Příprava aplikace k šíření

Tato kapitola shrnuje témata týkající se přípravy aplikace k distribuci a snadné instalaci na cílovém počítači.

Nebudeme se zde zabývat umísťováním aplikací na Internetu nebo WWW, protože tato specifická témata jsou pojednána v jiných kapitolách.

Postup přípravy aplikace

Autor aplikace postupuje takto:

1. vyvine a odladí aplikaci;
2. *exportuje* ji do souborů na disketě nebo jiném přenosném médiu;
3. podle potřeby upraví *definiční soubor aplikace*;
4. pokud chce spojit instalaci aplikace s instalací provozního systému **WinBase602**, připraví upravené diskety provozního systému.

Na co nezapomínat při vývoji aplikace

Vývojář se obvykle soustřeďuje na základní funkci aplikace. Existuje řada otázek, které často zůstávají na okraji pozornosti, ale před vytvořením definitivní verze aplikace musejí být zodpovězeny. Zde jsou některé z nich.

Jaké skupiny uživatelů budou s aplikací pracovat a jaká budou mít oprávnění?

Aplikace zpravidla má několik úrovní uživatelských oprávnění. Pro tuto úrovně by autor aplikace měl definovat *role*, aby pozdější správce provozu aplikace mohl do nich obsadit reálné uživatele a jejich skupiny a tím jim jednoduše přidělit potřebná oprávnění. Podrobněji se o tomto tématu píše v manuálu *Příručka správce*, v kapitole *Správa uživatelů a jejich práv*.

Jak je v aplikaci vyřešeno zálohování dat?

Součástí aplikace by měly být jednoduché nástroje, které zajišťují, případně i pravidelně vynucují zálohování dat. Nelze od uživatelů aplikace žádat, aby prováděli manuální export dat z tabulek, tyto akce musí být integrovány mezi správcovské nástroje aplikace. Aplikace musí být schopna obnovit svoji funkci ze záložních dat, pokud došlo k havárii.

Alternativou k integraci zálohování do aplikace je zálohování celé databáze. Zpravidla se lepší vyřešit zálohování již na úrovni aplikace, neboť databáze může obsahovat mnoho různých aplikací se zcela odlišnými požadavky na zálohování a obnovování. Podrobnosti

o zálohování a obnovování dat naleznete v *Příručce správce*, v *kapitole Provozní bezpečnost*, a také v *Příručce uživatele* v kapitole *Údržba a zabezpečení databáze*.

Jak aplikace zachází se zrušenými záznamy?

Pokud v aplikaci vzniká velké množství zrušených záznamů a není žádoucí, aby zbytečně zabíraly místo v databázi, je třeba zvolit některou z alternativ, jak s nimi zacházet:

- Provádět periodické uvolňování zrušených záznamů v těch tabulkách, kterých se problém týká, voláním funkce **`(cd_)Free_deleted`**. To může aplikace dělat automaticky nebo může být tím pověřen její správce.
- Provozovat aplikaci v režimu, v němž je každý zrušený záznam na konci transakce ihned uvolněn. Tento způsob práce lze nastavit pomocí funkce **`(cd_)Set_sql_option`**.
- Vytvořit v aplikaci vlastní zprávu neplatných záznamů. Záznamy pak místo rušení zneplatňovat a recyklovat.

Speciální vlastnosti aplikací

Omezení počtu uživatelů pracujících s aplikací

Do aplikace je možno zahrnout omezení počtu uživatelů, kteří s ní mohou současně pracovat. Toto lze využít k vytvoření různých verzí aplikace pro různý počet uživatelů a případně k implementaci správy licencí k aplikaci.

Funkce **`(cd_)Appl_inst_count`** vrací počet klientů, kteří pracují se stejnou aplikací jako klient, který funkci zavolal.

Přehled o klientech na serveru

Některé aplikace potřebují mít přehled o klientech přihlášených na databázový server. Tyto údaje lze získat voláním funkce **`(cd_)Get_logged_user`**. Pro každého uživatele lze zjistit jeho logovací jméno, se kterou aplikací pracuje a v jakém je právě stavu.

Kontrola integrity databáze z aplikace

Součástí aplikace může být také provedení kontroly integrity databáze. Nejde o funkci, kterou by bylo vhodné rutinně volat, může být však uživateli aplikace k dispozici, aby například mohl zjistit rozsah škod po selhání hardware na serveru. K tomu slouží funkce **`(cd_)Database_integrity`**.

Zvětšování databázového souboru

Aplikace, v níž neustále přibývají data, může mít integrovanou funkci pro zvětšování databázového souboru. Stejnou operaci lze provést i z provozního prostředí, ale její zavolání z prostředí aplikace může být pohodlnější. Databázový soubor se zvětšuje voláním **`(cd_)GetSet_fil_blocks`**.

Test rychlosti spojení se serverem

Do aplikací určených pro rozlehlé sítě je vhodné zahrnout informaci o rychlosti spojení se serverem, kterou lze získat od funkce **`(cd_)Connection_speed_test`**. Na základě testu rychlosti může aplikace uživatele upozornit, že musí očekávat pomalejší odezvu.

Optimalizace aplikací pro vzdálenou komunikaci

Je-li databázový klient připojen na server po telefonním okruhu nebo po Internetu, pak nižší přenosová rychlost těchto médií může nepříjemně prodloužit dobu odezvy aplikace při provádění některých operaci.. Je proto rozumné optimalizovat návrh aplikace s přihlédnutím k tomu, jak jsou které operace náročné na komunikaci.

Pokud klient zadá velmi složitý SQL příkaz a server jej 10 minut provádí, pak čas odezvy prakticky nezávisí na tom, zda klient je připojen v LAN nebo telefonem. Naproti tomu pokud klient listuje ve formuláři obsahujícím obrázky, pak server je zatížen minimálně, ale čas odezvy může být při telefonním nebo Internetovém spojení podstatně vyšší, než v LAN.

Autor aplikace, která bude provozována po telefonu nebo Internetu, může optimalizovat čas odezvy dvojím způsobem:

- minimalizací množství přenášených dat mezi klientem a serverem;
- minimalizací počtu požadavků klienta na server.

Množství přenesených dat za jednotku času je totiž omezeno přenosovou kapacitu spojení, zatímco počet splněných požadavků za jednotku času je omezen časem obrátky v síti.

Optimalizace počtu požadavků

Minimalizovat počet požadavků lze tak, že tam, kde je to možné, se:

- série aktualizací nahradí jedním SQL příkazem, který zpracuje množinu záznamů;
- čtení nebo zápis hodnot více sloupců v jednom záznamu nahradí čtením nebo zápisem záznamu jako celku.

Optimalizace objemu přenosu

Minimalizovat množství přenášených dat lze zejména při návrhu formulářů, v nichž uživatel bude listovat. Optimalizovat lze jak na úrovní záznamů, tak i sloupců.

Je-li ve formuláři více záznamů, musí se jich více načíst, kdykoli uživatel přejde na novou stránku.

Při načítání záznamu do formuláře se přečtou všechny jeho sloupce, bez ohledu na to, které se budou zobrazovat ve formuláři. Proto pokud do tabulky TAB s 50 sloupci vytvoříte formulář obsahující 2 sloupce AA a BB, pak se bude při listování ve formuláři načítat všech 50. Výrazného zrychlení dosáhnete, pokud formulář přesměrujete na kurzor vzniklý odpovědí na dotaz:

```
SELECT AA, BB FROM TAB
```

Zrychlení startu aplikace

Aplikace obsahující velký program startuje poměrně pomalu, protože se přeložený tvar programu přenáší z databáze na počítač klienta. Tomu se lze vyhnout tak, že se na počítači klienta zapne ukládání objektů do *cache* (viz *Příručka správce*, kapitola *Architektura a instalace WinBase602*). Klient pak čte přeložený program i definice ostatních objektů z lokálního souboru a nezatěžuje tím přenosovou cestu k serveru.

Vytvoření nápovědy pro aplikaci

Aplikaci ve WinBase602 **lze doplnit systémem nápověd určeným pro jejího uživatele. Nápovědy musí být vytvořeny podle konvencí *Windows*.**

Příprava aplikační nápovědy

Pro databázovou aplikaci, která má poskytovat vlastní nápovědu, je třeba vytvořit nápovědný soubor s příponou HLP vyhovující konvencím ***Windows***.

Struktura nápověd

Nápověda je členěna do stránek, mezi nimiž mohou vést odkazy. Jednotlivé stránky nápovědy jsou označeny čísly od 1 do 65535. Čísla menší než 10000 jsou ve **WinBase602** rezervována. Pro stránky aplikační nápovědy lze využít čísel od 10000 výše.

Mimo to je pro snazší orientaci zvykem označovat stránky nápověd identifikátory. Definice některých standardních identifikátorů stránek nápověd naleznete v souboru WBAPL.HPJ, další si můžete zavést, pokud je chcete používat z programu v externím jazyce.

Stavebnice uživatelských nápověd

Součástí **WinBase602 SDK** jsou soubory tvořící stavebnici nápověd pro databázové aplikace. Naleznete je v subadresáři HELP. Na disku SDK není kompilátor nápověd.

Jaké stránky tvoří nápovědu?

Aplikace, kterou vytvoříte, by měla obsahovat tři druhy stránek nápověd:

- nápovědu k standardním dialogovým oknům, např. nápovědu, která se objeví po stisknutí tlačítka **Nápověda** v okně pro přihlášení se uživatele;
- nápovědu k položkám systémových formulářů, např. formuláře s historii hodnot sloupce;
- nápovědu k složkám formulářů definovaných v aplikaci a specifické nápovědy k různým částem aplikace volané tlačítky nebo z menu.

Text nápověd první a druhé skupiny včetně obrázků je součástí stavebnice a můžete ho převzít beze změny do souboru nápověd vaší aplikace. Text je v souboru WBAPL.RTF, obrázky v souborech s příponou BMP. Texty stránek nápověd třetí skupiny vytvoříte sami.

Postup vytváření souboru nápověd

Výroba souboru nápověd probíhá obvykle v těchto krocích:

1. Pomocí textového editoru umožňujícího psát neviditelný text, poznámky pod čarou a exportovat text ve formátu RTF napíšete text jednotlivých stránek nápověd.
2. Jména souborů zaznamenáte do tzv. *help project file* do sekce [FILES]. Doporučujeme využít jako vzor soubor WBAPL.HPJ, který je součástí stavebnice.
3. Celek přeložíte kompilátorem nápověd, čímž vytvoříte soubor s příponou HLP použitelný v aplikaci.

Kompilace souboru nápověd

Nápovědy lze kompilovat buď v prostředí specializovaného programu (jako např. MS Help Workshop) nebo pomocí řádkového kompilátoru HC31. V druhém případě všechny soubory (texty nápověd, obrázky, soubor WBAPL.HPJ) umístíte do stejného adresáře a v něm zadáte příkaz:

```
HC31  WBAPL.HPJ
```

Tím spustíte kompilátor nápověd HC31.EXE, který vytvoří soubor nápověd WBAPL.HLP. Tento soubor pak můžete podle potřeby přejmenovat.

Použitelný kompilátor nápověd je k dispozici na Internetu v knihovnách shareware pod jménem HC505.EXE.

Bližší informace viz:

Formát zdrojových textů nápověd patří mezi standardy ***Windows*** a jeho detailní popis vzhledem ke svému rozsahu přesahuje možnosti tohoto manuálu. Popisy tohoto formátu i kompilátory nápověd jsou součástí řady vývojových prostředí pro ***Windows***.

Instalace nápověd

Při importu aplikace do **WinBase602** se nápovědný soubor (první nalezený soubor s příponou HLP) automaticky zkopíruje z instalační diskety do adresáře obsahujícího programy **WinBase602**. Díky tomu lze z importované aplikace přímo volat nápovědu.

Použití nápověd v aplikaci

Před voláním nápovědy musí aplikace určit, ve kterém souboru jsou stránky nápovědy obsaženy. K tomu zavolá funkci **`Help_file`**. Neuvede-li cestu, soubor nápověd se hledá v adresáři s programy **WinBase602**.

Je-li tato funkce zavolána z programu, pak se po doběhnutí programu **WinBase602** vrátí k vlastnímu souboru nápověd. Proto pokud má být uživatelská nápověda dostupná i mimo program, je nutno zavolat funkci **`Help_file`** mimo program (např. jako akci při otevření formuláře, v němž lze nápovědu volat).

Nápovědy aktivované z formuláře

V návrháři formulářů lze v seznamu vlastností libovolné složky v sekci **Chování složky** specifikovat její **Číslo nápovědy**. Toto číslo se uplatní při volání nápovědy (klávesou F1 nebo otazníkem na liště) v situaci, kdy je tato složka vybraná.

Číslo nápovědy lze v seznamu vlastností zadat i *globálně* pro celý formulář. Uplatní se při volání nápovědy na složce, která *nemá* specifikované *vlastní* číslo nápovědy.

V obou případech lze pomocí vlastnosti **Styl nápovědy** zvolit, zda se nápověda má otevřít v samostatném aplikačním okně nebo v popup okénku, které zmizí, jakmile uživatel klikne mimo ně.

Obecné modální formuláře mají v pravém rohu horního rámu tlačítko s otazníkem, které umožňuje vyvolat nápovědu i pro složku, která není právě vybraná.

Nápovědy aktivované akcí

Akce **Zobrazit nápovědu v samostatném okně** a **Zobrazit nápovědu v popup okénku** slouží k jednoduchému vyvolávání nápověd speciálním příkazem menu nebo tlačítkem ve formuláři či na liště.

Nápovědy aktivované příkazem

Nápovědu lze vyvolat v kterémkoli okamžiku procedurami vnitřního programovacího jazyka **`Show_help`** a **`Show_help_popup`**.

Vytváření vícejazyčných aplikací

Součástí **WinBase602** jsou nástroje pro vytváření vícejazyčných aplikací, které:

- umožňují, aby se v jednou navržených formulářích, sestavách, menu a pomocných oknech objevovaly nápisy v různých jazycích;

- umožňují odlišné chování programu nebo odlišnosti v uživatelském rozhraní aplikace v závislosti na jazyce tam, kde je to nutné;
- odstraňují nutnost paralelní údržby všech jazykových mutací aplikace.

Principy vícejazyčnosti

Vícejazyčnosti aplikace se ve **WinBase602** dosahuje tím, že:

- místo znakových řetězců v určitém jazyce se v aplikaci (ve formulářích, v menu, v programech, ...) používají metařetězce;
- součástí aplikace je překladová tabulka, která pro každý metařetězec obsahuje odpovídající řetězce ve všech podporovaných jazycích;
- podle toho, který jazyk je právě zvolen, se v aplikaci objevují místo metařetězců odpovídající řetězce v tomto jazyce.

Mimo to lze jazykové mutace odlišit pomocí podmíněného překladu.

Jak vytvořit vícejazyčnou aplikaci

Definice jazyků

Předpokládejme, že vytváříte aplikaci v české a anglické mutaci. Nejprve si připravíte překladovou tabulku takto:

1. Ve vývojovém prostředí v menu ***Nástroje***, submenu ***Jazyk***, vyberte příkaz **Přidat jazyk**;
2. Vyplňte název jazyka, v našem případě *Čeština*, a stiskněte **OK**;
3. Ještě jednou proveďte příkaz **Přidat jazyk**, vyplňte název jazyka *English* a stiskněte **OK**.

Tím vytvoříte překladovou tabulku se dvěma jazyky. Další jazyky můžete přidat kdykoli později.

Zadání jazykových variant řetězců

Nyní předpokládejme, že potřebujete mít v aplikaci formulář s nadpisem "Zadejte svoje jméno" obsahující editační pole s popiskou "Jméno:". Tyto dva řetězce mají v angličtině znít "Enter your name" a "Name:". Nejprve si připravte překlady řetězců v překladové tabulce takto:

- Ve vývojovém prostředí v menu ***Nástroje***, submenu ***Jazyk***, proveďte příkaz **Editovat texty**, tím otevřete překladovou tabulku;
- Do tabulky vložte nový záznam s těmito údaji: v poli **Id** bude identifikace "Vstup jména - nadpis", v poli **Čeština** bude "Zadejte svoje jméno", v poli **English** bude "Enter your name", vše bez uvozovek;
- Vložte další záznam s těmito údaji: v poli **Id** bude "Vstup jména - popiska", v poli **Čeština** "Jméno:", v poli **English** "Name:";
- Zavřete okno s překladovou tabulkou.

Metařetězce ve formuláři

Nyní můžete vytvořit formulář pro vstup jména. Místo skutečných řetězců v něm budou metařetězce začínající znakem @ a obsahující místo textu identifikaci řetězce. Při nastavování vlastností formuláře zadáte tyto údaje:

- Ve vlastnosti formuláře **Okno formuláře / Nadpis** bude "@Vstup jména - nadpis" (bez uvozovek);
- Ve vlastnosti popisky Nápis (neměnný) bude "@Vstup jména - popiska" (bez uvozovek).

Alternativně lze překladovou tabulku otevřít i pomocí šipky v políčku seznamu vlastností formuláře.

Volba jazyka

Chcete-li nyní používat českou verzi formuláře, pak v menu ***Nástroje***, submenu ***Jazyk***, proveďte příkaz **Čeština**. Chcete-li použít anglickou verzi formuláře, pak na stejném místě proveďte příkaz **English**.

Obecný princip:

Obecně tedy platí, že potřebujete-li, aby určitý řetězec znaků měl více jazykových mutací, pak místo něj uvedete metařetězec skládající se ze znaku @ a identifikátoru řetězce. Metařetězec je v uvozovkách právě tehdy, pokud by se v nich psal normální řetězec, proto v textu programu budou všechny metařetězce v uvozovkách, zatímco například nadpisy formulářů a složek zadané na panelu vlastností budou bez uvozovek. Identifikátory vícejazyčných řetězců a jim odpovídající překlady uvedete v překladové tabulce.

Používání vícejazyčných aplikací a změny jazyka

Zvolený jazyk je v menu ***Nástroje***, submenu ***Jazyk***, vždy označen černým kolečkem. Aplikace si pamatuje, který jazyk je v ní zvolen. Když aplikaci opustíte a znovu otevřete nebo když ji exportujete a importujete, zvolený jazyk zůstane zachován.

Pokud změníte jazyk, program se před příštím během automaticky přeloží. Pokud aplikace používá pouze přeložený kód programu (v souboru EXX) a neobsahuje zdrojový program, pak je v programu fixován ten jazyk, s nimž byl program překládán, a nelze jej změnit. Jazyk nelze také měnit za běhu programu ve vnitřním jazyce.

Překladová tabulka se jmenuje _MULTILING a objeví se mezi tabulkami vícejazyčné aplikace.

Hlubší odlišnosti jazykových mutací aplikace

V některých případech je nutné, aby rozdíly mezi jazykovými mutacemi aplikace byly větší, než pouhé přeložení znakových řetězců. Pak je nutno do programu začlenit více variant a rozlišit je pomocí předdefinovaného makra **_LANGUAGE_** například takto:

```
#ifdef _LANGUAGE_="Čeština"
... česká verze programu ...
#elif _LANGUAGE_="English"
... anglická verze programu ...
#else
... jiná verze programu ...
```

```
#endif
```

Hodnotnou symbolu **_LANGUAGE_** je řetězec znaků obsahující název zvoleného jazyka.

Je-li součástí aplikace formulář, který má vypadat jinak v různých jazykových mutacích, pak je nutno vytvořit více variant tohoto formuláře a otevírat je programem obsahujícím konstrukci podobnou výše uvedené.

Sdílení datové základny nebo uživatelského rozhraní mezi aplikacemi

Databázovou aplikaci lze rozdělit do dvou částí, na:

- *datovou základnu*, tvořenou databázovými tabulkami a daty v nich obsaženými, triggery k těmto tabulkám, dotazy, sekvencemi a procedurami uloženými na serveru a replikačními vztahy (tzv. BACK-END);
- *aplikační a prezentační část*, tvořenou aplikačními programy, formuláři, sestavami, menu a dalšími objekty, které souvisejí s prezentací dat, přístupem k datům a využitím dat z databáze (tzv. FRONT-END).

WinBase602 umožňuje *sdílet* front-end nebo back-end mezi aplikacemi, tedy přistupovat ke stejným datům přes více různých rozhraní nebo využívat jedno rozhraní pro více různých souborů dat.

Příklad 1: Uvažujme celopodnikový informační systém. Data popisující podnik mohou být do té míry provázána, že nemá smysl je dělit do více schémat. Nad celopodnikovým datovým schématem (back-end) pak lze vytvořit řadu aplikačních programů (front-end), řešících například pohyb materiálu, objednávky, fakturaci, personalistiku, mzdy atd. Tyto programy sdílejí v určité míře data, kromě nich však nemají nic společného, a jsou tedy rozděleny do různých databázových aplikací.

Příklad 2: Uvažujme účetní aplikaci, která dovoluje vést účetnictví pro více různých firem. Může být výhodné zcela oddělit data patřící různých firmám, protože tím se usnadní přenosy a zálohování dat po jednotlivých firmách a také přidělování uživatelských práv k datům jednotlivých firem. Proto vytvoříme společné uživatelské rozhraní pro všechny firmy, tedy front-end část aplikace, a pro každou firmu zvláštní instanci datové základny, tedy zvláštní back-end.

Principy sdílení ve WinBase602

Sdílení front-endu nebo back-endu mezi aplikacemi se realizuje ve **WinBase602** ve formě aplikací *závislých* na jiných aplikacích:

- Pro aplikaci lze předepsat, že nemá vlastní front-end a využívá front-end z jiné aplikace.
- Pro aplikaci lze předepsat, že nemá vlastní back-end a využívá back-end z jiné aplikace.

Na řídicím panelu aplikace A, která využívá data z jiné aplikace B, vidíte tabulky, dotazy, triggery, uložené procedury, sekvence a replikační vztahy z aplikace B. Na řídicím panelu aplikace A, která využívá front-end z jiné aplikace B, vidíte formuláře, sestavy, menu, programy, obrázky, relace, schémata, ODBC spojení a WWW objekty z aplikace B. Pro jakoukoli aplikaci vidíte na řídicím panelu pouze její vlastní role.

Sdílení back-endu

V situaci popsané v příkladu 1 vytvoříme jednu bázovou aplikaci obsahující kompletní back-end, tedy všechny tabulky, triggery, dotazy a procedury. Tato aplikace buď nemá žádný front-end, nebo pouze minimální, například pro zálohovací a správní funkce. Poté pro každou oblast nasazení informačního systému vytvoříme zvláštní aplikaci, která obsahuje pouze front-end - aplikační programy a uživatelské rozhraní pro tuto oblast, ale využívá datovou základnu z bázové aplikace.

Sdílení front-endu

V situaci popsané v příkladu 2 vytvoříme kompletní účetní aplikaci. Poté z této aplikace vezmeme pouze back-end a v databázi vytváříme jeho další instance. Pro každou z těchto instanci předepíšeme, že má používat front-end z původní aplikace.

Závislost aplikace na aplikaci

Jak fungují aplikace využívající front-end nebo back-end z jiné aplikace? Předpokládejme, že v aplikaci A je nastaveno využití back-endu nebo front-endu z aplikace B. Pak platí:

- při exportu aplikace A se neexportují objekty ani data z aplikace B; exportuje se však informace o tom, co se z aplikace B využívá a jméno aplikace B;
- při importu A se automaticky nastaví využití objektů z aplikace B (pokud aplikace B neexistuje, aplikace A nebude kompletní nebude fungovat);
- při zrušení aplikace A nebudou objekty z aplikace B nijak dotčeny;
- při zrušení aplikace B zmizí její objekty z aplikace A a aplikace A přestane fungovat;
- v aplikaci A lze modifikovat objekty z aplikace B, rušit je nebo vkládat nové. Změny se projeví v aplikaci B a ve všech aplikacích, které tyto objekty z aplikace B využívají. Tyto manipulace s cizími objekty lze omezit přístupovými právy.

Pokud v aplikaci A byl vytvořen nějaký objekt X před nastavením využití objektů stejné třídy z jiné aplikace B, pak objekt X nebude přístupný.

Jak ovládat sdílení částí aplikací?

Sdílení front-endu a back-endu se ve **WinBase602** řídí na stránce **Závislosti** v dialogovém okně otevřeném stiskem tlačítka **Vlastnosti** na řídicím panelu, na němž je vybraná aplikace.

Postup vytvoření aplikace využívající back-end z jiné aplikace:

1. vytvořte novou aplikaci;
2. tlačítkem **Vlastnosti** otevřete dialogové okno a vyberte záložku **Závislosti**;
3. v combu nadepsaném **Aplikace využívá data z aplikace** vyberte jméno aplikace, které obsahuje back-end.

Postup vytvoření nové instance back-endu aplikace:

1. exportujte aplikaci s příznakem **Exportovat pouze back-end** (příznak **Exportovat včetně dat v tabulkách** nastavte dle potřeby, obvykle nebude zatržen);
2. importujte vyexportovanou aplikaci a přidělte ji nové jméno - takto vytvořená aplikace bude mít na záložce **Závislosti** nastaveno v combu **Aplikace využívá uživatelské rozhraní z aplikace** jméno výchozí aplikace.

Role a závislé aplikace

Roli nelze sdílet mezi aplikacemi a lze ji přidělit práva pouze k objektům ze stejné aplikace. Proto v aplikaci A, která využívá front-end nebo back-end objekty z jiné aplikace B, nelze přidělit roli R z A právo k objektu X z B.

Pokud při práci s aplikací A uživatel potřebuje využívat kromě objektů z A také objekty z jiné aplikace B, je nutno jej obsadit současně do role RA z A, který má práva k objektům z A, a do role RB, která má práva k objektům z B.

Pokud v aplikaci A, která využívá front-end z jiné aplikace B, vytvoříte nový objekt patřící do front-endu, tento objekt se sice objeví v B a bude patřit do B, ale standardním rolím (v žádné aplikaci) se nepřidělí žádná práva k tomuto objektu. Totéž platí pro sdílení back-endu.

Export aplikace

Export aplikace je krokem, který odděluje vývoj aplikace od jejího šíření.

Šifrování objektů v aplikaci

Většina komponent databázové aplikace se za normálních okolností ukládá do databáze textovém tvaru a je tudíž čitelná pro každého uživatele, který má dostatečná práva. To platí i o dohotovených aplikacích distribuovaných klientům. **WinBase602** nabízí možnost uložení komponent v šifrovaném tvaru, aby se zamezilo riziku zneužití zdrojových textů aplikace. Autor aplikace vyznačí, které objekty chce chránit, a při exportu je nechá zapsat do souborů zašifrovaně.

který chce šířit aplikaci v šifrovaném stavu, postupuje takto:

- pro komponenty, které chce šifrovat, zatrhne ve spodní části řídicího panelu **Utajit objekt** (přímo může označit skupinu objektů a zatržení provést pro celou skupinu najednou);
- při exportu aplikace zatrhne čtverec **Exportovat zašifrovaně**.

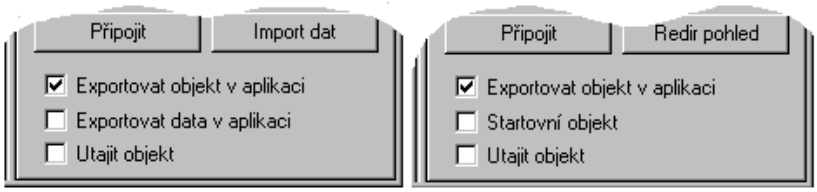
Po importu aplikace obsahující zašifrované komponenty:

- databázový klient dokáže šifrované komponenty aplikace dešifrovat a pracuje s nimi zcela stejně, jako kdyby šifrované nebyly;
- na šifrované komponenty nelze otevřít návrhář ani textový editor, nelze je exportovat ani duplikovat;
- na šifrované komponenty nelze vyvolat návrháře ani z programu pomocí funkcí jako **`Edit_view`** apod.;
- při přímém čtení definice komponent z databáze se jejich obsah jeví jako nesmyslná binární data;
- aplikaci nelze exportovat, přepínač **Utajit objekt** již nemá žádný význam;
- nové komponenty, které případně vznikají při provozu aplikace, zašifrované nejsou a lze s nimi pracovat bez omezení.

Pro šifrování je použit kryptograficky odolný generátor pseudonáhodných čísel. Každý objekt je šifrován pomocí jiného klíče, proto případná znalost zdrojového tvaru některých zašifrovaných objektů nenapomůže k dešifrování ostatních. Pokud se na straně databázového klienta využívá cache, objekty jsou v ní uloženy v zašifrovaném stavu. Šifrovat nelze obrázky, role, relace ani ODBC spojení.

Nastavení před exportem aplikace

Před exportem by měl autor aplikace ověřit, zda níže popsaná nastavení odpovídají jeho požadavkům.



Volba objektů pro export

V aplikaci lze vyznačit, které objekty se mají exportovat. Nové objekty jsou vždy označeny jako exportované. Obvykle se exportuje vše kromě pracovních objektů, které vznikají při provozu aplikace a nejsou potřebné při jejím uvádění do chodu.

U exportovaných tabulek lze dále vyznačit, zda se exportují s daty nebo bez nich. S daty se zpravidla exportují číselníky tvořící součást aplikace, naopak bez dat se exportují ty tabulky, které budou naplněny specifickými daty uživatele.

Export objektů se vyznačí na řídicím panelu, zatržením označovacích čtverců pod tlačítky ve spodní části pravé oblasti panelu. Zatržení lze provádět nejen pro jednotlivé objekty, ale také hromadně pro celé skupiny objektů vybrané v seznamu.

Funkce **`Chng_component_flag`** slouží k nastavení příznaku exportu z programu. Díky ní může program zakázat export těch dočasných objektů, které vytvoří.

Označení startovního objektu

V distribuované aplikaci musí být vyznačen její startovní objekt, aby bylo možno aplikaci spouštět bez nahlížení do jejího nitra.

Někdy je vhodné použít jiný startovní objekt pro inicializaci aplikace při prvním spuštění na uživatelském počítači a jiný pro běžný provoz. Svůj startovní objekt může aplikace změnit voláním funkce **`(cd_)Set_appl_starter`**.

Vyznačení šifrovaných objektů

Pomocí označovacího čtverce na řídicím panelu se označí objekty, které se mají při exportu šifrovat. Podmínkou šifrování však je zapnutí přepínače **Exportovat zašifrovaně** při vlastním exportu.

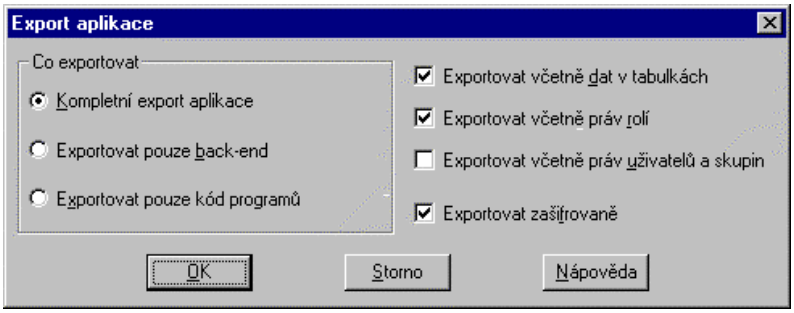
Volba krátkých nebo dlouhých jmen souborů

Jména souborů, do nichž se exportují komponenty aplikace, se odvozují z jmen těchto komponent. Mohou proto mít až 31 znaků před příponou. Přestože ***Windows*** pracují i s dlouhými jmény souborů, řada běžně používaného software (komprimační programy, síťové servery) vyžaduje použití krátkých jmen. Proto **WinBase602** obsahuje přepínač umožňující zapnout nebo vypnout využívání dlouhých jmen. Tento přepínač naleznete v okně parametrů (příkaz **Parametry** z menu *Nástroje*) pod položkou **Import a export**.

Je-li použití dlouhých jmen vypnuto, pak **WinBase602** zkrátí jména komponent na nejvýše 8 znaků. Přitom dbá, aby nevznikla duplicitní jména.

Export aplikace do souborů

Export aplikace se spouští z řídicího panelu. Před exportem se objeví dialog pro zadání parametrů exportu:



V levé části zvolte, zda se má exportovat aplikace jako celek, pouze back-end aplikace (tabulky, dotazy, triggery, uložené procedury, sekvence), nebo pouze kód programů. Export samostatného kódu již v této verzi vzhledem k možnosti šifrovat a cachovat objekty ztrácí na významu.

Vpravo můžete upravit obsah exportu takto:

Exportovat včetně dat v tabulkách – příznak určuje, zda se kromě definic tabulek do exportu zahrnou i *data*. Týká se pouze těch tabulek, pro něž je na řídicím panelu vyznačeno **Exportovat data v aplikaci**.

Exportovat včetně práv rolí – příznak určuje, zda se součástí exportu stanou i *práva ke komponentám aplikace nastavená pro role* **v** aplikaci. Bez něj se **v** aplikaci nastaví pouze standardní práva pro standardní role. Pokud jsou **v** aplikaci smysluplné role, je třeba exportovat i jejich práva.

Exportovat včetně práv uživatelů a skupin – příznak určuje, zda se součástí exportu stanou i *práva nastavená pro uživatele a skupiny* ke komponentám aplikace a také popis *obsazení uživatelů a skupin do rolí*. Pokud na cílovém počítači budou existovat jiní uživatelé a skupiny, nemá smysl tato práva exportovat.

Exportovat zašifrovaně – příznak určuje, zda se exportované objekty *zašifrují* při exportu. Týká se pouze těch objektů, které mají na řídicím panelu nastavený příznak **Utajit objekt**.

Po stisku **OK** zvolíte adresář, do něhož se aplikace vyexportuje do řady souborů.

Při exportu aplikace vzniknou tyto soubory:

- pro každou exportovanou komponentu aplikace soubor s její (případně zašifrovanou) definicí;
- pro každou tabulku, jejíž data se mají exportovat, soubor s jejím obsahem;
- tzv. DEFINIČNÍ SOUBOR APLIKACE (APL SOUBOR) popisující, co a jak se má importovat;
- soubor s příponou APX obsahující některé dodatečné informace o aplikaci.

Jména souborů při exportu

Jména souborů vznikajících při exportu aplikace mohou být buď stejná jako jména exportovaných objektů s příponou určující kategorii, anebo mohou být zkrácena na 8+3 znaků (případně odlišená znaky na konci, pokud by jinak došlo k duplicitě jmen). Zvolit jednu nebo druhou variantu lze pomocí příkazu **Parametry** z menu ***Nástroje***, položka **Import a Export**.

Úpravy exportované aplikace

Definiční soubor aplikace (soubor s příponou APL vzniklý při exportu) obsahuje výčet komponent aplikace a informace o jejich právech. Je to textový soubor a lze do něj zasáhnout.

Použití programu v přeloženém tvaru

Z distribuované aplikace a lze vyřadit zdrojový text programů a šířit pouze přeložený program. Tento obrat se používal ve starších verzích **WinBase602** k ochraně práce programátora, v současnosti jej lze nahradit šifrováním objektů.

Zdrojový tvar programu vyřadíte z importu prostě tak, že řádku, která jej popisuje, vypustíte z definičního souboru aplikace. Pak lze smazat soubor s programem. Obsahuje-li aplikace více programů, lze takto vyřadit pouze jeden.

Soubory s příponou EXX

Při exportu aplikace se každý její program přeloží a zapíše do souboru s příponou EXX. Tyto soubory vzniknou ve stejném adresáři, jako definiční soubor. V definičním souboru nejsou zmíněny, protože se do databáze nebudou importovat a při každém použití se přečtou ze souboru.

Použití programu v přeloženém tvaru

Program v přeloženém tvaru se používá tak, že se na příkazové řádce spouštějící provozní systém **WinBase602** uvede parametr **/C** následovaný cestou k souboru obsahujícímu tento program. Aby taková příkazová řádka při importu aplikace vznikla, musíte upravit v definičním souboru aplikace řádku začínající slovem ICON do této podoby:

```
ICON  jméno_zástupce[:jméno_složky] jméno_programu.EXX
```

Během importu aplikace se soubor *jméno_programu.EXX* zkopíruje do adresáře k programům **WinBase602** a do příkazové řádky ikony se doplní parametr **/C**, cesta a jméno souboru.

Při spuštění **WinBase602** s parametrem **/C** se ze zadaného souboru načte a spustí program EXX. Případný startovní objekt aplikace se ignoruje.

Příklad:

Aplikace ABC obsahuje program ABCPROG, jehož zdrojový text chce autor utajit. Exportem aplikace vznikne definiční soubor. Řádka začínající ICON se upraví např. do podoby:

```
ICON  wb602-abc  abcprog.exx
```

Nechť se aplikace importuje do **WinBase602**, která má programy v adresáři C:\WB602, na server WBDB. Pokud aplikace nebude při importu uživatelem přejmenována, vznikne ikona s nápisem **WB602-ABC** spouštějící program WB602.EXE s parametry:

```
&WBDB  ABC  /cC:\WB602\ABCPROG.EXX
```

První parametr udává server, na nějž se klient připojí, druhý parametr je jméno aplikace, která se otevře, třetí udává cestu k souboru obsahujícímu program v přeloženém tvaru. Tento program se spustí.

Aplikaci, která neobsahuje zdrojové kódy programů a spouští se proto postupem uvedeným v této sekci, není možné spustit z řídicího panelu.

Nestandardní ikona v zástupci aplikace

Pokud chcete, aby zástupce aplikace měl jinou než standardní ikonu, pak na stejnou disketu, jako vyexportovanou aplikaci, uložte soubor s touto ikonou. Jméno souboru se musí v prvních osmi znacích shodovat se jménem aplikace bez diakritiky s nealfanumerickými znaky nahrazenými znakem _ a musí mít příponu ICO. Ikona se při instalaci aplikace zkopíruje do adresáře k programům **WinBase602**.

```
ICON  jméno_zástupce[:jméno_složky]   parametry
```

Řádka definičního souboru začínající slovem ICON vytváří zástupce aplikace zadaného jména v zadané složce. Pokud dvojtečka a *jméno_složky* chybí, zástupce se umístí do složky **Aplikace602**. Složka, do níž se nakonec umístí zástupce aplikace, může být také určena nastavením volitelných parametrů **WinBase602** na tom počítači, kde import proběhne.

Přidání nápovědy k aplikaci

Pokud jste pro aplikaci vytvořili nápovědu odpovídající konvencím ***Windows***, pak zkopírujte soubor a nápovědou k vyexportovaným souborům aplikace. Jméno souboru s nápovědou musí být stejné jako jméno aplikace zkrácené na 8 znaků, bez diakritiky a s nealfanumerickými znaky nahrazenými znakem _, a musí mít příponu HLP.

Rozdělení aplikace na více disket

Je-li rozsah aplikace takový, že se nevejde na jedno médium, je třeba její komponenty rozdělit na více médií. Aplikaci exportujte na pevný disk a poté její soubory uložte po skupinách na diskety. Přitom respektujte tato pravidla:

1. Na první disketu uložte soubory s příponou APL a APX.
2. Na poslední disketu uložte kopii souboru s příponou APX a případně i soubory ICO (ikona aplikace), EXX (přeložený kód programu) a HLP (nápověda k aplikaci), pokud jsou součástí aplikace.

3. Pokud editujete soubor APL, řádku začínající ICO ponechte na jeho konci. Pořadí řádek popisujících komponenty aplikace můžete změnit, ale import dat do tabulky musí být umístěn až za importem definice této tabulky.
4. Ostatní soubory rozdělte mezi ***diskety v tom pořadí, v němž jsou uvedeny v souboru s příponou APL***.

Při importu aplikace se její komponenty přenášejí do databáze v tom pořadí, v němž jsou popsány v souboru APL. Pokud se některý soubor nenajde, uživatel bude vyzván k vložení další diskety. Pokud by soubory nebyly rozloženy na disketách v pořadí řádek v souboru APL, uživatel by musel měnit diskety zbytečně často a navíc by musel při každé změně pokusně nabízet všechny diskety až do nalezení souboru.

Přehled akcí při importu aplikace

Při importu aplikace se v první řadě založí nová aplikace. Nedochází-li ke konfliktu s nějakou již existující aplikací, použije se jméno a identifikace z definičního souboru, jinak se uživateli umožní zadat nové jméno a vytvoří se nová identifikace.

Pak se importují komponenty popsané v definičním souboru a případě také data z tabulek.

Dále se z distribučního média zkopírují do adresáře, obsahujícího instalaci **WinBase602**, tyto soubory (naleznou-li se):

- soubor s příponou ICO obsahující specifickou ikonu aplikace, jméno souboru musí být stejné jako upravené jméno aplikace;
- soubor s příponou HLP obsahující nápovědu k aplikaci, jméno souboru musí být stejné jako upravené jméno aplikace;
- soubor s příponou EXX obsahující kód programu, jehož jméno je uvedeno jako parametr na řádce začínající ICON.

Upraveným jménem aplikace se zde rozumní jméno aplikace zkrácené na 8 znaků, zbavené diakritiky a obsahující znaky _ místo nealfanumerických znaků.

Při zpracování řádky začínající ICON se vytvoří zástupce aplikace.

Dále se vyhledá a přečte soubor s příponou APX obsahující další informace o aplikaci.

Distribuce aplikací s provozním systémem

Aplikace pro **WinBase602** se dá distribuovat buď samostatně, nebo spolu s Provozním systémem.

První případ se hodí pro aplikace určené k provozu na síťovém SQL serveru **WinBase602** nebo na Personální databázi. Uživatel buď již má nainstalovaný anebo si zakoupí a nainstaluje příslušný produkt, a do něj pak importuje aplikaci.

Pokud je aplikace určena pro provoz v *nesíťovém* prostředí Provozního systému, pak lze její instalaci spojit s vhodně upravenou instalací Provozního systému. Tuto cestu lze použít také proto, že Provozní systém lze bez porušení licenčních podmínek volně šířit.

Odkud vzít provozní systém?

Provozní systém využitelný pro distribuci aplikací, lze okopírovat z distribučního média **WinBase602**, z adresáře **RunTime**. Provozní systém je rozdělen do adresářů tak, aby obsah každého adresáře vešel na jednu 1.44 MB disketu. Lze jej však zkopírovat i jako celek na médium s větší kapacitou.

V adresáři **RunTime** je také příklad souboru SETUP.INI upraveného pro distribuci aplikace.

Jak upravit provozní systém?

Postup instalace provozního systému lze podstatně upravit tak, aby vyhovoval požadavkům aplikace. Úprava se provádí přepsáním souboru SETUP.INI nacházejícího se ve stejném adresáři, jako program SETUP.EXE (na prvním instalačním disku).

V souboru SETUP.INI se upravuje sekce začínající řádkou `[WINBASE602]`. Každý parametr se zapisuje na zvláštní řádku.

Parametr	***Hodnota***	***Význam***
`Import`	Jméno aplikace	Po instalaci provozního systému se importuje aplikace tohoto jména ze stejně pojmenovaného souboru s příponou APL
`Title`	Text	titulek instalace provozního systému
`WelcomeTitle`	Text	titulek úvodního dialogu
`WelcomeText`	Text	začátek obsahu úvodního dialogu
`TargetDir`	adresář	přednastavený adresář pro instalaci provozního systému
`AskTarget`	0 nebo 1	je-li 0, uživatel nebude tázán na adresář pro instalaci a použije se adresář specifikovaný v TargetDir
`ServerName`	jméno databáze	jméno, které se objeví v dialogu pro výběr jména databáze
`AskDb`	0 nebo 1	je-li 0, uživatel nebude tázán na jméno databáze; použije se jméno zadané v ServerName; databáze se vytvoří v adresáři nastaveném v TargetDir
`ShortcutFolder`	jméno složky	jméno, které se nabídne v dialogu pro vytvoření zástupců programu
`AskFolder`	0 nebo 1	je-li 0, uživatel nebude tázán na jméno složky; použije se jméno zadané v ShortcutFolder
`AskClientProtocol`	0 nebo 1	je-li 0, uživatel nebude tázán na protokoly klienta

`ShowSummary`	0 nebo 1	je-li 0, nebude se zobrazovat dialog se shrnutím zadaných parametrů (použijte v případě CopyFIL=1 nebo jsou-li předchozí hodnoty Ask... rovny 0
`AskInsert`	0 nebo 1	je-li 0, nezobrazí okno s výzvou k vložení diskety s aplikací nebo databázovým souborem WB5.FIL. To má význam při instalaci z CD
`CopyFIL`	0 nebo 1	je-li 1, místo importu aplikace se zkopíruje připravený databázový soubor WB5.FIL (obsahující aplikaci). Při této instalaci se nevytvoří ikona spouštìjící aplikaci.

Parametr **`CopyFil`** dovoluje k aplikaci přibalit kompletní databázový soubor a místo importu aplikace jej prostě zkopírovat. Databázový soubor určený k distribuci touto cestou je vhodné předem zkompaktnit pomocí funkce **`(cd_)Compact_database`**.

Pokud chcete využít instalační diskety provozního systému k instalaci klienta (který se bude připojovat na síťový server) nebo pouze ODBC driveru pro **WinBase602**, můžete použít níže uvedené parametry. V takovém případě se import aplikace neprovádí.

`ClientOnly`	0 nebo 1	instaluje se pouze klient **WinBase602**
`ODBCOnly`	0 nebo 1	instaluje se pouze ODBC driver pro **WinBase602**

Příklady obsahu souboru SETUP.INI se nacházejí v elektronické nápovědě.

Databáze uložené na CD ROM

WinBase602 umožňuje vytváření databází uložených na CD ROM. Tyto databáze zpřístupňují uživateli po instalaci značné množství dat, aniž by tato data musela být zkopírována na pevný disk. Podstatnou vlastností takto vytvořených databází je, je uživatel může v omezeném rozsahu obsah databáze editovat, přičemž na pevný disk se zaznamenají pouze změny v datech, nikoli celá databáze.

Níže je popsán postup, jak vytvořit databázi na CD ROM a jak zajistit její instalaci. Stejný postup lze použít pro zřízení sdílené referenční databáze umístěné na síťovém disku bez práva zápisu.

Princip práce s databází uloženou na CD ROM

Uživatel, který si instaluje databází z CD ROM, pracuje s databázovým souborem rozděleným do dvou částí. První část je umístěna na CD ROM, obsahuje základ dat z databáze

a (přirozeně) není editovatelná. Druhá část je umístěna za uživatelově pevném disku, vznikne při instalaci databáze, a zapisují se do ní změny, které uživatel v databázi případně provede. Tato část musí existovat i v případě, že uživatel nechce provádět žádné změny v databázi.

Změny, které uživatel může v databází provádět, jsou omezeny interním limitem, který není jednoduše kvantifikovatelný. Vývojářům aplikací doporučujeme, aby se řídili tímto pravidlem: Bez omezení lze zakládat nové objekty a plnit nově založené tabulky daty. Editovat obsah existujících tabulek z CD ROM by mělo být dovoleno pouze výjimečně. Doporučujeme se vyhnout editaci tabulek z CD ROM tak, že aplikace vytvoří (např. při prvním spuštění) vlastní tabulku, jejíž záznamy budou svázány vztahem 1:1 se záznamy v určité tabulce na CD ROM. Editovat se pak budou pouze záznamy v nově vytvořené tabulce.

WinBase602 musí být na uživatelově počítači buď instalována předem (normálním instalačním programem **WinBase602**), nebo se bude při každém spuštění číst z CD ROM. Druhý případ má tyto omezující vlastnosti:

- **WinBase602** z CD ROM se spouští pomaleji než když je instalovaná na pevném disku, úměrně rozdílu v přístupové době na obě média;
- na počítači s nedostatkem paměti může **WinBase602** z CD ROM reagovat pomaleji i za běhu;
- použití **WinBase602** z CD ROM předpokládá, že na uživatelově počítači je instalováno ODBC.

WinBase602 z CD ROM se chová jako provozní systém, umožňuje tedy pracovat s obsahem databáze, ale neumožňuje vyvíjet aplikace.

Aplikace z databáze na CD ROM se na uživatelově počítači spouští pomocí svého startovního objektu, k němuž během instalace vznikne zástupce. Aplikace může mít kód programu v externím souboru a může mít vlastní soubor nápověd. Lze vytvořit více zástupců spouštějících více různých aplikací ze stejné databáze.

Příprava databáze pro CD ROM

Předpokládejme, že autor má hotovou databázovou aplikaci určenou k šíření na CD ROM. Aplikace je naplněna daty.

1. V aplikaci se označí její startovní objekt.
2. Pokud ve stejném databázovém souboru jsou ještě jiné aplikace, které na CD ROM nepatří, nebo pokud se databázový soubor skládá z více než jedné části, pak:

- aplikace se exportuje (včetně dat),
- založí se nový databázový soubor a
- aplikace se do něj importuje.

3. V opačném případě doporučujeme aplikaci exportovat také, za účelem zálohování dat.
4. Ve vývojovém prostředí připojeném na databázi, určenou k převedení na CD ROM, se z menu ***Nástroje / Server*** příkazem **Provozní parametry** otevře dialogové okno.
5. Pokud v databázovém souboru je mnoho volných clusterů, lze velikost souboru zmenšit stiskem tlačítka **Zkompaktnit**.
6. Ve stejném dialogu se databáze stiskem tlačítka **Připravit pro CD ROM** převede do needitovatelné podoby. Od tohoto okamžiku není dovoleno do databáze zapisovat.
7. Dialog a vývojové prostředí se uzavře a server se ukončí.

Nad stejnou databází se server nesmí více spouštět. Popsaným postupem vznikne soubor WB5.FIL určený k přenesení na CD ROM.

Příprava obsahu CD ROM

V další fázi se vytvoří adresář, který bude obsahovat vše, co má být přeneseno na CD ROM. Tento adresář může obsahovat podadresáře, aby jeho obsah byl přehlednější. Do adresáře (případně jeho podadresářů) umístíte tyto soubory:

1. Databázový soubor WB5.FIL vytvořený v předchozí fázi.
2. Soubor SETUPDB.EXE z **WinBase602 SDK** (tento soubor lze libovolně přejmenovat). Tento program provede na uživatelském počítači instalaci databáze.
3. Soubor SETUPDB.INI s obsahem popisujícím CD ROM a instalaci databáze u uživatele (viz dále).
4. Soubor obsahující ikonu, která se použije v zástupci vytvořeném na počítači uživatele pro spouštění databázové aplikace. (Souborů může být více, pokud se bude vytvářet více zástupců různých aplikací)
5. Soubor resp. soubory s uživatelskou nápovědou k aplikaci.
6. Případný soubor resp. soubory s přeloženým kódem těch aplikací, z nichž byl odstraněn zdrojový program.
7. Programy a knihovny **WinBase602**, pokud aplikace má být provozována na uživatelově počítači bez instalace **WinBase602**.

Takto připravený adresář se dá využít pro pokusné instalace databáze před jejím vypálením do CD ROM. Lze tak například ověřit správnost obsahu souboru SETUPDB.INI .

Obsah souboru SETUPDB.INI

SETUPDB.INI je textový soubor ve formátu INI-souborů pro ***Windows***. Tento soubor řídí činnost instalačního programu SETUPDB.EXE. Obsahuje sekce **WinBase602**, **Shortcut1** a případné další sekce **Shortcut2**, **Shortcut3** atd.

V sekci **WinBase602** jsou tyto údaje:

- **InitialMessage**=*text v úvodním okně*

Tento údaj je nepovinný. Pokud je uveden, pak se po spuštění programu SETUPDB.EXE objeví úvodní okno obsahující zadaný text a dvě tlačítka: **OK** a **Storno**. Stisk tlačítka **Storno** ukončí program, stisk **OK** umožní pokračování instalace. Pokud údaj není zadán, k instalaci se přistoupí ihned po spuštění programu.

- **RegisterInstallation**=*adresář obsahující programy WinBase602 na CD ROM*

Tento údaj je nepovinný. Pokud není uveden, pak se na uživatelském počítači bude hledat existující instalace **WinBase602** aktuální verze, a pokud se nenajde, instalace skončí neúspěšně. Adresář se uvádí bez úvodního písmene označujícího diskovou jednotu a bez následující dvojtečky.

- **DatabasePath**=adresář obsahující databázový soubor WB5.FIL na CD ROM

Tento údaj je povinný. Adresář se uvádí bez úvodního písmene označujícího diskovou jednotu a bez následující dvojtečky.

- **DatabaseName**=jméno databáze

Tento údaj je povinný. Pod tímto jménem se databáze zaregistruje na uživatelově počítači

- **PrivateDatabaseDir**=plná cesta k adresáři pro editovatelnou část databázového souboru

Tento údaj je nepovinný. Umožňuje zadat umístění a jméno adresáře, který se během instalace uživateli nabídne pro umístění editovatelné části databázového souboru. Pokud není uveden, nabídne se běžný adresář.

- **FinalMsg**=text závěrečné zprávy o úspěšné instalaci

Tento údaj je nepovinný. Není-li uveden, zobrazí se na závěr úspěšné instalace zpráva „Instalace databáze proběhla úspěšně“

Sekce Shortcutx

Každá sekce **Shortcut*x*** popisuje jednoho zástupce, který se má vytvořit na uživatelském počítači. Zástupci odpovídají aplikacím v databázovém souboru. Sekce musí být očíslovány sekvenčně od 1. V sekcích jsou tyto údaje:

- **ShortcutApplication**=jméno aplikace

Tento údaj je povinný. Specifikuje jméno databázové aplikace, která se má otevřít v databázi, a jejíž startovní objekt se má spustit.

- **ShortcutCaption**=popiska zástupce

Tento údaj je nepovinný. Specifikuje název zástupce. Pokud není zadán, bude název zástupce stejný, jako jméno aplikace.

- **ShortcutFolder**=jméno položky v menu Start / Programy, do níž se zástupce umístí

Tento údaj je nepovinný. Specifikuje umístění vytvářeného zástupce. Může označovat existující položku, do níž se zástupce přidá, nebo novou položku, která se při instalaci vytvoří. Pokud údaj není uveden, zástupce se vytvoří v položce **Database**.

- **ShortcutIcon**=cesta a jméno souboru s ikonou zástupce

Tento údaj je nepovinný. Obsahuje adresář a jméno souboru na CD ROM, v němž je soubor s ikonou zástupce. Adresář se uvádí bez úvodního písmene označujícího diskovou jednotu a bez následující dvojtečky. Pokud údaj není uveden, zástupce bude obsahovat ikonu **WinBase602**.

- **ApplicationHelp**=cesta a jméno souboru s aplikační nápovědou (soubor HLP)

Tento údaj je nepovinný. Obsahuje adresář a jméno souboru na CD ROM, v němž je soubor s nápovědou k aplikaci. Adresář se uvádí bez úvodního písmene označujícího diskovou jednotu a bez následující dvojtečky. Je-li tento údaj uveden, soubor se zkopíruje do adresáře, v němž je instalována WinBase602, kde jej běžící aplikace najde. Jsou-li na CD ROM programy WinBase602, pak soubor s nápovědou doporučujeme umístit do stejného adresáře, aby mohl být nalezen také při práci programů z CD ROM.

- **ApplicationCode**=cesta a jméno souboru s přeloženým kódem aplikace (soubor EXX)

Tento údaj je nepovinný. Obsahuje adresář a jméno souboru na CD ROM, v němž je soubor s přeloženým kódem aplikace. Adresář se uvádí bez úvodního písmene označujícího diskovou jednotu a bez následující dvojtečky. Je-li tento údaj uveden, soubor se zkopíruje do adresáře zvoleného pro editovatelnou část databázového souboru a odkaz na něj se vloží do příkazové řádky zástupce. Tento údaj se využije v případě, že autor aplikace chce utajit zdrojový text programu a odstranil jej z aplikace.

- **DisableLogin**=1, pokud se uživatel má přihlásit jako anonymní, nebo 0 pro normální přihlášení

Tento údaj je nepovinný. Pokud není uveden nebo pokud má hodnotu 0, uživatel se bude do databáze normálně přihlašovat svým jménem a heslem. Pokud má hodnotu 1, pak se přihlašovací dialog neobjeví a uživatel bude automaticky přihlášen jako anonymní.

Příklad obsahu souboru SETUPDB.INI:

```
[WinBase602]
```

```
InitialMessage=Tato instalace Vám umožní přístup k databázi motýlů
na CD ROM. Chcete pokračovat?
RegisterInstallation=WBINST
DatabasePath=DATABASE
DatabaseName=CDmotyli
PrivateDatabaseDir=c:\motyli
FinalMsg=Instalace proběhla úspěšně. V menu Start najděte složku
Motýli a spusťte Katalog motýlu
[Shortcut1]
ShortcutApplication=motyli
ShortcutCaption=Katalog motýlu
ShortcutFolder=Motýli
ShortcutIcon=IKONY\MOTYL.ICO
ApplicationHelp=WBINST\MOTYLI.HLP
ApplicationCode=KOD\MOTYLI.EXX
DisableLogin=1
```

Tento soubor zpřístupní databázi jménem **CDmotyli** uloženou na CD ROM v adresáři DATABASE. Vytvoří jednoho zástupce jménem **Katalog motýlů** umístěného v položce **Motýli** v menu Start / Programy. Zástupce bude mít ikonu umístěnou na CD ROM v souboru IKONY\MOTYL.ICO . Zástupce bude spouštět startovní objekt aplikace **motyli**, bez nutnosti přihlašovat se k databázi. Kód aplikace je v souboru MOTYLI.EXX, nápověda k aplikaci v souboru MOTYLI.HLP.

Obsah CD ROM odpovídající tomuto souboru:

Kořenový adresář:

- SETUPDB.EXE
- SETUPDB.INI (uvedený výše)

Subadresář WBINST:

- programy a knihovny tvořící instalaci WinBase602, tedy: WB602.EXE, WBSERVER.EXE, WBKERNEL.DLL, WBPREZEN.DLL, WBVIEWED.DLL, WBED.DLL a INSTSERV.DLL
- nápovědný soubor MOTYLI.HLP

Subadresář DATABASE:

- Soubor WB5.FIL obsahující aplikaci MOTYLI vytvořený postupem popsaným výše

Subadresář IKONY:

- Soubor MOTYL.ICO obsahující ikonu, která se použije v zástupci spouštějícím aplikaci.

Subadresář KOD

- Soubor MOTYLI.EXX s přeloženým kódem aplikace.

Postup instalace databáze na uživatelském počítači

Uživatel, který získá CD ROM a chce si zpřístupnit databázi, postupuje takto:

Vloží CD ROM do jednotky na svém počítači.
1. Spustí program SETUPDB.EXE.
2. Během instalace zvolí adresář, v němž na jeho počítači bude umístěna editovatelná část databázového souboru. Tento soubor vznikne až při prvním spuštění databáze.

Po instalaci se v menu ***Start / Programy*** objeví v některé složce zástupce nebo zástupci sloužící ke spouštění databázových aplikací.

Pokud se databáze instaluje z CD ROM umístěného na jiném počítači v síti nebo ze síťového disku, není nutno diskovou jednotu mapovat. K CD ROM nebo disku lze přistupovat pomocí dlouhého jména ve tvaru:

```
\\jméno_počítače\jmého_share\adresáře\soubor,
```

např.:

```
\\NWSERVER\PUBLIC\DATABASE\SETUPDB.EXE .
```

Instalační program SETUPDB.EXE pracuje takto:

1. Vyhledá soubor SETUPDB.INI ve stejném adresáři, v němž se sám nachází, přečte a verifikuje jeho obsah. V případě, že chybí některý povinný údaj nebo odkazovaný soubor, program vydá chybové hlášení a skončí.
2. Pokud je uveden údaj **InitialMessage**, zobrazí úvodní okno a umožní uživatel volit buď ukončení instalace nebo její pokračování.
3. Požádá uživatele, aby vybral na svém počítači adresář, v němž bude umístěna editovatelná část databázového souboru. Pokud je uveden údaj **PrivateDatabaseDir**, pak tento adresář nabídne jako výchozí.
4. Vyhledá na uživatelském počítači instalaci aktuální verzi **WinBase602**, a pokud ji nenajde, zaregistruje instalaci v adresáři **RegisterInstallation**. Není-li tento údaj uveden, skončí s chybou. (Programy **WinBase602** se v žádném případě nekopírují na počítač uživatele - budou se číst se při každém použití z CD ROM a předpokládají, že na počítači mají k dispozici instalované ODBC. Pokud je jejich provoz z CD ROM příliš pomalý nebo pokud na uživatelově počítači není instalováno ODBC, je nutno provést normální instalaci **WinBase602**.)
5. Zaregistruje databázi v adresáři **DatabasePath** a její pokračování v adresáři zvoleném uživatelem, vše pod jménem **DatabaseName**.
6. Pro každou sekci **Shortcut*x*** vytvoří zástupce s ikonou **ShortcutIcon** a textem **ShortcutCaption** v položce **ShortcutFolder**. Zástupce spouští startovní objekt aplikace **ShortcutApplication**. Je-li uvedeno **DisableLogin=1**, pak zástupce obsahuje příznak, že se má přeskočit přihlašování do databáze. Soubor s ikonou se překopíruje

do adresáře zvoleného v bodě 3. Jsou-li uvedeny soubory s kódem aplikace nebo s nápovědou, zkopíruje je na odpovídající místa.

7. Pokud instalace doběhne úspěšně do konce, zobrazí se v dialogovém okně text **FinalMsg** nebo standardní text, není-li tento údaj uveden.