto nastavení potřebujeme změnit. Provedeme to přes volbu **Nástroje** > **Volby** > **OpenOffice** > **Bezpečnost**, zde klikneme na tlačítko **Zabezpečení maker**, tím otevřeme dialogové okno pro zabezpečení makra a vybereme úroveň, která nám vyhovuje (viz obr. 18).



Obrázek 18: Dialogové okno pro nastavení úrovně zabezpečení *(Zdroj: Vlastní tvorba)*

Doporučuje se používat střední zabezpečení, tzn. že se nás AOO vždy zeptá, jestli obsažené makro v dokumentu chceme povolit, nebo zakázat. Můžeme také nastavit důvěryhodné umístění a certifikáty, které umožňují načíst dokumenty bez potvrzení. [10]

II. PRAKTICKÁ ČÁST

3 OPAKUJÍCÍ SE TABULKOVÉ OPERACE V ADMINISTRATIVNÍ PRAXI

3.1 Formátování buněk

Prvním praktickým příkladem makra je formátování buněk, což je asi nejčastěji vykonávaná funkce v Excelu, proto ji zautomatizuji. Formátování obsahuje výšku a šířku buňky, zalomení textu, sloučení či rozdělení buňky, ohraničení buňky, barva buňky, formát čísel, formát textu (barva, font, velikost), zarovnání a orientace textu. Celý postup jde samozřejmě nahrát pomocí záznamu, víme-li přesný formát buňky, který potřebujeme. Já projdu jednotlivé kroky formátování, jeho možnosti a ukáži, jak makro upravit editorem VBA. Ve většině ukázkových příkazech používám volbu buněk Selection (vybrané), protože formátování v Excelu často nebývá jednotvárné pro celý list, či nevíme, jak velká tabulka bude. Může se používat také Range ("čísla buněk"), Cells (souřadnice buněk), ActiveCell (aktivní buňka) a další.

1. **Výška a šířka buňky (Rows/Columns) –** nejprve určíme, zda chceme vybrat výšku a šířku buňky automaticky nebo pevně zadanou. Automatická výška/šířka buněk pro celý list je dána kódem:

```
With Cells
   .Rows.AutoFit
   .Columns.AutoFit
End With
```

   Pro pevně zadanou výšku (RowsHeight) zapíšeme kód:

```
With Cells
   .Rows.RowHeight = 20
End With
```

   Pokud upravujeme pouze vybrané řádky (buňku), pak:

```
Selection.RowHeight = 60
```

   Pevná šířka buněk (ColumnsWidth) sešitu se zadá:

```
With Cells
   .Columns.ColumnWidth = 12
End With
```

Následující kód při spuštění makra vypíše v dialogovém okně velikost vybrané buňky:

```
MsgBox "Buňka: Výška = " &Selection.RowHeight & vbCrLf & "Šířka
= " &Selection.ColumnWidth
```

2. **Zalomení textu –** zalomení textu (WrapText) ve vybrané buňce a automatické upravení výšky (EntireRow) buňky se použije především u delších textů:

```
Selection.WrapText = True
Selection.EntireRow.AutoFit
```

3. **Sloučení nebo rozdělení buňky –** sloučení (Merge) buňek do jedné je vhodné pro nadpis sloupce, který obsahuje více řádků:

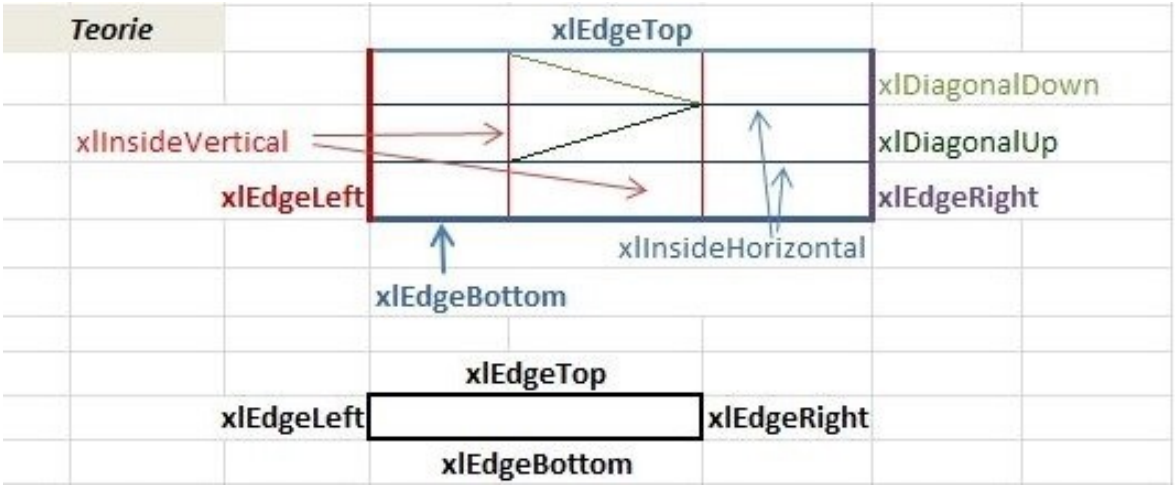
```
Selection.Merge
```

Pro rozdělení buňky:

```
Selection.UnMerge
```

4. **Ohraničení buňky (Borders) –** pomocí VBA se vytváří ohraničení pro různé hrany. Přesná specifikace umístění čár je:
   - xlDiagonalDown (ohraničení probíhá z levého horního rohu do pravého spodního rohu každé buňky),
   - xlDiagonalUp (ohraničení protíná každou vybranou buňku z levého dolního rohu do pravého horního rohu),
   - xlEdgeRight (ohraničení na pravém okraji rozsahu),
   - xlEdgeTop (ohraničení v horním okraji rozsahu),
   - xlEdgeLeft (ohraničení na levém okraji rozsahu),
   - xlEdgeBottom (ohraničení v dolním okraji rozsahu),
   - xlInsideVertical (vnitřní ohraničení všech vertikálních buněk v rozsahu),
   - xlInsideHorizontal (vnitřních ohraničení všech horizontálních buněk v rozsahu).[17]

Zobrazeno je to graficky na obrázku 19.



Obrázek 19: Ohraničení buněk ve VBA [16]

V druhém řádku kódu se nastavuje styl čáry (LineStyle). Je jich 6 základních stylů: xlContinuous, xlDot (tečkovaná), xlDashDotDot (čerchovaná), xlDash (čárkovaná), xlSlantDashDot (čerchovaná posunutá) a xlDouble (dvojitá).

Dalším parametrem je tloušťka čáry (Weight), která může mít hodnotu xlThin (tenká), xlMedium (střední), xlThick(silná). Barva ohraničení funguje jako u pozadí nebo barvy písma. Přiřadíme ji využitím Color (standardní barvy vbBlack, vbRed, vbGreen, vbYellow, vbBlue vbMagenta, vbCyan, vbWhite nebo celočíselná hodnota od 0 do 16581375), ColorIndex, či ThemeColor.Ohraničení buňky tenkou černou horní linkou pak vypadá takto:

```
With Selection.Borders(xlEdgeTop)
   .LineStyle = xlContinuous
   .Weight = xlThin
   .Color = vbBlack
End With
```

Jestli ohraničení chceme odstranit, použijeme:

```
Selection.Borders(xlEdgeTop).LineStyle = xlNone
```

U obou posledních kódů platí, že stačí měnit umístění linky, která je zapsána v závorce, a buď tím tedy konkrétní ohraničení změníme, nebo odstraníme.

5. **Výplň buňky (Interior) –** barvu (pozadí) buňky lze, jak jsem již zmínil, způsoby ColorIndex, Color nebo ThemeColor. Označené buňky se ve VBA přebarví:

```
Selection.Interior.Color = RGB(50, 30, 250)
```

Výplň odstraníme:

```
Selection.Interior.ColorIndex = xlNone
```

6. **Formát čísla –** formát čísla (NumberFormat) buněk se v praxi často upravuje. Přiřazení makru usnadní práci hlavně, pokud se využívá vlastní číselný formát. V Excelu je definováno několik druhů, které jsou vypsány v tabulce 1.

Tabulka 1: Druhy číselných formátů v Excelu [17]

Druh	VBA
Obecný	"General"
Číslo	"0.00"
Měna	"#,##0.00 $"
Účetnictví	"_($* #,##0.00_);_($* (#,##0.00);_($* ""-""??_);_(@_)"
Datum	"dd/mm/yyyy"
Čas	"h:mm:ss;@"
Procenta	"0.00%"
Zlomky	"# ?/? "
Matematický	"0.00E+00"
Text	"@"
Speciální	"000 00" (PSČ)
Vlastní	Např. "$#,##0.00_);[Red]($#,##0.00)"

Konkrétní příklady:

`Selection.NumberFormat = "0.00"` - změní vybrané buňky na číselné

`Cells(1,1).NumberFormat = "@"` - buňka v 1. řádku a v 1. sloupci se převede na textovou

Tyto číselné formáty lze také používat ve funkci Format. Dobrou ukázkou je vypsání aktuálního data a času:

```
Range("A1").Value = Format$(Now, "yyyy/mm/dd hh:nn:ss")
```

7. **Písmo (Font) –** u písma je spousta možností, co vše lze upravovat. Základní úpravou je motiv písma (Font Name), může to být jeden z fontů: Calibri, Times New Roman, Arial, atd. Následující kód nastaví vybranou oblast na písmo Arial:

```
Selection.Font.Name = "Arial"
```

Nastavení řezu písma (Font Style) je možno z: Regular (obyčejné), Bold (tučné), Italic (kurzíva), Bold Italic (tučná kurzíva). Změna textu na kurzívu jde jedním z kódu:

```
Selection.Font.FontStyle = "Italic"
Selection.Font.Italic = True
```

Změna velikosti fontu (Font Size) na 20:

```
Selection.Font.Size = 20
```

Vybarvení textu (Font Color) má stejné vlastnosti jako barva ohraničení. Pro určení barvy využít i RGB model barev. Chceme-li text červeně, použijeme:

```
Selection.Font.Color = RGB(255, 0, 0)
```
[18]

8. **Zarovnání a orientace textu –** zarovnání probíhá vodorovně a svisle. Vodorovné (Horizontal) zarovnání textu má hodnoty: xlGeneral, xlCenter, xlDistributed, xlJustify, xlLeft, xlRight. Pro vycentrování textu zapíšeme:

   ```
   Selection.HorizontalAlignment = xlCenter
   ```

   Zarovnání svisle (Vertical) je možno: xlBottom, xlCenter, xlDistributed, xlJustify, xlTop. Opět kód pro vycentrování textu:

   ```
   Selection.VerticalAlignment = xlCenter
   ```

   Orientace textu se nastaví hodnotou ve stupních od -90 do 90 nebo konstantami xlDownward, xlHorizontal, xlUpward, xlVertical.
   Orientace textu v buňce o 90°:

   ```
   Selection.Orientation = 90
   ```

   Odsazení textu o 1 bod:

   ```
   Selection.IndentLevel = 1
   ```

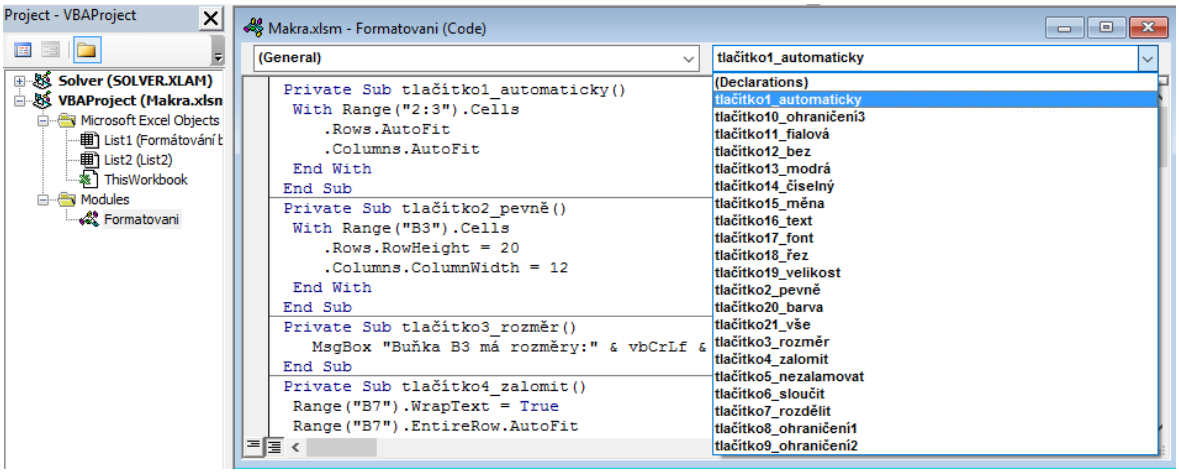
   Směr textu může mít pouze 3 hodnoty: xlRTL (zprava doleva), xlLTR (zleva doprava) nebo xlContext (kontext). Tento příklad kódu nastavuje směr psaní textu xlRTL (zprava doleva).

   ```
   Selection. ReadingOrder = xlRTL
   ```

Nyní k samotné tvorbě. Na tomto příkladu jsem chtěl ukázat, jaké jsou možnosti makra. Dokáží ušetřit spoustu času i při snadné a často prováděné činnosti, jakou je formátování. Jednotlivá makra jsem přidělil tlačítkům. Uživatel poté nemusí nikde nic hledat a pouhým klikem často využívané formáty aplikovat.

Nejdříve jsem si vytvořil nový sešitExcelu a rozvrhl funkce, které chci přidat. Následně otevřel přes kartu **Vývojář** VBE a vytvořil v aktuálním sešitě nový modul (pomocí **In-**

sert > **Module**). Do okna kódu jsem již zapisoval jednotlivá makra. Název byl volen podle toho, že makro bude přiřazeno tlačítku a jakou bude plnit funkci (např. tlačítko4_zalomit, tlačítko6_sloučit, tlačítko15_měna atd.). Lze to vidět na obrázku 21, který ukazuje prostředí vytvářeného kódu i s jednotlivými názvy.



Obrázek 20: Kód makra pro formátování ve VBA *(zdroj: Vlastní tvorba)*

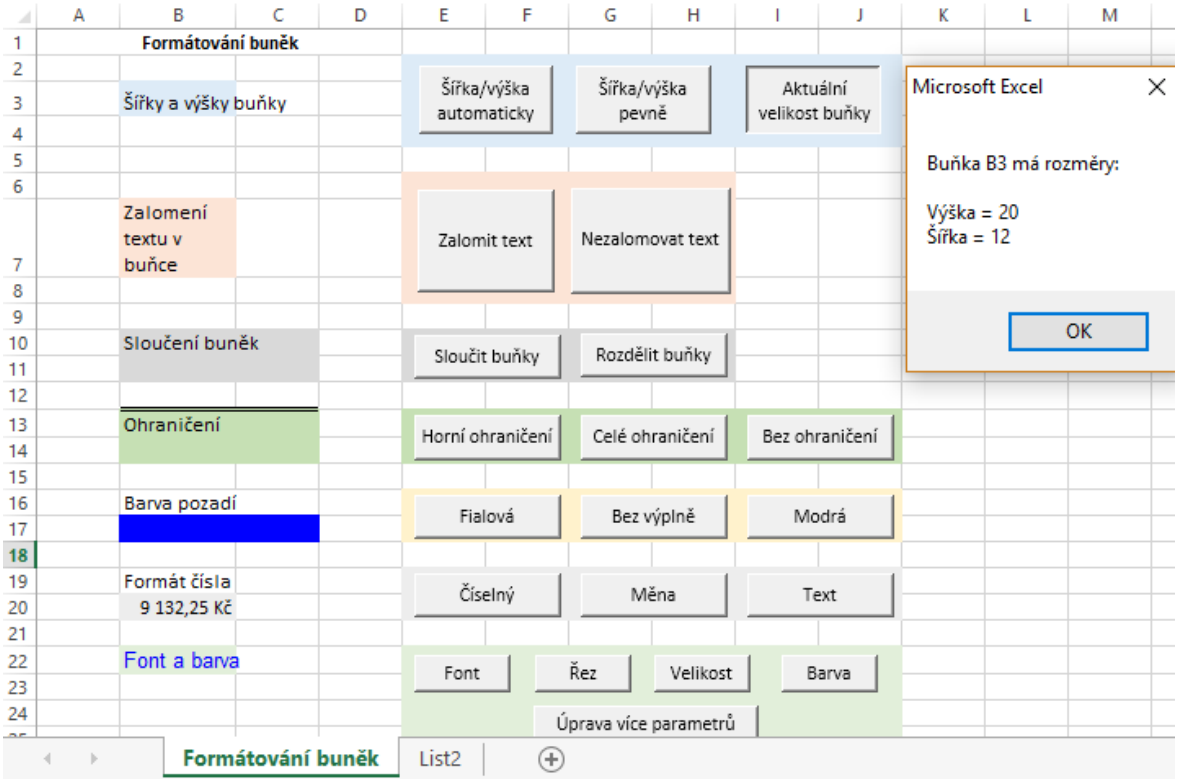
Celý kód makra vypadá následovně:

```
Private Sub tlačítko1_automaticky()
 With Range("2:3").Cells
    .Rows.AutoFit
    .Columns.AutoFit
 End With
End Sub
Private Sub tlačítko2_pevně()
 With Range("B3").Cells
    .Rows.RowHeight = 20
    .Columns.ColumnWidth = 12
 End With
End Sub
Private Sub tlačítko3_rozměr()
   MsgBox "Buňka B3 má rozměry:" & vbCrLf & vbCrLf & "Výška = " &
Range("B3").RowHeight   &   vbCrLf   &   "Šířka   =   "   &   Ran-
ge("B3").ColumnWidth
End Sub
Private Sub tlačítko4_zalomit()
 Range("B7").WrapText = True
 Range("B7").EntireRow.AutoFit
```

```
End Sub
Private Sub tlačítko5_nezalamovat()
 Range("B7").WrapText = False
 Range("B7").EntireRow.AutoFit
End Sub
Private Sub tlačítko6_sloučit()
 Range("B10:C11").Merge
End Sub
Private Sub tlačítko7_rozdělit()
  Range("B10:C11").UnMerge
End Sub
Private Sub tlačítko8_ohraničení1()
 Range("B13:C14").Select
With Selection.Borders(xlEdgeTop)
    .LineStyle = xlDouble
    .Weight = xlThick
    .ColorIndex = xlAutomatic
End With
End Sub
Private Sub tlačítko9_ohraničení2()
 Range("B13:C14").Select
With Selection.Borders
    .LineStyle = xlDash
    .Weight = xlThick
    .ColorIndex = xlAutomatic
End With
End Sub
Private Sub tlačítko10_ohraničení3()
 Range("B13:C14").Select
With Selection.Borders
    .LineStyle = xlNone
End With
End Sub
Private Sub tlačítko11_fialová()
Range("B17:C17").Interior.ColorIndex = 26
End Sub
Private Sub tlačítko12_bez()
```

```
 Range("B17:C17").Interior.ColorIndex = xlNone
End Sub
Private Sub tlačítko13_modrá()
Range("B17:C17").Interior.Color = RGB(0, 0, 255)
End Sub
Private Sub tlačítko14_číselný()
Range("B20").NumberFormat = "0.00"
End Sub
Private Sub tlačítko15_měna()
Range("B20").NumberFormat = "#,##0.00 $"
End Sub
Private Sub tlačítko16_text()
Range("B20").NumberFormat = "@"
End Sub
Private Sub tlačítko17_font()
Range("B22").Font.Name = "Symbol"
End Sub
Private Sub tlačítko18_řez()
Range("B22").Font.Bold = True
Range("B22").Font.Italic = True
End Sub
Private Sub tlačítko19_velikost()
Range("B22").Font.Size = 20
End Sub
Private Sub tlačítko20_barva()
Range("B22").Font.Color = RGB(255, 0, 0)
End Sub
Private Sub tlačítko21_vše()
Range("B22").Select
With Selection.Font
    .Color = RGB(0, 0, 255)
    .Italic = False
    .Bold = False
    .Size = 12
    .Name = "Arial"
End With
End Sub
```

Do sešitu jsem poté přidával jednotlivá tlačítka (volbou karta **Vývojář** > **Ovládací prvky** > **Vložit** > **Tlačítko**). Po nakreslení tlačítka nabízí Excel ihned možnost makro mu přiřadit, stačí zadat správný název již vytvořeného makra. Nakonec tlačítka vhodně pojmenovat a odzkoušet, zda vše správně funguje. Na obrázku 21 je možno vidět všechny použité formáty.



Obrázek 21: Makro pro formátování buněk v Excelu *(zdroj: Vlastní tvorba)*

3.2 Práce se sešity

Práce se sešity souvisí s jejich vytvářením, ukládáním, uzavřením, či tiskem. Sešit není pouze soubor aplikace Excel, je to také objekt v modelu objektů Excelu. To znamená, že můžeme na sešit odkazovat ve VBA a zautomatizovat vytváření nových sešitů, automaticky sešity ukládat a mnoho dalšího.

3.2.1 Vytvoření nového sešitu

Makro automaticky vytvoří nový sešit. Makro se využije, pokud potřebujeme z tabulky zkopírovat data a vložit je do nově vytvořeného sešitu. Následující makro zkopíruje řadu buněk B3:O22z vybraného listu (Tabulka pro zpracování) a vloží data do nového sešitu (ZákladníTabulka) v buňce A1:

```
Sub vytvořit_sešit()
'Krok 1 Kopírovat data
Sheets("Tabulka pro zpracování").Range("B3:O11").Copy
'Krok 2 Vytvořit nový sešit
Workbooks.Add
'Krok 3 Vložit data
ActiveSheet.Paste Destination:=Range("A1")
'Krok 4 Vypnout upozornění aplikace
Application.DisplayAlerts = False
'Krok 5 Uložit nově vytvořený sešit
ActiveWorkbook.SaveAs _
Filename:="C:\Temp\ZákladníTabulka.xlsx"
'Krok 6 Zapnout upozornění aplikace
Application.DisplayAlerts = True
End Sub
```

Jak makro funguje:

1. Jednoduše zkopíruje data z buněk od B3 do O22. Je vhodné specifikovat název listu, abychom při práci s více otevřenými sešity použili ten správný.
2. K vytvoření nového sešitu použijeme `Add`. Tento příkaz nahrazuje ruční metodu **Soubor** > **Nový** >**Prázdný sešit**.
3. Metodou `Paste` vložíme zkopírovaná data do buňky A1 nového sešitu. Kód odkazuje na objekt `ActiveSheet`, protože při přidání se nám nový sešit ihned zobrazí a stane se aktivní.
4. Upravením `DisplayAlerts` na hodnotu `False` vypneme varování aplikace Excel. Děláme to proto, že v dalším kroku kódu uložíme nově vytvořený sešit a neobtěžuje nás dialog.
5. Soubor uložíme pomocí `SaveAs` a zadáváme úplnou cestu k uloženému umístění včetně konečného názvu souboru.
6. Vracíme aplikaci z kroku 4, kdy jsme vypnuli upozornění. Povolení se provede změnou z `False` na `True`.

3.2.2 Uložení sešitu při změně buňky

Někdy můžeme pracovat na tak citlivých datech, které je potřeba uložit při každé změně určité buňky nebo rozsahu buněk. Toto makro umožňuje definovat rozsah buněk, které při jejich změně způsobí, že se sešit uloží.

Tajemstvím tohoto kódu je metoda `Intersect`. Nechceme list uložit, když se změní jakákoli buňka, ale pouze cílové buňky (C5: O11 v tomto případě).

`Intersect` vrací jednu ze dvou věcí: buď objekt `Range`, který definuje průsečík dané oblasti, nebo `Nothing`. Takže musíme dát cílovou buňku proti `Intersect` pro kontrolu hodnoty `Nothing`. V tomto okamžiku můžeme rozhodnout, zda se uloží sešit. [18] Celé makro se musí zapsat ve VBA do konkrétního listu, v mém případě List2 (Tabulka pro zpracování).

```
Private Sub Worksheet_Change(ByVal Target As Range)
'Krok 1: protíná změna zadaný rozsah?
If Intersect(Target, Range("C5:O11")) Is Nothing Then
'Krok 2: pokud ne, ukončit
Exit Sub
'Krok 3: pokud protíná, uložit
Else
ActiveWorkbook.Save
'zavře příkaz If
End If
End Sub
```

1. Krok 1 – kontrola, zda se cílová (změněná) buňka nachází v rozsahu určeném `Intersect`. Hodnota `Nothing` znamená, že cílová buňka nespadá do rozsahu.
2. Krok 2 – nutí makro k zastavení a ukončení, pokud mezi změněnou a zadaným rozsahem není průsečík.
3. Krok 3 – jestli změněná buňka spadá do zadaného rozsahu, provede se uložení aktivního sešitu a přepsání původního.
4. Krok 4 – uzavře příkaz `If`. Vždy, když vytvoříme instanci příkazu `If ... Then ... Else`, musí se také ukončit pomocí `End If`.

3.2.3 Uložení sešitu před uzavřením

Toto makro je vynikajícím způsobem, jak chránit uživatele před neúmyslným zavřením souboru bez uložení. Excel zpravidla varuje, pokud uzavíráme neuložený sešit, ovšem mnohdy se stane, že se omylem zvolí **Neukládat**, nebo stiskne klávesa „N“ a práci ztratíme. Když je makro implementováno, zajišťuje, aby se Excel automaticky uložil před zavřením sešitu.

Tento kód je spuštěn událostí `BeforeClose`. Když se pokusíme zavřít sešit, tato událost se spustí. Podstata kódu je jednoduchá - zeptá se uživatele, zda skutečně chce uzavřít sešit. Makro pak vyhodnotí, zda uživatel klepnul na tlačítko **OK** nebo **Zrušit**.

Vyhodnocení se provádí příkazem `Select Case`. Ten je alternativou příkazu `If ... Then ... Else`, který umožňuje provádět kontroly v makrech. [19] Aby makro správně fungovalo, musí být vepsáno do objektového pole **ThisWorkbook**. Konstrukce příkazu je:

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)
'Krok 1: Aktivuje okno se zprávou a spustí kontrolu
Select Case MsgBox("Uložit a zavřít?", vbOKCancel)
'Krok 2: Stlačeno Zrušit, zrušit uzavření
Case Is = vbCancel
Cancel = True
'Krok 3: Stisknuto OK, uloží sešit a zavře
Case Is = vbOK
ActiveWorkbook.Save
'Krok 4: Zavře příkaz Select Case
End Select
End Sub
```

1. V kroku 1 aktivujeme okno zprávy jako podmínku pro příkaz `Select Case`. Zde použijeme argument `vbOKCancel`, abychom zajistili, že tlačítka **OK** a **Zrušit** budou prezentována jako volby.
2. Pokud uživatel klikne na tlačítko **Zrušit** v okně zprávy, makro řekne aplikaci Excel zrušit událost `Workbook_Close`. To se stane tím, že zadáme `True` na Cancel Boolean.
3. Když uživatel v okně se zprávou kliknul na tlačítko **OK**, krok 3 se provede. Zde říkáme Excelu, aby sešit uložil. A protože se neprovedl příkaz `Cancel True`, Excel pokračuje zavřením.

4. V kroku 4 jednoduše uzavřeme příkaz `Select Case`. Pokaždé když vytvoříme instanci `Select Case`, musíme ji ukončit odpovídajícím výběrem konce.

3.2.4 Zavření všech sešitů najednou

Jednou z nejvíce obtěžujících činností v aplikaci Excel je uzavření mnoha sešitů najednou. Každý otevřený sešit je potřeba aktivovat, zavřít a potvrdit. Toto makro projde hromadně všemi otevřenými sešity, automaticky je uloží a zavře:

```
Sub zavření_sešitů()
'Krok 1: Deklarovat proměnné
Dim wb As Workbook
'Krok 2: Projít sešity, uložit a zavřít
For Each wb In Workbooks
wb.Close SaveChanges:=True
Next wb
End Sub
```

1. Deklarovat proměnnou objektu, která představuje objekt sešitu. To zachytí všechny otevřené sešity.
2. Projít otevřenými sešity, uložit je a zavřít. Pokud bychom sešit nechtěli uložit, změníme argument `SaveChanges` z `True` na `False`.

3.2.5 Vytisknout všechny sešity v adresáři

Pokud potřebujeme tisknout více sešitů v adresáři, můžeme použít toto makro.V něm použijeme funkci `Dir` k vrácení řetězce, který představuje název souboru. Zde `Dir` vyhledá všechny soubory přípony .xlsx v daném adresáři. Poté se nám otevře každý soubor, vytiskne a soubor zavře. Do modulu zapíšeme:

```
Sub vytisknutí_vAdresáři()
'Krok 1: Deklarovat proměnné
Dim MyFiles As String
'Krok 2: Určit cílový adresář
MyFiles = Dir("C:\Users\Public\*.xlsx")
Do While MyFiles <> ""
'Krok 3: Otevřít sešit jeden po druhém
Workbooks.Open "C:\Users\Public\" & MyFiles
ActiveWorkbook.Sheets("List1").PrintOut Copies:=1
```

```
ActiveWorkbook.Close SaveChanges:=False
'Krok 4: Další soubor v adresáři
MyFiles = Dir
Loop
End Sub
```

1. Krok 1 deklaruje `MyFiles` řetězcovou proměnnou, která zachycuje každý název souboru ve výčtu.
2. Krok 2 používá funkci `Dir` pro zadání adresáře a typu souboru, který hledáme. Kód zde hledá `* .xlsx`. To znamená, že projde pouze soubory .xlsx. Chceme-li soubory .xls, musíme to specifikovat (spolu s adresářem, který prohledáváme). Makro předává libovolný název souboru, který nalezne řetězcová proměnná `MyFiles`.
3. Krok 3 otevře soubor a poté vytiskne jednu kopii List1. Je samozřejmě možné změnit, které listy se vytisknou a také počet kopií.
4. Krok 4 přechází zpět k hledání dalších souborů. Pokud žádné další soubory nenajde, je proměnná `MyFiles` prázdná a makro se ukončí.

3.3 Automatizace Listů

Tato část se zabývá makry, která pracují na úrovni listu. Jsou to makra plnící činnosti, jakými jsou např. přidávání, mazání a přejmenování pracovních listů. Mohou provádět také seřazení listů, či vytvořit obsah pracovních listů.

3.3.1 Seřadit listy podle názvu

Obsahuje-li sešit spoustu listů, časem může být velmi problematické a zdlouhavé se v nich zorientovat. Bohužel Excel nemá žádnou nativní funkci, která by sešity ihned umožnila seřadit např. dle abecedy. Nechceme-li sešity ručně třídit, použijeme na to makro.

Toto makro může vypadat na první pohled složitě, ale jeho činnost je poměrně jednoduchá. Prochází přes jednotlivé listy v sešitu a porovnává aktuální list s předchozím. Je-li název předchozího listu větší než aktuální list (abecedně), makro přesune aktuální list před něj. Až se provedou všechny iterace, sešit bude seřazen. Makro vepíšeme do modulu:

```
Sub seřazení_abecedně()
'Krok 1: Deklarace proměnných
Dim CurrentSheetIndex As Integer
Dim PrevSheetIndex As Integer
```

```
'Krok 2: Nastavení počátečního indexu a začne opakování
For CurrentSheetIndex = 1 To Sheets.Count
For PrevSheetIndex = 1 To CurrentSheetIndex - 1
'Krok 3: Zkontroluje aktuální list proti předchozímu
If UCase(Sheets(PrevSheetIndex).Name) > _
UCase(Sheets(CurrentSheetIndex).Name) Then
'Krok 4: Pokud přesune aktuální list před předcházející
Sheets(CurrentSheetIndex).Move _
Before:=Sheets(PrevSheetIndex)
End If
'Krok 5 Vrací se zpátky a opakuje
Next PrevSheetIndex
Next CurrentSheetIndex
End Sub
```

1. Krok 1 deklaruje dvě proměnné typu Integer. `CurrentSheetIndex` obsahuje indexové číslo pro aktuální iteraci listu a `PrevSheetIndex` zahrnuje indexové číslo pro předchozí iteraci listu.
2. V kroku 2 začne makro počítat iteraci pro obě proměnné. Pro `PrevSheetIndex` je jedno číslo za `CurrentSheetIndex`. Po nastavení počtů začne opakování.
3. V kroku 3 kontrolujeme, zda je název předchozího listu větší než název aktuálního listu. Tento krok používá funkce `UCase`. Je zde k získání obou názvů ve stavu velkých písmen. Tím se zabrání chybám při třídění v důsledku rozdílných stavů písma.
4. Krok 4 je dosažen pouze, pokud je předchozí název listu větší než aktuální název listu. V tomto kroku použijeme metodu `Move` k přesunutí aktuálního listu před předchozí list.
5. V kroku 5 se vrátíme zpět k začátku smyčky. Každé opakování zvyšuje obě proměnné o jedno číslo, dokud nedojde k poslednímu listu. Po vyčerpání všech iterací makro skončí.

3.3.2 Vytvoření obsahu pracovních listů

Mimo setřídění listů je často požadovaným makrem aplikace Excel vytvoření tabulky obsahu pro listy v sešitu. Důvodem je lepší orientace v sešitech a jejich přehlednost. Často musíme pracovat se soubory, které mají více karet listů, a zabere spoustu času, než se do-

staneme k tomu správnému. Makro vytvoří nejen seznam názvů listů v sešitu, ale také hypertextové odkaz, takže můžeme jednoduše klepnout na list. Provádí několik činností:

- Odstraňuje předchozí tabulku obsahu
- Vytvoří novou tabulku obsahu
- Použije název každého listu a vloží jej do obsahu
- Přidá hypertextový odkaz na každou položku v obsahu

Celý kód se zapíše do modulu a postup je následující:

```
Sub vytvoření_obsahuSešitů()
'Krok 1: Deklarace proměnných
Dim i As Long
'Krok 2: Smaže předchozí obsah, pokud je
On Error Resume Next
Application.DisplayAlerts = False
Sheets("Obsah").Delete
Application.DisplayAlerts = True
On Error GoTo 0
'Krok 3: Přidá nový obsah na první list
ThisWorkbook.Sheets.Add _
Before:=ThisWorkbook.Worksheets(1)
ActiveSheet.Name = "Obsah"
'Krok 4: Spustí počítadlo i
For i = 1 To Sheets.Count
'Krok 5: Vybere další dostupný řádek
ActiveSheet.Cells(i, 1).Select
'Krok 6: Přídá název listů a hypertextový odkaz
ActiveSheet.Hyperlinks.Add _
Anchor:=ActiveSheet.Cells(i, 1), _
Address:="", _
SubAddress:=" '" & Sheets(i).Name & "'!A1", _
TextToDisplay:=Sheets(i).Name
'Krok 7: Vrací přírůstek i
Next i
End Sub
```

1. První krok deklaruje proměnnou typu Integer s názvem `i`, která slouží jako počítadlo, jak makro iteruje přes listy.

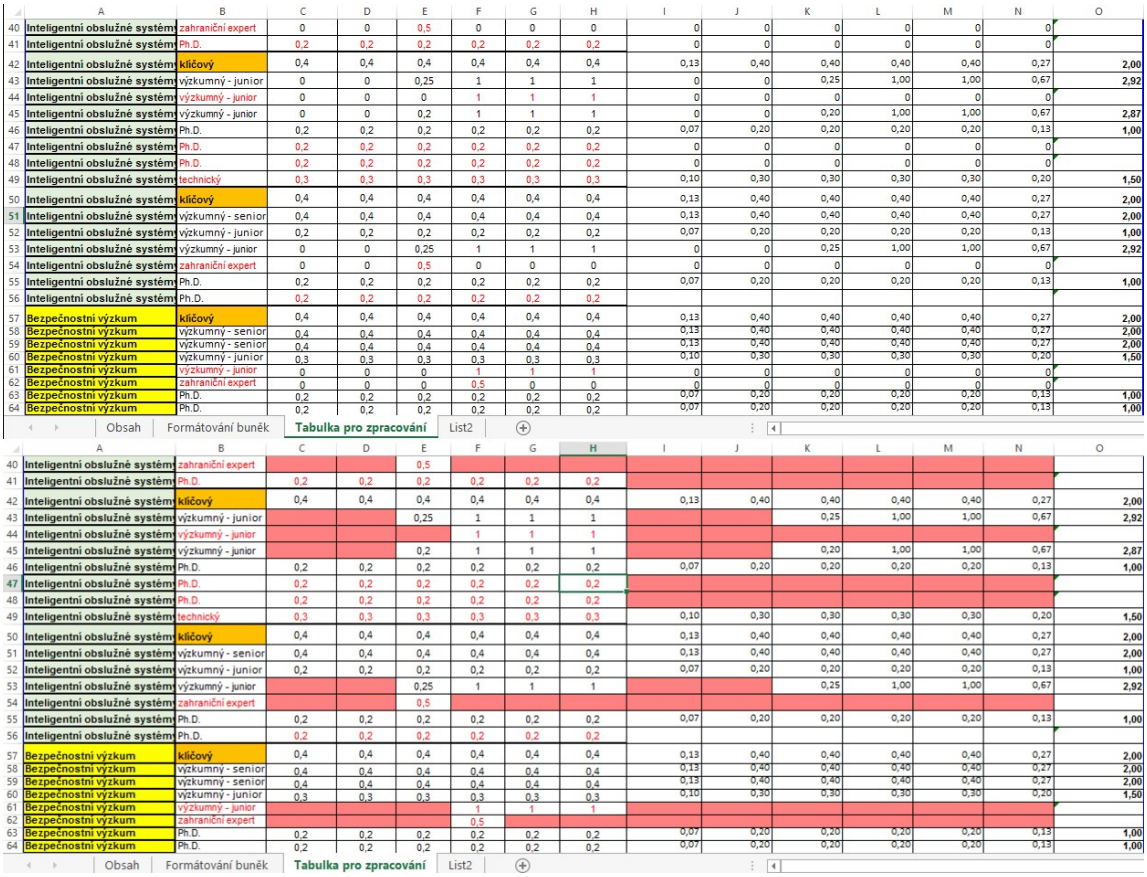
2. Druhý krok se v podstatě pokouší vymazat předchozí list nazvaný Obsah. Protože tento list nemusí být vytvořen, potřebujeme spustit Krok 2 s`On Error Resume Next`. Error handler zamezí programu Excel pokračovat v makru, pokud nalezne chybu. Poté odstraní list Obsah pomocí `DisplayAlerts`, který účinně vypíná varování aplikace Excel, takže nemusíme potvrzovat smazání. Nakonec resetujeme error handler zadáním `On Error GoTo 0`, aby mohl zachytit další případné chyby.
3. Ve třetím kroku přidáme do sešitu nový list. Použitím `Before` umístíme nový list jako první. Poté pojmenujeme list jako Obsah. Při přidání nového listu se automaticky stává aktivní. Tudíž celý postup v tomto kódu dále odkazující na Obsah používá už jen `ActiveSheet`.
4. Čtvrtý krok spustí počítadlo `i`na 1 a ukončí jej při maximálním počtu všech listů v sešitu.
5. Krok pátý vybírá příslušný řádek v sešitě Obsah. To znamená, že pokud je počítadlo `i`na 1, vybere první řádek v Obsahu. Pokud je počítadlo `i`ve 2, zvolí druhý řádek a tak dále.
6. Šestý krok používá `Hyperlinks.Add` pro přidání názvu listu a hypertextových odkazů vybraným buňkám.
7. Poslední krok v makru se vrátí k inkrementaci `i`. Když počítadlo `i` dosáhne čísla, které se rovná počtu listů v sešitu, makro končí.

3.4 Práce s daty

Tato část se týká správy dat: jejich vybírání, formátování a další. Při práci s informacemi v aplikaci Excel musíme často nějakým způsobem transformovat data.

3.4.1 Nahrazení hodnot v rozsahu

Může nastat situace, kdy budeme potřebovat v tabulce vyplnit prázdné buňky nulovými hodnotami (či jakýmkoli číslem), nebo naopak z nulových hodnot udělat buňky prázdné. Tohle makro ušetří spoustu práce především při velkém množství dat v tabulce. Příkladem je následující „Tabulka pro zpracování“, ve které bylo nutno udělat z nulových hodnot prázdné buňky a pro zviditelnění je podbarvit, jako to můžeme vidět na obrázku 22.



Obrázek 22: Úprava tabulky z nulových na prázdné buňky *(zdroj: Vlastní tvorba)*

Aplikovaný kód je:

```
Sub odstranění_nul()
'Krok 1: Zadaní rozsahu buněk
For Each cell In Range("C5:N99")
'Krok 2: Nastavení podmínky a opakovat
If cell.Value = "0" Then cell.Value = "": cell.Interior.ColorIndex
= 22
Next
End Sub
```

1. Nejdříve zadáme rozsah buněk, pro které se makro aplikuje.
2. Nastavujeme podmínku, že pokud má buňka hodnotu 0, pak dostane prázdnou hodnotu a zároveň je podbarven pomocí `cell.Interior.ColorIndex`.
3. Příkaz `Next` pokračuje další buňkou. Po projetí všech buněk v cílovém rozsahu makro skončí.

3.4.2 Zvýraznění duplikátů

Existuje mnoho způsobů, jak najít a zvýraznit duplikáty - manuální metody zahrnují vzorce, podmíněné formátování, třídění a tak dále. Toto makro zjednodušuje úkol, umožňuje najít a zvýraznit duplicity v datech kliknutím myši.

Pomocí příkazu `For Each` aktivuje každou buňku v cílovém rozsahu. Poté použijeme funkci `CountIf` k počítání, kolikrát se hodnota v aktivní buňce vyskytuje v zvoleném rozsahu.

Pokud je toto číslo větší než jedna, formátujeme žlutou buňku. Makro implementujeme do modulu:

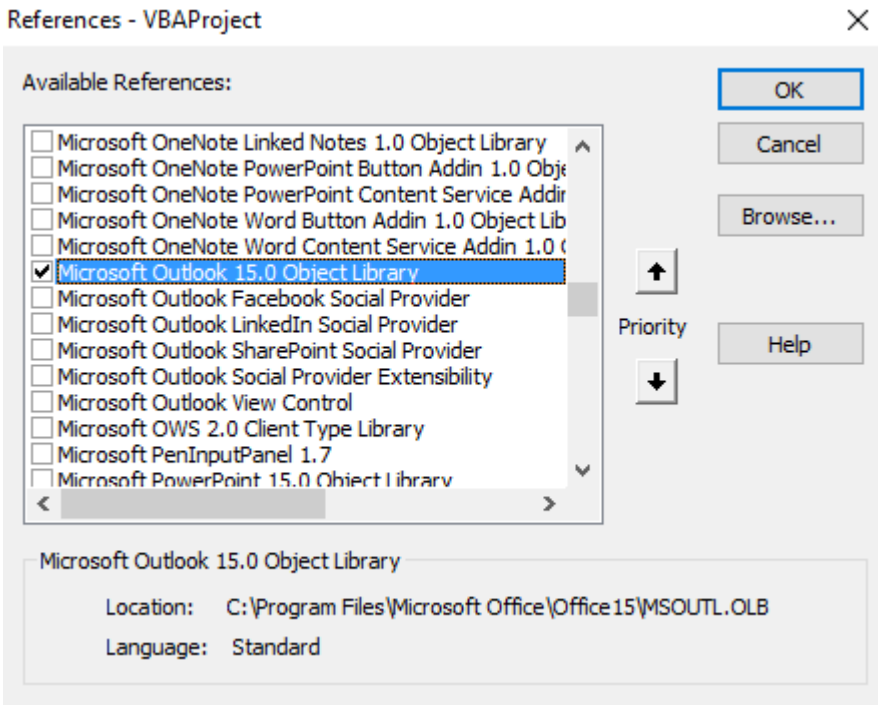
```
Sub zvýraznění_duplikátů()
'Krok 1: Deklarovat proměnné
Dim MyRange As Range
Dim MyCell As Range
'Krok 2: Definovat rozsah buněk
Set MyRange = Selection
'Krok 3: Začne procházet obsah
For Each MyCell In MyRange
'Krok 4: Kontrola, zda formátuje
If WorksheetFunction.CountIf(MyRange, MyCell.Value) > 1 Then
MyCell.Interior.ColorIndex = 36
End If
'Krok 5: Opakovat na další buňce
Next MyCell
End Sub
```

1. Deklaruje dvě proměnné objektu `Range` – `MyRange` pro udržení cílového rozsahu a `MyCell` pro každou buňku v rozsahu.
2. Vyplní proměnnou `MyRange` s cílovým rozsahem. Zde používáme vybraný rozsah, který byl označen v tabulce.
3. Začne procházet každou buňku v cílovém rozsahu a aktivuje každou buňku.
4. Objekt `WorksheetFunction` poskytuje způsob, jak lze spustit mnoho funkcí sešitu ve VBA. `WorksheetFunction` používáme ke spuštění funkce `CountIf`. Tady počítáme, kolikrát se aktivovaná hodnota buňky (`MyCell.Value`) nachází v daném rozsahu (`MyRange`). Pokud výraz `CountIf` vyhodnotí hodnotu větší než 1, makro změní vnitřní barvu buňky.

5. Vrací se k další buňce. Po aktivaci všech buněk v cílovém rozsahu makro skončí.

3.5 E-mailování z MS Excel

Tato část zahrnuje e-mailování z aplikace Excel. Poslat můžeme rozsah buněk nebo celý sešit jako přílohu. Při posílání i přijímání zpráv Excel využívá MS Outlook a je nutné mít jej nainstalovaný. Nastavit se musí i objektová knihovna MS Outlook. Provede se to v otevřeném editoru VBA, volba **Tools** > **References**. V dialogovém okně najdeme Microsoft Outlook XX Object Library (XX je verze Outlook), zaškrtneme a potvrdíme (viz obrázek 22).



Obrázek 23: Povolení Microsoft Outlook Object Library *(zdroj: Vlastní tvorba)*

3.5.1 Odeslat sešit jako přílohu

Nejdůležitější úkol aplikace Outlook, který můžeme provést pomocí automatizace, je odesílání e-mailu. Vytvoření toho makra lze rozdělit na 4 kroky:

1. Krok 1 - deklaruje dvě proměnné. `OLApp` je proměnná objektu, která vystavuje objekt aplikace Outlook. `OLMailis` je proměnná objektu, která obsahuje mailovou zprávu.
2. Krok 2 – aktivuje aplikaci Outlook a spustí novou relaci. Použitím `OLApp.Session.Logon` se přihlásíme k aktuálnímu MAPI (Messaging Application

Programming Interface) s výchozími pověřeními. Vytvoří také mailovou zprávu. Je to náhrada místo tlačítka Nová zpráva v Outlooku.

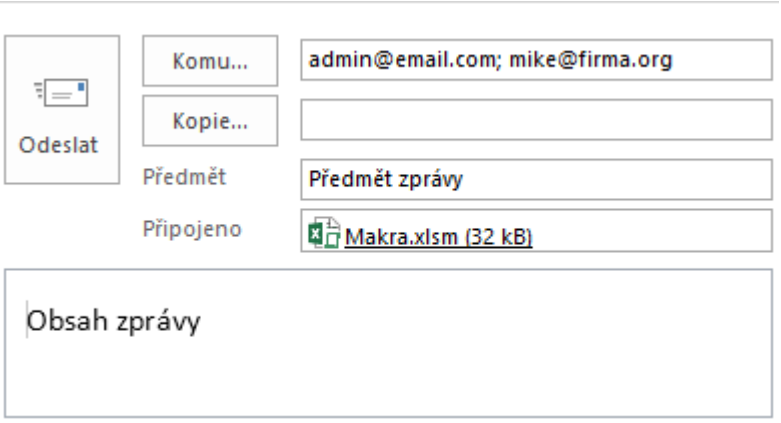
3. Krok 3 - sestaví profil naší mailové zprávy. Patří sem příjemci, příjemci CC (pro kopii), příjemci BCC (pro skrytou kopii), předmět, obsah a přílohy e-mailu. Při větším počtu příjemců se oddělují středníkem. Standardní syntaxe pro přílohu je `.Attachments.Add` a může za tím být cesta k souboru (pokud posíláme jinou přílohu). Zde specifikujeme cestu k souboru v aktuálním sešitu syntaxí `ActiveWorkbook.Fullname`. Tímto nastavíte aktuální sešit jako přílohu e-mailu. Po sestavení zprávy použijeme metodu `.Display` pro kontrolu e-mailu. Můžeme nahradit `.Display` za `.Send`, aby se e-mail poslal bez kontroly.
4. Krok 4 - Uvolnění objektů přiřazených k našim proměnným je obecně dobrou praxí. Snižuje to pravděpodobnost problémů způsobených objekty, které zůstanou spuštěny v paměti. Hodnotu nastavíme na `Nothing`.

V kódu je aktivní sešit odeslán jako příloha dvěma příjemcům e-mailu:

```
Sub odeslání_emailu()
'Krok 1: Deklarovat proměnné
Dim OLApp As Outlook.Application
Dim OLMail As Object
'Krok 2: Otevřít Outlook spustit nový email
Set OLApp = New Outlook.Application
Set OLMail = OLApp.CreateItem(0)
OLApp.Session.Logon
'Krok 3: Vytvořit email a odeslat
With OLMail
'adresát
.To = "admin@email.com; mike@firma.org"
'kopie pro
.CC = ""
'skrytá kopie pro
.BCC = ""
'předmět zprávy
.Subject = "Předmět zprávy"
'text zprávy
.Body = "Obsah zprávy"
'aktivní sešit jako příloha
.Attachments.Add ActiveWorkbook.FullName
```

```
'zobrazení okna se zprávou
.Display
End With
'Krok 4: Vyčištění paměti
Set OLMail = Nothing
Set OLApp = Nothing
End Sub
```

Při spuštění makra `odeslání_emailu` automaticky vyskočí přednastavené okno Outlooku pro odeslání e-mailu, jak vidíme na obrázku 23.



Obrázek 24: Okno Outlook pro odeslání e-mailu *(zdroj: Vlastní tvorba)*

3.5.2 Odeslat rozsah jako přílohu

Nechceme-li posílat celý sešit prostřednictvím e-mailu, můžeme poslat pouze určitý rozsah. Makro funguje tak, že zadaný rozsah zkopíruje a hodnoty a formáty vloží do dočasného souboru, který se po odeslání smaže.

Postup vytvoření má 6 kroků, některé se však opakují s předchozím:

1. Deklarace proměnných OLApp a OLMail.
2. Zkopírovat zadaný rozsah a vložit hodnoty a formáty do dočasného souboru aplikace Excel. Makro soubor uloží zadaného adresáře s názvem.
3. Třetí krok je stejný – spuštění nové relace a přihlášení.
4. Vytvoření profilu mailové zprávy probíhá stejně jako u předchozího. Změna je ve specifikaci cesty dočasně vytvořeného souboru `.Attachments.Add (ThisWorkbook.Path &"\TempRangeForEmail.xlsx")`.

5. Zavřít a smazat dočasný soubor dle cesty, kam jsme ho uložili `Kill ThisWorkbook.Path &"\TempRangeForEmail.xlsx"`.
6. Vyčištění paměti.

Následující makro odešle e-mailem v příloze buňky A1:O11 a následně soubor smaže:

```
Sub odeslání_výběru()
'Krok 1: Deklarovat proměnné
Dim OLApp As Outlook.Application
Dim OLMail As Object
'Krok 2: Zkopírovat, vložit do nového sešitu a uloži
Sheets("Revenue Table").Range("A1:O11").Copy
Workbooks.Add
Range("A1").PasteSpecial xlPasteValues
Range("A1").PasteSpecial xlPasteFormats
ActiveWorkbook.SaveAs ThisWorkbook.Path & " \ TempRangeFo-
rEmail.xlsx"
'Krok 3: Otevřít Outlook spustit nový email
Set OLApp = New Outlook.Application
Set OLMail = OLApp.CreateItem(0)
OLApp.Session.Logon
'Krok 4: Vytvořit email a odeslat
With OLMail
.To = "admin@email.com; mike@firma.org"
.CC = ""
.BCC = ""
.Subject = "Předmět zprávy"
.Body = "Obsah zprávy"
.Attachments.Add (ThisWorkbook.Path & " \ TempRangeForEmail.xlsx")
.Display
End With
'Krok 5: Smazat dočasný Excel soubor
ActiveWorkbook.Close SaveChanges:=True
Kill ThisWorkbook.Path & " \ TempRangeForEmail.xlsx"
'Krok 6: Vyčištění paměti
Set OLMail = Nothing
Set OLApp = Nothing
End Sub
```

3.6 Integrace aplikace Excel a dalších Office aplikací

Procesy orientované na data mají určitý tok aplikací, které přebírají data od vytvoření až po konečného uživatele. V mnoha případech se data přesouvají z databáze, jako je například aplikace Microsoft Access, analyzují a seskupují v Excelu a poté jsou distribuovány prostřednictvím dokumentu Word, prezentace PowerPoint nebo dokonce e-mailu.

3.6.1 Odesílání dat do dokumentu Word

Makro nahrazuje neustálé kopírování a vkládání dat do Wordu. Před samotným makrem je však důležité projít několik nastavení.

Před samotným procesem musíme mít již vytvořený dokument Word. V tomto dokumentu vytvoříme záložku označující umístění, kde chceme kopírovat data z Excelu. Chceme-li vytvořit záložku v dokumentu Word, umístíme kurzor tam, kam chceme záložku, zvolíme kartu **Vložit** a vybereme možnost **Záložka** ve skupině **Odkazy**. Tím se aktivuje dialogové okno **Záložka**, kde přidáme název záložky.

Také je třeba nastavit odkaz na knihovnu objektů Microsoft Word v Excelu. Postup je stejný, jako jsme prováděli při povolování MS Outlook. V editoru Visual Basic vybereme příkaz **Tools > References**. Nalezneme položku Microsoft Word XX Object Library a zaškrtneme políčko. Poté už můžeme zapisovat kód do modulu.

```
Sub odeslat_Word()
'Krok 1: deklarovat proměnné
Dim MyRange As Excel.Range
Dim wd As Word.Application
Dim wdDoc As Word.Document
Dim WdRange As Word.Range
'Krok 2: zkopírovat danou oblast
Sheets("Tabulka pro zpracování").Range("B4:H11").Copy
'Krok 3: otevřít cílový Word
Set wd = New Word.Application
Set wdDoc = wd.Documents.Open _
(ThisWorkbook.Path & "C:\Users\Public" & "Makra.docx")
wd.Visible = True
'Krok 4: nastavit na cílovou záložku
Set WdRange = wdDoc.Bookmarks("Tabulka").Range
'Krok 5: odstanit starou tabulku, vložit novou
```

```
On Error Resume Next
WdRange.Tables(1).Delete
WdRange.Paste 'vložit do tabulky
'Krok 6: upravit šířky sloupců
WdRange.Tables(1).Columns.SetWidth _
(MyRange.Width / MyRange.Columns.Count), wdAdjustSameWidth
'Krok 7: znovu vložit záložku
wdDoc.Bookmarks.Add "Tabulka", WdRange
'Krok 8: vyčistit pamět
Set wd = Nothing
Set wdDoc = Nothing
Set WdRange = Nothing
End Sub
```

1. Deklaruje čtyři proměnné: `MyRange` obsahuje cílový rozsah Excel, který chceme kopírovat; `Wd` je proměnná objektu, která vystavuje objekt `Word Application`; `WdDoc` je proměnná objektu, která vystavuje objekt `Word Document`; `wdRange` je proměnná objektu, která vystavuje objekt `Word Range`.
2. Zkopíruje rozsah z „Tabulka pro zpracování“.
3. Otevře existující cílový dokument Word, který slouží jako šablona. Nastavíme vlastnost `Visible` aplikace Word na hodnotu `True`. Tím zajistíme, že při spuštění kódu budeme moci vidět akci v aplikaci Word.
4. Používá Wordovský `Range` pro nastavení na cílovou záložku. Tímto způsobem v podstatě vybíráme záložku jako rozsah, což nám umožňuje provádět akce v tomto rozsahu.
5. Smaže libovolnou tabulku, která může existovat v záložce a potom vloží zkopírovaný rozsah. Pokud nejprve neodstraníme existující tabulky, zkopírovaný rozsah se připojí k existujícím datům.
6. Při vkládání souboru Excel do dokumentu Word nemusí šířky sloupců správně odpovídat obsahu v buňkách. Tento krok řeší problém úpravou šířky sloupců. Zde je šířka každého sloupce nastavena na číslo, které se rovná celkové šířce tabulky dělené počtem sloupců v tabulce.
7. Když vložíme rozsah Excel do cílové záložky, v podstatě přepíšeme záložku. Tento krok znovu záložku vytvoří, abychom zajistili, že při příštím spuštění tohoto kódu bude k dispozici.
8. Konečně makro uvolňuje objekty přiřazené proměnným, což snižuje pravděpodobnost problémů způsobených nechtěnými objekty, které mohou zůstat otevřené v paměti.

3.6.2 Odesílání dat do prezentace PowerPoint

Spousta PowerPointových prezentací obsahuje data, která byla zkopírována přímo z aplikace Excel. Jde o častou činnost, protože analyzovat a vytvářet grafy a pohledy dat v Exceluje je daleko snadnější, než je tomu v PowerPoint. Makro v této části umožňuje dynamicky vytvářet snímky aplikace PowerPoint, které obsahují data ze zadaného rozsahu.

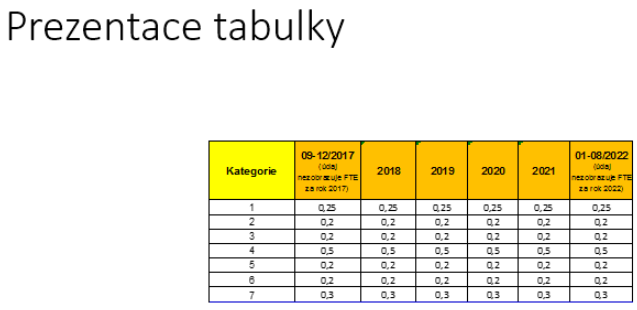
Protože je tento kód spuštěn z aplikace Excel, je třeba nastavit odkaz na knihovnu objektů Microsoft PowerPoint. Opět můžeme nastavit odkaz otevřením editoru jazyka v aplikaci Excel a výběrem **Tools** > **References**. Nalezneme položku Microsoft PowerPoint XX Object Library a zaškrtneme. Kód zadáme opět do modulu VBA.

```
Sub odeslat_PowerPoint()
'Krok 1: deklarovat proměnné
Dim PP As PowerPoint.Application
Dim PPPres As PowerPoint.Presentation
Dim PPSlide As PowerPoint.slide
Dim SlideTitle As String
'Krok 2: otevřít PowerPoint a vytvořit novou prezentaci
Set PP = New PowerPoint.Application
Set PPPres = PP.Presentations.Add
PP.Visible = True
'Krok 3: přidat nový snímek a aktivovat ho
Set PPSlide = PPPres.Slides.Add(1, ppLayoutTitleOnly)
PPSlide.Select
'Krok 4: kopírovat rozsah jako obrázek
Sheets("Tabulka pro zpracování").Range("B4:H11").CopyPicture _
Appearance:=xlScreen, Format:=xlPicture
'Krok 5: vložit obrázek a upravit polohu
PPSlide.Shapes.Paste.Select
PP.ActiveWindow.Selection.ShapeRange.Align msoAlignCenters, True
PP.ActiveWindow.Selection.ShapeRange.Align msoAlignMiddles, True
'Krok 6: přidat titulek
SlideTitle = "Prezentace tabulky"
PPSlide.Shapes.Title.TextFrame.TextRange.Text = SlideTitle
'Krok 7: vyčistit pamět
PP.Activate
Set PPSlide = Nothing
```

```
Set PPPres = Nothing
Set PP = Nothing
End Sub
```

1. Deklaruje čtyři proměnné: `PP` je proměnná objektu, která vystavuje objekt aplikace PowerPoint; `PPPres` je proměnná objektu, která vystavuje objekt prezentace PowerPoint; `PPSlide` je proměnná objektu, která vystavuje objekt snímku PowerPoint; `SlideTitle` je řetězcová (string) proměnná, která slouží k předání textu pro nadpis snímku.
2. Otevře aplikaci PowerPoint s prázdnou prezentací. Nastavíme vlastnost `Visible` aplikace PowerPoint na hodnotu `True`. Tak zajistíme, že při spuštění kódu bude akce zobrazena.
3. Přidá nový snímek do prezentace pomocí metody `Add` objektu`Slide`. Používáme `ppLayoutTitleOnly`, čímž zajistíme, že snímek bude vytvořen s rámečkem textového nadpisu. Pak uděláme další krok, kterým nastavíme, aby tento snímek vybral a činil ho aktivním.
4. Používá metodu `CopyPicture` k zkopírování cílového rozsahu jako obrázku. Rozsah, který se zde kopíruje, je rozsah B4 až H11v Prezentace tabulky.
5. Vloží obrázek do aktivního snímku a vycentruje vodorovně i svisle.
6. Ukládá text pro název v řetězcové proměnné a předá tuto proměnnou do aplikace PowerPoint, aby použil text do rámečku nadpisu textu.
7. Uvolňuje objekty přiřazené našim proměnným pro zamezení případných problémů s následujícími objekty.

Výsledně vloženou tabulkou jako obrázek v prezentaci můžeme vidět na obrázku 25.



Obrázek 25: Excelovská tabulka vložená do PowerPoint

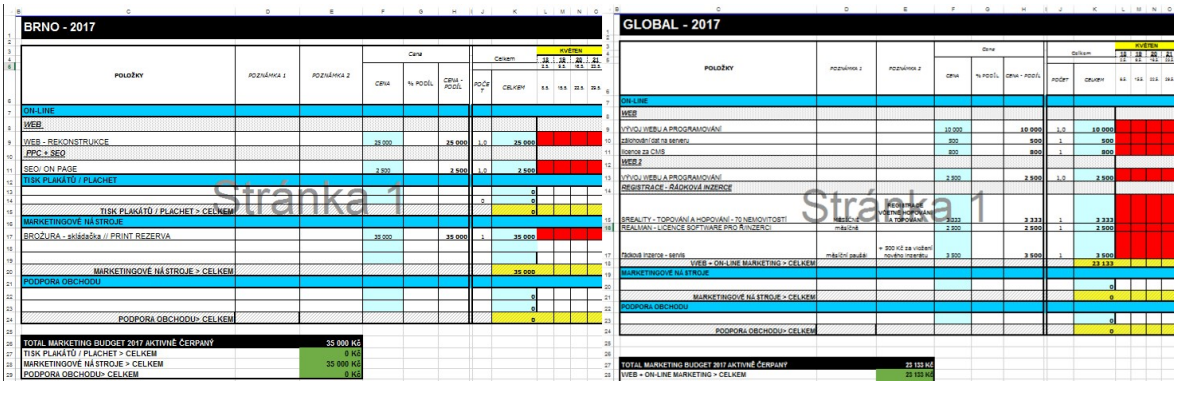
(zdroj: Vlastní tvorba)

4 NÁVRH AUTOMATIZOVANÉHO ŘEŠENÍ OPAKUJÍCÍCH SE TABULKOVÝCH OPERACÍ

Poslední kapitolou této práce je navrhnutí automatizovaného řešení. Příklad jsem vytvářel pro konkrétní úlohu, kdy zaměstnanec ve firmě dostává jednotlivé fakturace za společnosti a z nich musí vytvořit souhrn. Ten následně zformátuje a výslednou tabulku posílá svému nadřízenému e-mailem a sešit vytiskne.

Jelikož sešit budeme posílat e-mailem a využívat aplikace Outlook, před samotnou tvorbou kódu musíme povolit odkaz na objektovou knihovnu Outlook, jako jsme to prováděli u maker 3.5. a je to zobrazeno na obrázku 23.

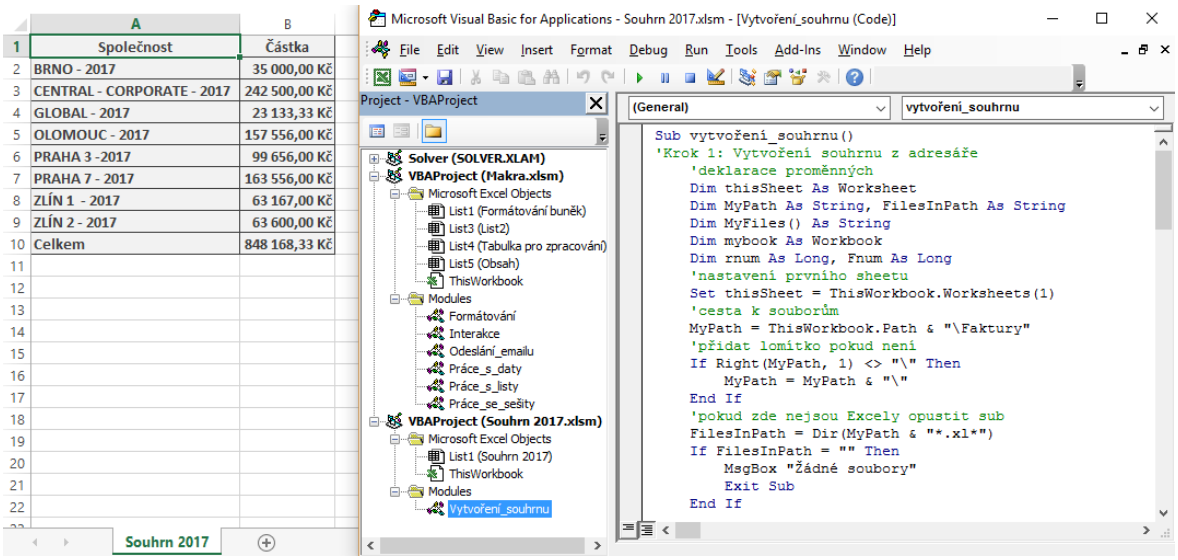
Krokem 1 bylo ze všech sešitů vytvořit souhrnnou tabulku, obsahující název společnosti a celkovou vyfakturovanou částku. Všechny soubory s fakturacemi jsou uloženy ve stejném adresáři (Faktury) a mají jeden list. Z těchto sešitů je tedy nutné sloučit potřebné údaje do jedné tabulky. Na první pohled jsou všechny tabulky téměř totožné, jak můžeme vidět na obrázku 26, název společnosti je obsažen vždy v buňce C1. Ovšem tabulky mají rozdílný počet řádků podle toho, kolik obsahují položek. Celková cena je vždy ve sloupci E, ale pokaždé na jiném řádku. Vycházel jsem z toho, že tento řádek obsahuje v každé tabulce název "TOTAL MARKETING BUDGET 2017 AKTIVNĚ ČERPANÝ", proto jsem pro druhou hodnotu využil příkazu `Cells(.Cells.Find("TOTAL MARKETING BUDGET 2017 AKTIVNĚ ČERPANÝ").Row, 5)` pro nalezení celkové částky.



Obrázek 26: Struktura tabulek pro automatizované řešení *(zdroj: Vlastní tvorba)*

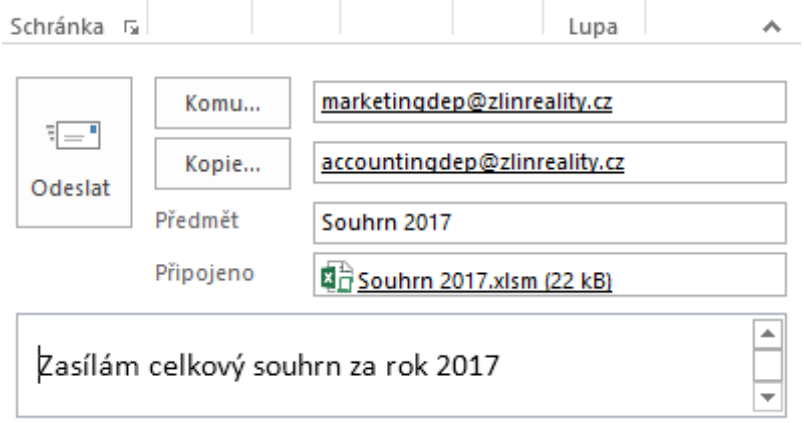
Tento krok nám pod sebe zkopíruje vybrané hodnoty, kde v 1. sloupci jsou názvy společností a ve 2. sloupci částky jednotlivých firem, ovšem tento výsledek vypadá stroze a je potřeba ho vhodně upravit.

Ve druhém kroku sečteme hodnoty 2. sloupce, abychom dostali celkovou sumu. Pomohl jsem si proměnnou `rnum`, která využívá poslední obsazený řádek + 1. Na tuto hodnotu jsem vytvořil v 1. sloupci text „Celkem“ a v 2. sloupci vypočítanou sumu pomocí `WorksheetFunction.Sum(thisSheet.Range("B1:B" & rnum - 1)`. První řádek je přidaný s textem "Společnost" a "Částka" a vycentrovaný. Celá tabulka je pak ve stylu "Výstup" a výsledek můžeme vidět na obrázku 27. List se přejmenuje na "Souhrn 2017" a vytiskne se jedna kopie.



Obrázek 27: Výsledná tabulka pro souhrn fakturací spolu s kódem *(zdroj: Vlastní tvorba)*

Posledním krokem je pak tento sešit odeslat e-mailem jako přílohu. Postup je téměř totožný jako u makra 3.5.1 – kód automaticky spustí Outlook a vytvoří novou relaci. Vyplní adresáta, kterým je tady vedoucí oddělení. Kopie se zasílá na účetní oddělení, nakonec vyplní předmět, obsah a přílohu zprávy. Při zapnutí makra samozřejmě proběhne celý proces automaticky, nám jen vyskočí na konci procesu okno s e-mailem, které zkontrolujeme a stiskneme **Odeslat** (viz obr. 28).



Obrázek 28: Okno pro odeslání souhrnu e-mailem

(zdroj: Vlastní tvorba)

Struktura kódu se může zdát na první pohled složitá, ale spousta kroků vychází již z předchozích maker. Kód je nahrán v modulu.

```
Sub vytvoření_souhrnu()
'Krok 1: Vytvoření souhrnu z adresáře
'deklarace proměnných
Dim thisSheet As Worksheet
Dim MyPath As String, FilesInPath As String
Dim MyFiles() As String
Dim mybook As Workbook
Dim rnum As Long, Fnum As Long
'nastavení prvního sheetu
Set thisSheet = ThisWorkbook.Worksheets(1)
'cesta k souborům
MyPath = ThisWorkbook.Path & "\Faktury"
'přidat lomítko pokud není
If Right(MyPath, 1) <> "\" Then
MyPath = MyPath & "\"
End If
'pokud zde nejsou Excely opustit sub
FilesInPath = Dir(MyPath& "*.xl*")
If FilesInPath = "" Then
MsgBox "Žádné soubory"
Exit Sub
End If
'naplnit pole (myFiles) soubory Excel ze složky
```

```
Fnum = 0
Do While FilesInPath <> ""
Fnum = Fnum + 1
ReDim Preserve MyFiles(1 To Fnum)
MyFiles(Fnum) = FilesInPath
FilesInPath = Dir()
Loop
'změnit ScreenUpdating, Calculation a EnableEvents
With Application
.Calculation = xlCalculationManual
.ScreenUpdating = False
.EnableEvents = False
End With
rnum = 1
'smyčka všemi soubory v myFiles
If Fnum > 0 Then
For Fnum = LBound(MyFiles) To UBound(MyFiles)
Set mybook = Nothing
Set mybook = Workbooks.Open(MyPath& MyFiles(Fnum))
If Not mybook Is Nothing Then
With mybook.Worksheets(1)
thisSheet.Cells(rnum, 1).Value = .Cells(1, 3).Value
thisSheet.Cells(rnum, 2).Value = .Cells(.Cells.Find("TOTAL
MARKETING BUDGET 2017 AKTIVNĚ ČERPANÝ").Row, 5).Value
End With
mybook.Close SaveChanges:=False
End If
rnum = rnum + 1
Next Fnum
End If
'obnovení ScreenUpdating, Calculation a EnableEvents
With Application
.ScreenUpdating = True
.EnableEvents = True
.Calculation = xlCalculationAutomatic
End With
'Krok 2: Výpočet sumy a formátování tabulky
```

```
thisSheet.Columns.AutoFit
rnum = thisSheet.Cells(thisSheet.Rows.Count, "A").End(xlUp).Row +
1
thisSheet.Cells(rnum, 1).Value = "Celkem"
thisSheet.Cells(rnum, 2).Value = WorksheetFuncti-
on.Sum(thisSheet.Range("B1:B" & rnum - 1))
thisSheet.Rows("1:1").Insert Shift:=xlDown, CopyOri-
gin:=xlFormatFromLeftOrAbove
thisSheet.Range("A1").Value = "Společnost"
thisSheet.Range("B1").Value = "Částka"
thisSheet.Range("A1:B1").HorizontalAlignment = xlCenter
thisSheet.Range(Cells(1, 1), Cells(rnum + 1, 2)).Style = "Výstup"
thisSheet.Cells(rnum + 1, 2).NumberFormat =
"$#,##0.00_);($#,##0.00)"
'přejmenovat sešit
thisSheet.Name = "Souhrn 2017"
'Krok 3: Vytisknutí sešitu
thisSheet.PrintOut Copies:=1
'Krok 4: Odeslání e-mailu
'deklarace proměnných
Dim OLApp As Outlook.Application
Dim OLMail As Object
'otevřít Outlook, začít email
Set OLApp = New Outlook.Application
Set OLMail = OLApp.CreateItem(0)
OLApp.Session.Logon
'vytvořit email a poslat
With OLMail
.To = "marketingdep@zlinreality.cz"
.CC = "accountingdep@zlinreality.cz"
.BCC = ""
.Subject = "Souhrn 2017"
.Body = "Zasílám celkový souhrn za rok 2017"
.Attachments.Add ActiveWorkbook.FullName
.Display
End With
'vyčistit pamět
```

```
Set thisSheet = Nothing
Set mybook = Nothing
Set OLMail = Nothing
Set OLApp = Nothing
End Sub
```

Toto makro nám v praxi ušetří opravdu spoustu času. Jde o celou sérii činností, která se provede pouze jedním klikem.

Pozn.: Cesta cílového adresáře je nastavena na podsložku "Faktury" aktuálně umístěného sešitu. Makro funguje pouze pro jednu složku, pro podsložky již soubory nevyhledává. Řešením je v kódu změnit konkrétní cílový adresář, přesunout sešit do nadřazené složky, nebo sešity ukládat do jedné složky, vhodně pojmenovávat a v kódu upravit, aby vyhledal pouze soubory začínající např. faktura, nebo rokem (2017). Pak v kódu změníme `FilesInPath = Dir(MyPath& "faktura*.xl*")`.

4.1 Ověření využitelnosti řešení v praxi

Ověření konkrétního řešení probíhalo u zaměstnance ve společnosti specializující se na služby v oblasti nemovitostí a facility managementu. Pracovní pozice, kterou zaměstnanec vykonává, je junior marketing manager v marketingovém oddělení. Tomu nasvědčují i tabulky, ze kterých jsem vycházel pro automatizované řešení. Náplní práce zaměstnance v administrativní činnosti je nahrávání podkladů do administrace, řešení účetních dokladů (zadávání a řešení nových incidentů) a objednávek, spárování nových účetních dokladů navedených pod marketing. Připravuje a dodává data k vyhodnocování kampaní vedoucímu marketingového oddělení – např. sběr informací k dané problematice, administrace spojená s vytvářením tabulek v Excelu. S využitím Excelu zaznamenává uskutečňování marketingového plánu nemovitostí z pohledu efektivního plnění marketingových, eventových a PR služeb. Zpracovává nabídky na nemovitosti na základě dodaných podkladů od obchodního oddělení. Zajišťuje kvalitní textový obsah ve všech médiích a tvorbu obsahové stránky marketingu.

S přechodem mých maker do firemního prostředí nebyl žádný problém. Makra jsem vytvářel ve verzi Office 15 a stejnou verzi používají i ve firmě. Navíc makra vytvořená ve VBA jsou v drtivé většině kompatibilní skrz všemi verzemi Office. Mou starostí z počátku bylo, zda mají zaměstnanci na firemních počítačích možnost makra používat. Tedy jestli není změněno výchozí zabezpečení na **Zakázat všechna makra bez oznámení** (viz kapitola

1.6.). To by znamenalo, že bez změny nastavení by využít nešly. Naštěstí takové zabezpečení zde nebylo a makra šla bez problémů povolit, přidat kartu **Vývojář** i nastavit **Důvěryhodná umístění**.

Nejvíce časově náročné bylo vysvětlení maker koncovému uživateli, jelikož s nimi dosud nepřišel do styku. Tedy vysvětlit mu alespoň nejdůležitější prvky karty **Vývojář**, k čemu všemu mohou makra sloužit, jak je spustit a popř. kde si v kódu upravit základní parametry (název listu, předmět, obsah a adresáta e-mailu, cílovou složku atd.).

Na základě aplikování do praxe jsem vyhotovil tabulku výhod a záporů jejích použití (tabulka 2).

Tabulka 2: Výhody a zápory použití maker ve firemním prostředí *(zdroj: Vlastní tvorba)*

Výhody	Nevýhody
Zrychlení činností	Práce s implementací
Snižuje administrativní zátěž	Zaškolení zaměstnanců
Umožní lepší využití zdrojů	Mohou být hrozbou pro počítač
Vyvaruje se lidské chybovosti	
Jednoduchost použití	
Spolupráce mezi aplikacemi MS Office	

Makra nabízejí mnoho výhod těm, kteří se je rozhodli používat. Mezi jeho nejlepší charakteristiky patří rychlost a jednoduchost použití. Snižují možnost vzniku lidské chyby, která se zvyšuje s mnoha opakovanými úhozy a úkoly. Makra zkracují dobu, kterou je potřeba vynaložit při provádění základních úloh, čímž uvolní zaměstnance pro komplexnější řešení problémů a vytváření nápadů. Rovněž usnadňují provádění složitých výpočtů. Snižovat manuální pracovní zatížení. Pracovníka činí účinnějším a produktivnějším. Při používání MS Office ve firmě je velkou výhodou, že makra spolupracují s jinými aplikacemi MS Office.

Nejčastější překážkou při používání Excelu je nedostatek času pro zdokonalování stávajícího řešení a inovace. Často tak firma raději používá starší, časově náročné, manuální řešení a nedokáže plně využít potenciál, který Excel nabízí. Zaměstnance je nutno proškolit,

jak makra plně využívat a v neposlední řadě, mohou být záměrně vytvořena pro vykrádání důvěrných informací, či poškození systému.

4.2 Vyhodnocení úspory času

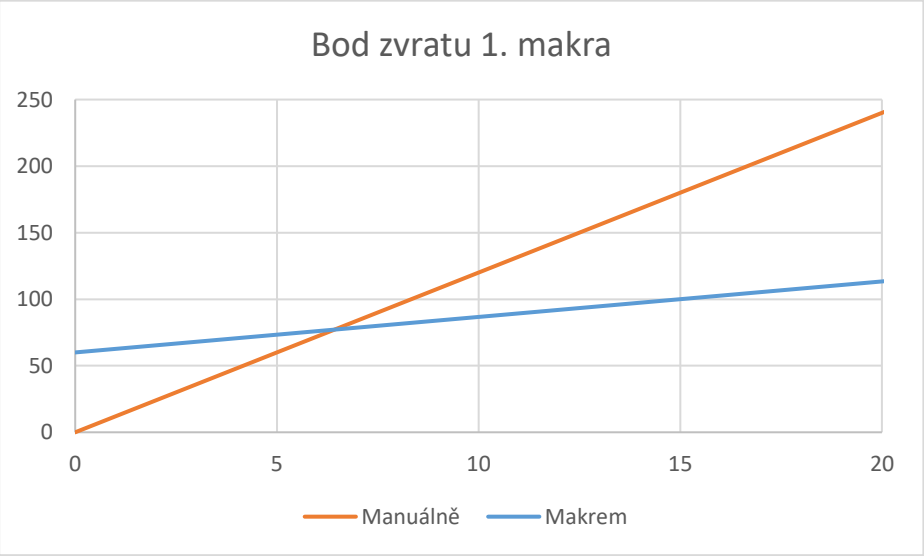
Poslední částí práce je vyhodnocení časové úspory při používání maker. Na základě grafů bude ukázáno, zda se pro potřeby administrativních činností vyplatí makro vytvářet či nikoliv.

Přesné číslo úspory se těžce vyhodnocuje - použití makra je při samotné činnosti vždy rychlejší (jedno kliknutí), než úkon provádět manuálně, ale je otázkou, kolik nám jeho vytvoření zabralo času. Pro tyto účely jsem použil hodnoty, které znázorňují:

Vynaložený čas na tvorbu makra a poté čas provedení konkrétní činnosti s makrem a manuálně. Hodnoty jsou vynásobeny četností těchto operací za měsíc.

1. **Vytvoření nového sešitu**

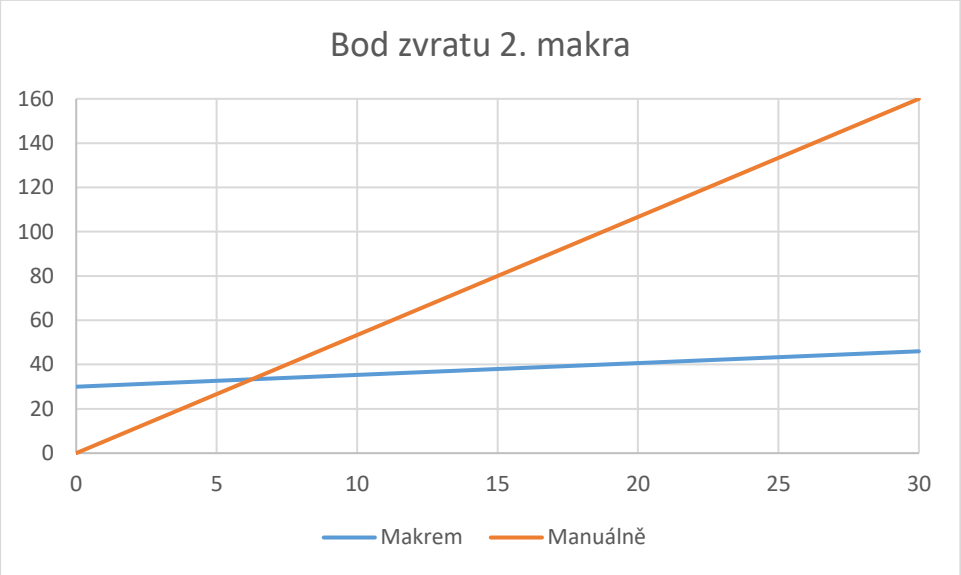
   Tato činnost manuálně vyžaduje následující kroky: označit vybraná data, kopírovat (Ctrl + C), kliknout na Soubor > Nový > Prázdný dokument, vložit data (Ctrl + V), Uložit, Procházet, vybrat umístění, pojmenovat, Uložit, OK. Celkem to zabere zhruba 18 sekund, makrem pouze 4 sekundy. Četnost této operace je 20 krát měsíčně. Jeho tvorba pojala 60 minut.



Graf 1: Bod zvratu pro vytvoření nového sešitu *(zdroj: Vlastní tvorba)*

2. **Uložení sešitu při změně buňky**

Tento krok se provede pouze jedním tlačítkem **Uložit**; ovšem výhoda spočívá především v tom, že se to provádí automaticky, a tím pádem ho uživatel skutečně nezapomene uložit. Délka kroku vychází manuálně 2 sekundy, četnost je zhruba 80 krát do měsíce – makrem 0,2 sekundy, jelikož se činnost provádí sama a tvorba pojala 30 minut.



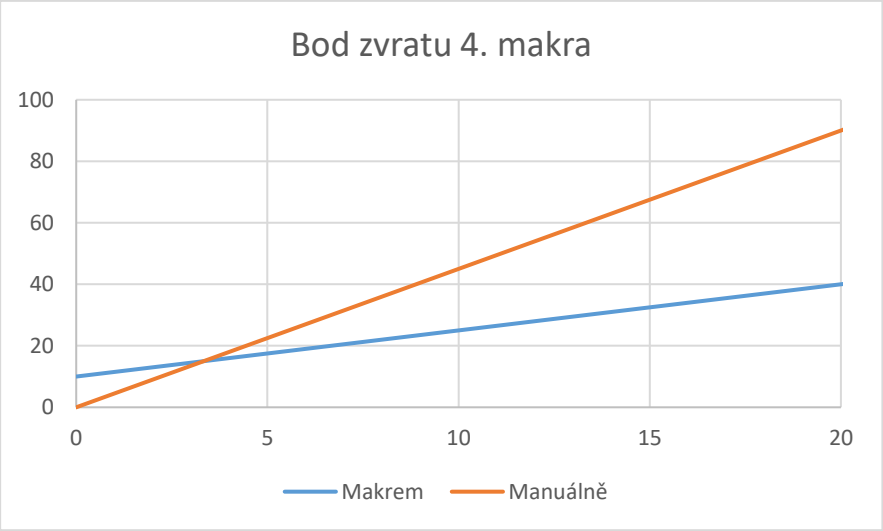
Graf 2: Bod zvratu pro uložení sešitu při změně buňky *(zdroj: Vlastní tvorba)*

3. **Uložení sešitu před uzavřením**

   Tato možnost nám žádný čas neušetří, figuruje zde spíše pro to, aby nedošlo k uzavření sešitu bez jeho uložení. Při uzavření jsme tázání, zda soubor chceme uložit. Provedení 24 krát měsíčně. Trvání 3 sekundy u obou případů. Vynaložený čas na tvorbu byl přibližně 40 minut. K bodu zvratu nikdy nedojde.

4. **Zavření všech sešitů najednou**

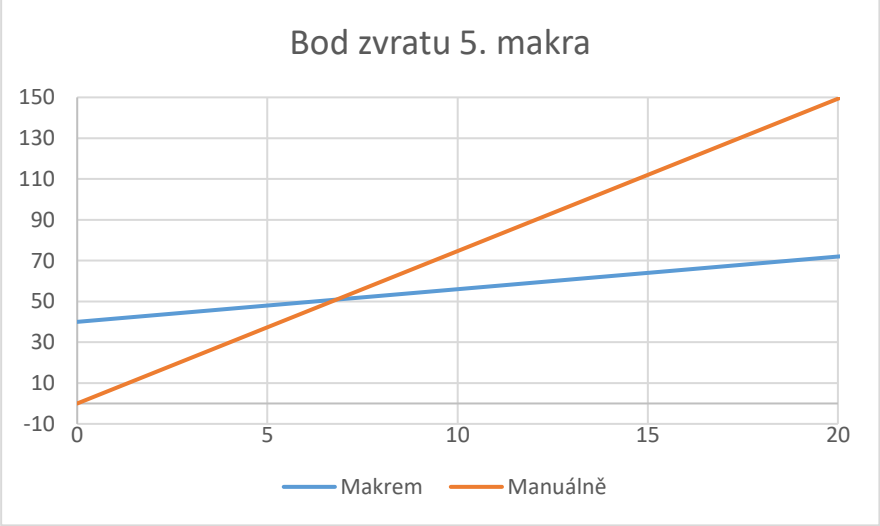
   Tato možnost záleží především na tom, kolik sešitů máme otevřených. U každého rozdělaného sešitu poté musíme manuálně kliknout na křížek a uložit každý soubor zvlášť. Při počtu 4 rozdělaných sešitů tato činnost zabere 9 sekund – makrem 3 sekundy. Četnost úkonu 15 krát měsíčně. Tvorba makra vychází na 10 minut.



Graf 3: Bod zvratu pro zavření všech sešitů *(tvorba: Vlastní zdroj)*

5. **Vytisknout všechny sešity v adresáři**

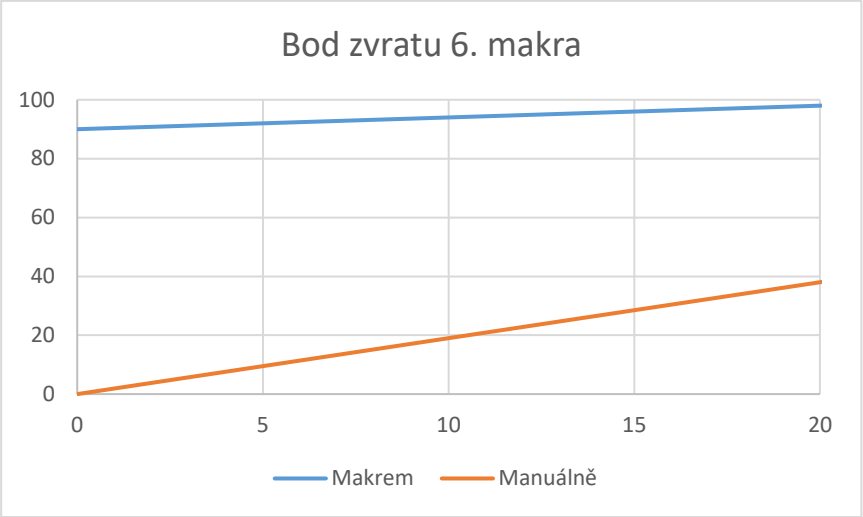
Tady opět záleží, kolik sešitů potřebujeme vytisknout. Čím je jich více, tím více ušetříme makrem času. Při počtu 3 sešitů činnost zabere zhruba 28 sekund - makrem ji provedeme za 6 sekund. Tvorba pojala 40 minut a četnost úkonu je 8 krát do měsíce.



Graf 4: Bod zvratu pro vytisknutí všech sešitů v adresáři *(zdroj: Vlastní tvorba)*

6. **Seřadit listy podle názvu**

Tato činnost se musí provádět manuálně, Excel nenabízí žádnou funkci pro seřazení. Při 6 listech v sešitu zabere aktivita 19 sekund, makrem 4 sekundy. Provádí se 3 krát do měsíce. Vytvoření kódu vzalo 90 minut. Na grafu 6 vidíme, že se nám makro kvůli úspoře času nevyplatí vytvářet. Usnadní nám však orientaci při práci v sešitu.



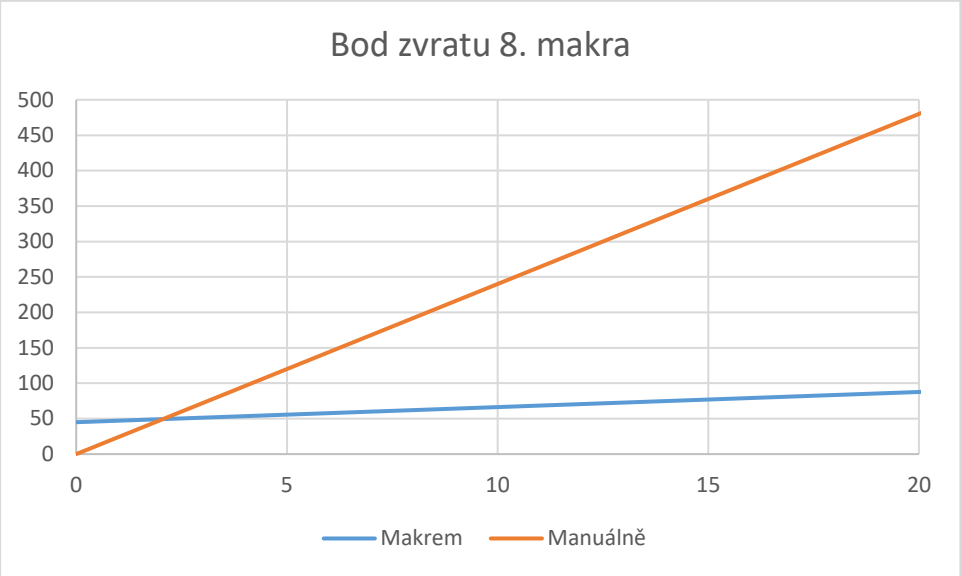
Graf 5: Vyhodnocení makra pro seřazení listů podle názvu *(zdroj: Vlastní tvorba)*

7. **Vytvoření obsahu pracovních listů**

   Plní podobnou funkci jako předchozí seřazení listů. Tento úkon usnadní orientaci v listech a je navíc velmi praktický, ale vyčíslitelnou hodnotu úspory času nám nedá. Zaměstnanec jej využívá maximálně 2 krát za měsíc. Ruční vytvoření obsahu listů je minutová práce – makrem je to 4 sekundy. Tvorba kódu zabrala zhruba 90 minut. Graf by měl stejnou podobu jako předchozí a k bodu zvratu nedojde.

8. **Nahrazení hodnot v rozsahu**

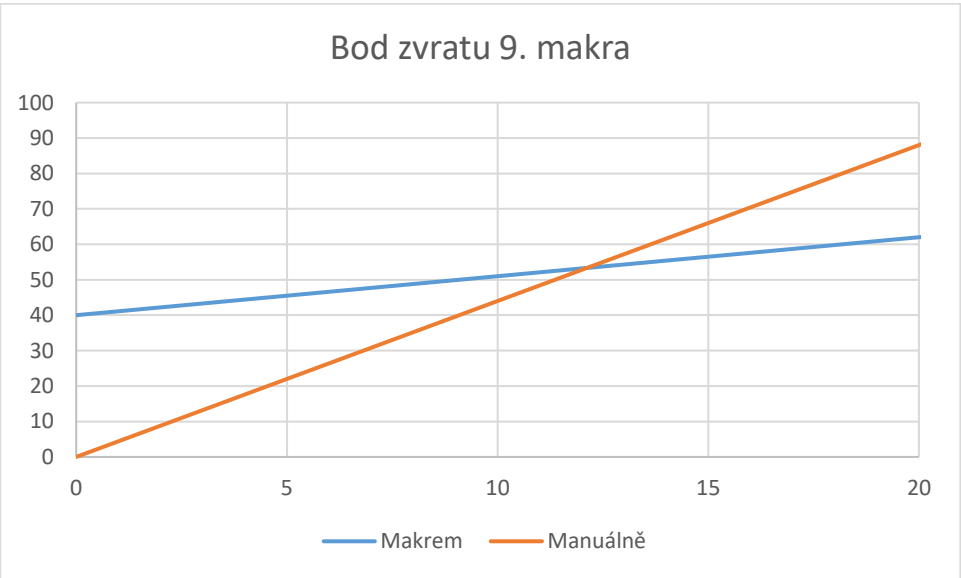
   Tento krok se musí manuálně provést označením buněk > Podmíněné formátování > Pravidla zvýraznění > Rovná se > 0 > OK. Poté buňky odmazat. Opět záleží na tom, jak velkou buňku máme, u vzorové (3.4.1.) tato činnost zabere až 90 sekund, jelikož vymazat buňky se musí ručně. Makro to zvládne za 4 sekundy a jeho vytvoření je záležitostí 45 minut. Tento úkon se provádí 16 krát do měsíce. Na grafu 8 lze pozorovat, že makro se skutečně vyplatí.



Graf 6: Bod zvratu pro nahrazení hodnot v rozsahu *(zdroj: Vlastní tvorba)*

9. **Zvýraznění duplikátů**

Makro plní podobnou funkci jako předchozí s rozdílem, že nic neodmazáváme. Provede se označení buněk > Podmíněné formátování > Pravidla zvýraznění > Duplicitní hodnoty > OK. Úkon pojme 11 sekund, makrem jen vteřiny 3. Vytváření makra vyšlo na 40 minut, krok se používá 12 krát za měsíc.

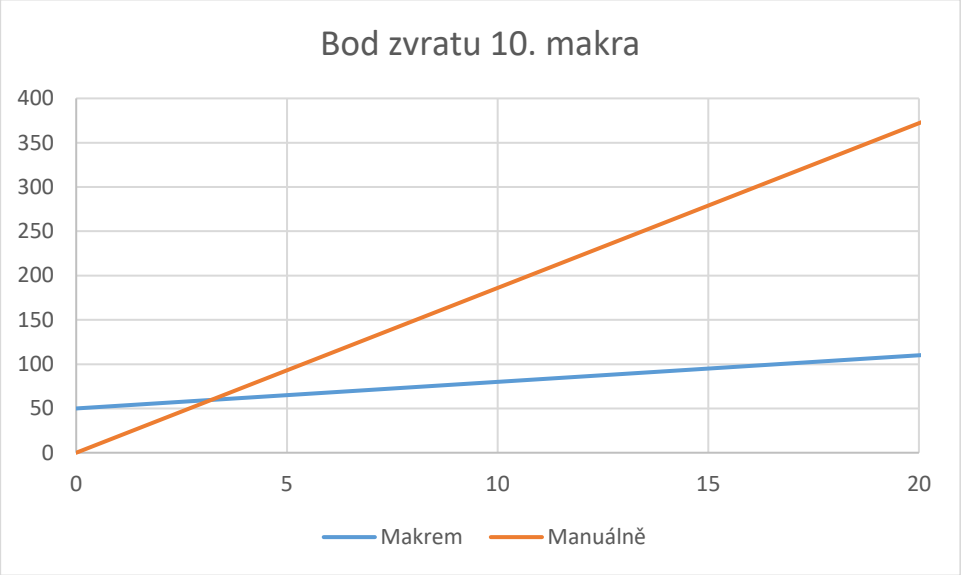


Graf 7: Bod zvratu pro zvýraznění duplikátů *(zdroj: Vlastní tvorba)*

10. **Odeslat sešit jako přílohu**

Krok se musí provádět v aplikaci MS Outlook, tzn. že se musí otevřít > Nový e-mail > Vyplnit adresáta, předmět, obsah zprávy > kliknout na Připojit soubor >Vyhledat > Odeslat a zavřít Outlook. Proces trvá manuálně 31 sekund, z toho nejdelší činností je

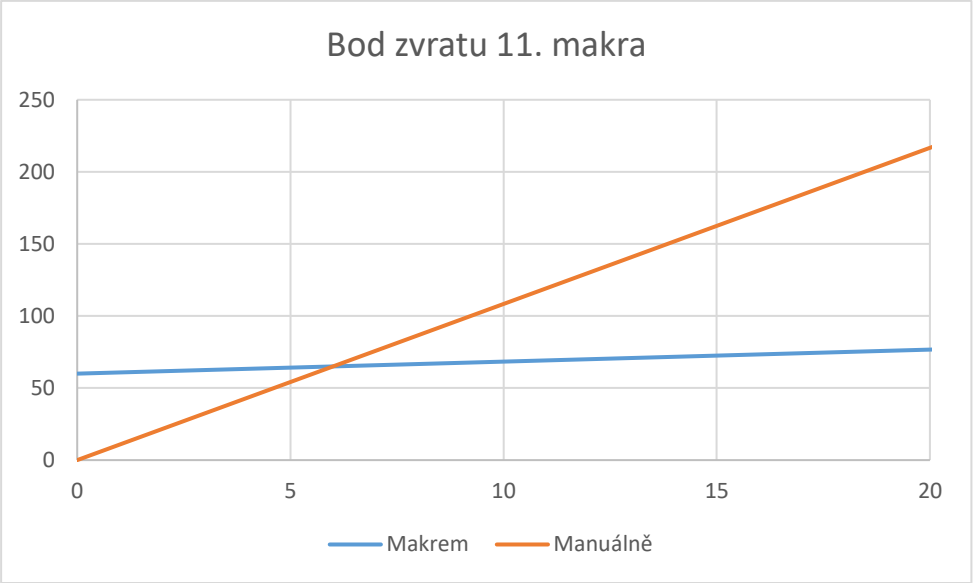
vyplnit pole pro odeslání e-mailu. Makrem stačí potvrdit odeslání tlačítkem OK za 5 sekund. Odesílání sešitu jako přílohu je používáno 18 krát měsíčně, kód byl vytvořen za 50 minut.



Graf 8: Bod zvratu makra odeslání sešitu jako přílohu *(zdroj: Vlastní tvorba)*

11. **Odeslat rozsah jako přílohu**

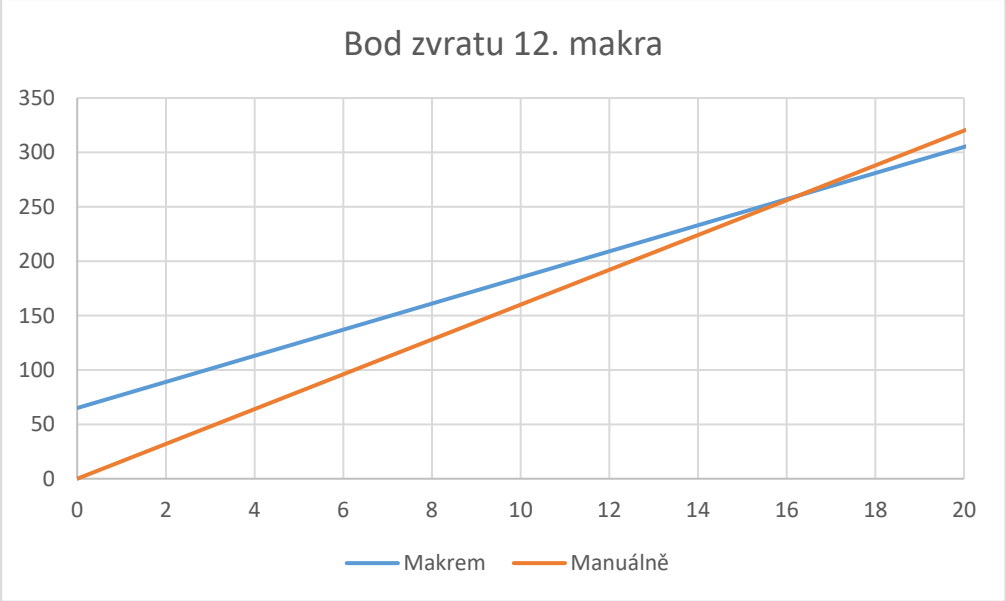
Tato činnost se může jevit jako totožná s předchozí, ale abychom zvládli to, co makro, musíme navíc rozsah zkopírovat do nového souboru, ten uložit, odeslat a následně smazat. Manuální postup je tedy vybrat rozsah tabulky > kopírovat > Otevřít nový sešit > Vložit > uložit si sešit - Uložit, Procházet, vybrat umístění, pojmenovat, Uložit > otevřít MS Outlook > Nový e-mail > Vyplnit adresáta, předmět, obsah zprávy > Kliknout na Připojit soubor > Vyhledat > Odeslat > Zavřít Outlook > Smazat vytvořený soubor. Zvolený zaměstnanec „junior marketing manager“ odesílá rozsah méně než sešit celý – 5 krát za měsíc. Úkon zabere 65 sekund, makrem 5 sekund a vypracován byl za 60 minut.



Graf 9: Bod zvratu pro odeslání rozsahu jako přílohu *(zdroj: Vlastní tvorba)*

12. **Odesílání dat do dokumentu Word**

S touto činností se v praxi pracovalo velice často. Manuálním postupem je vybrat data > kopírovat > otevřít nový Word > vložit data > Uložit, procházet, vybrat umístění, pojmenovat, Uložit. Tyto kroky vyjdou na 16 sekund. Makrem to však také není úplně automatizované - do Wordu nejdříve musíme vložit Záložku, tento proces zabere 12 sekund. Vypracovat makro pojalo 65 minut, v praxi byl pak úkon používán 30 krát za měsíc.



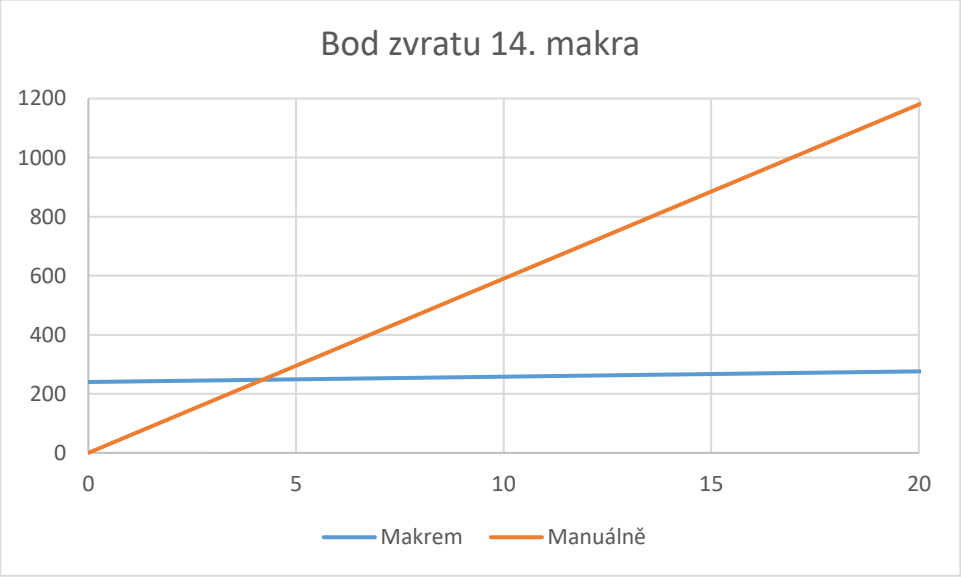
Graf 10: Bod zvratu makra pro odeslání dat do Wordu *(zdroj: Vlastní tvorba)*

13. **Odesílání dat do prezentace PowerPoint**

    Tento úkon je velmi podobný tomu předchozímu. Z výsledků vyplynulo, že přímky se protínají až ve velké vzdálenosti. Tento proces se ve zkoumaných firemních podmínkách téměř nevyužívá, a bod zvratu tady tudíž nenastane. Postup spočívá v krocích, jakými je vybrat data > kopírovat > otevřít nový PowerPoint > Vložit nový snímek > Vložit data > Uložit, Procházet, vybrat umístění, pojmenovat, Uložit. Úkon vychází na 17 sekund, pomocí maker 4 vteřiny a v praxi u zvoleného zaměstnance byl použit pouze jedenkrát. Tvorba pojala 65 minut.

14. **Automatizované řešení**

    Posledním bodem vyhodnocení úspory času je automatizované řešení. Vzhledem k tomu, že je tento úkon komplexnější, bude také zahrnovat více činností. Manuálně si nejdříve v sešitu nachystáme první řádek tabulky a následně otevíráme jeden soubor po druhém (8 sešitů) a hodnoty kopírujeme do tabulky. Po všech překopírovaných hodnotách provedeme sumu druhého sloupce. Tabulku zformátujeme, sešit uložíme a vytiskneme - Soubor > Vytisknout > Tisk. Poté otevřeme Outlook, klikneme na Nový e-mail, vyplníme okno e-mailu, přidáme přílohu a odešleme. Aby výsledek vypadal jako tabulka makra, zabralo by to necelých 5 minut, přesně 295 sekund, zatímco makro to zvládne za 9 sekund. Výroba řešení trvala 240 minut a využita byla 6 krát za měsíc.



Graf 11: Bod zvratu automatizovaného řešení *(zdroj: Vlastní tvorba)*

Z výsledků grafů je jasně vidět, že má smysl vytvářet makra pro úkony, které zabírají hodně času a také ty, které jsou hojně využívány a nezáleží ani tak na investovaném čase do vytváření řešení.

ZÁVĚR

Cílem bakalářské práce bylo seznámit čtenáře se základy tvorby maker a představit základy jazyka Visual Basic. Dalším záměrem práce bylo popsat podrobněji nejdůležitější a nejpoužívanější uživatelské úkony v administrativní práci, při kterých vytvořená makra mohou výrazně rozšířit možnosti tabulkového kalkulátoru Excel, zrychlit a usnadnit práci při jeho každodenním využívání.

Možnosti programu Excel jsou obrovské a tomu odpovídá také rozmanitost prostředků, které jsou v jazyku Visual Basic for Application k dispozici. Programování maker je do značné míry záležitostí praxe. Nejlepší cestou k dobrému zvládnutí jazyka VBA je, stejně jako u většiny lidských činností, především zkušenost. K vytvoření jednoduchého makra je třeba se naučit jen základní syntaxi a lze začít kódovat za 2 hodiny. Je-li však potřeba vytvořit komplikovanější programy, je nezbytné vědět o všech funkcích programu Excel. Tento proces může zabrat až jeden měsíc. Z toho vyplývá, že nejprve je nutno definovat, co potřebujeme, a až poté se začít problematiku učit.

Praxí vytvářením maker bylo prokázáno, že již při začátku tvorby je rozumné dbát na volbu názvů pro makra a proměnné. Při sestavování kódu postupné kroky vhodně komentovat, aby usnadnily jejich pozdější úpravu.

Při použití maker u zaměstnance vykonávajícího administrativní činnost ve firemním prostředí se potvrdilo, že tato především zrychlují činnosti a snižují administrativní zátěž. K dalším výhodám aplikovaných maker patří úspora času, efektivnější využívání lidských zdrojů, předchází lidské chybovosti a navíc jsou jednoduše použitelná. Naopak náročnost spočívala v implementaci maker a v zaškolení uživatele, aby byl schopen makra správně využívat. Rizikovým faktorem je, že makra mohou být záměrně vytvořena pro různá zneužití.

Výsledné šetření bakalářské práce je založeno na srovnání vytvořených grafů. Bylo zjištěno, že se makra v praxi vyplatí především pro úkony zabírající hodně času, či pro činnosti, které se provádí velmi často a opakovaně. Na základě posuzovaných údajů vyplynulo, že makra jsou efektivnější. Vynaložený čas na tvorbu maker nemá na výsledný graf až takovou váhu. Pouze určí, zda se vyplatí makro vytvářet či nikoliv. Konkrétně lze makry optimalizovat práci spojenou s možnou úsporou času především při složitějších úkonech, jako je práce s více sešity najednou, interakce s dalšími aplikacemi MS Office, nebo provádění více posloupných činností.

SEZNAM POUŽITÉ LITERATURY

[2] BENÁČANOVÁ, Helena. Tvorba aplikací v MS Excel. Praha: Oeconomica, 2007. ISBN 978-80-245-1271-6.

[2] BARILLA, Jiří, Pavel SIMR a Květuše SÝKOROVÁ. Microsoft Excel 2016: podrobná uživatelská příručka. Brno: Computer Press, 2016, 456 s. Podrobná uživatelská příručka. ISBN 978-80-251-4838-9.

[3] ČERNÝ, Jaroslav. Excel 2000-2007: záznam, úprava a programování maker. 2., aktualiz. vyd. Praha: Grada Publishing, 2008, 183 s. Průvodce. ISBN 978-80-247-2305-1.

[4] LAURENČÍK, Marek a Michal BUREŠ. Programování v Excelu 2010 & 2013: záznam, úprava a programování maker. Praha: Grada, 2013, 198 s. Průvodce. ISBN 978-80-247-5033-0.

[5] Automate tasks with the Macro Recorder. Microsoft [online]. [cit. 2017-03-03]. Dostupné z: https://support.office.com/en-us/article/Automate-tasks-with-the-Macro-Recorder-974ef220-f716-4e01-b015-3ea70e64937b?ui=en-US&rs=en-US&ad=US

[6] BENÁČANOVÁ, Helena. Tvorba aplikací v Microsoft Office Excel. V Praze: Oeconomica, 2009. ISBN 978-80-245-1602-8.

[7] Přechod z Microsoft Office na OpenOffice.org 2.0: Průvodce pro středně pokročilé a pokročilé uživatele [online]. 2005 [cit. 2017-03-25]. Dostupné z: https://i.iinfo.cz/files/root/k/Prechod_z_Microsoft_Office_na_OOo.pdf

[8] Porting Excel/VBA to Calc/StarBasic [online]. [cit. 2017-04-02]. Dostupné z: http://www.openoffice.org/documentation/HOW_TO/various_topics/VbaStarBasicXref.pdf

[9] Malá makra – úvod do programování. OpenOffice.cz: portál uživatelů OpenOffice a OpenLibre [online]. 2010 [cit. 2017-03-26]. Dostupné z: https://www.openoffice.cz/doplnky/mala-makra-v-openoffice-org

[10] PITHONYAK, Andrew. OpenOffice.org Macros Explained: OOME Third Edition [online]. 2016, , 672 [cit. 2017-03-28]. Dostupné z: http://www.pitonyak.org/OOME_3_0.pdf

[11] Historie hackingu: Vývoj virů v dokumentech. COMPUTERWORLD: FROM IDG [online]. 2016 [cit. 2017-04-02]. Dostupné z: http://computerworld.cz/securityworld/historie-hackingu-vyvoj-viru-v-dokumentech-52846

[12] VirusInfo : The Virus Encyclopedia[online]. 2007 [cit. 2017-04-04]. Dostupné zhttp://virus.wikia.com/wiki/Main_Page

[13] Makroviry jsou zpět!. FEEDIT.CZ [online]. 2014 [cit. 2017-04-04]. Dostupné z: https://feedit.cz/2014/07/24/makroviry-jsou-zpet/

[14] Makra v OpenOffice.org Calc: Petr Ponížil [online]. [cit. 2017-04-10]. Dostupné z: http://www.gymkrom.cz/web/ict/materialy/Makra_OO-Calc.pdf

[15] Vlastnosti VBA. LibreOffice Help [online]. [cit. 2017-04-10]. Dostupné z: https://help.libreoffice.org/Common/VBA_Properties/cs#Spustiteln.C3.BD_k.C3.B3d

[16] Formátování buněk - kódy Excel VBA. Microsoft Office: ..ať pracuje za vás.. [online]. [cit. 2017-04-15]. Dostupné z: http://office.lasakovi.com/excel/vba-listy-bunky/formatovani-bunek-excel-vba/

[17] Formatting a Range of Cells In Excel VBA. *Excel How To* [online]. [cit. 2017-04-25]. Dostupné z: http://www.excelhowto.com/macros/formatting-a-range-of-cells-in-excel-vba/

[18] ALEXANDER, Michael T. 101 ready-to-use excel macros (mr. spreadsheet). Indianapolis, IN: Wiley Pub., 2012. ISBN 1118281217.

[19] LIEW VOON KIONG. Excel VBA made easy: a concise guide for beginners. Lexington, KY: Liew Voon Kiong, 2009. ISBN 1449959628.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

MS Microsoft (společnost)

VBA Visual Basic for Applications (programovací jazyk od Microsoftu)

VBE Visual Basic Editor (editor v MS Office pro práci s kódem VBA)

XML eXtensible Markup Language (rozšiřitelný značkovací jazyk)

AOO Apache OpenOffice (název kancelářského balíku)

OO OpenOffice

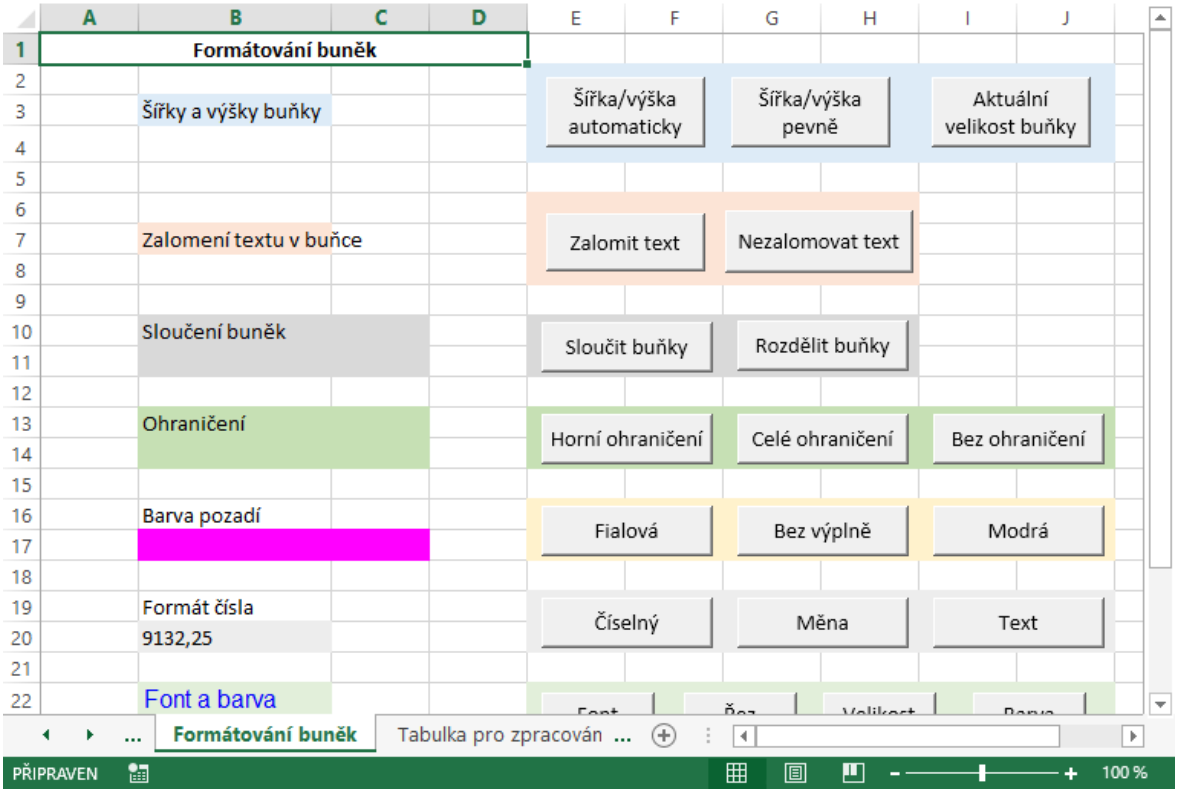
SEZNAM OBRÁZKŮ

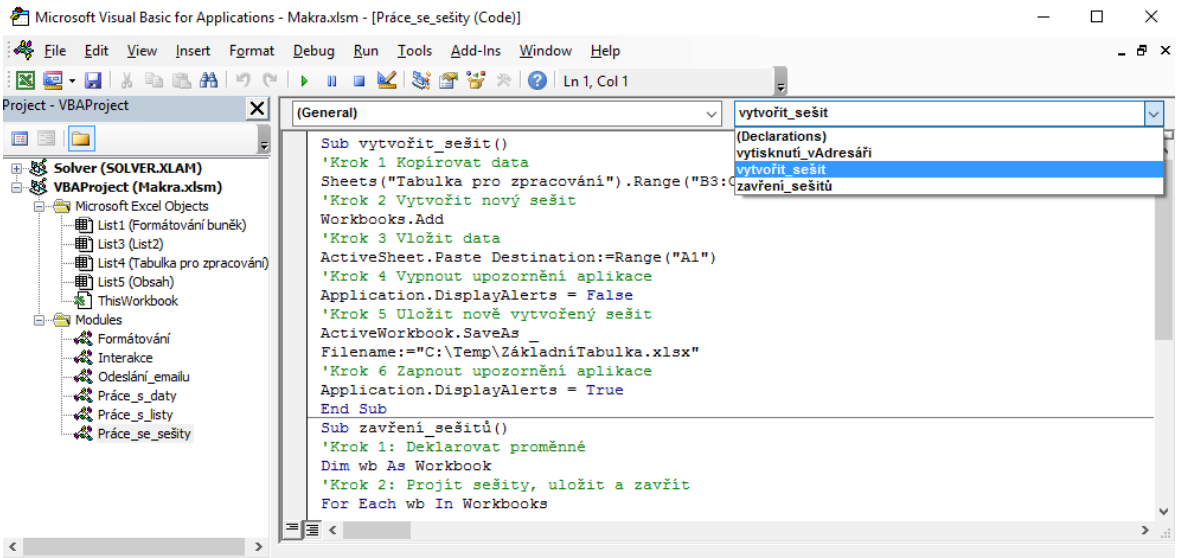
SEZNAM TABULEK

SEZNAM GRAFŮ

SEZNAM PŘÍLOH

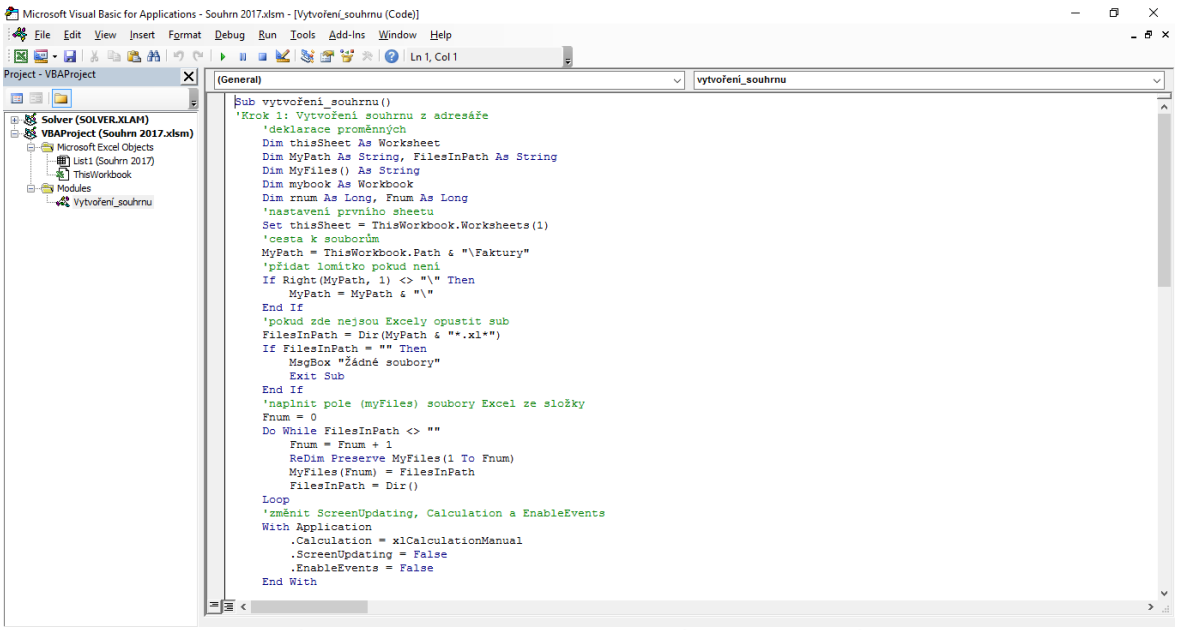
PŘÍLOHA P I: SOUBOR MAKRA.XLSM





Zdroj: Vlastní tvorba

PŘÍLOHA P II: SOUBORSOUHRN 2017.XLSM



	A	B
1	Společnost	Částka
2	BRNO - 2017	35 000,00 Kč
3	CENTRAL - CORPORATE - 2017	242 500,00 Kč
4	GLOBAL - 2017	23 133,33 Kč
5	OLOMOUC - 2017	157 556,00 Kč
6	PRAHA 3 -2017	99 656,00 Kč
7	PRAHA 7 - 2017	163 556,00 Kč
8	ZLÍN 1 - 2017	63 167,00 Kč
9	ZLÍN 2 - 2017	63 600,00 Kč
10	Celkem	848 168,33 Kč

Souhrn 2017

Zdroj: Vlastní tvorba

PŘÍLOHA P III: ZDROJOVÉ TABULKY PRO SOUHRN 2017.XLSM

Název	Typ	Velikost
BRNO - 2017	List aplikace Microsoft Excel	15 kB
CENTRAL - CORPORATE - 2017	List aplikace Microsoft Excel	17 kB
GLOBAL - 2017	List aplikace Microsoft Excel	17 kB
OLOMOUC - 2017	List aplikace Microsoft Excel	18 kB
PRAHA 3 -2017	List aplikace Microsoft Excel	17 kB
PRAHA 7 - 2017	List aplikace Microsoft Excel	18 kB
ZLÍN 1 - 2017	List aplikace Microsoft Excel	18 kB
ZLÍN 2 - 2017	List aplikace Microsoft Excel	17 kB

BRNO

POLOŽKY	POZNÁMKA 1	POZNÁMKA 2	Cena: CENA	Cena: % PODÍL	Cena: CENA - PODÍL	Celkem: POČET	Celkem: CELKEM	KVĚTEN 8.5.	KVĚTEN 15.5.	KVĚTEN 22.5.	KVĚTEN 29.5.
ON-LINE											
WEB											
WEB - REKONSTRUKCE			25 000		25 000	1,0	25 000				
PPC + SEO											
SEO/ ON PAGE			2 500		2 500	1,0	2 500				
TISK PLAKÁTŮ / PLACHET											
							0				
						0	0				
TISK PLAKÁTŮ / PLACHET > CELKEM							0				
MARKETINGOVÉ NÁSTROJE											
BROŽURA - skládačka // PRINT REZERVA			35 000		35 000	1	35 000				
MARKETINGOVÉ NÁSTROJE > CELKEM							35 000				
PODPORA OBCHODU											
							0				
							0				
PODPORA OBCHODU> CELKEM							0				

TOTAL MARKETING BUDGET 2017 AKTIVNĚ ČERPANÝ	35 000 Kč
TISK PLAKÁTŮ / PLACHET > CELKEM	0 Kč
MARKETINGOVÉ NÁSTROJE > CELKEM	35 000 Kč
PODPORA OBCHODU> CELKEM	0 Kč

BRNO

Zdroj: Vlastní tvorba