Ciele projektu

Naším cieľom je vytvorenie textového editora primárne orientovaného na úpravu zdrojových kódov, ktorý bude využívať grafické prvky na zjednodušenie a zefektívnenie práce programátora. Špecifikácia editora zahŕňa nasledovné funkcionálne a nefunkcionálne požiadavky:

- práca s blokmi – umiestnenie editovaného kódu do grafických elementov, ktoré zodpovedajú logickým blokom, umožní najmä:
    - rýchle označovanie blokov na požadovanej úrovni programu
    - jednoduché presúvanie a vymieňanie častí programu
    - nastaviteľné skrývanie blokov a ich skupín
    - rozšírené zvýrazňovanie syntaxe (zmena farby, tvaru, správania bloku)
- tvorba dokumentácie – umožniť písanie dokumentácie priamo do zdrojového kódu, a poskytnúť:
    - priame formátovanie textu
    - samostatný aj kombinovaný výstup pre kompilovateľný program a dokumentáciu
- rozšíriteľnosť – možnosť pridávania podpory pre rôzne programovacie a značkovacie jazyky pomocou skriptov bez nutnosti zásahov do programu editora
- multiplatformovosť – použitie platformovo nezávislých technológií, testovanie minimálne pre Windows a Linux
- otvorenosť – použitie *OpenSource* technológií a možnosť využitia navrhnutých konceptov v iných podobne zameraných projektoch

Výsledkom nemá byť vývojové prostredie poskytujúce obrovské množstvo funkcií, ale jednoduchý editor ilustrujúci výhody zaobalenia logických blokov textu do grafických prvkov.

Za hlavný prínos navrhovaného editora považujeme zefektívnenie práce založené na sprehľadnení, zjednodušení a zrýchlení tvorby, ale najmä úprav a zlepšovania (refactoring) existujúceho kódu. Chceme tiež poukázať na dosiaľ málo preskúmané možnosti využitia vizuálnych elementov pri písaní programov (mimo vizuálneho programovania).

Problémová oblasť

Pre tvorbu zdrojových kódov existuje množstvo editorov. Veľa z nich podporuje skrývanie častí kódu (napríklad funkcií), dokonca *Notepad++* poskytuje možnosť vytvorenia vlastného bloku kódu, ktorý bude možné skryť. Žiadny z editorov však plne nevyužíva možnosti, ktoré poskytuje sémantická a syntaktická analýza. Žiadny neposkytuje rady programátorovi typu, že daný kód je už príliš hlboko vnorený, a že je vhodné prehodnotiť napísaný kód za účelom optimalizácie.

Veľa editorov využíva *rich text* na zvýraznenie kľúčových slov daného jazyka. Tento spôsob je veľmi zaužívaný, ale nevyužíva všetky možnosti grafických komponentov. Okrem zvýrazňovania syntaxe bude možné zvýrazniť hlavné bloky (napríklad funkcie, cykly) aj grafickými elementmi, predovšetkým obdĺžnikmi. Takéto zvýrazňovanie by malo prispieť k zvýšeniu prehľadnosti zdrojového kódu.

Je dôležité nemýliť si náš editor s vizuálnym programovaním typu *Simulink* v Matlabe. Nejde o programovanie pomocou grafických elementov, ale o sprehľadnenie a umožnenie iného pohľadu na zdrojový kód.

Za významnú tiež považujeme podporu hlavnej myšlienky *literate programming*-u, že kód samotný je dokumentáciou, toto nepodporuje žiadny z dostupných editorov. Bude umožňovať vkladanie obrázkov (najčastejšie asi UML diagramy) priamo do okna zdrojového kódu. Do formátu

.pdf bude možné exportovať to, čo má používateľ práve neskryté, teda len to čo potrebuje. Programátor si jednoducho nastaví, čo chce vidieť a už nebude musieť prepínať do iných aplikácií, všetko bude na jednom mieste, v jednom editore.

Prehľad riešenia

Celkový návrh riešenia sa skladá z troch častí, ktoré postupne rozoberieme:

- analýza programu a manažment hierarchie blokov,
- vizualizácia a správanie blokov,
- podpora dokumentácie a generovanie výstupov.

Väčšina navrhovanej funkcionality je postavená na rozdelení editovaného kódu do štruktúr logických blokov, kde logický blok predstavuje akúkoľvek syntakticko-lexikálnu jednotku daného jazyka. Na analýzu kódu využívame skriptovací jazyk *Lua*, konkrétne knižnicu *LPeg*. Pomocou gramatiky definovanej v skripte sa vygeneruje abstraktný syntaktický strom (AST). Každý uzol AST je platný blok. AST obsahuje celý text a originálny kód je z neho kedykoľvek rekonštruovateľný.

Pridať podporu nového jazyka znamená zapísať jeho gramatiku do skriptu pomocou relatívne jednoduchej syntaxi *LPeg*-u. Gramatika musí byť rozšírená tak, aby pokrývala celý text programu vrátane komentárov a bielych znakov a obsahovala tiež niektoré ďalšie potrebné elementy, napríklad kľúčové slová. Každá gramatika je doplnená skupinou formátovacích pravidiel, ktoré sa používajú na zvýrazňovanie syntaxe jazyka.

Gramatika pre otváraný súbor je zvolená automaticky podľa jeho typu alebo dodatočne manuálne. AST je potrebné počas písania textu pravidelne aktualizovať. Gramatiky sú preto navrhnuté tak, aby bolo možné analyzovať aj samostatné časti programov. Výhodou nie je ani tak vysoká rýchlosť (optimalizácie *LPeg*-u), ako jednoduchšie odchytávanie syntaktických chýb – nebude označený ako neplatný celý program, ale len časť obsahujúca danú chybu.

Ďalší možný spôsob využitia analýzy zdrojového kódu je upozorňovanie používateľa na pachy v zdrojovom kóde. Pri tejto analýze by malo byť možné detegovať niektoré základné pachy:

- dlhá metóda,
- veľká trieda,
- dlhý list parametrov,
- komentáre,
- zdvojený kód.

Používateľské rozhranie a vizualizácia grafických elementov je riešená pomocou multiplatformovej knižnice *Qt*. Bloky sú zobrazované ako rôznofarebné rámy obkolesujúce text. Pri prechode myšou cez blok sa objavia ovládacie prvky pre daný blok (napr. skrytie, presunutie alebo zvýraznenie bloku). K textu, ktorý je súčasťou viacerých vnorených blokov, je možné pristúpiť zvolením najšpecifickejšieho z nich. Pri presune bloku sú ako cieľové ponúkané pozície na rovnakej alebo príbuznej úrovni v AST, ako príklad uveďme nemožnosť presunutia príkazu mimo telo funkcie. Sémantické pravidlá nie sú zohľadňované. K niektorým blokom AST nemá používateľ možnosť pristupovať, ide hlavne o koreňový blok a bloky zodpovedajúce medzerám a tabulátorom.

Za najvhodnejšiu reprezentáciu bloku sme po skúmaní možností knižnice *Qt* zvolili prvok *QGraphicsItem*, ktorý je schopný niesť *rich text* informáciu ako i obrázok. Tento prvok je umiestnený na scéne, ktorá zabezpečuje jeho vykresľovanie. Scéna umožňuje natívnu drag-and-drop podporu pre bloky.

Každý znak textu v akomkoľvek bloku je štandardne editovateľný klávesnicou. Pri presune kurzora medzi blokmi sa naposledy upravovaný blok analyzuje a v prípade chyby je farebne zvýraznený. Text je automaticky formátovaný podľa pravidiel pripojených ku gramatike. Editačná oblasť podporuje techniku drag-and-drop, a to aj v prípade importu textu z iných aplikácií.

Editor umožňuje vkladať okrem komentárov daného jazyka aj špeciálne dokumentačné bloky. Vyznačujú sa tým, že text v nich podporuje základné formátovanie (font, veľkosť, farba, tučné, kurzíva…). Pri ukladaní sa dokumentačné bloky automaticky ošetria tak, aby nenarušovali kompilovateľnosť programu, najčastejšie sa vložia do štandardných komentárov. Formátovanie bude v súbore zaznamenané pomocou jednoduchých značiek.

Výstupom editora sú v prvom rade súbory zdrojových kódov (čistý text) určené na kompiláciu. Ďalej je možné generovať neformátovanú a formátovanú dokumentáciu a rôzne kombinácie.

Súčasný stav

Logika práce s blokmi a funkcionalita, ktoré bude náš editor poskytovať, sú už navrhnuté a z časti implementované. Vytvorená je základná kostra editora *TrollEdit* (názov odvodený od spoločnosti TrollTech, ktorá vyvíjala *Qt*) spolu s analýzou zdrojového kódu. Zatiaľ máme vytvorené dve gramatiky C a zjednodušené XML. Vieme vytvoriť abstraktný syntaktický strom zo zdrojového kódu. Zatiaľ to testujeme len v prostredí operačného systému Windows.

Základ grafickej časti je implementovaný, vieme vytvoriť scénu a umiestňovať do nej jednotlivé bloky, tak isto vieme pracovať s hierarchiou blokov. Pri presúvaní rodiča sa presúvajú aj všetci jeho následníci. Tak isto to funguje aj pri skrývaní. Blok sa vie automaticky zväčšiť, keď sa mení pri jeho editácii, alebo keď mu pribudne väčší podblok ako je on sám. Pri presúvaní blokov sa bloky automaticky umiestňujú na správnu pozíciu. Toto zatiaľ funguje iba pri blokoch na rovnakej úrovni.

Plán ďalšieho postupu

Do konca semestra budeme implementovať hlavne grafickú stránku editora:

- implementovať otvorenie zdrojového kódu editorom, aby sa správne umiestnili bloky do grafickej podoby,
- správne načítavanie blokov pri písaní zdrojového kódu pomocou nášho editora,
- ukladanie zdrojových súborov,
- exportovanie dokumentácie,
- hlbšia analýza zdrojového kódu za účelom hľadania pachov kódu (code smells),
- vytvorenie gramatiky pre jazyk JAVA.