



# **DEDIT**

## **INTEGROVANÝ NÁSTROJ PROSTŘEDÍ**

### **STUDIOWIN**

Popis nástroje  
Tvorba grafiky pro aplikace automatů zobrazení  
Praktické příklady

edice 06.2015  
verze 2.0

DEDIT – integrovaný programovací nástroj

© MICROPEL s.r.o. 2015  
Ing. Zdeněk Rozehnal,

všechna práva vyhrazena  
kopírování publikace dovoleno pouze bez změny textu a obsahu  
<http://www.micropel.cz>

# OBSAH

|                                          |    |
|------------------------------------------|----|
| 1. Použití .....                         | 3  |
| 2. Grafické prvky .....                  | 4  |
| 2.1. Lišta příkazů.....                  | 5  |
| + Uložit.....                            | 5  |
| + Položka.....                           | 5  |
| + Edit .....                             | 5  |
| 2.2. Kategorie položek .....             | 5  |
| + Výchozí paleta .....                   | 5  |
| + Písma .....                            | 6  |
| + Pozadí .....                           | 6  |
| + Grafické prvky.....                    | 6  |
| + Indexy barev .....                     | 6  |
| + Speciální znaky.....                   | 7  |
| + Datové položky .....                   | 7  |
| 3. Příklady .....                        | 8  |
| 3.1. Pozadí.....                         | 8  |
| 3.2. Ikony a jejich použití.....         | 11 |
| 3.3. Písma.....                          | 14 |
| 3.4. Písma jako symboly .....            | 16 |
| 3.5. Animace pomocí filmového pásu ..... | 17 |
| 3.6. Obrázek jako textura .....          | 19 |
| 3.7. Indexované barvy.....               | 21 |
| 3.8. Datové položky .....                | 24 |

# 1. Použití

DEDIT je nástroj umožňující přípravu grafických objektů (fontů, znaků a bitmap) pro grafické displeje na PLC MICROPEL. Pracovní soubor má příponu \*.DED, obsahuje všechna vložená zdrojová data (obrázky apod.) a též konfigurační nastavení. Výstupem je \*.STP soubor v syntaxi jazyka SIMPLE4, začlenitelný jako knihovní modul do projektů v prostředí StudioWin.

Typický postup prací při integraci grafických objektů do obsluhy displeje:

- spustit prostředí StudioWin
- otevřít existující nebo založit nový projekt a vložit typ automat vybavený grafickým displejem
- do složky grafické modulu vložit soubor nebo soubory obdobným způsobem jako do složky knihovní moduly
- příkazem nebo poklepáním na jméno souboru otevřít soubor k editaci
- upravit obsah souboru a označit položky, které chceme použít v aplikaci
- doplnit program aplikace o vykreslení grafických prvků
- aplikaci přeložit a otestovat funkčnost

## Pravidla pro tvorbu grafických prvků

Automaty MICROPEL jsou vybaveny grafickými displeji s různým rozlišením, je tedy vhodné používat grafické prvky, které jsou s rozlišením kompatibilní. To platí především pro podkladové obrázky, které jsou vykreslovány přes celý displej. Automat tyto obrázky vykreslí pouze tehdy, pokud jejich rozměr přesně odpovídá. Aby tomu tak bylo, postará se o vhodnou velikost samotný nástroj DEDIT, který obrázky upraví na požadovanou velikost. Úprava spočívá v oříznutí části přesahující rozměr displeje a doplnění chybějících částí barvou podkladu. Výsledek většinou neodpovídá požadavkům aplikace a proto je nanejvýš vhodné používat obrázky na míru, tj. v rozlišení, které přesně odpovídá rozměrům displeje použitého automatu.

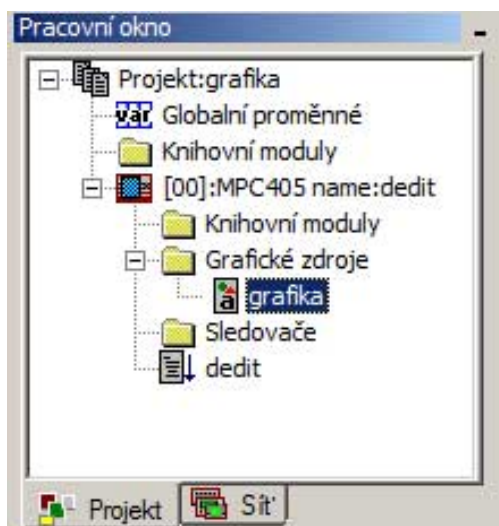
Poněkud jednodušší je situace u obrázků typu ikona nebo u písem. Zde jsou například automaty MPC405 a MT424 kompatibilní, protože fyzická velikost obrázku je na displejích obou automatů přibližně stejná. Stejně tak jsou kompatibilní automaty MT425 a MT470. Vzájemný poměr rozlišení u těchto skupin kompatibilních automatů způsobí, že obrázky kreslené pro MT424 budou na displeji MT425 vykresleny v poloviční velikosti a tudíž budou většinou příliš malé.

## Pojmenování grafických prvků

Protože se jména grafických prvků používají v běžném zdrojovém textu aplikace psaném v jazyce Simple 4, je nutné, aby jména těchto prvků splňovala požadavky kladené na zápis jmen proměnných. Jména proměnných označujeme slovem symbol, protože je nelze tvořit libovolně. Pro symbol platí, že může obsahovat pouze písmena latinky od „A“ do „Z“ nebo od „a“ do „z“, číslice a podtržítka. Symbol musí vždy začínat písmenem nebo podtržítkem. Program DEDIT při zadávání jména grafického prvku provádí kontrolu, zda zadané jméno těmto omezením odpovídá. Vzhledem k tomu, že takto definovaná jména mohou být poněkud strohá, je možné ke každému grafickému prvku připojit obecný popis, případně nápovědu, která je rozšířením tohoto popisu. Nápověda se hodí v případech, kdy se daný soubor grafických prvků používá více uživateli jako knihovní modul. Zde je vhodné, aby měl každý uživatel úplnou informaci o použití prvku ve funkcích knihovny.

## 2. GRAFICKÉ PRVKY

Okno s grafickými prvky je hlavním výchozím bodem pro veškeré definice a úpravy grafických objektů. Dle zvolené platformy cílového displeje jsou v okně zobrazeny dostupné typy grafických objektů.. Např. u modelu MPC300 jsou k dispozici pouze definice uživatelských znaků (cca 8 možných znaků). U modelu MPC400, MT424 atp. s barevným grafickým displejem je k dispozici tvorba celých fontů, obrázků pro podkladovou plochu i obrázků ve formě objektu.



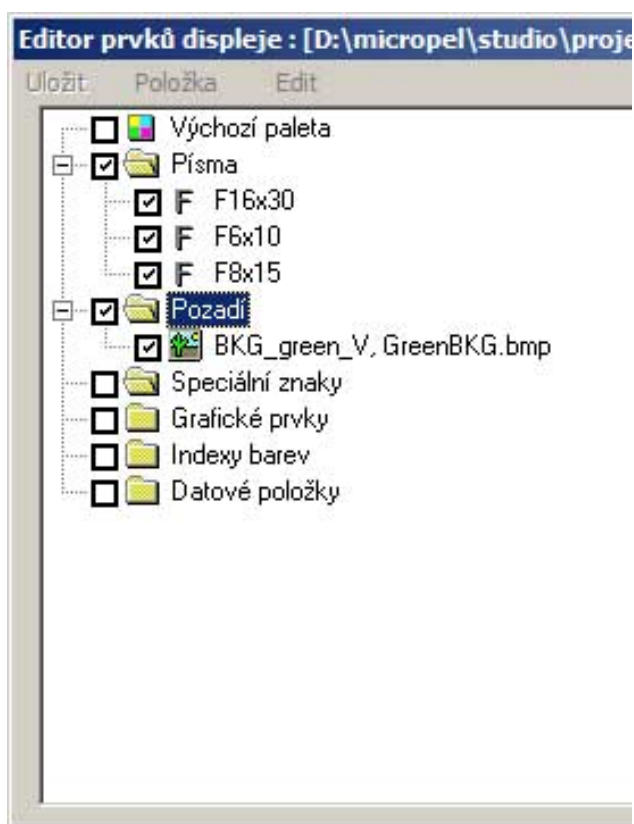
Obr. 2.1 Pracovní okno prostředí StudioWin

Na Obr. 2.1 je zobrazeno pracovní okno prostředí StudioWin s vloženým automatem typu MPC405 a vloženým souborem „grafika“ ve složce „grafické zdroje“. Soubor se vkládá stejným postupem, jako soubor složky knihovnických modulů.

Poklepáním na jméno souboru nebo příkazem otevřít z lokální nabídky, otevřeme soubor pro editaci ve vestavěném editačním nástroji DEDIT tak, jak ukazuje Obr. 2.2.

Okno má v horní části lištu příkazů a na jeho hlavní ploše jsou zobrazeny kategorie objektů (které jsou dostupné pro zvolenou platformu displeje).

Zaškrťovací box před kategorií určuje, zda se všechny objekty dané kategorie budou či nebudou konvertovat/generovat do cílového souboru se zdrojovým textem. Zaškrťovací boxy před jednotlivými položkami pak totéž určují pro jednotlivé položky.



Obr. 2.2 Editací okno nástroje DEDIT

## 2.1. Lišta příkazů

Okno editoru je vybaveno lištou sdružující nezbytné příkazy pro editaci souboru grafických prvků.

### + Uložit

Příkaz slouží k uložení editačních změn grafických prvků do souboru. Příkaz je aktivní pouze v případě, že byly nějaké změny provedeny.

### + Položka

Příkazem vyvoláme nabídku, sdružující příkazy pro editaci položek adresářové struktury, do níž jsou jednotlivé grafické prvky tříděny. Nabídka obsahuje tyto příkazy:

- ☐ **Nová** – Založí do vybrané kategorie (označené modrým zvýrazněným řádkem) novou prázdnou položku - prázdný čistý obrázek, nebo font apod. Takovouto položku je pak třeba naplnit vestavěným editorem programu DEDIT.
- ☐ **Přidat** – Založí do vybrané kategorie (označené modrým zvýrazněným řádkem) položku s předvyplněným obsahem na základě již existujícího obrázku (ve formátu BMP) nebo existujícího fontu (nainstalovaného ve Windows).
- ☐ **Smazat** - Smaže označenou položku.
- ☐ **Import** – Umožní vložit vybrané položky ze souboru nástroje DEDIT

### + Edit

- ☐ **Editovat** – Spustí nad označenou položkou vestavěný editor obrázku nebo editor fontu, znaku, konstanty atp.
- ☐ **Přejmenovat** – Umožní změnit jméno a popis položky. Jméno slouží k identifikaci položky ve zdrojovém textu aplikace, popis pak ke snazší identifikaci prvku co do funkce a použití.
- ☐ **Kopírovat, Vystřihnout, Vložit** – Příkazy představují standardní editační funkce s pomocí schránky systému
- ☐ **Zpřístupnit/Znepřístupnit pro aplikaci** – Přepínač, který umožňuje označit vybranou položku jako nepřístupnou pro editaci v knihovně aplikace. Ve výchozím nastavení je každá položka souboru přístupná k editaci v knihovně aplikace

## 2.2. Kategorie položek

Sdružují položky stejného typu a jsou ve výpisu označeny symbolem adresáře viz Obr. 2.2. Jednotlivé položky představující balíky grafických dat (Písma, Pozadí, Grafické prvky) mají svá jména, nastavitelná s pomocí příkazu „Přejmenovat“. Tato jména slouží jako symboly ve výsledném programu v SIMPLE4, používají se jako odkazy na datové struktury při volání systémových grafických funkcí. Je u nich tedy třeba dodržet konvence pro tvorbu symbolů (nepoužívat mezery, diakritiku ani žádné zvláštní znaky nebo oddělovače).

### + Výchozí paleta

Tvoří samostatnou kategorii. Zobrazuje výchozí paletu barev použitou v automatu a pomocí editoru umožňuje nastavit 16 uživatelských barev, pro které je v paletě vyhrazeno místo. Tyto barvy odpovídají symbolům user\_0 až user\_16 definovaným pro standardní grafické rozhraní automatů MICROPEL.

## + **Písma**

Editor písem (fontů). Co položka, to jedno písmo - sada 256 znaků. Umožňuje buď import hotových písem instalovaných ve Windows (příkazem "Přidat"), nebo vytvoření prázdného písma se zadanými parametry (příkazem "Nová"). Příkazem "Editovat" se otevře editor pro úpravu jednotlivých znaků.

## + **Pozadí**

Rastrové obrázky pro použití ve vrstvě BACKGROUND, mohou mít barevnou hloubku 256 nebo 65535 barev. Co položka, to jeden obrázek. Je vhodné aby byl obrázek ve velikosti přesně odpovídající formátu daného displeje. Pokud připravujeme data pro použití na různých typech automatů a terminálů je vhodné, vytvořit pro každou platformu separátní velikost podkladového motivu, abychom předešli ořezání či doplnění obrázku o body s pevnou základní barvou pozadí na chybějících místech.

Obrázky můžeme importovat z hotového obrázku v některém, z podporovaných formátů příkazem "Přidat", nebo vytvořit prázdnou kreslicí plochu (příkaz "Nová"). Příkazem "Editovat" se otevře editor pro úpravu obrázku. Editor obrázku pozadí umožňuje volbu barevné hloubky (256 barev nebo 65535 barev). Obrázky pozadí se v programu vykreslují funkcí GdiBkgBMP.

## + **Grafické prvky**

Rastrové obrázky pro použití ve vrstvě GRAPHIC, mají barevnou hloubku 256 barev.

Co položka, to jeden obrázek, s nastavenou velikostí. Umožňuje buď import hotového obrázku v některém z podporovaných formátů (příkazem "Přidat"), nebo vytvoření prázdné plochy pro kreslení (příkazem "Nová"). Příkazem "Editovat" se otevře editor pro úpravu obrázku. Obrázky se v programu vykreslují funkcí GdiBMP.

Importovat a kreslit lze obrázky i ve formě tzv. "filmového pásu" - nastavením počtu obrázků více než 1. Pak se výška bitmapy rozdělí počtem obrázků (obrázky jsou de-facto kresleny těsně pod sebou v jedné bitmapě) a pomocí řídicí datové struktury bmpmode, předávané do funkce GdiBMP, je specifikováno i číslo obrázku který má být vykreslen.

## + **Indexy barev**

Každá položka kategorie, představuje sadu indexů 16ti barev, kterou je možné použít pro modifikaci obrazových bodů zvoleného obrázku přímo za běhu aplikačního programu. Obrázek kategorie grafické prvky obvykle tvoříme pomocí pevných barev. Spolu s těmito pevnými barvami však můžeme použít až 16 indexovaných barev, jejichž finální barva bude doplněna až za běhu aplikace. Pro tyto barvy se buď použijí uživatelsky definované barvy nebo odkazy na barvy předané ke kreslení právě zvolenou položkou indexů barev. Pomocí indexů barev je například možné měnit v ikoně média jeho barvu. Takovou ikonu pak můžeme použít pro indikaci studené vody, teplé vody nebo plynu. Dále můžeme zobrazit i různý stav či další fyzikální parametry média jako je například průtok, tlak apod. popisované řešení je mnohem méně náročné na paměť, protože vyžaduje uložit pouze jeden obrázek namísto řešení bez indexování barev, kde musíme vytvořit tolik obrázků, kolik

se rozhodneme zobrazovat kombinací stavů. Příkladem může být médium proudící či neproudící se současnou indikací mezního tlaku a teploty. Při indikaci pouze kritické a běžné hodnoty potřebujeme při řešení bez indexů barev 8 obrazů zatímco při indexaci barev, nám postačí obrázek jediný

### + **Speciální znaky**

Editor znaků systémového fontu. Jméno znaku se používá pro zavolání vygenerované funkce, která znak s daným ASCII kódem předefinuje v systémovém fontu displeje. Např. PLC typu MPC300 umožňují redefinici prvních 8 znaků s kódy 00-07, PLC typu MT201 nebo MPC405 umožňují předefinování všech 256 znaků fontu. Mechanismus tvorby a použití je nejlépe patrný po otevření vytvořeného výstupního \*.STP souboru.

### + **Datové položky**

Tato kategorie je určena především pro tvorbu knihovnic souborů aplikací. Položkou této kategorie je editor hodnoty, textu, bloku textů atp. Tento editor pak umožní uživateli modifikovat běh knihovnic funkcí podle požadavků uživatele. Vhodnou aplikací může být též tvorba aplikace s různými jazykovými mutacemi, či různým nastaveními pro řízení zařízení z vybrané sady typů



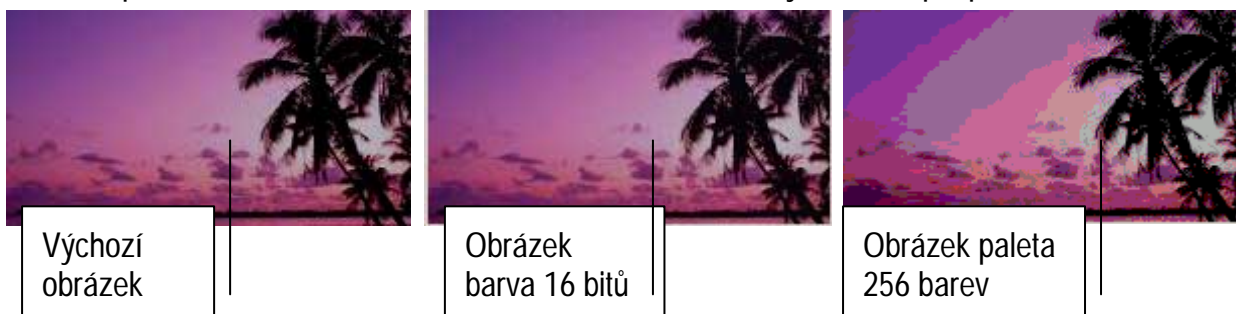
### 3. PŘÍKLADY

Následující odstavce jsou věnovány příkladům použití grafického nástroje DEDIT pro tvorbu dat pro použití při tvorbě aplikace v prostředí StudioWin. Příklady předpokládají znalost základních principů grafického systému automatů MICROPEL. Grafický systém je popsán v příručce „MPC405 – programování v Simple“

#### 3.1. Pozadí

Pozadí grafiky je v grafickém systému MICROPEL tvořeno buď plnou barvou nebo podkladovým obrázkem (tapetou). Podkladový obrázek je vykreslen pouze tehdy, když jeho rozměry plně odpovídají rozměrům displeje automatu. V opačném případě grafický systém obrázek nevykreslí. Grafický nástroj DEDIT automaticky upravuje rozměry generovaného podkladového obrázku podle rozměrů displeje, takže problém, kdy systém podkladový obrázek nevykreslí, není v takovém případě způsoben chybou velikosti. V naprosté většině případů je obrázek překryt neprůhlednou vrstvou textové obrazovky nebo vrstvou grafiky.

I když je možné jednoduchý obrázek nakreslit i pomocí grafického nástroje, je nanejvýš vhodné pro jeho přípravu použít některý ze standardních editorů obrázků. Grafický nástroj DEDIT akceptuje obrázek ve formátu BMP, JPG, GIF, TIF a PNG. V případě že máme obrázek k dispozici vložíme ho do souboru grafiky příkazem „Přidat“. Příkaz spustí standardní dialog pro výběr vhodného souboru. Vyhledáme požadovaný soubor a stiskem tlačítka „Otevřít“ soubor načteme. Na tomto místě je potřeba zdůraznit, že grafický nástroj obrázek načte a následně převede do reprezentace maximální podporované barevné hloubky (16 bitů). Při tomto převodu může dojít k tomu, že některé obrazové body nebudou vypadat v celkovém kontextu obrázku dobře. Proto můžeme pomocí vestavěného editoru nevhodně barevné body a oblasti přizpůsobit.

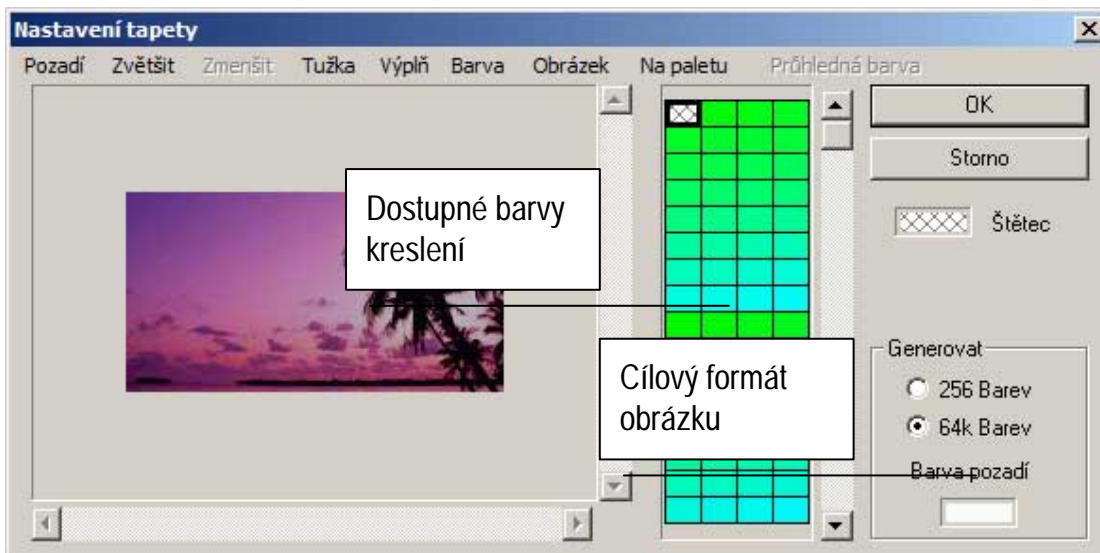


**Obr. 3.1** Náhled zpracování obrázku

Na Obr. 3.1 je ukázán náhled obrázku tak, jak ho zpracuje grafický nástroj pro zobrazení v 16bitové barvené hloubce a v paletě 256 barev. Z obrázku je zřejmé, že pokud chceme použít jako podkladový obrázek fotografii, je nanejvýš vhodné použít formát se 16bitovou barevnou hloubkou. Pokud podkladový obrázek tvoří „pérovka“, vystačíme i s paletou 256 barev. Použitím palety 256 barev se zmenší nároky na paměť konstant automatu na polovinu.

Na Obr. 3.2 je zobrazen editor tapety (podkladového obrázku) z grafického nástroje DEDIT. Horní lišta sdružuje všechny dostupné editační příkazy.





Obr. 3.2 Editor podkladového obrázku (tapety)

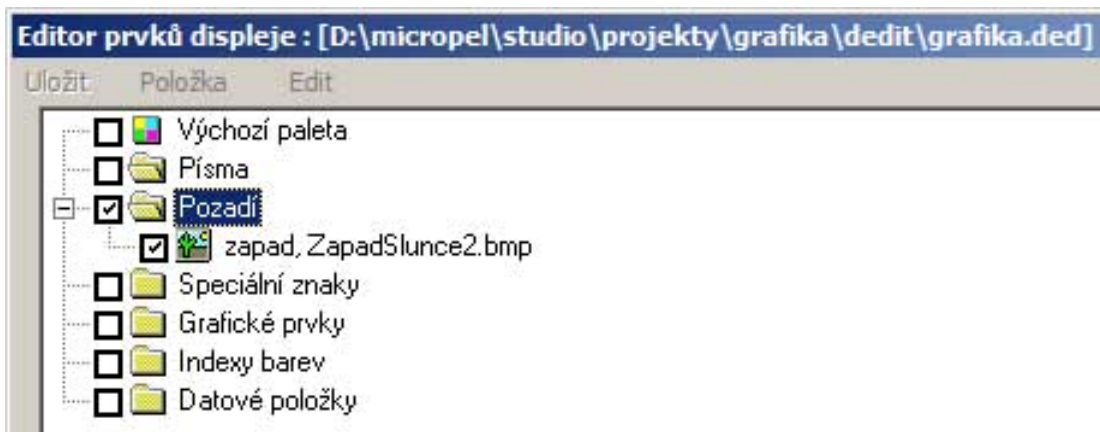
### Příkazy editoru tapety

- Pozadí – příkaz nastavuje barvu pozadí, která bude použita, v případě, že nebude souhlasit rozměr obrázku s rozměrem displeje cílového automatu. Grafický nástroj v takovém případě použije barvu pozadí pro všechny doplňované body do výsledného obrázku
- Zvětšit, Zmenšit – příkazy nastavují velikost zobrazení cílového obrázku pro potřeby editace. Maximální zvětšení umožňuje editaci jednotlivých bodů obrázku, přičemž tyto body jsou v náhledu ohraničeny černou linkou.
- Tužka – editační nástroj pro kreslení jednotlivého bodu
- Výplň – editační nástroj, který nahradí barvu bodů v oblasti novou barvou. Okraje oblasti jsou vyhledány na základě barvy bodu, na který umístíme kurzor a stiskneme levé tlačítko myši.
- Barva – editační nástroj s jehož pomocí nastavujeme barvu štětce (barvu, kterou kreslíme) podle barvy zvoleného bodu
- Obrázek – sdružuje příkazy potřebné pro vložení obrázku
  - Vložit – vloží obrázek z vybraného souboru do pracovní vrstvy editoru. V této vrstvě nastavíme požadovanou pozici obrázku vůči kreslicí ploše. Po nastavení pozice umístíme obrázek do plochy příkazem „Umístit“. Po umístění je obrázek trvale vložen do podkladu a již není možné měnit jeho umístění.
  - Vystředit – umístí střed vloženého obrázku na střed podkladové plochy. Pomocný příkaz pro umístění obrázku
  - Umístit – příkazem se obrázek trvale vloží do podkladové plochy
- Na paletu – příkazem přepínáme zobrazení podkladového obrázku mezi zobrazením pomocí palety 256 barev a 16 bitovou barevnou hloubkou. Příkaz můžeme používat pro náhled, jak by podkladový obrázek vypadal, kdyby se vykreslil pomocí palety barev nebo v případě, že chceme obrázek použít v zobrazení přes paletu. Pokud budeme v zobrazení pře paletu obrázek editovat, budeme ve výběru kreslicí barvy omezeni paletou barev

- Průhledná barva – příkaz vyhledá a zvolí jako kreslicí průhlednou barvu

### Vygenerování podkladového obrázku

Vygenerování podkladového obrázku a jeho použití v kódu aplikace je konečným cílem. Aby byl podkladový obrázek dostupný pro aplikaci, potřeba v editačním okně nástroje DEDIT zaškrtnout položku podkladového obrázku tak, jak to ukazuje Obr. 3.3.

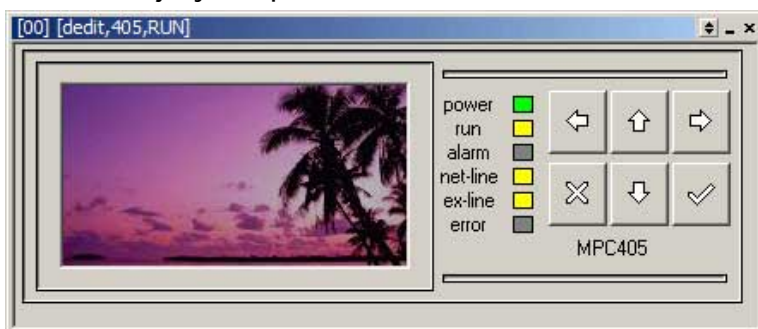


Obr. 3.3 Označení položky v editoru

Pokud máme položku označenou, můžeme s ní pracovat ve zdrojovém textu aplikace. Pokud budeme sledovat popisovaný jednoduchý příklad, budeme předpokládat, že podkladový obrázek se nebude měnit za chodu aplikačního programu. V takovém případě můžeme nastavení podkladového obrázku provést přímo v sekvenci resetu. Celý zdrojový text příkladu bude vypadat:

```
GdiBkgBMP(zapad) ; nastavení podkladoveho obrazku  
reset = 0  
end
```

Zdrojový text přeložíme a zatáhneme do simulátoru, Výsledek ukazuje Obr. 3.4.



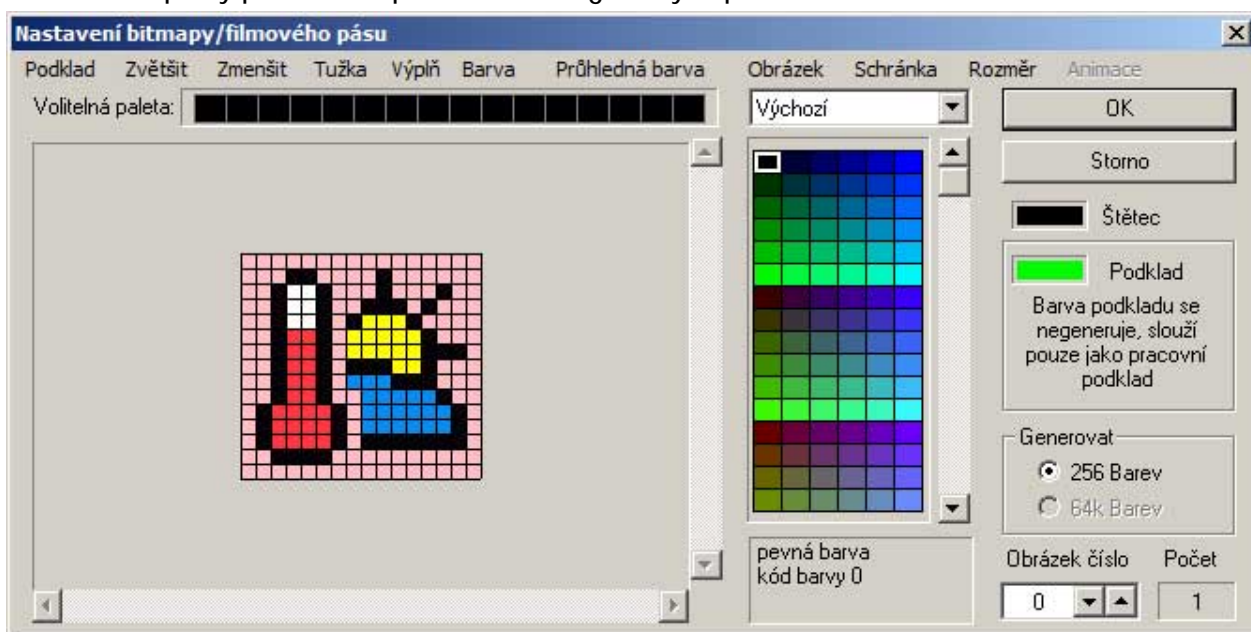
Obr. 3.4 Simulace zdrojového textu vložení podkladového obrázku

### Shrnutí

Pro vložení podkladového obrázku je nutné nejprve obrázek vložit do grafického souboru nástroje DEDIT. Dále je třeba obrázek označit zaškrtnutím, aby se vygenerovala potřebná data. Dále vhodným způsobem umístíme volání systémové podprogramu GdiBkgBMP(„jmeno obrazku“). Pokud potřebujeme obrázky podkladu měnit podle kontextu, je nutné volání podprogramu GdiBkgBMP doplnit vhodným rozhodovacím kódem zdrojového textu.

## 3.2. Ikony a jejich použití

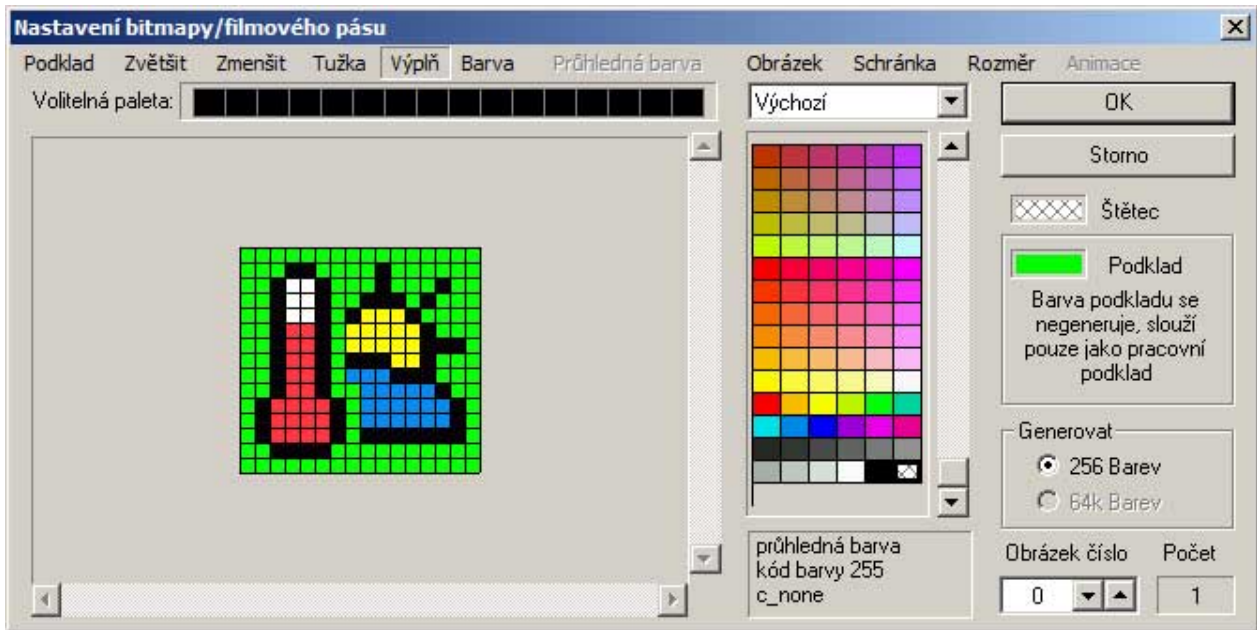
Ikony jsou grafické prvky, které vykresluje v grafické zobrazovací vrstvě. Tato vrstva zobrazuje v paletě 256 barev s tím, že 16 barev z této palety není pevně definováno a uživatel si je může vybrat z barev 16bitové barevné hloubky. Zmíněných 16 barev také slouží k tzv. zobrazování pomocí indexace barev, kdy zobrazovací funkce v průběhu kreslení zaměňují zvolenou barvu za barvu jinou podle seznamu indexů barev. Tato technika je účinná pro indikaci stavu zařízení pomocí barev. Ikony jako takovou doporučujeme vytvořit pomocí vhodného programu pro tvorbu obrázků. Pro běžné potřeby vystačíme s programem „Kreslení“ operačního systému. Po té co ikonu vytvoříme a uložíme do souboru, můžeme ji zařít zpracovávat pomocí nástroje DEDIT. V základním zobrazení editoru grafických prvků zvolíme položku „Grafické prvky“ a z nabídky „Položka“ pomocí příkazu „Přidat“ otevřeme dialogové okno se seznamem podporovaných grafických souborů. Vybereme požadovaný soubor a stiskneme tlačítko otevřít. V následujícím dialogu vybereme volbu „jednotlivé políčko“, přijmeme nabízený rozměr ikony (ten je zjištěn přímo ze souboru) a doplníme jméno ikony tak, aby bylo v souladu s požadavky na symbol jazyka Simple 4. Pokud rozměr upravíme dojde k oříznutí obrázku nebo jeho doplnění potřebnými obrazovými body tak, aby byl dodržen požadovaný rozměr. V průběhu vložení, přepočítá nástroj DEDIT původní barvy obrazových bodů na barvy indexované do palety podporované grafickým systémem automatů. Závěrečné úpravy provedeme pomocí editoru grafických prvků dle.



Obr. 3.5 Editor grafických prvků

Protože požadujeme aby editovaná ikona byla čtvercová ale aby jednotlivé segmenty ikony byly „vyřezány“ podle okrajů, obsahuje obrázek světle růžové body, které pomocí editoru bitmapy nahradíme body s průhlednou barvou. Postup je jednoduchý. Nejprve příkazem „Průhledná barva“ nastavíme průhlednou barvu pera a výplně. V dalším kroku zvolíme příkazem výplň, kreslicí nástroj s jehož pomocí nahradíme barvu zvolených bodů za barvu průhlednou. Nahrazené body poznáme tak, že se zobrazí barvou podkladu.

Pokud se nám nepodaří nahradit všechny body najednou, nahrazujeme je postupným vybarvováním požadované oblasti. Výsledek nahrazení je ukázán na Obr. 3.6. Pokud pomocí příkazu „Podklad“ změníme barvu podkladu, přebarví se vybranou barvou podkladu všechny průhledné body.

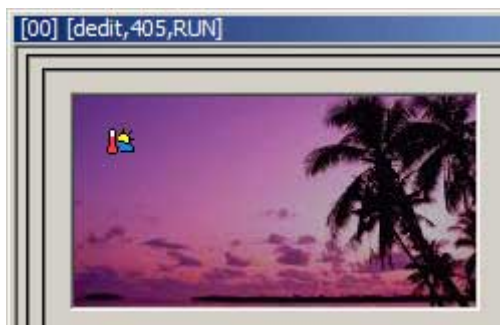


Obr. 3.6 Použití průhledné barvy

Pokud jsem s nahrazením spokojeni uzavřeme editor tlačítkem OK, uložíme změny do souboru nástroje DEDIT.

V dalším kroku umístíme ikonu na podkladový obrázek z příkladu **Chyba! Nenalezen zdroj odkazů..** Kód pro vložení obrázku bude vypadat takto:

```
code bmpmode temp_ext_pos = (
(15,15), ; pozice bitmapy
(16,15), ; velikost pozadovana na displeji
0,      ; cislo zobrazovaneho obrazku
_bmp_00 ; styl zobrazeni - prosty bez otoceni a zrcadleni
)
GdiBkgBMP(zapad) ; nastaveni podkladoveho obrazku
GdiBmp(temp_ext,temp_ext_pos) ; volani zobrazeni
reset = 0
end
```



Obr. 3.7 Zobrazení ikony na podkladu

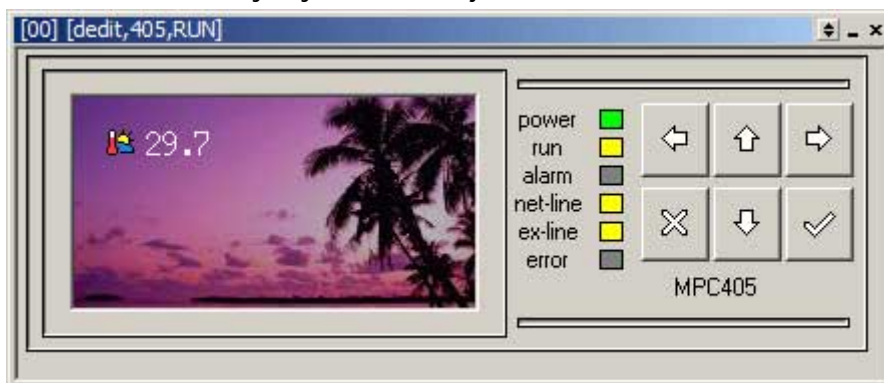
Při editaci uváděného příkladu nesmíme zapomenout na zaškrtnutí ikony v seznamu editačního nástroje okna DEDIT. Pokud ikonu nezaškrtneme, pak editační nástroj nevygeneruje její data a překladač nebude schopen zdrojový kód přeložit. Po zatažení příkladu do simulátoru bude výsledné zobrazení odpovídat Obr. 3.7.



V dalším vylepšení příkladu můžeme doplnit text představující aktuální hodnotu teploty. Pro tuto úlohu je potřeba doplnit zdrojový text o nastavení barvy písma a barvy podkladu pro písmo, dále převést číselnou hodnotu na text a ten výsledně natisknout do grafické vrstvy. Zdrojový text řešení, které není složité bude vypadat takto:

```
code bmpmode temp_ext_pos = (
(15,15), ; pozice bitmapy
(16,15), ; velikost pozadovana na displeji
0,      ; cislo zobrazovaneho obrazku
_bmp_00 ; styl zobrazeni - prosty bez otoceni a zrcadleni
)
code pxy temp_ext_text = (35,15) ; umisteni textu
GdiBkgBMP(zapad) ; nastaveni podkladoveho obrazku
GdiBmp(temp_ext,temp_ext_pos) ; volani zobrazeni
GdiBColor(c_none) ; barva pozadi textu bude pruhledna
GdiFColor(c_white) ; barva textu bude bila
position = _text_virtual_line ; pozice pro tisk virtualniho radku
format = 0x1001 ; tisk na jednodesetinne misto
Display(temp) ; tisk hodnoty na virtualni radek
GdiDisplayText(temp_ext_text) ; tisk textu z virtualniho radku
reset = 0
end
```

Potřebná úprava zdrojového textu vyžaduje pouze pár kroků. Nejdůležitějším poznatkem je tisk do virtuálního řádku. Tisk se provádí stejnými formátovacími funkcemi jako při tisku standardního textu. Jediné co se změní, je nastavení proměnné position. Virtuální řádek začíná na pozici 10000. Toto číslo tedy nastavíme před formátováním tisku. Dále nastavíme formát tisku na jedno desetinné místo a pomocí funkce Display vytiskneme hodnotu teploty na virtuální řádek. Poslední krok spočívá v přenesení obsahu virtuálního řádku do grafické vrstvy. O to se postará funkce GdiDisplayText. Funkci voláme s parametrem typu pxy (souřadnice bodu), který udává počáteční bod tisku hodnoty. Výsledek tisku je vidět na Obr. 3.8.



Obr. 3.8 Dotisk textu do grafické vrstvy

### Shrnutí

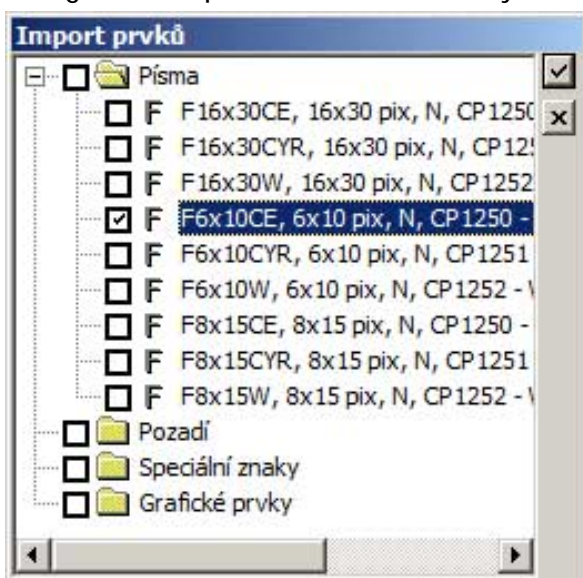
Pro tvorbu grafického zobrazení veličin je vhodné používat grafické prvky ikon doplněné podkladovým obrázkem a textem s hodnotou sledované veličiny. Pro tisk hodnoty využíváme s výhodou znalosti a funkce pro tisk do textové vrstvy obrazovky. Výsledné umístění textu pak volíme pomocí počátečního bodu tisku. V náročnějších případech můžeme využít i ořezový obdélník pro tisk textu, včetně otáčení textu.

### 3.3. Písma

V kapitole 3.2 je ukázán tisk hodnoty teploty u příslušné ikony. Tisk v tomto příkladě nevypadá příliš dobře, protože tisknutý text je příliš vysoký a tedy v nevhodném poměru k ikoně. V takovém případě můžeme buď přizpůsobit velikost ikony nebo textu.

Pokud zvolíme jako řešení změnu velikosti textu, musíme nejprve získat řez písma v požadované velikosti. Písmo je v grafickém prostředí automatů MICROPEL reprezentováno vždy 256 znaky společné výšky. Pokud mají všechny znaky šířku totožnou, jedná se o písmo s pevnou šířkou znaku. V případě, že nastavíme šířku jednotlivých znaků individuálně, získáme písmo proporcionální. Při editaci znaku je nutné počítat s tím, že vyhrazený prostor pro jednotlivý znak je uvažován včetně mezery mezi znaky a mezery mezi řádky.

Pro automaty MICROPEL jsou k dispozici písma o velikosti 6x10, 8x15, 16x30 v jazykových sadách „Latin1-CP1252“, „Latin 2-CP1250“ a „Cyrilic-CP1251“. Tato písma jsou k dispozici v instalačním adresáři StudioWin v souboru „font.ded“. Abychom mohli příslušné písmo použít v našem příkladě, použijeme příkaz „Položka→Import“. Po vyvolání příkazu se otevře standardní dialogové okno pro otevření souboru. Vyhledáme soubor „font.ded“ a otevřeme ho.



Obr. 3.9 Dialogové okno pro import prvků

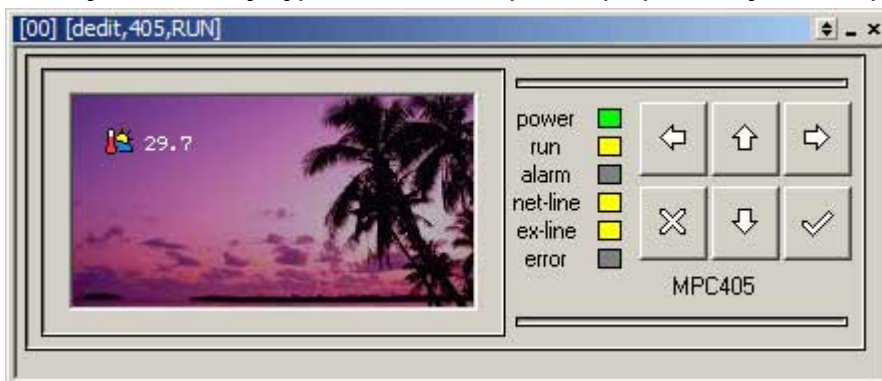
Tím vyvoláme dialog pro import prvků, který je uveden na Obr. 3.9. Zde vybereme ze složky „Písma“ položku „F6x10CE, 6x10 pix, N, CP1250 - Central Europe (Latin2)“ a stiskneme potvrzovací tlačítko pro import . Tím jsme importovali do našeho projektu písmo o velikosti 6x10 obrazových bodů kódové stránky 1250. Vzhledem k tomu, že standardní velikost písma pro automaty řady MPC400 je 8x15 obrazových bodů, bude při použití písma 6x10 výška textů 5 obrazových bodů nižší.

Aby se text vytiskl tímto novým písmem, musíme položku písma v projektu označit ke generování a také musíme do zdrojového textu před tiskovou funkci „GdiDisplayText“ vložit řádek

```
.....  
GdiSetFont(F6x10CE)  
.....
```

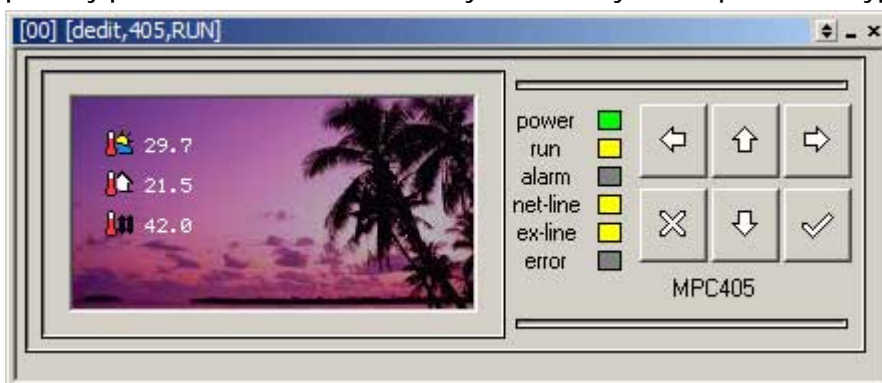
Projekt přeložíme a zatáhneme do simulátoru. Po spuštění programu je vidět, že bude potřeba upravit svislé umístění textu. Změníme tedy souřadnici Y v deklaraci

„code pxy temp\_ext\_text = (35,15)“ z hodnoty 15 na 18. Projekt přeložíme a spustíme. Na je vidět výsledek, který vypadá mnohem lépe než při použití výchozího písma 8x15.



Obr. 3.10 Použití písma 6x10

Vhodnou úlohou pro samostatné zkoušení je doplnění stávajícího příkladu o zobrazení teploty interiéru a teploty okruhu TUV. V první fázi řešení této úlohy si připravíme vhodné obrázky pro ikony. Ty přidáme pomocí grafického nástroje DEDIT do projektu, zdrojový text doplníme o položky pro rozmístění ikon a tisknutých textů. Výsledek pak může vypadat podle Obr. 3.11.



Obr. 3.11 Doplnění zobrazení vnitřní teploty a teploty TUV

### Shrnutí

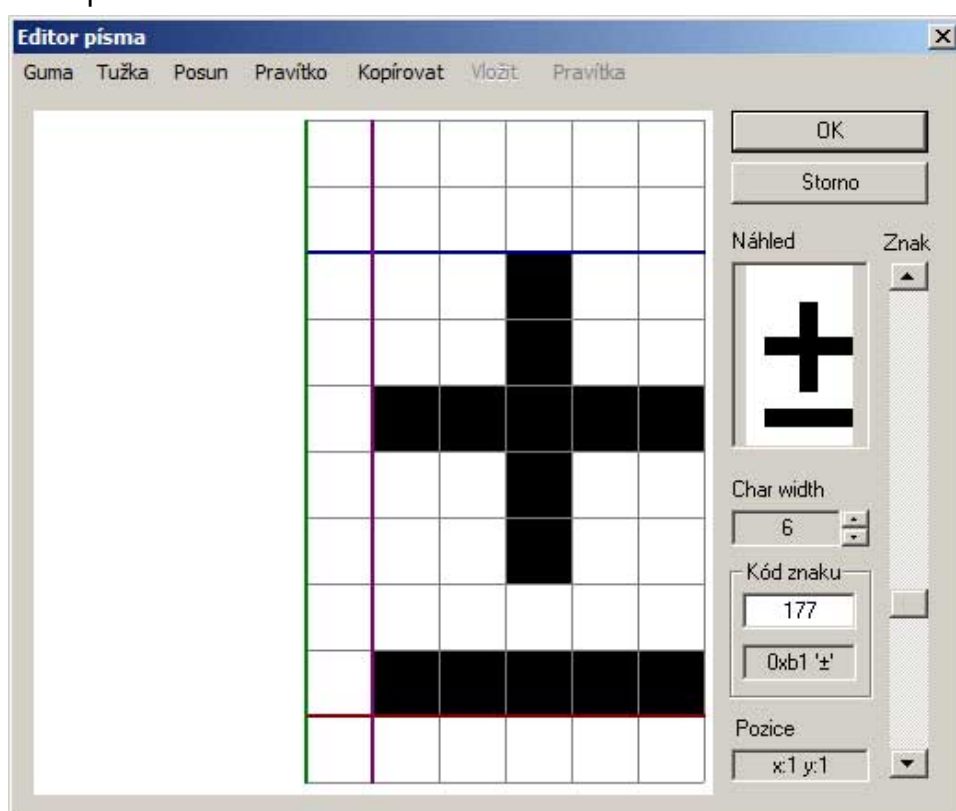
Příklad ukazuje import grafického prvku, použití písma malé velikosti a zobrazení většího množství údajů na displeji automatu. Zde je na místě poznamenat, že písma různých velikostí a barev můžeme při tisku na displej kombinovat. Pomocí písma též můžeme zobrazovat různé speciální symboly podle vlastních představ. Změna písma jeho barvy případně barvu podkladu nijak neprodlužuje zpracování programu. Je však zřejmé, že každý grafický prvek alokuje potřebnou část paměti pro konstantní data. Tato paměť sdílí společný paměťový prostor s pamětí pro program a je tedy zřejmé pravidlo, že čím více grafických prvků tím kratší program může automat pojmout.



### 3.4. Písma jako symboly

V odstavci 3.3 je demonstrováno použití vlastního písma při tisku hodnot teploty v řídicím programu aplikace. Grafické prostředí umožňuje vytvořit vlastní písmo a to buď zcela od začátku příkazem „Položka→Nový“ a nebo pomocí vybraného řezu písma instalovaného v počítači za použití příkazu „Položka→Přidat“. Je nutné podotknout, že i když využijeme možnosti vložení s pomocí instalovaného písma, bude finální úprava písma poměrně zdlouhavý proces díky nekompatibilitě v definici písma automatů MICROPEL s písmi instalovanými v počítači.

Proto MICROPEL dodává s instalací prostředí StudioWin fonty s vhodnou velikostí pro použití na všech typech grafických displejů vyráběných automatů. Možnost editace písma, kterou editor grafiky DEDIT umožňuje, využijeme spíše pro doplnění standardních písem znaky a symboly, které ta či ona aplikace vyžaduje. S výhodou na to využijeme volných znaků podobně jako u definice vlastních znaků pro textové displeje řady MPC300. Příkazem „Edit→Editovat“ otevřeme editor písma dle Obr. 3.12.



Obr. 3.12 Editor písma a jeho znaků

Editor písma zobrazuje náhled editovaného znaku, jeho kód v tabulce znaků a editační okno znaku. Dále obsahuje lištu s volbou editačního nástroje „Guma“ pro mazání bodu znaku, „Tužka“ pro vybarvení bodu znaku, „Posun“ pro posun celého znaku v rámci editační plochy a „Pravítko“ pro posun vybraného pravítka. Příkazem „Kopírovat“ kopírujeme znak do schránky a příkazem „Vložit“ vložíme znak ze schránky na pracovní plochu editoru. Nabídka „Pravítka“ umožňuje zobrazit až 3 svislá a 3 vodorovná pravítka, pro lepší orientaci při editaci znaků.

#### Shrnutí

Pomocí editace znaků zvoleného písma můžeme vytvořit pomocné symboly podle potřeb aplikace. Pokud odpovídají velikosti editovaného písma a písma textové vrstvy, je možné upravené písmo nastavit jako písmo textové obrazovky a používat tak vlastní znaky i v textovém režimu.


### 3.5. Animace pomocí filmového pásu

Pod pojmem filmový pás chápeme řadu stejně velkých obrázků (ikon) umístěných v řadě za sebou. Takto chápou libovolný obrázek i vestavěné funkce grafického rozhraní automatu. S pomocí řídicí datové struktury pro zobrazení obrázků můžeme volit, který obrázek (políčko) filmového pásu bude vykresleno. Tento princip je základem pro realizaci jednoduchých animací.



Obr. 3.13 Animace běhu kotle

Na Obr. 3.13 je ukázka filmového pásu, který se dá použít pro indikaci běhu kotle pomocí animace. Pokud bude kotel vypnut (nebude natápět vodu) budeme zobrazovat první políčko filmového pásu. Pokud bude kotel vodu natápět tj. poběží, budeme s vhodnou časovou základnou zobrazovat střídavě druhý a třetí obrázek.

Jako první krok v řešení příkladu vytvoříme filmový pás. Vložíme tedy nový obrázek do složky „Grafické prvky“. Abychom nemuseli nastavovat velikost, provedeme vložení obrázku pomocí příkazu „Přidat“. Budeme vkládat jednotlivý obrázek , který bude symbolizovat vypnutý kotel. Obrázek otevřeme v editoru a růžové obrazové body nahradíme průhlednou barvou. V dalším kroku pomocí příkazu „Obrázek→Přidat za“ přidáme další obrázky filmového pásu. Všimneme si, že po každém vložení se zvýší počet obrázků o jeden. Jednotlivé obrázky a jejich zobrazení a editaci volíme prvkem „Obrázek číslo“. Příkazem „Animace“ můžeme vyvolat náhled, jak bude animace vypadat na displeji reálného automatu.

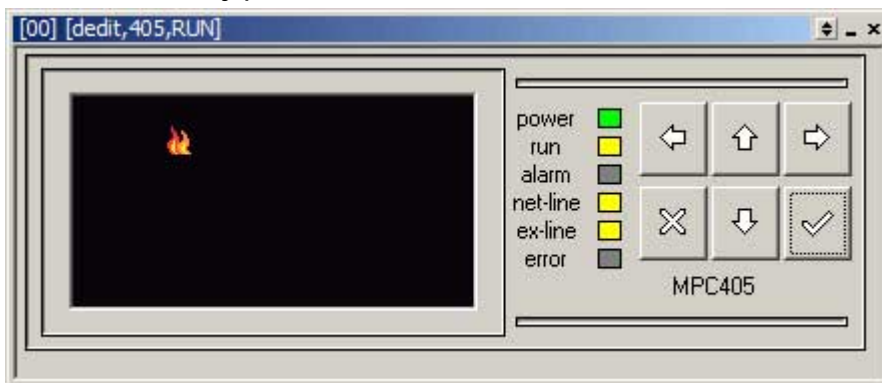
Animaci vyzkoušíme na následujícím zdrojovém kódu.

```
var bmpmode flame_pos;
var longword flame_time
var bit flame_enable
if reset then begin
    flame_pos.org.x = 45 ; inicializace ridici promenne pro animaci
    flame_pos.org.y = 15
    flame_pos.sz.x = 16
    flame_pos.sz.y = 15
    flame_pos.mode = _bmp_00
    flame_pos.pn = 0
flame_enable = 0
end
; spousteni animace stiskem klavesy ENTER
if kbcodes = KB_ENT then flame_enable = flame_enable'

if flame_enable then begin
if ((GetTickCount() - flame_time) > 500) then begin
    flame_pos.pn = flame_pos.pn + 1 ; rizeni animace a casovani
    if flame_pos.pn > 2 then flame_pos.pn = 1
    flame_time = GetTickCount();
end
end
else begin
flame_time = GetTickCount();
flame_pos.pn = 0 ; stav vypnuto
end
```

```
GdiBmp(flame,flame_pos) ; volani zobrazeni
reset = 0
end
```

Pro potřeby animace musíme deklarovat proměnou „flame\_pos“ typu „bmpmode“, kterou budeme řídit zobrazení zvoleného obrázku z filmového pásu. Ze struktury této proměnné použijeme pouze položku „pn“, která určuje číslo zobrazovaného obrázku. Ostatní položky zůstanou neměnné tak jak je inicializujeme po resetu automatu. Spouštění a vypínání kotle budeme simulovat stiskem klávesy „Enter“. Při stisku budeme negovat pomocnou bitovou proměnnou „flame\_enable“. Pro potřeby časování animace budeme deklarovat proměnnou „flame\_time“ typu longword. Časování animace je patrně nejsložitější část celého zdrojového textu. Princip spočívá v odměření časového intervalu pro přepínání jednotlivých obrázků animace. K tomuto účelu využijeme funkci GetTickCount(), která vrací počet milisekund, které uběhly od spuštění automatu. Pokud tedy od aktuálního času vráceného touto funkcí odečteme čas počáteční, zjistíme časový interval, který uběhl od počátečního času do okamžiku volání funkce GetTickCount. Pak již stačí tento rozdíl porovnat se zvolenou hodnotou (zde 500ms). Pokud je rozdíl větší než tato hodnota, znamená to, že požadovaný časový interval uplynul. V tomto okamžiku přepneme na následující obrázek animace a do proměnné flame\_time poznameneáme nový počátek měření. V případě, že je proměnná flame\_enable nulová (kotel je vypnutý), zobrazujeme trvale obrázek s indexem 0 a při každém průchodu nastavujeme počáteční čas animace. To je nezbytné abychom správně odčasovali časový první interval animace.



Obr. 3.14 Ukázka obrazovky v okamžiku animace

### Shrnutí

Na Obr. 3.14 je ukázán okamžik, kdy je v programu spuštěna animace. Animaci spouštíme a zastavujeme stiskem klávesy ENTER. Uvedený zdrojový text je pouze náznakem pro využití filmového pásu a má skutečné možnosti pouze demonstrovat. Vhodným příkladem pro animaci může být zobrazení proudícího média, běžících motorů a mnohé další.

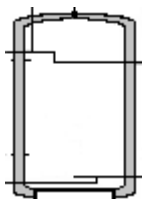
### 3.6. Obrázek jako textura

Textura se hodí na zobrazení média či materiálu a to ať už se jedná o hladinu, či pohyb média v potrubí apod. Ve funkci textury použijeme obrázek, který zobrazuje základní motiv symbolizující daný materiál. Na Obr. 3.15 je motiv použitý pro vodu. Motiv je zvětšen, protože obrázek má rozměr 6x4 obrazové body.



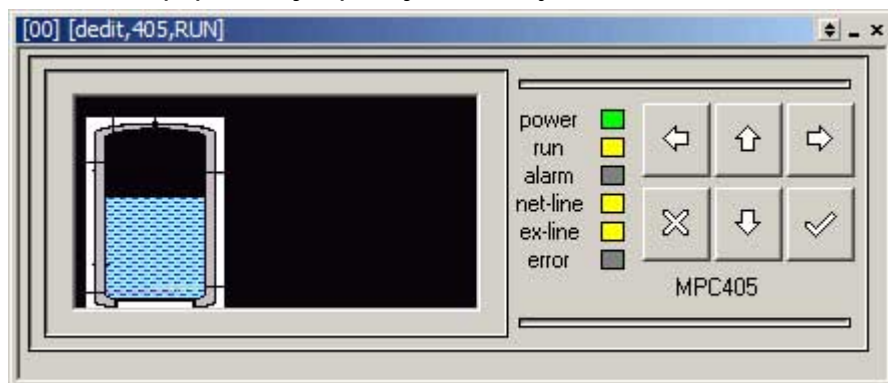
Obr. 3.15 Textura pro zobrazení hladiny vody

V úloze k demonstraci zobrazení hladiny vody v nádrži použijeme proměnnou sekundy, jejíž hodnotu budeme zobrazovat jako hladinu v nádrži. Hodnota proměnné sekund tedy slouží jako simulace výstupu čidla pro výšku hladiny. Pro zobrazení nádrže použijeme Obr. 3.16.



Obr. 3.16 Obrázek nádrže

Princip zobrazení hladiny je postaven na kreslení obdélníku vyplněného texturou vody. Šířka obdélníku bude odpovídat největší šířce uvnitř nádoby, kterou budeme chtít vyplnit. Výšku obdélníku budeme řídit proměnnou „second“ Po nakreslení obdélníku s texturou, nakreslíme přes tento obdélník obrázek nádrže. Tím dojde k naklívání obou obrázků na sebe tak, že v místech, kde má obrázek průhledné body, bude vidět obdélníček s texturou vody. Výsledek simulace zobrazení s popisovaným překrytím ukazuje Obr. 3.17



Obr. 3.17 Zobrazení hladiny v nádrži

Zdrojový text je ve svém principu jednoduchý. Nejprve si připravíme oba obrázky pomocí grafického nástroje. Dále zapíšeme tisk dvojice obrázků přes sebe s tím, že provedeme nastavení výšky obdélníku s texturou vody podle hodnoty proměnné „second“. Největší složitost představuje výpočet správné vzájemné pozice obou obrázků, protože obrázky vyžadují zadat výchozí bod, který je umístěn vlevo nahoře a rozměr, který se počítá od tohoto bodu směrem vpravo a dolů. Zobrazovaná hladina jde ale proti tomuto systému souřadnic, protože roste směrem vzhůru. Tak bude mít proměnná second vliv nejen na výšku obdélníku s texturou ale i na jeho počáteční bod ve svislém směru. Zdrojový text tedy bude vypadat takto.

```

code bmpmode nadrz_pos = (
(5,10), ; pozice bitmapy
(69,100), ; velikost pozadovana na displeji
0, ; cislo zobrazovaneho obrazku
_bmp_00 ; zobrazeni - proste bez otoceni a zrcadleni
)
var bmpmode voda_pos;
if reset then begin
    voda_pos.org.x = 10 ; inicializace ridici promenne pro animaci
    voda_pos.org.y = 100
    voda_pos.sz.x = 55
    voda_pos.sz.y = 0
    voda_pos.mode = _bmp_00 | _bmp_tile ; rezim vyplne bitmapou
    voda_pos.pn = 0
end
voda_pos.org.y = int(100 - second) ;nastaveni pozice
voda_pos.sz.y = int(second) ;nastaveni vysky
GdiBmp(voda,voda_pos) ; volani zobrazeni vyplne
GdiBmp(nadrz,nadrz_pos) ; volani zobrazeni nadrze
reset = 0
end

```

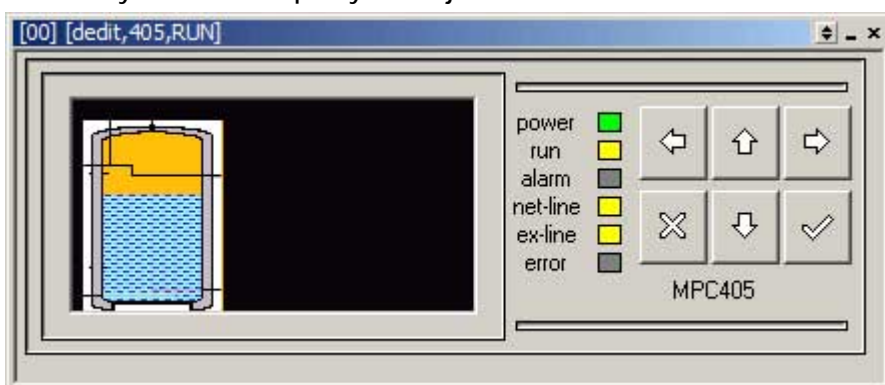
Uvedené řešení má slabinu v tom, že místo nad hladinou vody v nádrži je vyplněno zvolenou barvou podkladu případně podkladovým obrázkem., To je proto, že je celý vnitřní prostor nádrže vyplněn průhlednou barvou. Abychom zmíněný jev potlačili, přidáme pod obrázky textury a kotle obdélník s vhodnou podkladovou barvou. Do zdrojového kódu tedy stačí přidat před kreslení obrázků dva řádky pro vykreslení obdélníku.

```

. . . . .
voda_pos.sz.y = int(second) ;nastaveni vysky
GdiFillColor(c_orange)
GdiFillRect(nadrz_pos.sz,nadrz_pos.org)
. . . . .

```

Výsledek této úpravy ukazuje Obr. 3.18.



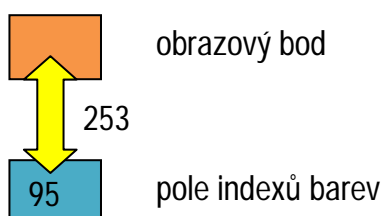
Obr. 3.18 Úprava kreslení podkladu nádrže

### Shrnutí

Jak je ukázáno v příkladu, představuje kreslení ve vrstvách s využitím průhledných bodů silný nástroj pro tvorbu názorných vizualizací přímo na displeji automatu. Pokud spojíme možnosti výplně texturou, průhledné obrazové body a animaci, můžeme dosáhnout velmi dobré úrovně zobrazení vizualizačních schémat.

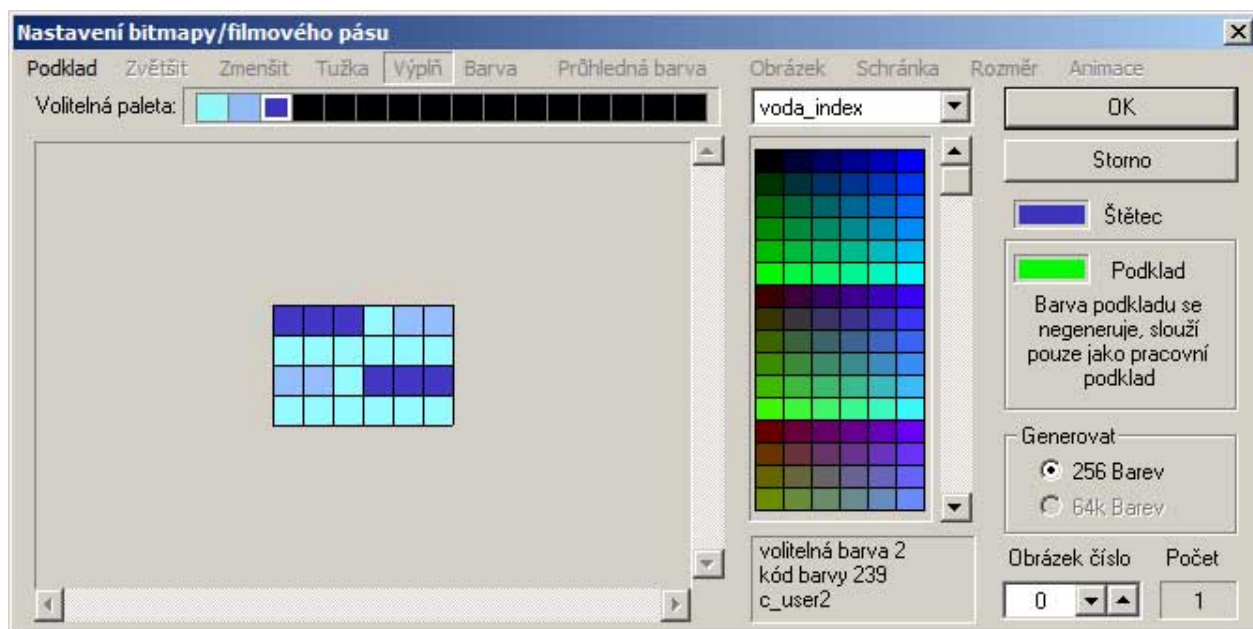
### 3.7. Indexované barvy

Indexované barvy představují účinný nástroj, pro minimalizaci počtu potřebných obrázků při indikaci různých stavů zařízení. Využití pak naleznou i v případě textur použitých pro zobrazení různých médií a jejich stavů. Grafický systém automatů MICROPEL nabízí až 16 barev, které je možné použít indexovaně. Tyto barvy jsou situovány do oblasti palety do 16ti uživatelsky definovaných barev. Indexy barev této oblasti jsou od 237 do 252. V případě indexovaných barev se tento prostor indexů palety využívá ve smyslu odkazu na barvu palety.



Obr. 3.19 Princip indexace barvy

Na Obr. 3.19 je ukázán princip indexace barev použitý ve funkci pro vykreslení obrázku do grafické vrstvy displeje. Původní obrazový bod obrázku má index 253 tj. první s uživatelských barev a tato uživatelská barva je v paletě nastavena na oranžovou. Funkci pro kreslení obrázku předáme krom datové struktury obrázu ještě pole 16 bajtů, které obsahuje indexy barev. První byte tohoto pole má hodnotu 95 a představuje tak modrou barvu umístěnou v paletě na indexu 95. V případě, že kreslicí funkce narazí na bod, který má v původním obrázku index 253 a představuje tak první uživatelskou barvu, nahradí tuto barvu barvou z pole indexů barev. Je to jako kdyby byl bod v původním obrázku aktuálně nakreslen barvou s indexem 95. Je zřejmé, že tímto postupem je možné efektivně měnit barvy vybraných bodů a ploch v obrázku.

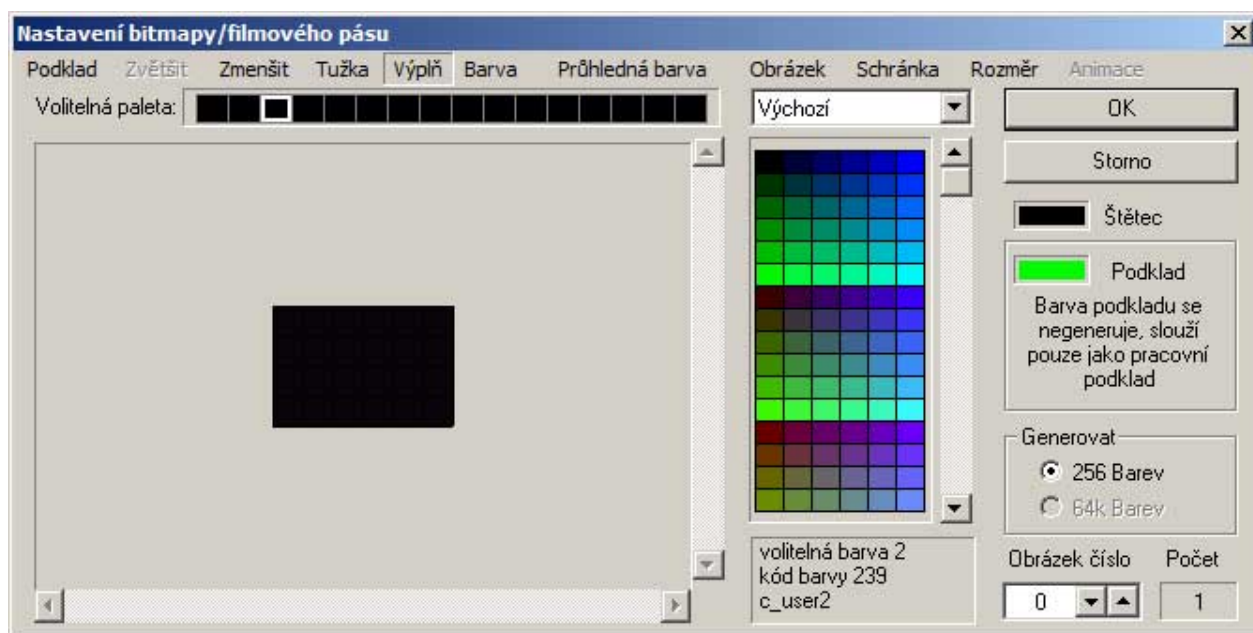


Obr. 3.20 Použití indexů barev při editaci obrázku

Abychom mohli používat indexování barev, musíme nejprve upravit obrázek pro použití indexovaných barev a následně definovat potřebný počet polí indexů barev. To vše uděláme



snadno v grafickém nástroji. Vyjdeme z příkladu z odstavce 3.6 a texturu vody nakreslíme v indexovaných barvách.



**Obr. 3.21 Změna volitelné palety na výchozí**

Nejprve tedy budeme definovat indexované barvy pro vodu. Obrázek vyžaduje tři barvy modré (kód 143, 137 a 46) a tak vlastně využijeme z pole 16-ti indexů pouze první tři položky. Otevřeme editor grafických prvků a přidáme do složky „Indexy barev“ položku „voda\_index“ a první tři barvy nastavíme na kódy 143, 137 a 46 a uzavřeme editační okno. Nyní zkopírujeme obrázek voda do nové položky medium a nový obrázek medium otevřeme v editačním okně obrázku.

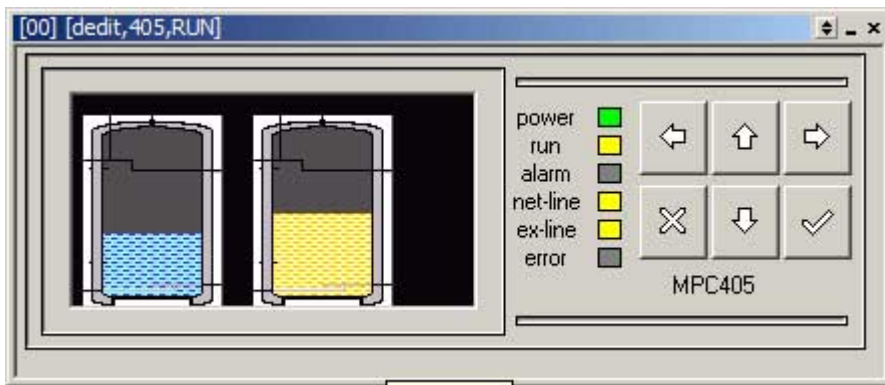
**Nyní zvolíme v okně indexů barev vytvořené pole indexů „voda\_index“. V položce volitelná paleta se objeví definované barvy pole „voda\_index“. S pomocí této volitelné palety přebarvíme obrazové body podle původní předlohy. Situaci ukazuje**

**Obr. 3.20. Na první pohled se tedy nic nezměnilo. Pokud se ale změním volitelnou paletu z „voda\_index“ na výchozí. Překreslí se nám všechny obrazové body na černou barvu, protože výchozí volitelná paleta obsahuje ve všech polích uživatelských barev (volitelné palety) černou barvu. Tato změna je ukázána na**

Obr. 3.21

Nyní můžeme definovat podobnou paletu například pro plyn. Zde se užívá obecně žlutá barva a tak pro plyn nadefinujeme tři odstíny žluté. Pro vykreslení hladiny vody a plynu v zásobnících nám bude tedy stačit textura „medium“ kreslená ve volitelné paletě, obrázek nádrže a podkladový obdélník, který v tomto případě nakreslíme vhodnější tmavošedou barvou.





Obr. 3.22 Použití indexovaných barev

Výsledek zobrazení ukazuje Obr. 3.22. Zdrojový text představuje oproti tomu z odstavce 3.6 principiálně pouze malé úpravy. Zde je.

```

code bmpmode nadrz_voda_pos = (
(5,10), ; pozice bitmapy
(69,100), ; velikost pozadovana na displeji
0, ; cislo zobrazovaneho obrazku
_bmp_00 ; zobrazeni - proste bez otoceni a zrcadleni
)
code bmpmode nadrz_plyn_pos = (
(90,10), ; pozice bitmapy
(69,100), ; velikost pozadovana na displeji
0, ; cislo zobrazovaneho obrazku
_bmp_00 ; zobrazeni - proste bez otoceni a zrcadleni
)
var bmpmode voda_pos;
var bmpmode plyn_pos;
if reset then begin
    voda_pos.org.x = 10 ; inicializace ridici promenne pro animaci
    voda_pos.org.y = 100
    voda_pos.sz.x = 55
    voda_pos.sz.y = 0
    voda_pos.mode = _bmp_00 | _bmp_tile
    voda_pos.pn = 0
    Copy(@plyn_pos,@voda_pos) ; kopie nastavení
    plyn_pos.org.x = 95 ; úprava umístění ve vodorovném směru
end
GdiFillColor(c_darkgray)
GdiFillRect(nadrz_voda_pos.sz,nadrz_voda_pos.org) ; podklad pro vodu
GdiFillRect(nadrz_plyn_pos.sz,nadrz_plyn_pos.org) ; podklad pro plyn
voda_pos.org.y = int(100 - second) ; vypocet výšky media pro vodu
voda_pos.sz.y = int(second)
plyn_pos.org.y = int(100 - (second + 10))
plyn_pos.sz.y = int(second + 10) ; vypocet výšky media pro plyn
GdiBmp(medium,voda_pos,voda_index) ; zobrazeni vody volitelnou paletou
GdiBmp(medium,plyn_pos,plyn_index) ; zobrazeni plynu volitelnou paletou
GdiBmp(nadrz,nadrz_voda_pos) ; zobrazeni nadrze pro vodu
GdiBmp(nadrz,nadrz_plyn_pos) ; zobrazeni nadrze pro plyn
reset = 0
end

```

## Shrnutí

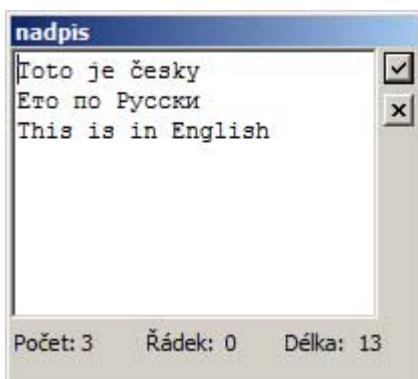
Z uvedeného příkladu je zřejmá výhoda dynamické změny barev obrázku. Obrázek můžeme mít pouze jeden a volitelnou paletou ho můžeme přizpůsobit pro různé druhy zobrazení. Dále je vhodné upozornit na možnost kombinace volitelné palety a animace a současně i na možnost používat ve volitelné paletě průhlednou barvu a též stejnou barvu pro více položek palety. Tím se násobí seznam všech variant, která nám tento systém dovolí použít při vykreslování obrázku.

## **3.8. Datové položky**

Datové položky jsou tvořeny různými typy editorů a umožňují volbu a editaci konstanty, řetězce, tabulky konstant a datových struktur. Tyto položky využijeme v případě, že tvoříme grafický soubor jako doplněk programové knihovny. Zdrojový text knihovny může s výhodou využívat data z grafického souboru a tudíž můžeme chování knihovny na tato data s výhodou navázat a umožnit tak úpravu chování knihovny. Z tohoto důvodu může vypadat použití editovatelné datové položky jako zvláštní, nicméně pokud je knihovna hotova a napojení na data je korektní, uplatní se editory položek včetně validačních funkcí editované hodnoty.

K dispozici jsou následující typy editorů.

- ☐ **Numerický** – editor číselné konstanty libovolného datového typu, umožňuje editovat v pevné nebo plovoucí řádové čárce, k dispozici je i formát času a data
- ☐ **Předvolba** – editor představuje datový typ longword editovaný po jednotlivých bitech, které jsou pojmenovány a editor nastavuje vždy pouze jeden z těchto bitů
- ☐ **Seznam textů** – editorem vytváříme běžnou tabulku textů
- ☐ **Tabulkový** – editor umožňuje navrhnout tabulku editorů stejného nebo různých typů. Tabulka může mít až tři rozměry tj. pokud jsou položky ve všech buňkách třírozměrné tabulky totožné, obdržíme tabulku konstant deklarovanou se třemi indexy
- ☐ **Textový** – editor vytváří samostatný textový řetězec, rozdíl oproti „Seznamu textů“ spočívá ve vygenerovaném zdrojovém textu, kde se na řetězec odvoláváme jménem, zatímco u seznamu se odvoláváme jménem seznamu a indexem řetězce
- ☐ **Výběrový** – editor představuje datový typ longword editovaný po jednotlivých bitech, které jsou pojmenovány a editor umožňuje nastavit libovolnou kombinaci pojmenovaných bitů, ostatní bity zůstávají vynulovány.
- ☐ **Výčtový** – editor obsahuje seznam pojmenovaných hodnot a umožňuje tak nastavit některou z nich podle jména



**Obr. 3.23 Editor textových řetězců**

U všech editorů musíme zadat symbol, pod kterým bude editační konstanta dostupná ve zdrojovém textu. Dále máme k dispozici položku „popis“, kde můžeme stručně naznačit co dotýčný editor nastavuje či edituje. Musíme též vyplnit validační parametry tj. například minimum a maximum atp. Pokud připravujeme editory pro použití ve spolupráci s knihovnou můžeme využít i položka nápověda, s jejíž pomocí doplníme detailní popis funkce, kterou editorem modifikujeme. Popis nápovědy umožňuje vkládat obrázky, volit barvu a styl písma.

Použití editorů si ukážeme na tvorbě aplikace pro různé jazykové mutace. Nejprve tedy vložíme do datových položek editor typu „Seznam textů“. Maximum řetězců nastavíme na předpokládaný počet podporovaných jazyků. Maximální délku řetězce nastavíme na hodnotu podle předpokládané maximální délky řetězce. Ta je obvykle daná formátováním textu na obrazovce. Dále vyplníme seznam textů jednotlivými texty podle Obr. 3.23. Druhý text v pořadí je v ruštině a abychom ho mohli napsat, musíme mít nainstalovanou ruskou klávesnici.



**Obr. 3.24 Vložená písma potřebných kódových stránek**

Abychom mohli texty korektně zobrazit, musíme též ze souboru fonty importovat potřebná písma v daných kódových stránkách. Pro jednoduchost budeme importovat písma v rozměru 8x15, která můžeme použít i v textové vrstvě. Import ukazuje Obr. 3.24.

Na závěr vložíme editor pro výběr jazyka, zvolíme výčtový typ a jednotlivým položkám přiřadíme hodnoty 0 pro česky, 1 pro rusky a 2 pro anglicky. Výčtový editor ukazuje Obr. 3.25.



**Obr. 3.25 Výčtový editor volby jazyka**

Nyní ukážeme jednoduchý zdrojový text, který zobrazuje na displeji textový řetězec ve zvolené jazykové mutaci. Zdrojový text je:

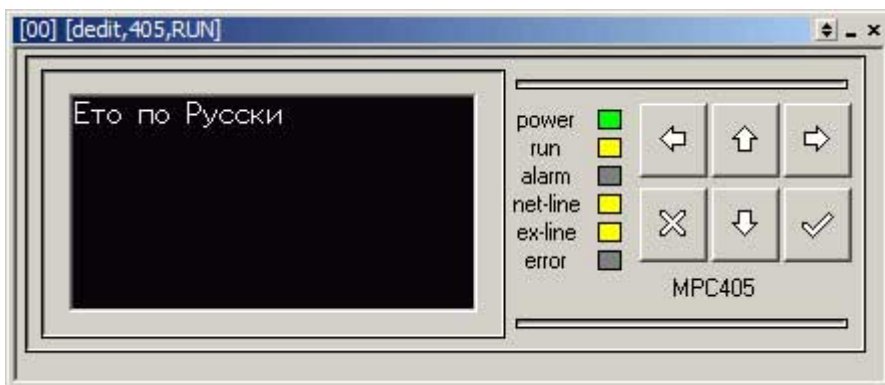
```
if reset then begin
```

```

switch(jazyk)
case 0: DisplaySetFont(f8x15ce)      ; volba kodove stranky podle jazyka
case 1: DisplaySetFont(f8x15cyr)
default:DisplaySetFont(f8x15w)
end
end
position = 0
Display(nadpis[jazyk]) ; zobrazení textu ve zvoleném jazyce
end
reset = 0
end

```

Na Obr. 3.26 je ukázán výsledek příkladu volby jazyka na simulátoru.



Obr. 3.26 Výsledek zobrazení ve zvoleném jazyce na simulátoru

### Shrnutí

Uvedený příklad uvádí pouze ilustrativně použití datových položek a není rozhodně vyčerpávající. Na druhou stranu jednoduchým způsobem ukazuje princip jejich použití.