Microsoft Excel, Borland C++ Builder, Delphi

Chytrost nejsou žádná kouzla

Abychom nebyli stále monotematičtí, a tedy nepsali pouze o produktech Microsoftu, rozšířili jsme červencové tipy a triky o dva příspěvky pro vývojáře používající prostředí Borland C++ Builder a Delphi.

V Excelu můžeme zapisovat, formátovat (tzn. zobrazovat) nejen čísla, ale i datum a čas, a to vícero způsoby. Díky jejich vnitřnímu uložení lze s nimi provádět snadno výpočty a výsledek zobrazit v požadovaném tvaru.

Práce s datem a časem v Excelu 2000 cz

Datum a čas formátujeme v buňce na listu sešitu příkazem FORMÁT|BUŇKY, kdy na kartě "Číslo"

vybereme druh "datum" nebo "čas" a ze seznamu vhodný předdefinovaný formát. Je-li v buňce zapsaný nějaký údaj, zobrazí se na kartě "Číslo" v oddíle "Ukázka" jeho podoba pro vybraný formát. Můžeme však vybrat i druh "vlastní" a do políčka "Typ" zapsat pomocí kódů vlastní formátování. Některé kódy jsou uvedeny dále.

Před formátováním může být označena jak jedna buňka, tak celá, i nesouvislá oblast buněk, a to i na několika listech. Označené buňky jsou potom zformátovány, je upraveno zobrazení v nich zapsaných hodnot. Pokud na vyznačené oblasti buněk stiskneme pravé tlačítko myši a z místní nabídky vybereme příkaz FORMÁT BUNĚK… nebo zadáme klávesovou zkratku Ctrl+Shift+1 (jedničku nad klávesami s písmeny), zobrazí se totéž dialogové okno jako v prvém způsobu.

Pro formátování dat a času lze použít kódy: d (kód pro den, day), m (pro zobrazení měsíce, month), y (pro rok, year, v předchozí verzi to byl kód "r"). Pro čas jsou potom vyhrazena písmena h (pro vyjádření hodiny, hour), m (minuty, minute) a s (pro sekundy, second). Písmeno m je jak pro měsíce, tak pro minuty. K záměně nemůže dojít, neboť před minutou musí být vždy kód h.

Interně je datum vyjádřeno počtem dnů od začátku kalendářního systému a tím je pro PC 1. 1. 1900 a pro počítače Macintosh 2. 1. 1904. Čas je vyjádřen interně jako zlomek dne. Čas 0:00:00 má hodnotu 0,0 a 24:00:00 potom 1,0. Tedy např. datum a čas zapsané jako 1.1.2000 12:00:00 jsou vyjádřeny číslem 36526,50. Tento způsob se používá již od "dávnověku" tabulkových procesorů (tehdy např. již od prvních verzí zaniklého kalkulátoru SuperCalc). Tento způsob umožňuje s datem a časem počítat.

Zapíšeme-li kód pro formátování příslušné části dne dvakrát za sebou, znamená to, že číslice od jedné do deseti bude uvedena s vodicí nulou. Např. po zápisu kódu dd.mm.yyyy a zápisu data 1.1. 2000 se v buňce zobrazí 01.01.2000. Zapíšeme-li však kód jen jednou, nebude vodicí nula doplněna. Zobrazí se tedy 1.1.2000.To je nejčastější způsob. Po zápisu data dojde k implicitnímu zformátování na d.m.yyyy.

Formátování data

V češtině by se měla mezi částmi data psát mezera. Implicitně se mezera nezobrazuje. Toho můžeme dosáhnout použitím mezery mezi částmi kódu pro datum. Např. místo d.m.yyyy zapíšeme d. m. yyyy, a potom se po zápisu, např. 21.1.2000, zobrazí 21. 1. 2000.

Uvedeme-li kód pro formátování za sebou třikrát, je rozdíl u kódu dnů a měsíců. U dnů se zobrazí dvouznaková zkratka dne v týdnu: po, út, st, … Při použití u měsíce se měsíc zobrazí římskými číslicemi. Tak např. při kódování ddd. mmm. yyyy se pro zapsané datum 1.4.2000 zobrazí so. IV. 2000.

Uvedeme-li znak čtyřikrát, zobrazí se plný název dne v týdnu a plný název měsíce. Tedy např. sobota duben 2000, zapíšeme-li kód při formátování dddd mmmm yyyy do buňky 1.4.2000.

U dnů se větší počet znaků než čtyři redukuje na čtyři. U měsíců se při pěti znacích zobrazí první písmeno měsíce, tedy l, ú, b, d, … Více než pět znaků se zredukuje na čtyři znaky "m".

U roku lze zapsat pouze yy nebo yyyy. Tedy náš letopočet lze zkrátit na "00" (což může vzbuzovat asociaci malé místnůstky) nebo v úplném znění "2000". Zapíšeme-li jen "y", dojde k automatickému doplnění na "yy", při třech znacích se tyto znaky změní na "yyyy" a pro více než čtyři znaky jsou tyto znaky automaticky redukovány na čtyři.

Pořadí kódů pro formátování data můžeme podle potřeby zaměnit a doplňovat další oddělovací znaky. Lze tak např. zapsat yyyy/mm/d a při zápisu 1.4.2000 se zobrazí 2000/04/1. nebo

při formátovacím kódu mmmm yyyy, dddd se pro zapsané datum 1.4.2000 zobrazí duben 2000, sobota.

Kódy pro formátování můžeme použít i opakovaně. Tak např. po zápisu d. mmm. yyyy (dddd) obdržíme v buňce po zápisu data 1.4.2000 hodnotu 1. IV. 2000 (sobota). Můžeme též odkázat na buňku obsahující datum a ve zformátované buňce potom obdržíme požadovaný tvar. Viz obrázek.

Poznámka. V předchozích verzích Excelu byl pro rok použit kód “r”. Změna může v některých situacích způsobit problémy, neboť se místo roku zobrazí písmena rrrr. Údaj musíme přeformátovat. Ve většině případů dochází k automatické konverzi.

Formátování času

U formátování času je situace obdobná. Jeden formátovací znak – před číslem není vodicí nula a dva znaky určují, že před číslicí od jedné do deseti je vodicí nula. Příklad: buňku zformátujeme kódem h:mm:ss a po zápisu 5:5:5 obdržíme 5:05:05. Jako oddělovač je nejčastěji použita dvojtečka, méně často pomlčka. Lze však použít i jiný znak.

Čas můžeme ještě doplnit kódy dop./odp., resp. AM/PM, am/pm. Obdržíme čas doplněný zkratkou, zda jde o dopoledne (dop.), nebo odpoledne (odp.). Tedy dvanáctihodinový cyklus. Anglické zkratky se automaticky konvertují na český text.

Formátovat datum a čas lze více méně libovolně. Zápis je však kodifikován. Den můžeme zapsat s oddělovači: “.” (tečka), “/” (lomítko) nebo “-“ (pomlčka) a čas zapisujeme pouze s oddělovači “:” (dvojtečka), např. 9:25:55. Jiný zápis vede k zápisu data do buňky jako textu. Pokud při zápisu zapíšeme mezi částmi data mezeru, je implicitně odstraněna a datum se zobrazí bez mezer.

Zapíšeme-li čas běžným způsobem, dojde k implicitnímu zformátování na h:mm:ss. Použijeme-li při zápisu číslice pro hodiny do 12, můžeme zápis doplnit o určení úseku. Za zapisovaný čas doplníme dop., resp. odp.

U sekund lze počítat na zlomky sekund, uvedeme-li za kód “s” čárku a nuly. Např. h:mm:ss,00. Zachytíme tak setiny eskundy. Po zápisu např. 0:5:7,14 se při formátování kódem h:mm:ss,00 zobrazí čas 0:05:07,14.

Při zápisu zlomků sekund dojde k implicitnímu zformátování podle kódu mm:ss,0. Je to logické, na zlomky se bude většinou počítat jen u minut. Znamená to však, že při zápisu hodin nebudou hodiny zobrazeny. Např. 1:25:50,25 se zobrazí jako 25:50,2. Zlomek 0,05 se zaokrouhluje směrem dolů. Pouze v situaci, kdy předchází pětka, např. 5,25, dojde k zaokrouhlení na 5,3.

To byly základní kódy pro formátování času. Lze však použít i kódy uvedené do hranatých závorek. Můžeme zapsat [h], [m] a [s]. Kódy v hranatých závorkách nepřevádí hodnoty na dvanáctihodinový, šedesátiminutový nebo šedesátisekundový cyklus. Zde však jsou pravidla:

· nelze zapsat dva tyto kódy vedle sebe,

· nelze napsat kód v hranatých závorkách a před ním kód pro čas bez hranatých závorek,

· můžeme zapsat kód v hranatých závorkách a za ním kód bez hranatých závorek.

Nemůžeme tedy buňku zformátovat na [h]:[mm]:[ss] nebo h:[mm], ale např. na [h]:mm:ss, [m]:ss, [h], [m], [s]. Viz obrázek.

Zapsaný údaj se převede na celý počet zadaných jednotek. Tedy 1:00:00 (1 hodina) bude při kódu [m] zobrazena jako 60 a při kódu [s] jako 3600.

V situaci, kdy potřebujeme zapsat datum nebo čas jako text, musíme před zápisem buňku zformátovat na text (FORMÁT|BUŇKY, karta “Číslo”, druh ”text”). Použijeme-li vlastní formátování, zapíšeme jako kód “zavináč” “@”.Ten lze zapsat Alt sekvencí Alt+064.

Vzhledem k interní konverzi textu na čísla lze v Excelu i s takto zapsanými daty počítat. Např. lze vypočítat rozdíl dat. U času zapsaného jako text obdržíme po výpočtu zlomek. Ten však můžeme zformátovat např. na [s] a obdržíme počet sekund mezi oběma časovými hodnotami.

Abychom nemuseli formáty stále zapisovat, můžeme je kopírovat. Pro kopírování lze použít tyto postupy:

· Na zdrojové buňce zadáme kombinaci kláves Ctrl+C a v cílové buňce zadáme příkaz ÚPRAVY|VLOŽIT JINAK|FORMÁTY. Příkaz lze použít za sebou opakovaně.

· Na zdrojové buňce stiskneme tlačítko “Kopírovat formát” v panelu nástrojů “Formát” a na cílové buňce stiskneme levé tlačítko myši. Poklepáním na tlačítko “Kopírovat formát” můžeme potom klepnout postupně na více cílových buněk – dokud nestiskneme levé tlačítko myši znovu nebo klávesu Esc.

Několik slov k funkcím pracujícím s časem

Výše uvedené kódy lze použít jako argumenty ve funkci HODNOTA.NA.TEXT.

Chcete vědět, který den jste se narodili? Nic snazšího. Do jedné buňky, např. B2, napište svoje datum narození a do druhé funkci odkazující na tuto buňku. Zapište =HODNOTA.NA.TEXT(B2;"dddd"). Pro zapsané datum vrátí funkce den v týdnu.

Nakdy připadne Silvestr letošního roku? Můžeme zapsat přímo =HODNOTA.NA.TEXT("31.12.2000";"dddd"), funkce vrátí "neděle".

Interní kódování dne a hodin umožňuje spočítat délku trvání nějaké akce.

Např. do buňky B3 zapíšeme začátek akce a do buňky C3 konec akce. Jak dlouho bude akce trvat? Stačí do buňky, např. D3, zapsat =C3-B3, a máme počet dnů mezi oběma daty. Výsledek však musíme zformátovat na číslo. Jinak se nám zobrazí podivné datum kolem roku 1900. Již víme proč; číslo se převedlo na počet dnů od začátku kalendářního systému.

Tak např. kolik dnů máme na projekt zahájený 1.4.2000 (zapíšeme do buňky B3) a ukončený 30.4.2000 (v buňce C3). Vzorec =C3-B3 vrátí 29.1.1900 a po zformátování na číslo (FORMÁT|BUŇKY, karta "Číslo", druh "číslo") vrátí 29,00.

Kolik to ale bude pracovních dnů? Nic snazšího, použijeme funkci NETWORKDAYS. Zapíšeme =NETWORKDAYS(B3;C3) a obdržíme 20 dnů. Ještě to však není ono, 24. 4. 2000 je volno. Funkci tedy upravíme na =NETWORKDAYS(B3;C3;"24.4.2000"), a nyní je již výsledek "OK".

Nebo jiný výpočet. Začneme na projektu pracovat 1. 4. 2000 a máme na něj 45 pracovních dnů. Kdy skončíme? Zapíšeme funkci =WORKDAY(B3;45). Funkce vrátí číslo 36679. Výsledku se již nelekneme, zformátujeme ho na datum ((FORMÁT|BUŇKY, karta "Číslo", druh "datum" a vybereme typ vracející den, měsíc a letopočet) a obdržíme 2.6.2000. Opět chyba, mezi začátkem a koncem jsou tři svátky. 24. 4., 1. 5. a 8. 5. Tak tedy upravíme funkci. Dva a více argumentů však již musíme zapsat do složených závorek. Funkce bude vypadat: =WORKDAY(B3;45; {"24.4.2000";"1.5.2000";"8.5.2000"}) a nyní obdržíme 7. 6. 2000.

Výpočty s časovými údaji můžeme použít např. pro výpočty výsledků soutěže. Do jednoho sloupečku vedle startujících napíšeme čas startu, ten je např. po pěti minutách, a do další buňky zapíšeme čas v cíli. Výpočet je již jasný, do čtvrtého sloupečku uvedeme rozdíl. A po seřazení tabulky podle výsledku výpočtu máme hned pořadí. A můžeme, takřka ihned, vyhlásit vítěze – a zarmoutit "hlemýždě".

Milan Brož

Borland C++Builder, Delphi

Rychlost kreslení

Nedávno se na mne obrátil jeden z čtenářů, který podle knihy o C++Builderu napsal program pro kreslení fraktálů. Pokud jde o zdrojový kód, prakticky se nelišil od uveřejněné verze, běžel však asi 10x pomaleji.

Po jistém pátrání se ukázalo, že příčinou byla vlastnost Autosize komponenty Image, která obsahovala obrázek. Zmíněný čtenář nastavil už v době návrhu vlastnost Autosize na true a tím způsobil, že se po vykreslení každého bodu přepočítávala velikost obrázku.

Poznamenejme, že nástroje pro kreslení jsou v C++Builderu a v Delphi zapouzdřeny do tzv. canvasu (plátna). To je třída, která nabízí mj. metody pro vytvoření úsečky, kružnice apod. Vedle toho ale obsahuje i vlastnost Pixels, což je dvourozměrné pole jednotlivých grafických bodů kreslicí plochy. Kreslení po jednotlivých pixelech není příliš obvyklé, je ale možné a v některých případech – jako třeba při kreslení fraktálů – je nezbytné. Při kreslení po jednotlivých bodech obvykle známe velikost obrázku předem, takže je zbytečné dávat vlastnosti Autosize hodnotu true.

Zaškrtnutí komponenty CheckBox

Jednou z běžných součástí programů pro Windows je zaškrtávací políčko (checkbox). V Delphi a v C++Builderu je reprezentováno komponentou TCheckBox.

Chceme-li zjistit, zda je některé políčko zaškrtnuté, stačí zjistit hodnotu jeho vlastnosti Checked. Stejně lze programově změnit stav políčka, stačí přiřadit této vlastnosti podle potřeby hodnotu true (zaškrtnuté) nebo false (volné).

Klepneme-li na tuto komponentu myší, její stav se přepne a zároveň nastane událost OnClick; to asi nikoho nepřekvapí. Zajímavé ale je, že událost OnClick nastane i v případě, že stav políčka změníme programově, přiřazením hodnoty vlastnosti Checked. Událost OnClick ovšem nastane pouze v případě, že přitom dojde ke změně stavu políčka.

To ale znamená, že pokud bychom chtěli měnit stav políčka programově v handleru, který reaguje na událost OnClick, mohl by vzniknout nekonečný cyklus. Typickým příkladem může být následující ukázka:

```
void __fastcall TForm1::CheckBox1Click(TObject *Sender)
{
  CheckBox1->Checked =!CheckBox1->Checked;
}
```

Zde při události OnClick měníme stav políčka na opačný, tedy snažíme se naprogramovat to, co políčko samo již umí. V tomto případě bude políčko po klepnutí chvíli blikat a pak se program zhroutí pro vyčerpání zásobníku: Klepnutí na toto políčko způsobí změnu stavu a vyvolá se událost OnClick. Program zavolá handler, který způsobí změnu stavu a vyvolá událost OnClick, což způsobí změnu stavu atd.

Borland C++Builder

Deklarace friend

Borland C++Builder 4 a 5 nedokáže za jistých okolností správně přeložit deklaraci spřátelené funkce (friend). Můžeme se s tím setkat v situaci, kdy jednu funkci – např. f() – deklarujeme jako spřátelenou ve dvou různých třídách, např. C a D, a v těle jedné z nich zapíšeme i definici f(). Překladač pak nebude v těle f() znát formální parametry:

```
// Soubor pok.cpp
class C
{
public:
  friend int f(int); // !
};

class D
{
public:
    friend int f(int e)
    {
        return e; // ?
    }
};
```

Na řádku označeném otazníkem v komentáři ohlásí překladač naprosto nesmyslnou chybu

```
[C++ Error] pok.cpp(12): E2451 Undefined symbol 'e'
```

Přitom na skutečných identifikátorech tříd, funkce a jejích parametrů zřejmě nezáleží.
Tento problém má snadné řešení – stačí definovat funkci f() mimo tělo třídy C.

```
class C
{ /* stejné jako předtím */ };

class D
{
public:
    friend int f(int e);
};

inline int f(int e)
{  return e; }
```

Ovšem i původní varianta je syntakticky správná a konkurenční překladače (IBM VisualAge C++ 4.0, Microsoft Visual C++ 6.0, Watcom C++ 11) ji bez problémů přeloží.

Na závěr bych rád poznamenal, že uvedený příklad sice vypadá jako programátorské zvěrstvo, ale vznikl zjednodušením skutečného programu, ve kterém šlo o přetížený operátor +, který měl za úkol sečíst instance dvou různých tříd.