Skriptování v DetStudiu

Abstrakt

Aplikační poznámka ukazuje příklady využití skriptu v aplikaci.

Autor: Zbyněk Říha
Dokument: ap0023_cz_01.pdf

Příloha

Obsah souboru: -

-	Není

Obsah

Historie revizí

Verze	Datum	Změny
001	14. 9. 2009	Nový dokument

Související dokumentace

1) Nápověda k návrhovému prostředí DetStudio
   soubor: DetStudioHelp.chm

1. Skript

Pomocí skriptu lze v editoru obrazovek, na základě událostí, které v řídicím systému nastaly (nezávisle na vykonávání procesů), ovlivňovat vlastnosti prvků na obrazovkách a hodnoty proměnných či aliasů. Takovým způsobem lze např. dynamicky ovlivňovat sled jednotlivých obrazovek, vzhled jednotlivých prvků.

Bez skriptu je možné s Basic prvky DetStudia vytvářet rozsáhlé, avšak funkčně omezené aplikace spíše ovládacího typu (s omezenou zpětnou odezvou). Pozice prvků, jejich viditelnost, jazyk a další vlastnosti jsou předem dány, sled jednotlivých obrazovek nelze dynamicky ovlivňovat. Bez skriptu lze jen těžko vytvářet reakce na chybové stavy a jejich následné kvitace.

Pří správném nasazení skriptu jsme schopni vytvářet mnohem efektnější a funkčně bohatší aplikace, než s Basic prvky. Obecně lze říci, že funkčnost, která není přímo obsažena v prvcích pro parametrizaci obrazovek v DetStudiu lze doprogramovat pomocí skriptu.

K prvkům na obrazovkách se přistupuje jako k objektům, které mají své jméno (shodné se jménem prvku), své vlastnosti a metody.

1.1. Skriptovací jazyk

Skriptovací jazyk použitý v editoru obrazovek je typu ST (Structured text) a má následující vlastnosti:

- příkazy jsou oddělené znakem středník (";")
- může být více příkazů na řádku
- ve skriptu se lze odvolávat na vybrané vlastnosti prvků na libovolné obrazovce a na hodnoty proměnných
- nejsou podporovány deklarace jakýchkoliv proměnných či deklarace konstant
- pro větvení programu je možné použít pouze příkaz if

Při psaní kódu lze také s výhodou použít menu intellisense vyvolané klávesovou zkratkou Ctrl+J. Po vyvolání menu z něj můžeme vybírat jednotlivé vlastnosti, prvky atd. Výběr se mění v závislosti na místě vyvolání ve skriptu. Pokud napíšeme za jménem prvku tečku, zobrazí se seznam všech pro prvek dostupných metod a vlastností.

Pomocí skriptovacího jazyka lze v obrazovkách vytvářet také vlastní procedury.

Veškeré další informace o možnostech skriptovacího jazyka, včetně popisu práce s ním, lze nalézt v nápovědě k návrhovému prostředí DetStudio.

2. Příklady využití skriptu

2.1. Vyvolání alarmové obrazovky

Při obsluze terminálu je v aplikacích vyžadováno, aby v případě, kdy nastane alarm, byla uživateli okamžitě zobrazena alarmová obrazovka, nezávisle na obrazovce, ve které se momentálně uživatel nachází. Toto lze řešit pomocí obrazovky Global a její události OnRefresh().

Za předpokladu, že máme nadefinovánu např. proměnnou „Err_Alarm“, jejíž jednotlivé bity odpovídají různým alarmům v technologii, by kód pro zobrazení požadované obrazovky (např. s názvem Alarmy) vypadal následovně.

```
event Global_OnRefresh()
     If Err_Alarm > 0 Then //zjištění, zda je nějaký alarm
          Alarmy.Show();   //pokud je alarm, otevře se obrazovka Alarmy
     EndIf;
end;
```

Reakční doba řídicího systému, na zobrazení obrazovky s názvem Alarmy, je dána časem, dosazeným ve vlastnostech obrazovky Global za parametr `RefreshPeriod`.

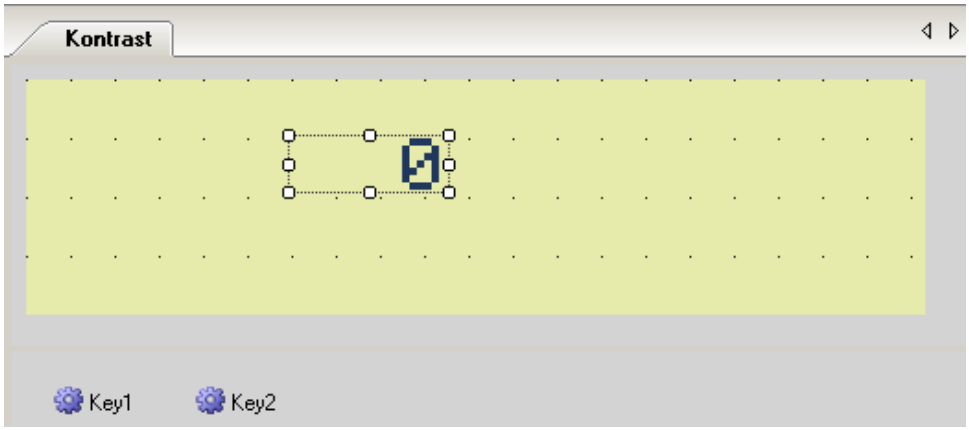
Pozor
Pokud je požadovaná obrazovka již otevřená, znovu se neotvírá. Pro uživatele to znamená, že při snaze otevřít již otevřenou obrazovku se nevykoná ani její událost OnClose() ani OnOpen().

2.2. Nastavení jasu a kontrastu

Nastavení jasu podporují pouze vybrané typy řídicích systémů/terminálů. Kontrast se u některých řídicích systémů/terminálů nastavuje pomocí trimru. U ostatních řídicích systémů/terminálů lze kontrast nastavit pouze pomocí skriptu.

Nastavení jasu a kontrastu lze naprogramovat pomocí prvků „NumericEdit“, kdy by uživatel přímo zadával hodnotu jasu či kontrastu v rozsahu 0 .. 100 %. Další možností (která je realizována v této aplikační poznámce) je např. využití kláves ↑ a ↓ (**Up** a **Down**) pro zadání jasu či kontrastu a zobrazení nastavené hodnoty pomocí prvku „NumericView“.

Založíme obrazovku, kterou pojmenujeme Kontrast, vložíme do ní prvek „NumericView“ (s nastaveným formátem ###) a dva prvky „Key“ dle následujícího obrázku.



Obr. 1 - Obrazovka pro nastavení kontrastu

Prvkům „Key“ zadáme za jejich parametr `KeyCode` klávesy ↑ a ↓. Do události OnKeyDown() prvku s přiřazenou klávesou ↑ pak napíšeme následující skript.

```
event Key1_OnKeyDown()
     If Application.Contrast < 100 Then //kontrola zda není zadán maximální kontrast
          Application.Contrast = Application.Contrast + 1; //Zvýšení kontrastu
          NumericView1.Value = Application.Contrast; //Zobrazení kontrastu
          NumericView1.Refresh(); //Refresh prvku NumericView
     EndIf;
end;
```

Ve skriptu nejprve ověříme, že již není zadána maximální hodnota kontrastu. Pokud ano, skript nevykonáváme. Pokud ne, přičteme k aktuálně nastavenému kontrastu hodnotu 1. Takto upravenou hodnotu uložíme do prvku „NumericView1“, který ji zobrazí na obrazovce.

Pozor
Prvky, u kterých chceme využívat parametr `Value` musí mít v okně Vlastnosti dosazenu za parametr `Variable` hodnotu „(none)“. V případě, že bude za parametr `Variable` dosazena proměnná, bude se prvek chovat dle této proměnné. Parametr `Value` ve skriptu má nižší prioritu než parametr `Variable` v okně Vlastnosti.

Závěrem provedeme refresh prvku „NumericView1“ aby se přiřazení hodnoty za parametr `Value` ihned projevilo i na displeji terminálu/řídicího systému. Pokud bychom refresh nenaprogramovali, projevila by se změna hodnoty až za dobu, která je nastavena parametrem `RefreshPeriod` ve vlastnostech obrazovky.

Obdobně pak budeme postupovat ve skriptu pro prvek „Key2“ s přiřazenou klávesou ↓. U tohoto prvku však budeme kontrast snižovat.

```
event Key2_OnKeyDown()
     If Application.Contrast > 0 Then
          Application.Contrast = Application.Contrast - 1;
          NumericView1.Value = Application.Contrast;
          NumericView1.Refresh();
     EndIf;
end;
```

Nastavení jasu lze u systémů, které to podporují provést stejným způsobem. Místo výrazu „Application.Contrast“ však použijeme výraz „Application.Brightness“.

2.3. Přepočet zobrazované/editované proměnné

Velkému množství modulů, které pracují např. s časem v programové části DetStudia se časový údaj zadává v milisekundách. Při zadávání dlouhých časových úseků (např. v hodinách) je pak zadání v milisekundách pro uživatele velmi nepřehledné. Pro zjednodušení, lze pomocí skriptu naprogramovat zobrazení a editaci takovýchto časových úseků přímo v hodinách.

V následujícím příkladu provedeme zobrazení a editaci přepočtené hodnoty pomocí prvku „NumericEdit“.

Přepočet musíme učinit již při otevření obrazovky, na které požadujeme zadání časového úseku např. v hodinách.

```
event Prepocet_OnOpen()
     Prepocet.FocusFirstControl();
     NumericEdit1.Value = Cas / 3600000;
end;
```

První řádek skriptu („NázevObrazovky.FocusFirstControl();“) je generován DetStudiem automaticky na každé vytvořené obrazovce. Díky tomuto skriptu je na každé otevřené obrazovce vždy již při otevření obrazovky zaměřen prvek s nejnižší hodnotou parametru `TabIndex` (viz nápověda k návrhovému prostředí DetStudio). Díky této funkčnosti může uživatel ihned po zobrazení obrazovky pomocí klávesy ↵ (**Enter**) spustit editaci takového prvku.

V dalším kroku provedeme přepočet milisekund na hodiny a výsledek uložíme do prvku „NumericEdit1“ (pomocí jeho parametru `Value`). Na obrazovce se tedy daný čas zobrazí v hodinách.

Pozor
Prvky, u kterých chceme využívat parametr `Value` musí mít v okně Vlastnosti dosazenu za parametr `Variable` hodnotu „(none)“. V případě, že bude za parametr `Variable` dosazena proměnná, bude se prvek chovat dle této proměnné. Parametr `Value` ve skriptu má nižší prioritu než parametr `Variable` v okně Vlastnosti.

Abychom mohli čas v hodinách pomocí prvku „NumericEdit“ také zadávat, musíme využít jeho událost OnEditComplete(). Do této události vložíme kód pro přepočet z hodin na milisekundy.

```
event NumericEdit1_OnEditComplete()
     Cas  = NumericEdit1.Value * 3600000;
end;
```

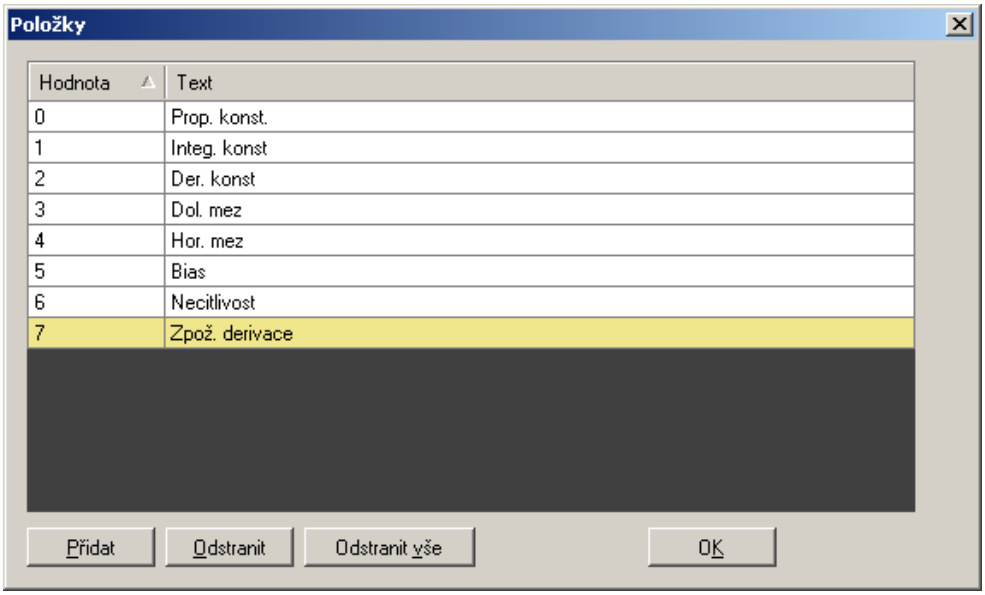
Výše napsanými skripty jsme dosáhli zobrazení aktuální hodnoty při otevření obrazovky a zápisu požadované hodnoty pomocí prvku „NumericEdit1“. Pokud však někdo změní hodnotu např. z vizualizace, neprojeví se tato změna, dokud obrazovku nezavřeme a znovu neotevřeme. Aby se změna hodnoty projevila, aniž bychom museli obrazovku zavírat a otvírat, použijeme událost OnRefresh() dané obrazovky. Do této události pak vložíme stejný skript jako do události OnOpen().

```
event Prepocet_OnRefresh()
     NumericEdit1.Value = Cas / 3600000;
end;
```

2.4. Editace celé matice jedním prvkem NumericEdit

V některých případech je užitečné editovat všechny buňky matice na jedné obrazovce pomocí jednoho editačního prvku. Toto je výhodné např. při definici parametrů PID regulátoru apod. Pro nastavení všech parametrů PID regulátoru bychom postupovali následovně.

Vytvoříme si obrazovku, na kterou umístíme prvek „Label“ s textem „Parametry PID“. Každý řádek matice s parametry modulu PID má svůj význam. Pro přehlednost budeme pro jednotlivé řádky matice zobrazovat různé texty, popisující význam daných řádků. Na obrazovku umístíme prvek „CaseLabelView“, na který dvakrát klikneme a nadefinujeme mu texty dle následujícího obrázku.

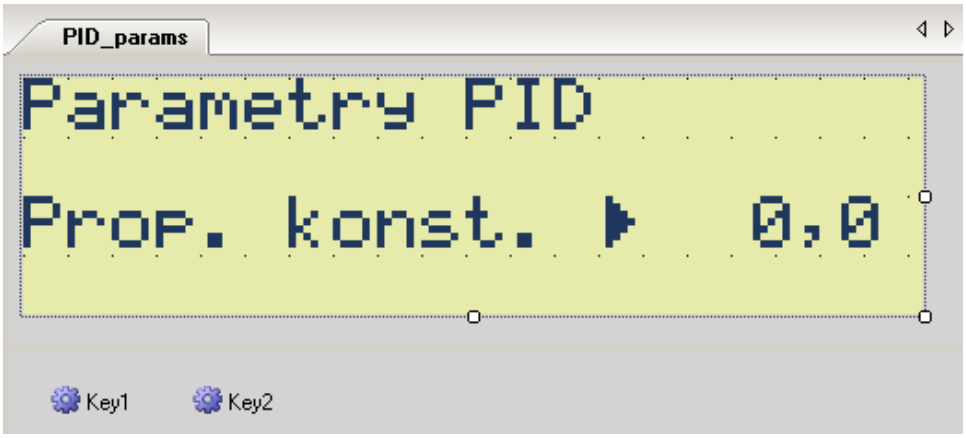


Obr. 2 - Nastavení prvku CaseLabelView

Dále na obrazovku umístíme prvek „NumericEdit“. Dvakrát na něj klikneme a přiřadíme mu možnost editace nulté buňky matice PID_params (matice rozměru 8 × 1, ve které jsou parametry modulu PID).

Závěrem na obrazovku umístíme dva prvky „Key“, kterým za parametr `KeyCode` nastavíme klávesy ↑ a ↓ (**Up** a **Down**). Pomocí těchto kláves se budeme pohybovat mezi jednotlivými parametry modulu PID.

Návrh obrazovky by měl nyní vypadat následovně.



Obr. 3 - Návrh obrazovky pro nastavení PID

Pomocí skriptu nyní naprogramujeme posun po jednotlivých řádcích matice v závislosti na stisku kláves ↑ nebo ↓. Zároveň s pohybem po jednotlivých řádcích budeme měnit i text, zobrazovaný prvkem „CaseLabelView“.

V prvním kroku musíme zobrazit pomocí prvku „CaseLabelView“ správný text dle aktuálně zobrazovaného řádku matice PID_params. K tomuto využijeme událost obrazovky OnOpen().

```
event PID_Params_OnOpen()
     PID_Params.FocusFirstControl();
     CaseLabelView1.Value = NumericEdit1.Row;
end;
```

První řádek skriptu (NázevObrazovky.FocusFirstControl();) je generován DetStudiem automaticky na každé vytvořené obrazovce. Díky tomuto skriptu je na každé otevřené obrazovce vždy již při otevření obrazovky zaměřen prvek s nejnižší hodnotou parametru `TabIndex` (viz nápověda k návrhovému prostředí DetStudio). Díky této funkčnosti může uživatel ihned po zobrazení obrazovky pomocí klávesy ↵ spustit editaci takového prvku.

V dalším řádku přiřadíme prvku „CaseLabelView1“ hodnotu čísla řádku (pomocí parametru `Value`), který je aktuálně zobrazován prvkem „NumericEdit1“. Tím docílíme zobrazení různých textů při editaci různých řádků matice.

Pozor
Prvky, u kterých chceme využívat parametr `Value` musí mít v okně Vlastnosti dosazenu za parametr `Variable` hodnotu „(none)“. V případě, že bude za parametr `Variable` dosazena proměnná, bude se prvek chovat dle této proměnné. Parametr `Value` ve skriptu má nižší prioritu než parametr `Variable` v okně Vlastnosti.

Stiskem klávesy ↓ se prvkem „NumericEdit1“ posuneme vždy o řádek níže a změníme text prvku „CaseLabelView1“ pomocí následujícího skriptu.

```
event Key2_OnKeyDown()
     If NumericEdit1.Row < 3 Then  //Kontrola, zda se nacházíme v mezích
          NumericEdit1.Row = NumericEdit1.Row + 1; // Posun na další řádek
          CaseLabelView1.Value = NumericEdit1.Row; //Výběr odpovídajícího textu
          CaseLabelView1.Refresh(); //Překreslení prvku
     EndIf;
end;
```

Pozor
Při pohybu po jednotlivých řádcích a sloupcích matice je vždy nutné hlídat, zda nepřekračujeme rozměr matice. ***V případě, že se kód skriptu bude snažit zobrazit neexistující buňku matice, dojde k restartu řídicího systému****.*

V události stisku klávesy ↓ nejprve provedeme kontrolu, zda se nacházíme v mezích, daných rozměrem matice. Pokud ano, posuneme se prvkem „NumericEdit1“ na další řádek. Poté přiřadíme číslo aktuálně zobrazeného řádku za parametr `Value` prvku „CaseLabelView1“ čímž dosáhneme zobrazení požadovaného textu. Závěrem provedeme refresh prvku „CaseLabelView1“ aby se přiřazení hodnoty za parametr `Value` ihned projevilo i na displeji terminálu/řídicího systému. Pokud bychom refresh nenaprogramovali, projevila by se změna textu až za dobu, která je nastavena parametrem `RefreshPeriod` ve vlastnostech obrazovky.

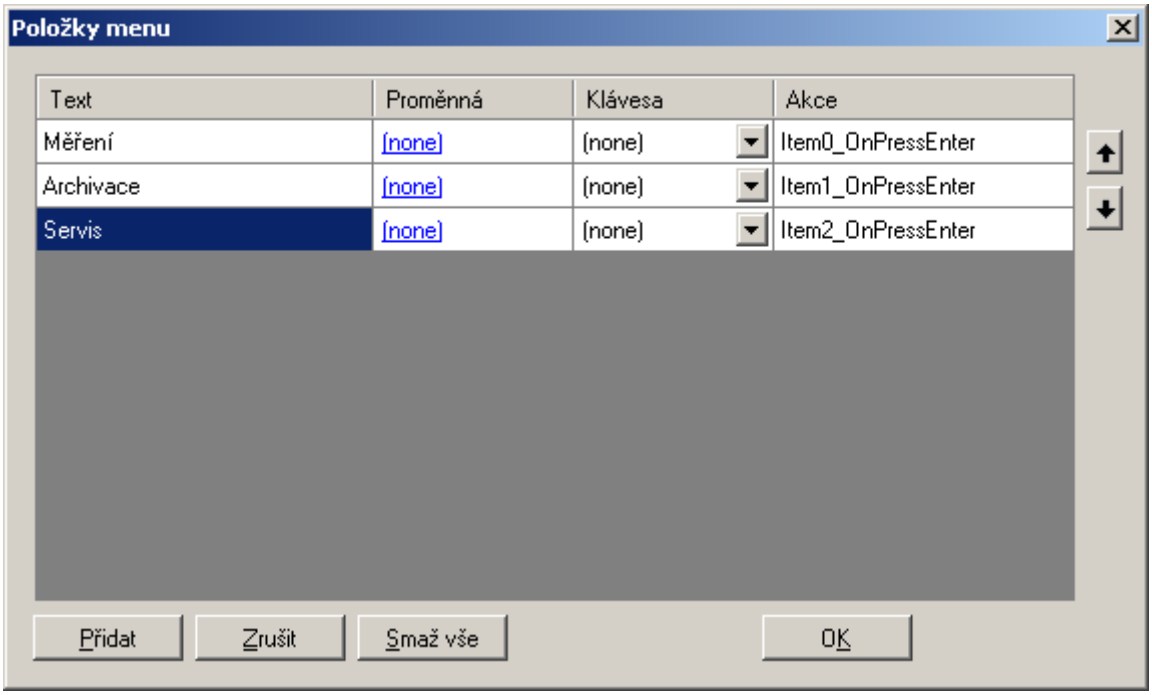
Obdobně naprogramujeme skript pro stisk klávesy ↑.

```
event Key1_OnKeyDown()
     If NumericEdit1.Row > 0 Then //Kontrola, zda se nacházíme v mezích
          NumericEdit1.Row = NumericEdit1.Row - 1; // Posun na další řádek
          CaseLabelView1.Value = NumericEdit1.Row; //Výběr odpovídajícího textu
          CaseLabelView1.Refresh(); //Překreslení prvku
     EndIf;
end;
```

Poznámka
Obdobný postup lze naprogramovat i v případě posunu po sloupcích matice. Pro posun po sloupcích matice lze využít u prvků ve skriptu parametr `Col`.

2.5. Použití položky menu dle přihlášeného uživatele

Pokud chceme uživateli umožnit použít určitou položku menu pouze v případě, kdy je přihlášen do řídicího systému s vyššími právy, je nutné využít skriptu společně s prvkem „Menu“ ze sekce General. Prvek „Menu“ oproti prvku „MenuScreen“ ze sekce Basic generuje událost „Byla vybrána položka menu“ a není tedy přímo vázán na obrazovky. Pokud chceme u jednotlivých položek rozeznávat aktuálně přihlášeného uživatele, umístíme na obrazovku prvek „Menu“ a nastavíme mu jednotlivé položky (např. dle následujícího obrázku).

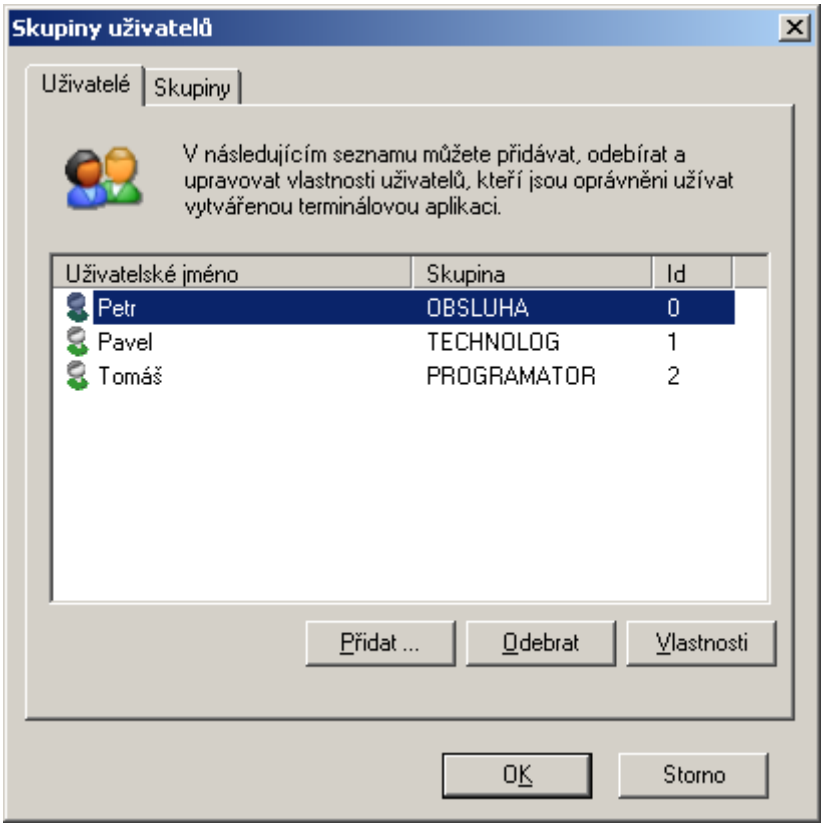


Obr. 4 - Nastavení položek prvku Menu ze sekce General

Po potvrzení tlačítkem **OK** dojde k vytvoření tří událostí a k otevření skriptovací části DetStudia. Jednotlivé události odpovídají vybrání jednotlivých položek menu. První dvě položky (Měření a Archivace) smí využívat uživatel s jakýmikoliv právy. Do jednotlivých událostí tedy vložíme přímo skript pro otevření požadovaných obrazovek.

```
event Menu1_Item0_OnPressEnter()
     Mereni.Show();
end;
event Menu1_Item1_OnPressEnter()
     Archivace.Show();
end;
```

Poslední položku však bude moci použít pouze uživatel, který má vyšší práva než Obsluha. Pro náš příklad nadefinujeme uživatele dle následujícího obrázku.



Obr. 5 - Uživatelé nedefinovaní v projektu

Pokud bude přihlášen Petr, nestane se po výběru položky Servis v menu vůbec nic. Pokud bude přihlášen Pavel nebo Tomáš, zobrazí se servisní obrazovka. K této funkčnosti využijeme indexu jednotlivých uživatelů (sloupec „Id“ v definici uživatelů) v kombinaci s následujícím skriptem napsaným do události výběru třetí položky menu.

```
event Menu1_Item2_OnPressEnter()
     If Application.ActualUser == 1 or Application.ActualUser == 2 Then
          Servis.Show();
     EndIf;
end;
```

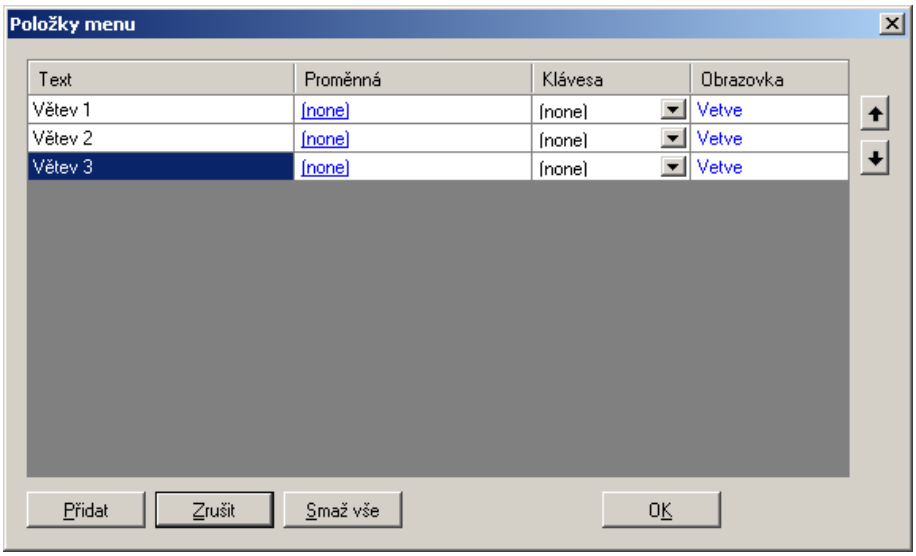
Pozor
Pokud není k řídicímu systému nikdo přihlášen, je index aktuálně přihlášeného uživatele 65535.

2.6. Optimalizace skriptem

Skript lze s výhodou využít také pro optimalizace. Častokrát se stává, že je v jedné aplikaci např. několik topných větví, které chceme ovládat přes terminál. Jednotlivé obrazovky na terminálu jsou pro většinu topných větví totožné, pouze zobrazují hodnoty různých topných větví. Pomocí skriptu lze skupinu parametrů více topných větví zobrazovat/nastavovat pouze pomocí jedné jediné obrazovky. Podmínkou pro takovéto zobrazení parametrů více topných větví pomocí jedné obrazovky je uložení parametrů topných větví v maticích.

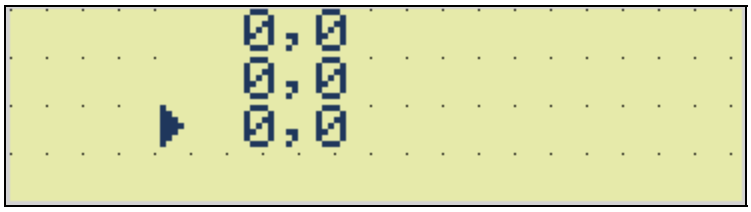
Pro výběr editované topné větve využijeme prvek „MenuScreen“ z prvků Basic. Mějme např. tři topné větve, u kterých chceme v jedné obrazovce zobrazit měřenou teplotu topné vody, žádanou teplotu topné vody a chceme zadat ekvitermní konstantu.

Založíme obrazovku s názvem „Menu“ a obrazovku s názvem „Vetve“. Na obrazovku „Menu“ vložíme prvek „MenuScreen“. Dvakrát na něj klikneme a vyplníme mu jednotlivé položky dle následujícího obrázku.



Obr. 6 - Pojmenování jednotlivých položek menu

Na obrazovku „Vetve“ vložíme dva prvky „NumercView“ a jeden prvek „NumericEdit“ dle následujícího obrázku.



Obr. 7 - Obrazovka pro ovládání tří topných větví

Tři parametry třech topných větví budeme mít v matici o rozměru 3 × 3, kde každý sloupec bude jeden parametr a každý řádek bude jedna topná větev. Jednotlivým prvkům tedy přiřadíme např. buňky matice v prvním řádku.

V obrazovce „Vetve“ vložíme do události OnOpen(), skript dle následujícího příkladu.

```
event Vetve_OnOpen()
     Vetve.FocusFirstControl();
     NumericEdit1.Row = Menu.MenuScreen1.SelectedIndex;
     NumericView1.Row = Menu.MenuScreen1.SelectedIndex;
     NumericView2.Row = Menu. MenuScreen1.SelectedIndex;
     Vetve.Refresh();
end;
```

Tím dojde při každém otevření obrazovky „Vetve“, k přiřazení hodnot matice do prvků v závislosti na vybrané položce prvku „MenuScreen“ v obrazovce „Menu“.

3. Technická podpora

Veškeré informace ohledně použití skriptu v DetStudiu, Vám poskytne oddělení technické podpory firmy AMiT. Technickou podporu můžete kontaktovat nejlépe prostřednictvím emailu na adrese **support@amit.cz**.

4. Upozornění

AMiT spol. s r.o. poskytuje informace v tomto dokumentu, tak jak jsou, nepřejímá žádné záruky, pokud se týče obsahu tohoto dokumentu a vyhrazuje si právo měnit obsah dokumentu bez závazku tyto změny oznámit jakékoli osobě či organizaci.

Tento dokument může být kopírován a rozšiřován za následujících podmínek:

1. Celý text musí být kopírován bez úprav a se zahrnutím všech stránek.

2. Všechny kopie musí obsahovat označení autorského práva společnosti AMiT, spol. s r. o. a veškerá další upozornění v dokumentu uvedená.

3. Tento dokument nesmí být distribuován za účelem dosažení zisku.

V publikaci použité názvy produktů, firem apod. mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.