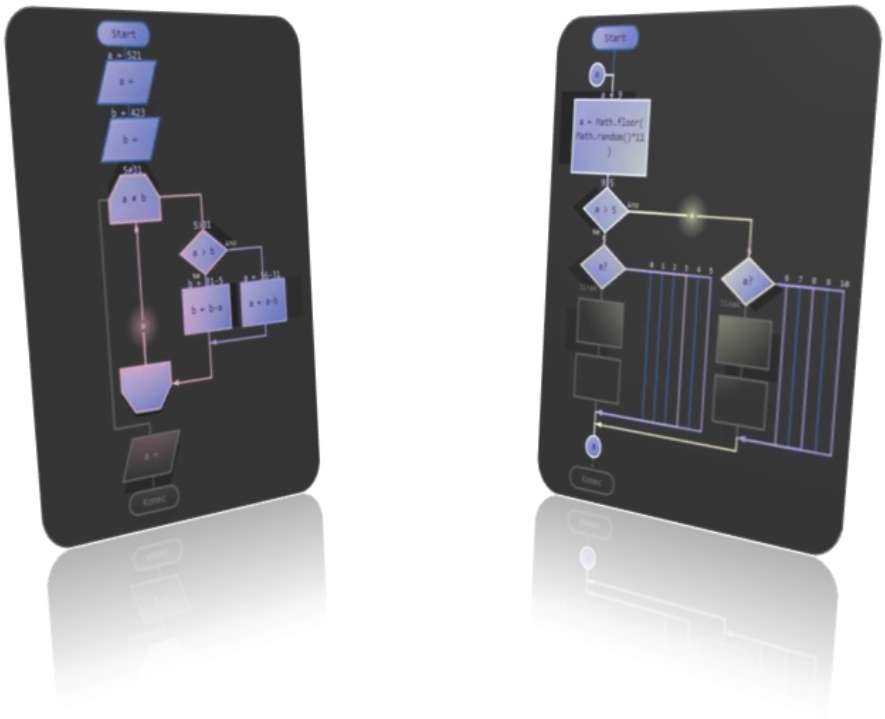


UŽIVATELSKÁ PŘÍRUČKA K APLIKACI PS DIAGRAM

autor: Miroslav Bartyzal
kontakt: miroslavbartyzal@gmail.com

OBSAH

1 Úvod

Tato uživatelská příručka slouží jako manuál a nápověda k aplikaci PS Diagram. Aplikace PS Diagram slouží k vytváření vývojových diagramů a vizualizaci jejich algoritmického průběhu pro výuku algoritmizace.

Díky zabudovanému layoutu aplikace poskytuje velmi rychlý způsob, jak vývojový diagram sestavit, přičemž nabízí širokou škálu použitelných symbolů z oblasti vývojového diagramu programu, navrženou v souladu s českou státní normou ČSN ISO 5807. Atraktivní a přehledný způsob vizualizace jejich průchodu pak do výuky přináší vhodný nástroj pro procházení algoritmů.

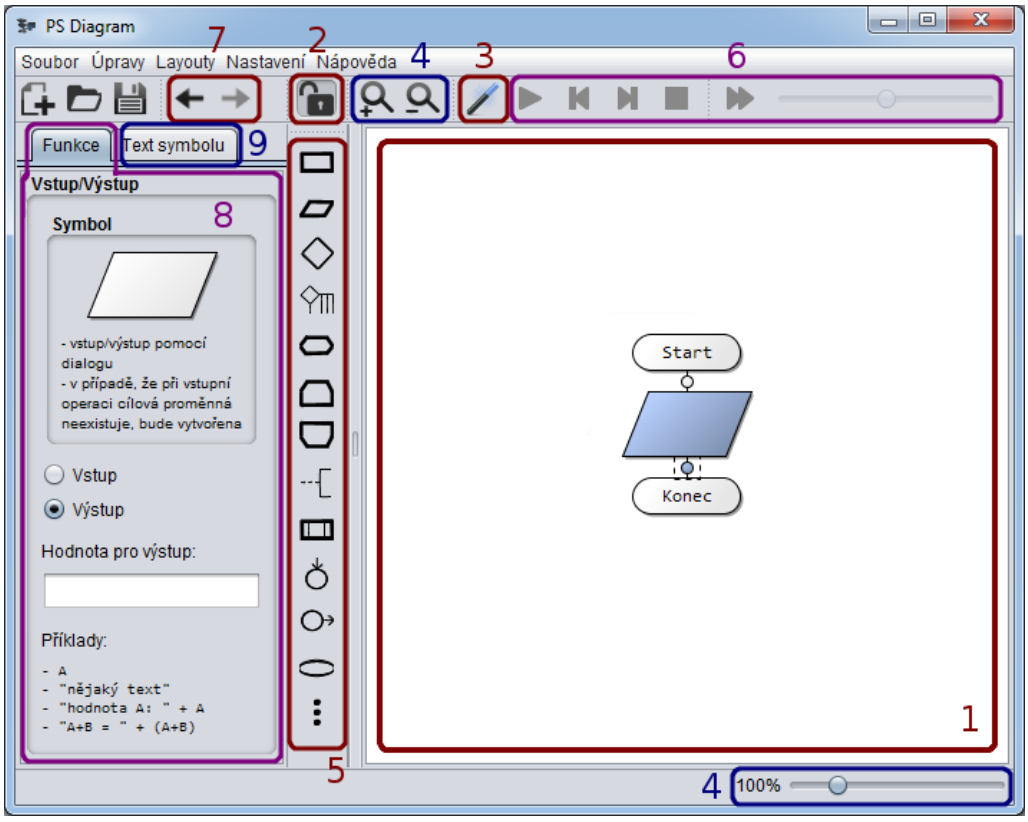
1.1 Charakteristika aplikace

- ✓ Vývojový diagram lze díky jeho vektorovému formátu kdykoliv přiblížit či oddálit, aniž bychom přišli o kvalitu jeho vykreslení (*kapitola 2.2*).
- ✓ Aplikace je přehledně rozdělena do třech oddělených režimů: režim náhledu (*kapitola* 2), editační režim pro samotné vytváření vývojových diagramů (*kapitola* 3) a režim animační, zajišťující vizualizaci jejich algoritmického průchodu (*kapitola* 4).
- ✓ Aplikace disponuje inteligentním zarovnáváním vkládaných symbolů vývojového diagramu do zvoleného layoutu. Symboly jsou zároveň automaticky korektně propojovány spojnicemi a uživatel je tak zproštěn jakýchkoli starostí s formátováním diagramu (*kapitola 3.1*).
- ✓ Přidávání a editace funkce či textu symbolů vývojového diagramu je díky *vkládacím bodům* a *párovému označování* velice svižné a efektivní (*kapitola 3.1.2*).
- ✓ Textové hodnoty symbolů vývojového diagramu jsou generovány automaticky (s ohledem na co nejvyšší výstupní uživatelskou srozumitelnost) na základě jeho vyplněné funkce (*kapitola 3.4.1*).
- ✓ Průchod algoritmického průběhu vývojového diagramu je možné realizovat více způsoby, přičemž při *krokování* je při průchodu poskytnuta i nadstandardní funkce kroku zpět (*kapitola 4.2*).

✓ Během průchodu vývojového diagramu je zobrazován jak výpis aktuálních proměnných, tak mezivýpočty funkcí symbolů. Celá vizualizace průchodu je navíc obohacena o barevné rozlišení symbolů a spojnic na základě toho, zda jimi tok algoritmu prošel, prošel jednou či prošel vícekrát. Je možné i zpětně odhadnout, o kolik průchodů se jednalo (*kapitola 4.1*).
✓ Vývojový diagram je možné importovat (vygenerovat) z vloženého zdrojového kódu některého z podporovaných programovacích jazyků (*kapitola* 7) nebo vývojový diagram naopak do zdrojového kódu exportovat (*kapitola* 8).

1.2 GRAFICKÉ ROZHRANÍ APLIKACE

Na obrázku Obr. 1 je zobrazeno hlavní okno aplikace. Čísly jsou označeny jeho hlavní ovládací prvky, jejichž vysvětlivky následují níže.



OBR. 1 GRAFICKÉ ROZHRANÍ APLIKACE

1. Plátno s vývojovým diagramem
2. Přepínání mezi aplikačním režimem editace / náhledu
3. Přepínání mezi aplikačním režimem animace / náhledu
4. Přibližování / oddalování vývojového diagramu
5. Dostupné symboly vývojového diagramu programu
6. Panel pro ovládání animačního režimu aplikace
7. Tlačítka zpřístupňující funkce Undo (vrátit akci) a Redo (obnovit akci)
8. Záložka editace funkce vybraného symbolu
9. Záložka editace textu vybraného symbolu

1.3 Režimy aplikace

Aplikace obsahuje celkem 3 různé režimy zobrazení vývojového diagramu. Je to režim náhledu, editační režim a režim animační. Diagram lze v jednu chvíli zobrazovat vždy jen v jednom z těchto režimů.

Jednotlivými režimy se v této příručce budeme podrobněji zabývat v samostatných kapitolách, které následují.

2 Režim náhledu

V režimu náhledu se nacházíme automaticky, jsou-li editační a animační režimy ve vypnutém stavu (Obr. 2, popisek 1 a 2).

Vývojový diagram je v tomto režimu zobrazován v prosté formě tak, jak ho lze následně exportovat (viz kapitola 6). Tato prostá forma zobrazení neumožňuje jakoukoliv editaci vývojového diagramu, poskytuje však uživateli přehledný způsob, jak vývojový diagram zkontrolovat vůči případným logickým chybám apod.

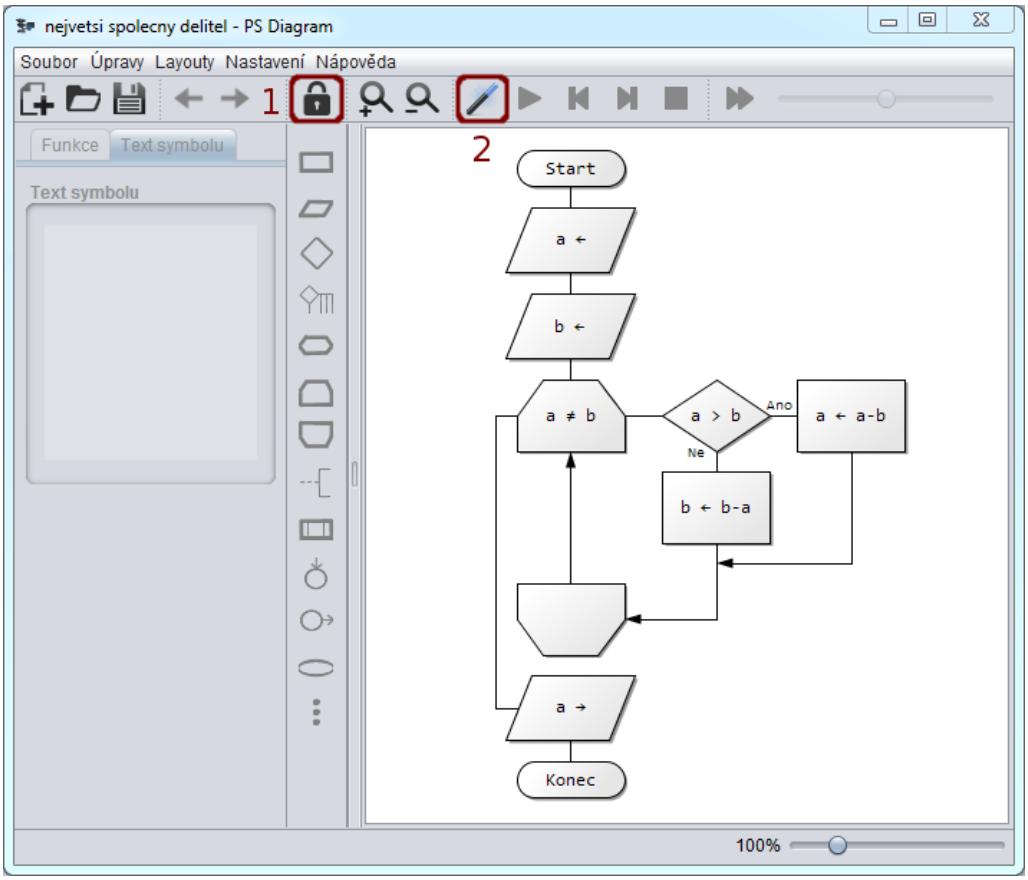
2.1 Manipulace s vývojovým diagramem

Režim náhledu zároveň umožňuje vývojovým diagramem intuitivně manipulovat.

Použijeme-li kolečko myši, můžeme diagram rolovat nahoru a dolů, tak jak jsme zvyklí z ostatních aplikací. Přidržíme-li navíc klávesu Alt, docílíme rolování doleva a doprava.

Pohodlnější způsob jak manipulovat s vývojovým diagramem je přidržením levého tlačítka myši nad plátnem (Obr. 1, popisek 1). Pohybujeme-li pak myší do různých směrů, kurzor myši se nám změní v ručičku a pozice diagramu kopíruje její pohyb.

Plátno pro vývojový diagram je nekonečné, nejsme tak omezeni jeho velikostí a můžeme si dovolit vytvořit libovolně složité algoritmické struktury. Při manipulaci s nekonečným plátnem nám však nehrozí, že ztratíme přehled o pozici našeho diagramu. Vývojovým diagramem lze totiž pohybovat vždy maximálně v okolí zobrazeného plátna + velikost diagramu, diagram se tak nachází vždy maximálně o jeden pixel za hranící viditelného plátna.



OBR. 2 REŽIM NÁHLEDU

2.2 PŘIBLIŽOVÁNÍ

Vývojový diagram lze kdykoliv přiblížit či oddálit. Veškerá grafika, která je vykreslena na plátně, je realizována vektorově. Při přibližování tak nepřicházíme o kvalitu vykreslení, jak by tomu bylo u grafiky bitmapové.

Přiblížení nebo oddálení vývojového diagramu lze ovládat třemi způsoby. První dva způsoby ilustruje Obr. 1 (popisek 4). Třetí způsob spočívá v přidržení klávesy Ctrl + použití kolečka myši nad plátnem vývojového diagramu (Obr. 1, popisek 1).

3 EDITAČNÍ REŽIM

Do editačního režimu vstoupíme sepnutím tlačítka zámku (Obr. 1, popisek 2) v hlavním panelu aplikace.

Tento režim nám zpřístupní panel dostupných symbolů (Obr. 1, popisek 5) a my tak můžeme vývojové diagramy vytvářet. K vytváření diagramů využijeme i záložku Funkce symbolu (Obr. 1, popisek 8) a Text symbolu (Obr. 1, popisek 9), jejichž účelem se budeme zabývat později.

K tomu, abychom porozuměli systému vkládání symbolů, si v následující kapitole představíme layout.

Poznámka:
Editační režim spolupracuje s režimem náhledu, s vývojovým diagramem tak lze nadále manipulovat obvyklým způsobem.

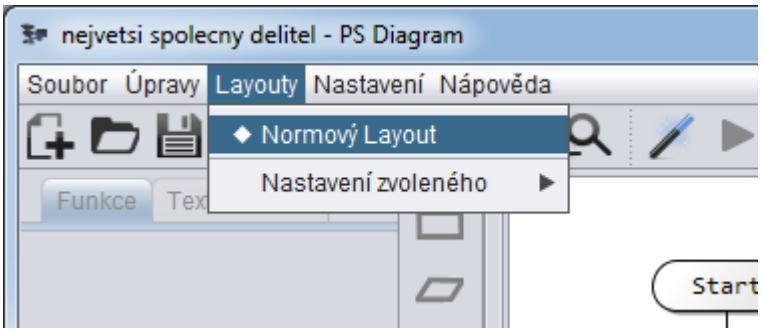
3.1 LAYOUT

Aplikace PS Diagram disponuje inteligentním zarovnáváním vkládaných symbolů do zvoleného layoutu. Symboly jsou zároveň automaticky korektně propojovány spojnicemi a uživatel je tak zproštěn jakýchkoli starostí s formátováním diagramu.

Tím, že při vytváření vývojového diagramu odpadají starosti s jeho formátováním, vzniká uživateli prostor pro koncentraci na samotný vytvářený algoritmus a nalezení správného řešení dané úlohy. Vytváření diagramů se tímto zároveň stává otázkou okamžiku.

3.1.1 DOSTUPNÉ LAYOUTY

Požadovaný layout lze zvolit v hlavní nabídce aplikace pod tlačítkem Layouty (Obr. 3). V dolní části této nabídky lze navíc nalézt případná další nastavení právě zvoleného layoutu.



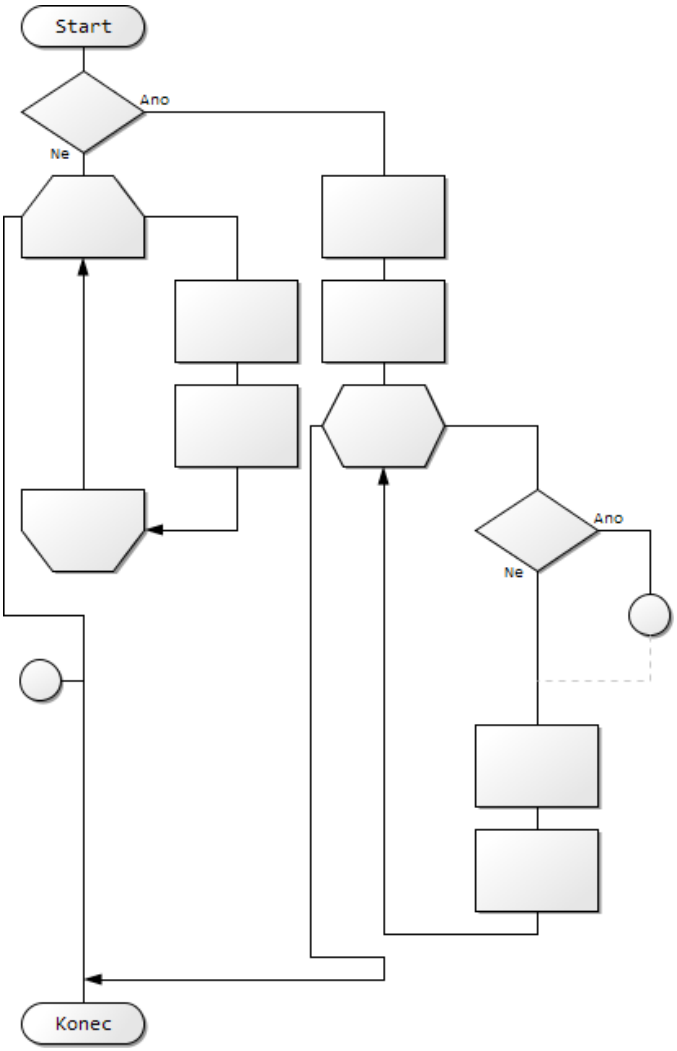
OBR. 3 VOLBA LAYOUTU

Poznámka:
Aplikace zatím obsahuje pouze layout Normový, v případě zájmu je však s rozšířením nabídky layoutů počítáno.

3.1.1.1 NORMOVÝ LAYOUT

Normový layout (Obr. 4) představuje výchozí layout aplikace. Byl navržen tak, aby co nejvíce odpovídal předpisům české státní normy ČSN ISO 5807 týkající se vývojových diagramů.

Směr toku tohoto layoutu směřuje zleva doprava a shora dolů, přičemž spojnice do symbolů vstupují zleva nebo shora a vystupují vpravo nebo dole. Ke spojnicím jsou automaticky doplňovány směrové šipky, směřují-li vzhledem k počátku jinam než do IV. kvadrantu.



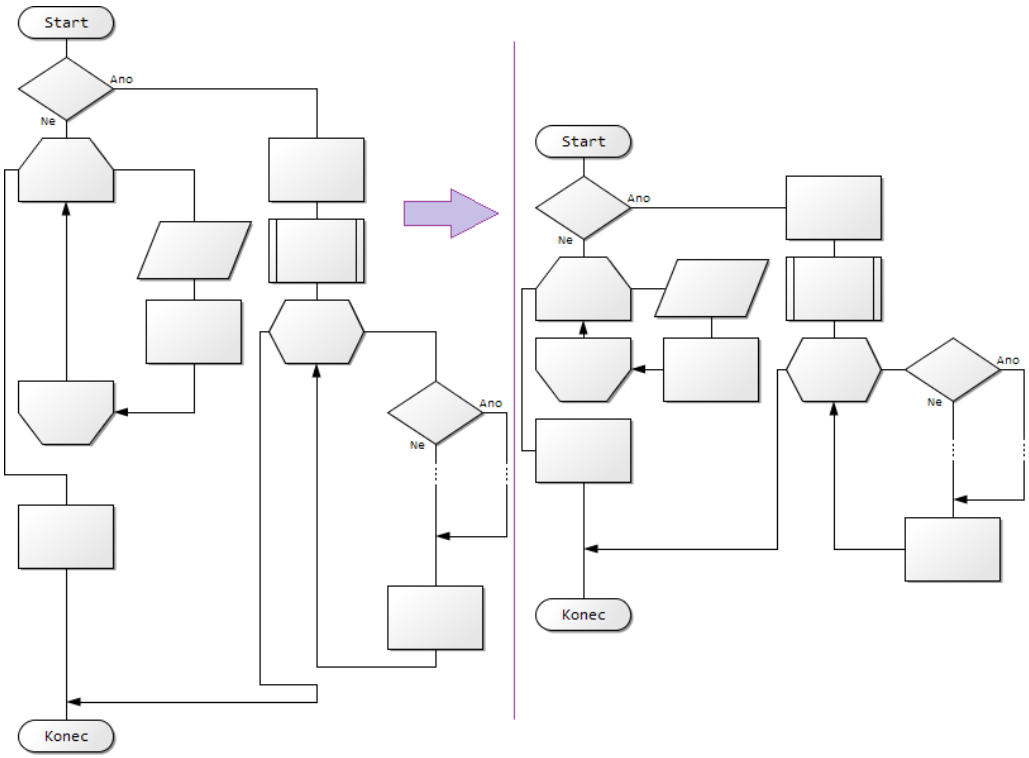
OBR. 4 NORMOVÝ LAYOUT

3.1.1.1.1 MOŽNÁ NASTAVENÍ

Normový layout poskytuje dvě možná přizpůsobení.

3.1.1.1.1.1 SMRŠŤOVÁNÍ

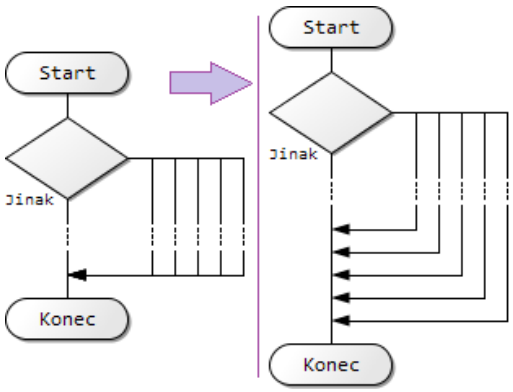
Možnost „*Smršťování*“ nám umožní symboly v sloupcích skládat vedle sebe a navazovat postranní šipky cyklů na následující symboly zleva (Obr. 5). Tuto možnost využijeme zejména tehdy, chceme-li např. vývojový diagram vytisknout a potřebovali bychom zmenšit jeho velikost. Při zapnutém smršťování lze vývojový diagram rovněž editovat, nicméně tento režim je méně přehledný, než editace při vypnutém smršťování.



OBR. 5 SMRŠŤOVÁNÍ

3.1.1.1.1.2 Expanze switch symbolu

Druhé přizpůsobení nese název „*Expanze switch symbolu*“. Switch symbol, označovaný také jako „vícenásobné rozhodování“, se od ostatních symbolů odlišuje vyšším počtem spojnic, které ze symbolu vycházejí. Obecně existuje více způsobů, jak tyto spojnice zobrazovat - PS Diagram spojnice defaultně vykresluje paralelně. Povolením této volby se konce těchto spojnic navrací jednotlivě, tak jak je to možné pozorovat na obrázku (Obr. 6).



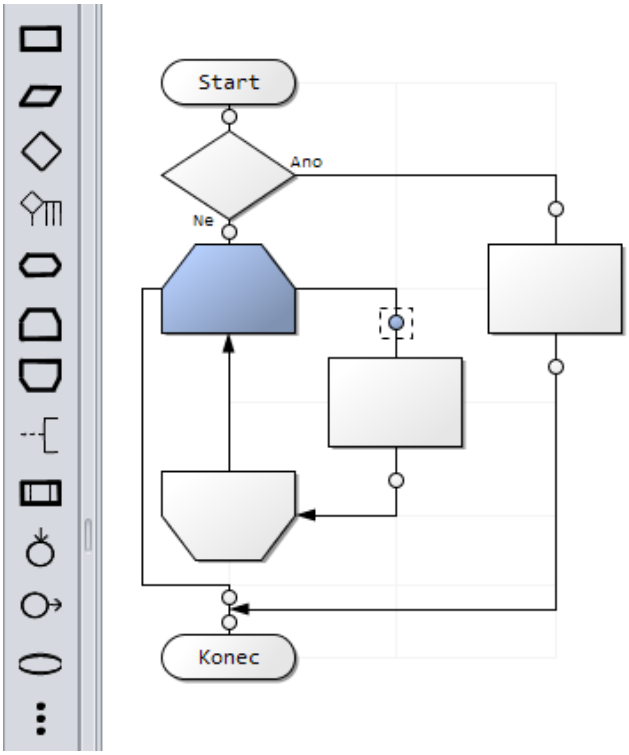
Obr. 6 Expanze switch symbolu

3.1.2 Přidávání symbolů

Přidávání symbolů se v aplikaci PS Diagram realizuje velmi specificky.

Nacházíme-li se v Editačním režimu aplikace, na místech mezi symboly jsou vykresleny tzv. *vkládací body* (Obr. 7). Vkládací body nám označují místa, kam lze potenciálně vložit nový symbol. Označíme-li si požadovaný vkládací bod a klikneme-li poté v panelu dostupných symbolů (Obr. 7, panel vlevo) na požadovaný symbol, dojde k jeho automatickému začlenění do vývojového diagramu.

Symbol můžeme do diagramu umístit také pomocí tažení myši. Stačí si z palety symbolů (Obr. 7, panel vlevo) jeden symbol vybrat, stlačit nad ním tlačítko myši a přetáhnout jej nad kterýkoliv vkládací bod. Po uvolnění tlačítka je symbol automaticky začleněn do vývojového diagramu.



OBR. 7 VKLÁDACÍ BODY

3.1.2.1 PÁROVÉ OZNAČOVÁNÍ

Všimněte si, že v jednom okamžiku je vždy označen pár symbol-vkládací bod (označení značí modrá barva). Tato funkcionalita zajišťuje svižnost při vkládání a editaci symbolů, neboť umožňuje tyto dvě činnosti provádět naráz, bez dalšího označování. Po vložení symbolu tak můžeme ihned editovat např. jeho text či funkci, přičemž je aplikace stále připravena na přidání symbolu následujícího.

V systému označování figuruje i označení rámované. Toto označení určuje místo, kam bude vložen případný přidávaný symbol komentáře.

3.2 ÚPRAVY

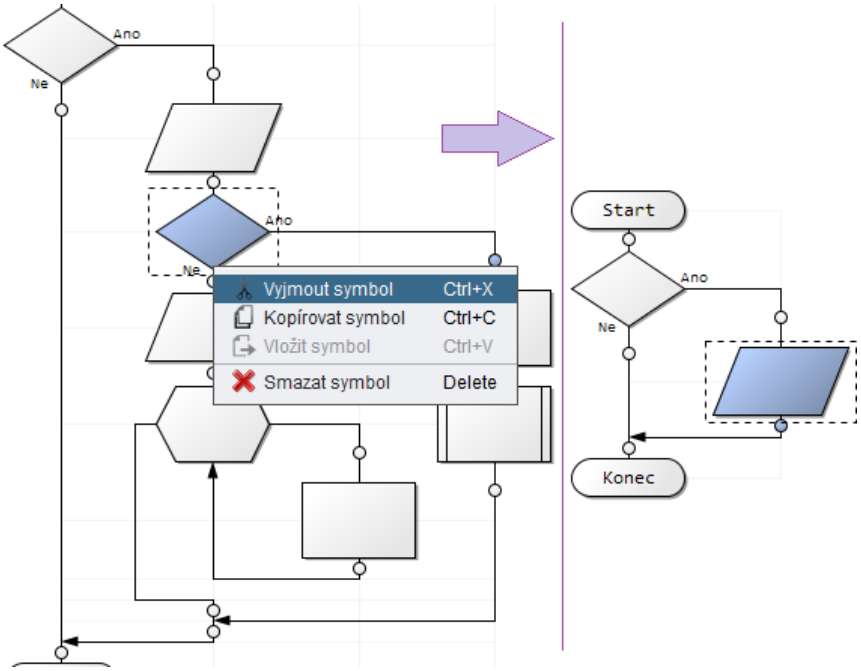
Editační režim aplikace podporuje většinu zaběhlých příkazů úprav, které bychom pro komfort editace vývojového diagramu mohli očekávat.

Menu úprav můžeme při zapnutém režimu editace vyvolat buď v hlavním menu pod položkou s názvem „*Úpravy*“ nebo vyvoláním kontextové nabídky po kliknutí na některý symbol či vkládací bod vykreslený na plátně.

3.2.1 KOPÍROVÁNÍ, VYJMUTÍ, VLOŽENÍ, MAZÁNÍ SYMBOLŮ

Při použití funkce kopírování, vyjmutí, vložení či smazání symbolu, aplikace bere zřetel na logickou strukturu našeho vývojového diagramu.

Chceme-li tedy vyjmout např. symbol podmínky, je po aktivaci této funkce vyjmut nejen samotný symbol této podmínky, ale i jeho vnitřní větve se všemi symboly vnořenými (Obr. 8).



OBR. 8 VYJMUTÍ SYMBOLU

Pro vložení symbolu pak již stačí jen označit požadovaný vkládací bod a stejným způsobem zvolit funkci „*Vložit symbol*“.

Tip:
Pro urychlení práce při úpravách vývojového diagramu využijte přetahování symbolů myší. Jednoduše chytněte za symbol, který chcete přemístit a pusťte jej na některém z vkládacích bodů. Pro kopírování symbolů postupujte obdobně, jen navíc přidržte klávesu Ctrl.

Tip:
Pro úpravy diagramu můžete použít i klávesové zkratky Ctrl+c pro kopírování, Ctrl+x pro vyjmutí, Ctrl+v pro vložení, resp. klávesu Delete pro smazání symbolu.

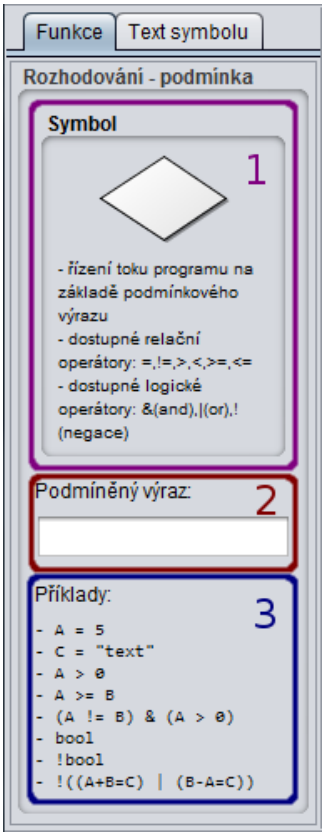
3.2.2 UNDO/REDO FUNKCE

Aplikace rovněž podporuje funkce „vrátit akci“ a „obnovit akci“. Jejich aktivaci vyvoláme buď příslušnými tlačítky v hlavní liště aplikace (Obr. 1, popisek 7) nebo v nabídce „*Úpravy*“. Můžeme využít i klávesových zkratek.

3.3 FUNKCE SYMBOLU

Záložka Funkce (Obr. 1, popisek 8) slouží k nastavení funkcionality zvoleného symbolu. Každý symbol má svou vlastní, specifickou funkcionalitu. Jednotlivým funkcím symbolů se budeme věnovat v kapitole *3.5 Dostupné symboly*.

Formulář nastavení funkce symbolu se skládá ze tří hlavních částí (Obr. 9):



OBR. 9 FORMULÁŘ FUNKCE SYMBOLU

1. Panel s grafickým znázorněním symbolu, kterého se nastavení týká spolu se stručným popisem jeho funkce
2. Komponenty pro nastavení samotné funkce symbolu

3. Příklady, popisující několik tipů ohledně možného nastavení funkce symbolu

Tip:
Zdají-li se vám textová pole pro vyplnění funkce symbolu příliš krátká, můžete celý panel Funkce rozšířit. Pro jeho rozšíření slouží lišta mezi plátnem vývojového diagramu (Obr. 1, popisek 1) a panelem dostupných symbolů (Obr. 1, popisek 5).

Poznámka:
Nastavené funkce jsou vyhodnocovány pomocí JavaScriptu Rhino od společnosti Mozilla. Je tak možné přímo využívat balíčků, které tento interpretovaný jazyk nativně obsahuje. Velice užitečná je pak například třída Math, která nám zprostředkovává některé složitější matematické operace (reference zde).

3.3.1 Proměnné a jejich hodnoty

V aplikaci PS Diagram se proměnné, pokud již neexistují, deklarují při přiřazování hodnot automaticky. Proměnným nedefinujeme ani jejich typ, jediné, co je tedy nutné do formuláře funkce symbolu vyplnit, je název proměnné, případně její hodnota k přiřazení (dovoluje-li to daný symbol).

Chceme-li jako hodnotu proměnné přiřadit textový řetězec, je nutné ho ohraničit dvojitými uvozovkami (*"textová hodnota"*).

Poznámka:
Proměnné jsou v aplikaci PS Diagram case-sensitive – citlivé na malá a velká písmena. Znamená to, že když přiřadíme proměnné s názvem „a“ hodnotu 1 a proměnné „A“ hodnotu 2, ve výsledku budeme mít v paměti uloženy proměnné dvě: „a“ s hodnotou 1 a „A“ s hodnotou 2. Identifikátor „a“ se totiž nerovná identifikátoru „A“.

3.3.1.1 Pole

Výjimku v automatické deklaraci proměnných tvoří odkaz na index pole, které ještě nebylo vytvořeno. Pole je vždy nutné před vložením jeho *n*-tého prvku nejprve obsadit alespoň žádným prvkem (hodnota vyplněna jako „[]“ či „[5,6,7, …]“ popřípadě „["první", "druhý", …]“).

Velikost pole není fixní, můžeme tak do pole přidávat libovolné množství dalších prvků, aniž bychom museli deklarovat jeho velikost. Prvky pole jsou indexovány od *nuly*.

Aplikace podporuje i pole vícerozměrná. Stačí jako hodnotu proměnné přiřadit pole, v němž jako jednotlivé prvky specifikujeme libovolné množství polí vnořených (hodnota vyplněna jako „[[5,3],[7,1],[3,5], …]“). Přejeme-li si pak přečíst hodnotu nultého prvku prvního pole (číslo 7), zápis bude vypadat takto: „*název_pole*[1][0]“.

3.3.2 Syntaktické filtry

Textová pole pro nastavení funkcí symbolů (Obr. 9, popisek 2) obsahují tzv. syntaktické filtry. Tyto filtry nám při úpravách funkce symbolu asistují prostřednictvím zpráv tak, aby příkazy a identifikátory byly zapsány ve správné formě.

Téměř ve všech programovacích jazycích identifikátor proměnné nesmí začínat číslicí. Této konvence se drží i PS Diagram a název proměnné tak může obsahovat pouze tyto znaky:

- číslice (v pořadí > 1. znak)
- písmena bez diakritiky
- znak podtržítka („_“)
- znak dolaru („$“)

Vždy, když přidáním konkrétního znaku do textového pole funkce symbolu vznikne syntaktická chyba, jsme na to syntaktickým filtrem upozorněni zobrazením zprávy s informacemi o dané chybě a označením jejího umístění v příkazu.

Poznámka:
Syntaktické filtry je možné v nastavení aplikace vypnout, je však silně doporučeno nechat tuto volbu povolenu.

3.4 Text symbolu

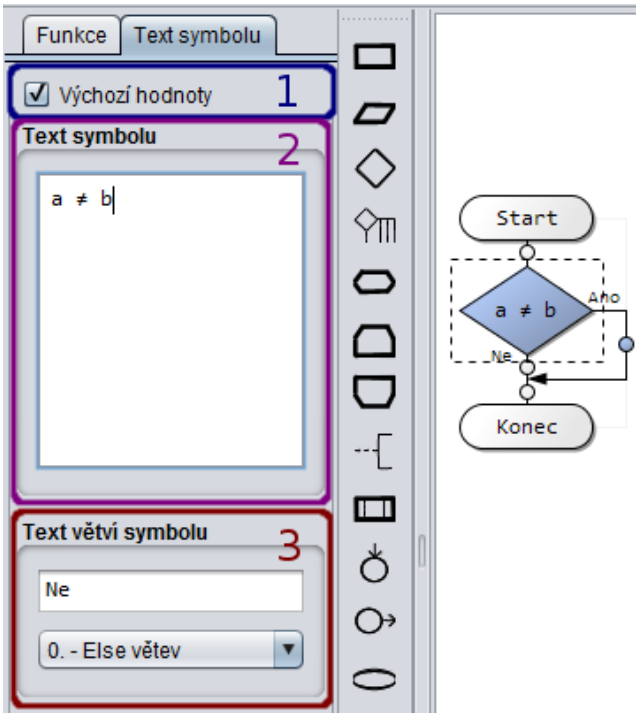
Záložka Text symbolu (Obr. 1, popisek 9) slouží k nastavení textové hodnoty právě označeného symbolu. Tyto hodnoty jsou rozděleny do dvou kategorií: výchozí hodnoty a hodnoty uživatelské.

3.4.1 Výchozí hodnoty

Je-li kompletně vyplněna funkce symbolu (Obr. 9, popisek 2), textová hodnota je pro tento symbol vygenerována automaticky. Automatické

generování textových hodnot bylo navrženo s ohledem na co nejvyšší výstupní srozumitelnost, některé znaky jsou proto konvertovány do čitelnější podoby. Jedná se především o tyto znaky:

- „nerovná se“
- „větší nebo rovno“, „menší nebo rovno“
- negace
- znaménko přiřazení hodnoty do proměnné



OBR. 10 FORMULÁŘ TEXTU SYMBOLU

3.4.2 UŽIVATELSKÉ HODNOTY

Text symbolu, ať už je vygenerovaný nebo není, můžeme ručně upravovat. K tomuto účelu je v tomto formuláři připravena textová oblast (Obr. 10, popisek 2). Jakékoliv upravené hodnoty, které neodpovídají případným hodnotám vygenerovaným, jsou považovány za uživatelský text.

Je-li automaticky vygenerovaný text přístupný, v horní části formuláře pro editaci textu symbolu se nám zobrazí zatrhávací políčko (Obr. 10, popisek 1) pro přepínání vygenerovaného textu, mezi textem uživatelským. Úpravou

vygenerovaného textu o tuto hodnotu tedy nepřicházíme a můžeme se k ní kdykoliv navrátit.

Umožňuje-li symbol editaci textu svých výstupních větví, v dolní části formuláře je pro jejich úpravy zobrazeno textové pole s možností výběru konkrétní větve k editaci (Obr. 10, popisek 3).

Tip:
Je-li automaticky vygenerovaný text příliš dlouhý a symbol tak přesahuje přípustné rozměry, můžeme použít editaci textu symbolu pro jeho optimální odřádkování.

3.5 Dostupné symboly

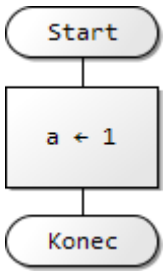
Aplikace nabízí širokou škálu použitelných symbolů. V rámci normy vývojového diagramu programu pokrývá všechny popsané symboly, krom symbolu paralelního zpracování.

Jednotlivé symboly si popíšeme v následujících podkapitolách této příručky.

3.5.1 Zpracování

Symbol zpracování nám umožňuje přiřazení dané hodnoty do cílové proměnné.

Znaménko přiřazení ve vygenerovaném textu symbolu má pro přehlednost podobu šipky.

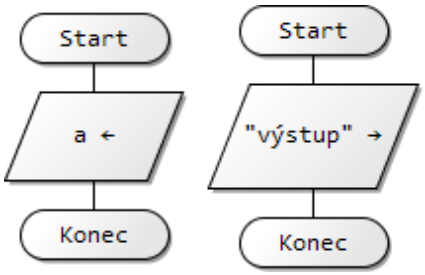


Obr. 11 Symbol zpracování

3.5.2 Vstup/Výstup

Tento symbol zajišťuje vstupně-výstupní operace pomocí dialogu (v animačním režimu aplikace). Vstupem je myšleno ruční zadání hodnoty (z klávesnice), výstupní operace zobrazí definovaný řetězec na výstupní zařízení (monitor).

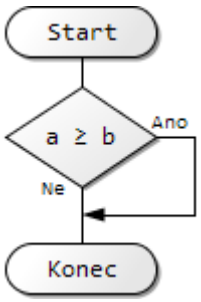
Vygenerovaný text pro rozlišení vstupu a výstupu používá opačně orientované šipky, kdy šipka směřující dovnitř symbolu značí vstupní a šipka ven výstupní operaci.



OBR. 12 SYMBOL VSTUP/VÝSTUP

3.5.3 ROZHODOVÁNÍ – PODMÍNKA

Symbol rozhodování neboli podmínka řídí tok programu na základě zadaného podmínkového výrazu. Je-li tento výraz vyhodnocen jako logická pravda, algoritmus pokračuje výstupní větví označenou jako „Ano“. V opačném případě, kdy je výraz vyhodnocen jako logická nepravda, se tok ubírá větví pod popiskem „Ne“ (else větev).



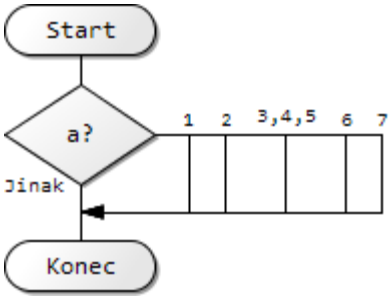
OBR. 13 SYMBOL ROZHODOVÁNÍ - PODMÍNKA

3.5.4 ROZHODOVÁNÍ – VÍCENÁSOBNÉ

Aplikace implementuje i rozhodování vícenásobné, které je v programování reprezentováno příkazem „switch“ či „case“.

Tento příkaz vyhodnotí cílovou proměnnou vůči zadaným konstantám. Konstanty mohou představovat číselné hodnoty nebo textové hodnoty ohraničené uvozovkami. Konstant je možno jedné větvi určit více, v takovém případě je nutné je oddělit čárkami.

Odpovídá-li cílová proměnná některé ze zadaných konstant, tok programu bude nasměrován k té větvi, které konstanta náleží. Neodpovídá-li cílová proměnná žádné z konstant, tok programu bude pokračovat větví označenou jako „Jinak“ (default větev).



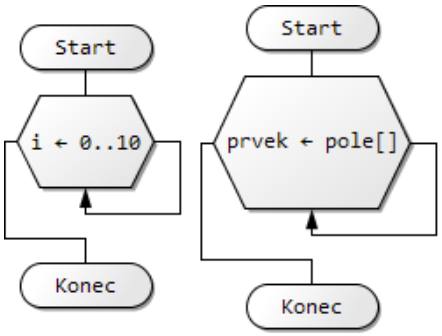
OBR. 14 SYMBOL ROZHODOVÁNÍ - VÍCENÁSOBNÉ

3.5.5 PŘÍPRAVA – CYKLUS PEVNÝM POČTEM OPAKOVÁNÍ

Symbol přípravy v aplikaci představuje dva specifické cykly s pevným počtem opakování.

První cyklus známe z programování jako klasický „for cyklus“, kdy je proměnné cyklu přiřazována číselná hodnota z definovaného „počítadla“ (Obr. 15, vlevo).

Druhý typ cyklu je z programování znám jako „for-each cyklus“, kdy jsou proměnné cyklu postupně přiřazeny všechny prvky cílového pole (Obr. 15, vpravo).

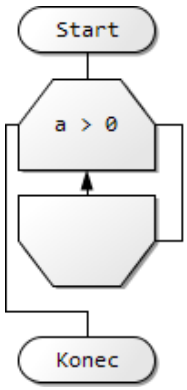


OBR. 15 SYMBOL PŘÍPRAVY- CYKLUS S PEVNÝM POČTEM OPAKOVÁNÍ

3.5.6 CYKLUS S PODMÍNKOU NA ZAČÁTKU

Symbol cyklu s podmínkou na začátku je definován mezními značkami. Horní mez cyklu řídí tok programu podobně jako symbol rozhodování.

Je-li podmínkový výraz symbolu vyhodnocen jako logická pravda, algoritmus pokračuje větví těla cyklu. Pokud je dosaženo dolní meze cyklu, tok programu je přesunut zpět k jeho horní hranici, aby zde byl znovu vyhodnocen podmínkový výraz. Je-li podmínkový výraz vyhodnocen jako nepravda, cyklus je přerušen a následuje příkaz pod tímto cyklem.

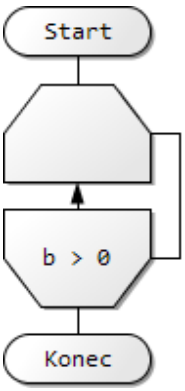


OBR. 16 SYMBOL CYKLUS S PODMÍNKOU NA ZAČÁTKU

Poznámka:
Všimněte si, že popsané chování cyklu je patrné i ze způsobu, jak jsou hranice cyklu propojeny spojnicemi.

3.5.7 CYKLUS S PODMÍNKOU NA KONCI

Symbol cyklu s podmínkou na konci je rovněž definován mezními značkami. Na rozdíl od cyklu s podmínkou na začátku, je však v tomto cyklu řídící podmínkový výraz umístěn v jeho dolní hranici. Toto umístění podmínkového výrazu nám zaručí, že tělo cyklu bude vykonáno minimálně jednou.



OBR. 17 SYMBOL CYKLUS S PODMÍNKOU NA KONCI

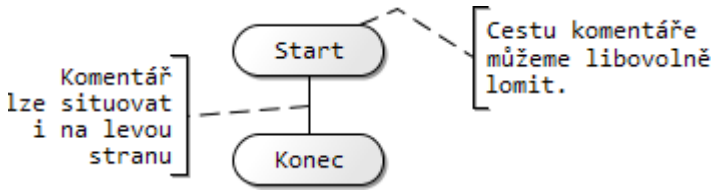
Poznámka:
Všimněte rozdílů ve spojnicovém propojení hranic cyklů s podmínkou nahoře a s podmínkou dole.

3.5.8 KOMENTÁŘ

Do vývojového diagramu lze umístit i symboly komentářů.

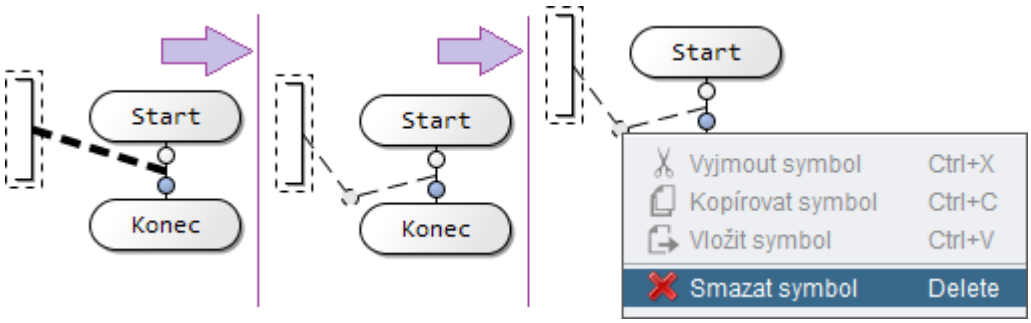
Komentář je obecně možné přiřadit buď na vlastní pozici uvnitř toku vývojového diagramu, nebo jej přiřadit ke kterémukoliv jinému symbolu.

Symbolem komentáře lze v editačním režimu aplikace dále manipulovat. Komentář můžeme umístit na libovolnou pozici na plátně, přičemž je nám umožněno i modulování jeho spojnice.



OBR. 18 SYMBOL KOMENTÁŘ

Úprava spojnice komentáře (Obr. 19) je realizována intuitivní cestou,



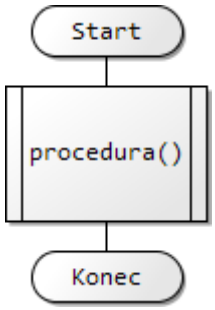
OBR. 19 ÚPRAVA SPOJNICE KOMENTÁŘE

kdy stačí myší najet nad požadovanou spojnici k editaci. Tato spojnice je pak zvýrazněna, a nám již stačí metodou „táhni a pusť“ (drag&drop) spojnici zlomit podle našich představ. Chceme-li později tento zlom upravit, stačí opět myší najet nad tento spoj. Podobným způsobem je možné spoj odstranit, kdy po najetí myši nad zlom spojnice vyvoláme kontextovou nabídku, v níž vybereme možnost jeho odstranění.

3.5.9 Předdefinované zpracování

Tento symbol představuje pojmenované zpracování, které se skládá z jedné nebo více operací, jež jsou specifikovány jinde. Jedná se tedy např. o volání funkcí či procedur.

Aplikace PS Diagram (zatím) procedury a funkce nepodporuje, tento symbol tedy nedisponuje nastavením své funkce.



Obr. 20 Symbol předdefinované zpracování

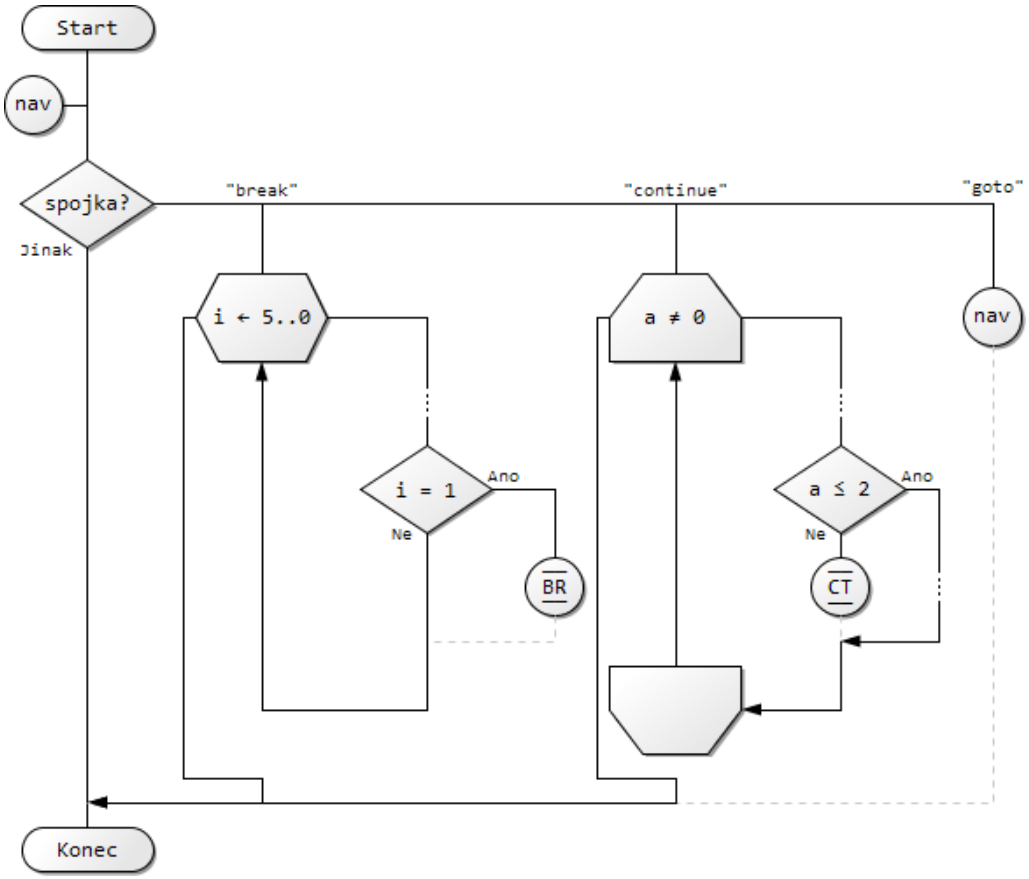
3.5.10 Spojka – break, continue, goto

Tento symbol může v aplikaci nabývat třech různých funkcí.

První funkce nese název „break“, stejnojmenného příkazu známého z programování. Příkaz break slouží k okamžitému opuštění cyklu tak, že je vykonán následující příkaz za cyklem. Symbolu je zároveň vygenerován automatický text.

Druhá funkce symbolu pod názvem „continue“ opět vykonává příkaz známý z programování, kdy je aktuální smyčka cyklu přerušena a tok programu je předán řídícímu symbolu cyklu. Symbolu je rovněž vygenerován automatický text.

Třetí, poslední funkcí tohoto symbolu je příkaz s názvem „goto“. Tento příkaz způsobí, že je tok programu přesměrován na místo označené symbolem návěští (kapitola 3.5.11) stejného identifikátoru. Je tedy nutné vyplnit text symbolu, který bude zároveň sloužit jako identifikátor.



OBR. 21 SYMBOL SPOJKA

Poznámka:
Všimněte si, že spojnice vedoucí od tohoto symbolu jsou zobrazeny čárkovaně. Takto jsou označena místa, kde se tok programu nemůže nikdy vyskytnout.

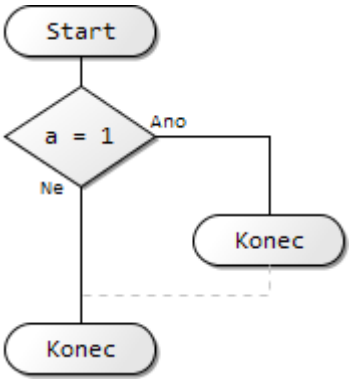
3.5.11 SPOJKA – NÁVĚŠTÍ

Symbol spojky v podobě návěští, představuje cíl pro přesměrování toku programu, je-li vstoupeno do symbolu s příkazem „goto“ (kapitola 3.5.10). Grafickou reprezentaci tohoto symbolu lze zpozorovat na Obr. 21 (druhý symbol shora).

3.5.12 MEZNÍ ZNAČKA

Mezní značka představuje začátek nebo konec programu. Každý algoritmus by měl být ohraničen právě těmito značkami, proto jsou v aplikaci automatickou součástí každého nově vytvořeného vývojového diagramu.

Tento symbol lze do vývojového diagramu přidat i vícekrát, v takovém případě vždy představuje konec programu.

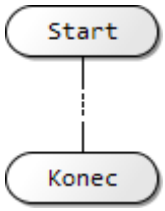


OBR. 22 SYMBOL MEZNÍ ZNAČKA

Poznámka:
Všimněte si, že spojnice vedoucí od tohoto symbolu jsou zobrazeny čárkovaně. Takto jsou označena místa, kde se tok programu nemůže nikdy vyskytnout.

3.5.13 VÝPUSTKA

Výpustku využijeme tehdy, chceme-li znázornit, že dochází k vypuštění symbolu nebo skupiny symbolů, kde ani druh ani jejich počet nemusí být definován.



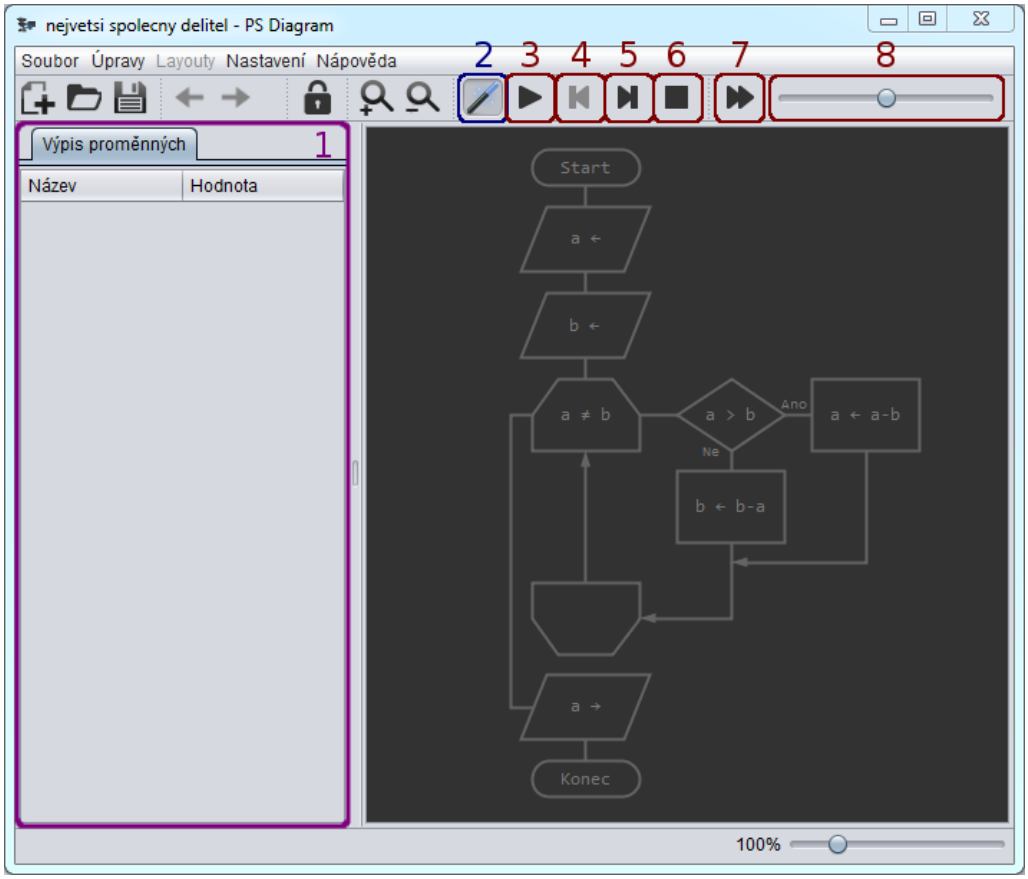
OBR. 23 SYMBOL VÝPUSTKA

4 Animační režim

Do animačního režimu aplikace vstoupíme sepnutím tlačítka magické hůlky (Obr. 24, popisek 2) v hlavním panelu aplikace.

Animační režim zprostředkovává vizualizaci algoritmického průběhu námi vytvořeného vývojového diagramu. Díky němu je pak možné pozorovat, jak se např. mění parametry cyklu, jakým způsobem algoritmus rozhoduje a jak se mění proměnné.

Vstupem do tohoto režimu se nám zpřístupní panel pro jeho ovládání (Obr. 24, popisky 3-8) a panel výpisu proměnných (Obr. 24, popisek 1). Panel s výpisem proměnných se nám bude hodit po spuštění samotného průchodu, kdy v něm budou přehledně zobrazeny aktuálně existující proměnné (kapitola 4.1.2).



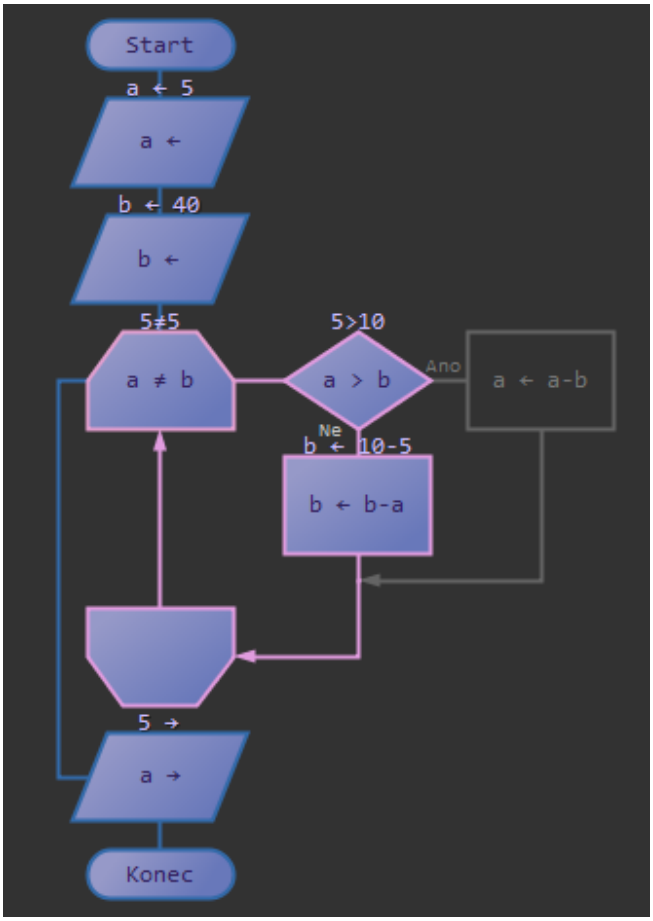
Obr. 24 Animační režim

Poznámka:
Animační režim spolupracuje s režimem náhledu, s vývojovým diagramem tak lze nadále manipulovat obvyklým způsobem.

4.1 Vizualizace průchodu vývojovým diagramem

Jistě jste postřehli, že vývojový diagram je v animačním režimu vykreslen jiným způsobem, než je tomu u ostatních režimů aplikace. Symboly s jejich spojnicemi jsou vykresleny prozatím nenápadně, to se ale mění po jejich aktivaci.

Vykoná-li symbol svou funkci, změní svou barvu tak, že lze tuto skutečnost jasně rozeznat. Uživatel tak zřetelně pozná, kudy se algoritmus vytvořeného vývojového diagramu ubírá či ubíral (Obr. 25).



Obr. 25 Vizualizace průchodu vývojovým diagramem

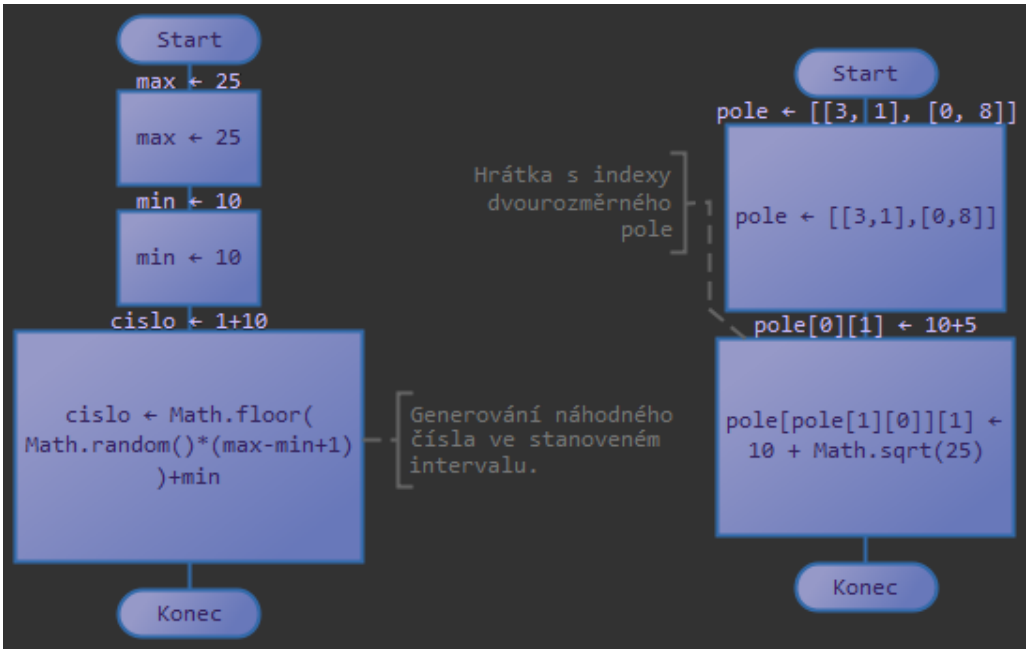
Z Obr. 25 můžeme navíc vypozorovat, že některé symboly provedly svou funkci více než jednou. Vykoná-li totiž symbol svou přiřazenou funkci, je barva jeho okraje zbarvena do odlišné barvy, než tomu bylo předtím. Stejně reagují i spojnice mezi symboly. Tato barva je upravována až do limitní bílé, která ve výchozím stavu odpovídá 25. aktivaci.

Tip:
Chcete-li s procházením vývojového diagramu začít od začátku, použijte tlačítko s popiskem „*Reset*“ (Obr. 24, popisek 6).

4.1.1 Mezivýpočty funkcí symbolů

Na Obr. 25 je možné pozorovat i zobrazení mezivýpočtů funkcí symbolů. Tyto mezivýpočty dávají uživateli jasnou představu o tom, jak bylo výsledku funkce daného symbolu dosaženo.

Mezivýpočty zobrazují vždy výsledky dílčích příkazů či hodnoty daných proměnných. Tento způsob zobrazení nám často může rozuzlit i jinak komplikované příkazové struktury (Obr. 26).



Obr. 26 Mezivýpočty složitějších struktur

4.1.2 Výpis proměnných

V panelu s výpisem proměnných se zobrazují aktuálně existující proměnné a jejich hodnoty (Obr. 27).

Proměnná, která byla posledním vykonáním funkce symbolu přidána nebo upravena, je vyobrazena tučným písmem. Je tak pro uživatele snazší se ve výpisu orientovat. Proměnné jsou navíc vypsány v jejich abecedním pořadí.

Speciálnímu nakládání se těší proměnné polí, kdy jsou jejich prvky na jednotlivých indexech zobrazeny ve stromové struktuře.

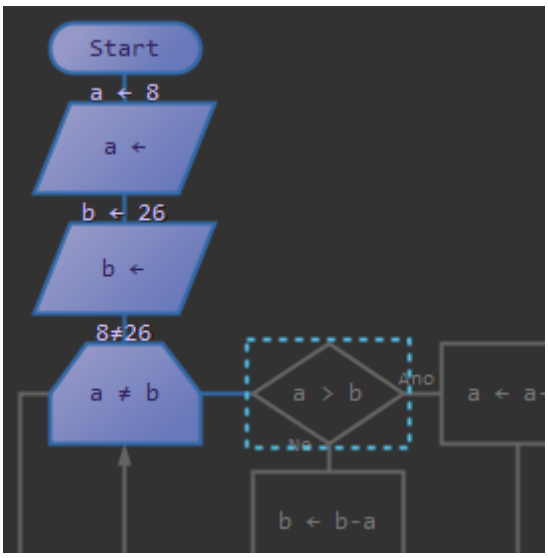
Název	Hodnota
cislo	5
novacek	**1**
▼ pole	[5, 2, 8]
[0]	5
[1]	2
[2]	8
prom	"moje hodnota"

OBR. 27 VÝPIS PROMĚNNÝCH

4.2 KROKOVÁNÍ

Procházení algoritmu vývojového diagramu můžeme ovládat několika způsoby. Jeden z těchto způsobů je právě funkce krokování.

Stiskem tlačítka krok vpřed (Obr. 24, popisek 5) aktivujeme symbol čekající na zpracování. Symbol čekající na zpracování je vždy modře ohraničen a my se tak vždy můžeme připravit na pozorování výsledku vykonání jeho funkce (Obr. 28).



OBR. 28 SYMBOL ČEKAJÍCÍ NA ZPRACOVÁNÍ

Aplikace PS Diagram disponuje i nadstandardní funkcí kroku zpět (Obr. 24, popisek 4), kdy je celý algoritmus po jeho (částečném) vykonání možné vrátit do požadovaného dřívějšího stavu.

Tip:
Pro urychlení krokování vývojovým diagramem využijte klávesových zkratek popsaných v nápovědách konkrétních tlačítek (zobrazí se po najetí myši).

4.3 Funkce spustit rychle

Dalším ze způsobů procházení algoritmu vývojového diagramu, je funkce jeho rychlého spuštění.

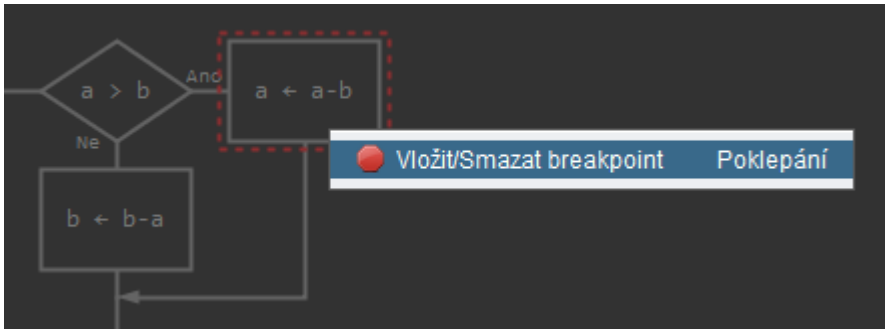
Po stisknutí odpovídajícího tlačítka (Obr. 24, popisek 7) je celý diagram procházen automaticky co nejvyšší možnou rychlostí. Tato funkce tedy simuluje chování skutečné aplikace, kdy není umožněno její krokování a konečný výsledek je zobrazen téměř okamžitě.

V případě, že bychom chtěli tento spuštěný proces pozastavit, můžeme tak učinit buď tlačítkem „*Pozastavit animaci*“ (Obr. 30, popisek 1) nebo využít funkce breakpoint, popsanou v následující podkapitole.

4.3.1 Funkce breakpoint

Breakpoint neboli zarážka je označení místa, kde dojde k zastavení spuštěného rychlého procházení diagramu. Používá se pro automatické pozastavení procházení v místě, kde chceme provést algoritmickou inspekci.

Pro vložení breakpointu vyvolejte v animačním režimu kontextovou nabídku nad některým z požadovaných symbolů, který bude touto zarážkou obklopen (Obr. 29). Breakpointů je možné definovat libovolné množství.



Obr. 29 Vložení breakpointu

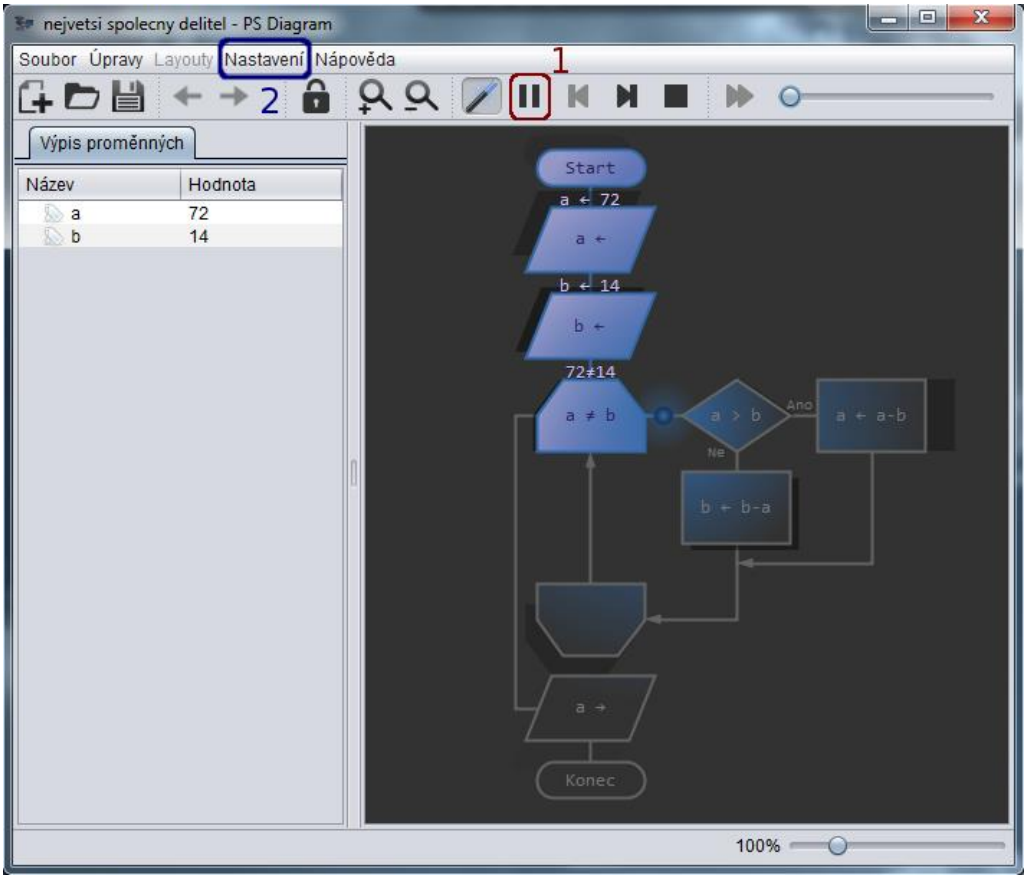
Tip:
Pro rychlé vložení breakpointu jednoduše poklepejte na požadovaný symbol.

4.4 ANIMACE

Animace je nejatraktivnější způsob průchodu vývojovým diagramem.

Aktivujeme-li ji tlačítkem s popiskem „*Spustit animaci*“, v místě symbolu aktuálně čekajícího na zpracování se zobrazí tzv. *průchodová kulička* (Obr. 30). Tato zářící kulička nás pak plynule provází celým průchodem, dávajíce nám jasně najevo, kudy se algoritmus našeho vývojového diagramu ubírá.

Animaci průchodu vývojovým diagramem můžeme kdykoliv pozastavit tlačítkem s popiskem „*Pozastavit animaci*“ (Obr. 30, popisek 1). Animaci lze zároveň popohnat, a to stiskem tlačítka pro krok vpřed (Obr. 24, popisek 5). Jak již jeho název napovídá, způsobí okamžitý přesun průchodové kuličky k symbolu, jež čeká na zpracování.



OBR. 30 ANIMACE

Tip:
Pracujeme-li s animací, je obzvlášť výhodné použití klávesových zkratek pro její ovládání. Využijte klávesu mezerník k jejímu spuštění/pozastavení.

4.4.1 Pohyb průběhové kuličky

Průběhová kulička se nepohybuje konstantní rychlostí. Její rychlost se mění v závislosti na prošlé trase vůči její délce a je tak docíleno vyšší záživnosti celé animace.

Rychlost, kterou se průběhová kulička pohybuje, je možno dále dynamicky upravovat. K tomuto účelu použijte posuvník v pravém horním rohu aplikace (Obr. 24, popisek 8).

4.5 Možnosti nastavení

Animační režim aplikace disponuje několika nastavitelnými položkami, které mají vliv na jeho chování. Tyto položky lze upravit v nastavení aplikace (Obr. 30, popisek 2) pod záložkou „*Animační režim*“.

V nastavení animačního režimu pak lze např. změnit přístup k proměnným, určit dosah záře průběhové kuličky či ji zcela vypnout. Nastavit můžeme i počet snímků za sekundu, kterým se animace bude vykreslovat.

4.5.1 Přístup k proměnným

Aplikace umožňuje k proměnným přistupovat dvěma způsoby, jež si popíšeme v následujících podkapitolách.

4.5.1.1 Globální přístup

Při globálním přístupu vytvořené proměnné existují globálně - nezanikají a po jejich vytvoření je s nimi možné pracovat kdykoliv a kdekoliv v rámci celého vývojového diagramu.

Tento přístup je znám například z programovacího jazyka Pascal, kdy jsou proměnné deklarovány v hlavičce zdrojového kódu.

4.5.1.2 Blokový přístup

Při blokovém přístupu vytvořené proměnné existují jen v rámci svého bloku (větve symbolu) a ve větvích do něj vnořených. Jakmile se tok programu ocitne mimo tento blok (nebo jeho vnořené bloky), proměnná zaniká.

Tento přístup k proměnným známe z většiny moderních (objektových) programovacích jazyků, deklarujeme-li proměnnou uvnitř těla metod (funkcí, procedur).

5 UKLÁDÁNÍ A OTEVÍRÁNÍ

Vytvořené vývojové diagramy lze standardně ukládat a otevírat ve formátu XML. K tomuto účelu můžeme využít tlačítek umístěných v menu „*Soubor*“ v horním levém rohu aplikace nebo tlačítek v hlavní programové liště pod touto nabídkou.

Uložené diagramy mají příponu souboru „*.psdiagram*“, kterou lze asociovat s aplikací PS Diagram. Otevření takového diagramu je pak jen otázka dvojího poklepání.

Poznámka:
Asociace je aktivní automaticky po prvním spuštění PS Diagramu. Pro její vypnutí či zapnutí navštivte nastavení aplikace.

Využít lze i zaběhlých klávesových zkratek nebo technologie „táhni a pusť“ (drag&drop). V tomto případě stačí soubor, který chceme otevřít, myší přetáhnout do aplikačního plátna (Obr. 1, popisek 1) a vývojový diagram je pak automaticky otevřen.

6 GRAFICKÝ EXPORT

Grafickou podobu vývojového diagramu lze kdykoliv exportovat do některého z podporovaných formátů.

V možnostech nastavení aplikace navíc nalezneme volby týkající se transparentnosti a velikosti okraje výstupního vývojového diagramu.

6.1 OBRÁZEK

Jednou z možností exportu vývojového diagramu je právě export do bitmapového obrázku. Formulář pro tuto funkci vyvoláme stiskem tlačítka „*Export do obrázku*“, které se nalézá v menu „*Soubor*“ v levém horním rohu aplikace.

6.2 PDF

Další možností, jak vývojový diagram exportovat, je jeho uložení do formátu PDF. V tomto případě je vývojový diagram uložen vektorově a my tak na rozdíl od grafiky bitmapové (export do obrázku), při jeho přibližování nepřicházíme o jeho obrazovou kvalitu.

7 IMPORT ZE ZDROJOVÉHO KÓDU

Zajímavou funkcí, kterou PS Diagram disponuje, je možnost reverzního inženýrství.

Tato funkcionalita nám dovoluje vývojový diagram vygenerovat z vloženého zdrojového kódu některého z podporovaných programovacích jazyků[1].

Formulář pro tuto funkci vyvoláme tlačítkem „*Import ze zdroj. kódu*“, umístěném v menu „*Soubor*“ v levém horním rohu aplikace.

Tip:
Import ze zdrojového kódu se stává vysoce užitečným zejména v případech, kdy bychom rádi prezentovali algoritmus, který již máme napsán v některém z podporovaných programovacích jazyků.

Poznámka:
Podporován je prozatím jen programovací jazyk Pascal.

8 EXPORT DO ZDROJOVÉHO KÓDU

Další nadstandardní funkcí, kterou PS Diagram disponuje, je možnost vývojový diagram naopak do zdrojového kódu exportovat.

Zhotovený vývojový diagram tak lze okamžitě převést do funkční aplikace některého z podporovaných programovacích jazyků[1].

Poznámka:
Podporován je prozatím jen programovací jazyk Pascal.

[1] Správnost vygenerovaného vývojového diagramu (nebo zdrojového kódu) není vždy stoprocentní, je proto doporučeno výstup ještě překontrolovat, případně poopravit.