Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Vývojové prostředí pro paradigmata programování

2023

Martin Půda

Vedoucí práce:
doc. RNDr. Michal Krupka Ph.D.

Studijní program: Aplikovaná informatika,
prezenční forma

Bibliografické údaje

Autor:	Martin Půda
Název práce:	Vývojové prostředí pro paradigmata programování
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2023
Studijní program:	Aplikovaná informatika, prezenční forma
Vedoucí práce:	doc. RNDr. Michal Krupka Ph.D.
Počet stran:	26
Přílohy:	elektronická data v úložišti katedry informatiky
Jazyk práce:	český

Bibliographic info

Author:	Martin Půda
Title:	Integrated Development Environment for the Programming Paradigms course
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2023
Study program:	Applied Computer Science, full-time form
Supervisor:	doc. RNDr. Michal Krupka Ph.D.
Page count:	26
Supplements:	electronic data in the storage of department of computer science
Thesis language:	Czech

Anotace

Tato bakalářská práce se zaměřuje na vytvoření vývojového prostředí pro předmět Paradigmata programování. Toto prostředí je vytvořeno v jazyce Common Lisp s využitím implementace Lispworks a cílem bylo vytvoření jednoduchého a uživatelsky přívětivého prostředí pro studenty tohoto předmětu. Práce se zabývá návrhem a implementací vývojového prostředí.

Synopsis

This bachelor thesis focuses on the development of an Integrated Development Environment (IDE) for the Programming Paradigms course. This environment is created in Common Lisp language using Lispworks implementation with the aim of providing a simple and user-friendly environment for students of this course. The thesis describes the design and implementation of the IDE.

Klíčová slova: vývojové prostředí; Lisp; Common Lisp; Lispworks

Keywords: Integrated Development Environment; IDE; Lisp; Common Lisp; Lispworks

Odevzdáním tohoto textu jeho autor/ka místopřísežně prohlašuje, že celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

Obsah

1 Úvod

Tato práce se zabývá návrhem a implementací vývojového prostředí pro předmět Paradigmata programování. Cílem bylo vytvořit takovou aplikaci, která by byla jednoduchá na ovládání, ale zároveň umožňovala přizpůsobení požadavkům předmětu a vyučujícího.

V rámci práce byla vytvořená desktopová aplikace pro Windows, která by měla nahradit aktuálně používané vývojové prostředí (LispWorks Personal Edition).

Základem aplikace je uživatelské rozhraní vytvořené s využitím knihovny CAPI, které obsahuje listener a editor zdrojového kódu jazyka Common Lisp. Pro uživatelské rozhraní bylo nutné implementovat základní funkcionalitu (v listeneru zápis výrazu a jeho vyhodnocení, práci s textem a ošetření chyb, v editoru práci s textem a soubory). Při implementaci tohoto chování bylo jako vzorové vývojové prostředí použito právě LW Personal Edition, výsledná aplikace se pak v detailech liší a obsahuje také další funkce (vyhodnocení výrazu z editoru v listeneru, uživatelská nastavení).

Ve spolupráci s vedoucím byly následně přidány další nástroje pro přizpůsobení vývojového prostředí a usnadnění práce studenta. Tyto nástroje jsou založeny na komunikaci aplikace se serverem vyučujícího a stahování různých souborů. Jde o hlavní rozdíl ve srovnání s LW Personal Edition, které takové funkce neobsahuje — bylo tedy nutné je navrhnout a implementovat.

Jedním z těchto nástrojů je volba jazyka (vybraná podmnožina funkcí a maker Common Lispu v závislosti na předmětu) — student si může zvolit jazyk vývojového prostředí a vyučující pak může tento jazyk měnit a přidávat vlastní funkce, makra a proměnné.

Dále je to práce s učebními texty a materiály (kódy, příklady). Aplikace dokáže tyto materiály stahovat od vyučujícího do adresáře studenta, který je má ihned k dispozici a může je prohlížet, případně upravovat.

Server vyučujícího dále obsahuje seznam chybových hlášek. Tyto hlášky se používají v listeneru a je možné je měnit, aby byly pro studenty srozumitelnější.

Funkcionalitu uživatelského rozhraní a chování jeho prvků lze měnit pomocí nastavení klávesových zkratek.

Všechny tyto nástroje umožňují přizpůsobení vývojového prostředí požadavkům předmětu a vyučujícího. Ačkoliv byla aplikace vyvíjena s ohledem na předmět Paradigmata programování (výběr jazyků, práce s učebními materiály), lze ji snadno modifikovat a použít i pro jiné kurzy zaměřené na výuku v Common Lispu.

2 Požadavky na vývojové prostředí

2.1 Paradigmata programování

Paradigmata programování je série předmětů vyučovaných na Univerzitě Palackého v Olomouci. Cílem těchto předmětů je seznámit studenty s principy programování bez úzké vazby na konkrétní programovací jazyk. Studijní plán obsahuje následující oblasti:

- funkcionální paradigma
- objektové (objektově orientované) paradigma
- paralelní programování
- logické programování

Jako modelový jazyk je většinou použit Common Lisp. Jde o standardizovaný dialekt jazyka Lisp, který kromě funkcionálního podporuje také objektové paradigma (CLOS) a lze jej snadno rozšířit pomocí maker. Jako vývojové prostředí se pak používá LispWorks.

2.2 LispWorks

LispWorks je komerční software vlastněný firmou LispWorks Ltd., který zahrnuje implementaci jazyka Common Lisp a vývojové prostředí. Je dostupný ve třech variantách:

- LispWorks Personal Edition
- LispWorks
  - Hobbyist Edition
  - Professional Edition
  - Enterprise Edition
- LispWorks for Mobile Runtime

LispWorks Personal Edition lze stáhnout zdarma na `www.lispworks.com`.

Vývojové prostředí LispWorks obsahuje kompiler pro ANSI Common Lisp (standard Common Lispu), další knihovny, editor zdrojového kodu, listener, debugger, inspector, rozhraní pro jazyky Java a C aj.

2.3 Vývojové prostředí

Studentům Paradigmat programování je doporučeno používat bezplatnou verzi LispWorks (LispWorks Personal Edition). Tato verze má ve srovnání s plnou verzí LispWorks následující nevýhody:

- je omezena velikost používané paměti (heap size)
- po pěti hodinách používání se vývojové prostředí samo vypne, bez uložení práce
- některé funkce (save-image, deliver, load-all-patches) nejsou dostupné

Kromě toho je toto vývojové prostředí pro studenty složité na ovládání a nelze ho přizpůsobit pro účely předmětu.

2.4 Cíle práce

Cílem bakalářské práce bylo naprogramovat s užitím LispWorks nové vývojové prostředí, které by bylo možné použít ve výuce Paradigmat programování.

Pro tvorbu aplikace byly stanoveny následující cíle:

- naprogramovat jednoduchý listener
- naprogramovat jednoduchý editor zdrojového kódu Lispu
- přidat jednoduché možnosti ladění (srozumitelná chybová hlášení)
- umožnit variabilní nastavení prostředí podle semestru příp. vyučovaného tématu (výběr podmnožiny jazyka, automatické načtení používaných knihoven)

3 Uživatelská dokumentace

3.1 Základní informace

Aplikace LW-IDE slouží jako jednoduché vývojové prostředí pro Paradigmata programování. Zahrnuje listener, editor zdrojového kódu a různé možnosti přízpůsobení podle předmětu a úrovně uživatele.

Důležitou součástí práce s aplikací je také využívání souborů ze serveru vyučujícího. Předpokládá se, že takový server bude obsahovat následující typy souborů:

- definice jazyka (podmnožina jazyka Common Lisp)
- definice chybových hlášek
- materiály – texty a zdrojové kódy pro výuku
- knihovny

Vývojové prostředí proto řeší také komunikaci s tímto serverem, stahování souborů a jejich využití a/nebo zobrazení uživateli.

3.2 Listener

Po spuštění programu se zobrazí okno s listenerem. Listener je nástroj pro zápis výrazu (*expression*) v Common Lispu a jeho následné vyhodnocení. Výraz se zapisuje za prompt (znak >) a jeho vyhodnocení se provede stisknutím klávesy Enter. Před stisknutím klávesy Enter je nutné umístit kurzor na konec výrazu, v opačném případě je část výrazu následující za kurzorem umístěna na nový řádek.

Po stisknutí klávesy Enter je zobrazen výsledek vyhodnocení zapsaného výrazu. Výsledek také může obsahovat data zapisovaná na výstup (funkce `fomat`, `print` aj.) nebo více hodnot (funkce `values`, vyhodnocení funkce vracející `values`).

Pokud je kurzor umístěn mimo aktuální prompt, na dané místo nelze zapisovat ani vkládat text.

Listener dále obsahuje několik proměnných pro snadnější práci s výrazy a jejich výsledky (proměnné a jejich význam jsou dány standardem):

Po vyhodnocení výrazu je výsledek uložen do proměnné $\star$ (předchozí výsledek pak do $\star\star$, ještě starší je uložen do $\star\star\star$) a tak je možné ho použít v následujícím výrazu, aniž by bylo nutné jej ručně kopírovat. V případě, že výraz vrací více hodnot, je uložena pouze první hodnota.

Podobně pak fungují proměnné `/`, `//` a `///`, které obsahují seznam se všemi výsledky vyhodnocení daného výrazu.

Hodnotou proměnné – (znak pro minus) je pak výraz aktuálně vyhodnocovaný v listeneru.

Pokud při vyhodnocování výrazu dojde k chybě, je o tom uživatel informován chybovou hláškou. Text chybové hlášky je možné změnit — vývojové prostředí využívá soubor ze serveru vyučujícího, ve kterém má každý typ chyby přiřazenou určitou chybovou hlášku.

3.3 Menu listeneru

Okno s listenerem dále obsahuje menu pro práci se soubory (`File`) a menu pro práci s textem (`Edit`). Akce těchto menu lze spustit kliknutím na jednotlivou akci nebo stisknutím klávesové zkratky pro danou akci.

- File
  - New (Ctrl + N) – otevření okna s novým dokumentem
  - Open – vybrání souboru, který se následně otevře v novém editoru
  - Load – vybrání souboru, který je následně načten do aktuálního prostředí
  - Exit – zavření celého programu
- Edit
  - Undo (Ctrl + Z) – zrušení poslední akce
  - Cut (Ctrl + X) – vyjmutí textu
  - Copy (Ctrl + C) – zkopírování textu
  - Paste (Ctrl + V) – vložení textu

3.4 Editor

Okno s editorem zdrojového kódu lze otevřít z menu listeneru akcemi `New` (editor s novým dokumentem) nebo `Open` (editor s otevřeným souborem). Slouží k zobrazení a upravování delšího zdrojového kódu.

Je také možné vybrat určitý výraz z editoru a vyhodnotit ho v listeneru. V takovém případě je nutné umístit kurzor v listeneru za aktuální prompt (znak >) (pokud se za tímto znakem již nachází nějaký text, je nutné jej smazat), následně umístit kurzor v editoru za vybraný výraz a stisknout Ctrl + E. Vybraný výraz se zkopíruje do listeneru a vyhodnotí se stejným způsobem, jako kdyby byl zapsán přímo do listeneru.

Pokud se v editoru nachází nějaký výraz, jehož čtení skončí chybou (nemusí jít o výraz vybraný uživatelem), je vyhodnocování výrazu ukončeno a do listeneru je vypsána chybová zpráva.

3.5 Menu editoru

Menu editoru se shoduje s menu listeneru, kromě toho však obsahuje další akce, které lze opět spustit klávesovou zkratkou nebo kliknutím na akci v menu:

- File
  - Save (Ctrl + S) – uložení dosud neuloženého nebo otevřeného souboru
  - Save As – uložení otevřeného souboru pod novým názvem

Tlačítkové menu editoru obsahuje kromě tlačítka pro uložení také tlačítko pro zkompilování textu v editoru. V takovém případě dojde ke kompilaci všech výrazů v editoru a výsledek každé kompilace je zapsán do listeneru (úspěšná kompilace / chyba / výsledek kompilovaného výrazu). Ve srovnání s výběrem jednoho výrazu z editoru a jeho následným vyhodnocením není nutné umístit kurzor v listeneru za znak > nebo mazat text za tímto promptem.

3.6 Tools – preferences

Menu `Preferences` umožňuje nastavit aktuální jazyk (viz dále) nebo font. Výběr je nutné potvrdit tlačítkem `OK`. Změna nastavení se projeví ve všech otevřených oknech uživatelského rozhraní (pokud je například otevřeno více editorů, není nutné nastavovat font pro každý editor zvlášť) a přetrvává i při dalším spuštění programu.

3.7 Commands

Položka menu `Commands` obsahuje některé další akce pro listener a/nebo editor. Tyto akce jsou dostupné také použitím dané klávesové zkratky.

3.8 Jazyk

Důležitou součástí vývojového prostředí je také možnost nastavení jazyka (podmnožina jazyka Common Lisp zvolená v závislosti na obsahu předmětu). Výchozí hodnotou je jazyk `LW-IDE`, toto nastavení lze změnit v `Preferences → Language`. Pro účely výuky Paradigmat programování nastavení obsahuje čtyři možnosti (`PP1` – `PP4`). Název právě používaného jazyka se také zobrazuje pod listenerem.

Popis funkcí obsažených ve vybraném jazyce by měl být zapsán v souboru umístěném na serveru vyučujícího. Soubor nemusí obsahovat pouze podmnožinu funkcí jazyka Common Lisp, lze také definovat nové funkce (příp. makra nebo proměnné) nebo změnit dokumentaci určité funkce.

Při změně jazyka vývojového prostředí dojde ke stažení tohoto souboru a načtení určených funkcí.

3.9 Dokumentace

Klávesovou zkratkou `Ctrl + I` lze zobrazit dokumentaci aktuálně dostupných funkcí, maker, speciálních operátorů a proměnných (v závislosti na právě používaném jazyce). Tato možnost je dostupná z listeneru i editoru.

3.10 Materials

Tlačítkem `Materials` lze zobrazit dostupné materiály pro aktuální jazyk. Tyto materiály se stahují ze serveru vyučujícího a opětovné stažení lze provést tlačítkem `Reload`. Následně lze materiál otevřít dvojklikem- soubory typu `lisp` se otevírají přímo ve vývojovém prostředí, ostatní soubory pak v programu, který je nastaven jako výchozí pro jejich otvírání v rámci operačního systému.

Pokud není možné se k serveru vyučujícího připojit, uživatel je o tom informován chybovou hláškou.

4 Podrobná dokumentace

4.1 Základní informace

LW-IDE je desktopová aplikace určená pro operační systém Windows 10, která umožňuje vývoj programů v jazyce Common Lisp.

Samotná aplikace byla napsána v jazyce Common Lisp s použitím implementace LispWorks (verze 8.0). Kromě funkcí, maker, speciálních operátorů a proměnných popsaných standardem ANSI proto využívá také knihovny a funkce, které jsou součástí zvolené implementace. Při vývoji byl kladen důraz na výslednou velikost celého programu, aplikace proto nepoužívá žádné balíčkovací nástroje (Quicklisp) a externí knihovny.

4.2 Použité knihovny a packages LispWorks

- CAPI

  CAPI (Common Application Programmer's Interface) je knihovna určená pro tvorbu grafických uživatelských rozhraní. Poskytuje základní sadu prvků uživatelského rozraní, nástroje k popisu jejich chování a umožňuje definovat nové prvky. Tato knihovna byla použita pro vytvoření uživatelského rozhraní aplikace.

- editor

  Výchozí chování editoru LispWorks je popsáno v dokumentu Editor User Guide. Tento dokument také popisuje možnosti, jak toto chování změnit. Třída `editor-pane` z CAPI využívá stejný model a proto byly pro její definování použity funkce a makra z package `editor`.

- graphics-ports

  Package `graphics-ports` je určen pro práci s rastrovou grafikou a pro vytváření obrázků a animací, které mohou být zobrazeny s použitím `output-pane` z CAPI. V rámci LW-IDE se používá k práci s fonty a zobrazování ikon.

- comm

  Package `comm` poskytuje rozhraní pro práci s TCP/IP. Umožňuje připojení k serveru nebo jeho implementaci.

- system

  Package `system` je v rámci aplikace použit práci se souborovým systémem a pro vykonávání příkazů pomocí `cmd.exe`.

- external-format

  Package `external-format` je použit pro převod mezi různými formáty dat.

4.3 Soubory lw-ide

Bakalářská práce lw-ide zahrnuje tyto soubory se zdrojovým kódem (popsáno v pořadí, v jakém jsou načteny při load):

- `package.lisp`
- `variables.lisp`
- `utils.lisp`
- `docs.lisp`
- `editor.lisp`
- `listener.lisp`
- `request.lisp`
- `actions.lisp`
- `callbacks.lisp`
- `materials.lisp`
- `preferences.lisp`
- `lw-ide.lisp`

Následuje popis těchto souborů v abecedním pořadí.

4.3.1 actions.lisp

Obsahuje definice pro akce dostupné z menu `ide-editor-interface` (interface editoru) nebo `lw-ide` (interface listeneru).

4.3.2 callbacks.lisp

Obsluha kláves stisknutých v listeneru je definovaná v souboru `callbacks.lisp`. Tyto funkce (spolu s klávesou, která je spouští) jsou přidány do output-modelu listeneru v jeho `initialize-instance` metodě. Jde o tyto klávesy:

- Return
  `return-callback`
- Backspace
  `backspace-callback`
- Delete
  `delete-callback`

Dále je zde definována funkce `send-to-listener` (a její pomocné funkce) pro editor, která se volá při stisknutí klávesy uvedené v souboru `shortcuts.txt`.

4.3.3 docs.lisp

Funkce obsažené v souboru `docs.lisp` slouží k zobrazení dokumentace. Tato funkcionalita je dostupná užitím klávesové zkratky Ctrl + I, případně tlačítkem v menu.

Po použití této zkratky je zobrazeno dialogové okno se seznamem dostupných funkcí, maker a proměnných a po kliknutí na vybraný symbol se zobrazí název, typ (funkce, makro, speciální operátor, proměnná), seznam argumentů a dokumentace.

4.3.4 editor.lisp

Obsahuje definice tříd `ide-editor-interface` a `ide-editor`.

4.3.5 listener.lisp

Obsahuje definici třídy `ide-listener`.

4.3.6 lw-ide.lisp

Soubor `lw-ide` obsahuje definici interface `ide` (zobrazeno hned po spuštění aplikace) a funkci `run-ide`, která inicializuje uživatelské proměnné a zobrazí uživatelské prostředí.

4.3.7 materials.lisp

Funkce obsažené v souboru `materials.lisp` slouží ke stahování učebních materiálů z daného serveru. Pro jejich následné zobrazení uživateli je použito interface `materials-view`.

4.3.8 package.lisp

Obsahuje definici výchozího package LW-IDE a definice výukových packages :PP1, :PP2, :PP3 a :PP4.

4.3.9 preferences.lisp

Preference lze nastavit akcí z menu `ide` nebo `ide-editor-interface` (pod Tools). Zahrnují:

- Language – změna jazyka
- Font – změna typu a velikosti fontu
- Resources-Url – změna adresy serveru vyučujícího (výchozí hodnota: `http://krupka.i` (pouze pro uživatele `:admin`)

Kliknutím na položku menu `Preferences` je vyvoláno dialogové okno, které umožňuje změnit hodnoty jednotlivých předvoleb. Okno lze zavřít bez uložení změn křížkem nebo tlačítkem Cancel, tlačítkem OK se uloží nastavené hodnoty.

4.3.10 request.lisp

Obsahuje funkce pro provedení HTTP requestu. Tyto funkce jsou dále použity v souboru `materials.lisp`.

4.3.11 utils.lisp

Soubor `utils.lisp` obsahuje pomocné funkce pro čtení a dekódování bytů a práci s datem a časem. Tyto funkce se následně používají ve funkcích obsažených v souborech `request.lisp` a `materials.lisp`.

4.3.12 variables.lisp

Obsahuje definice globálních proměnných pro práci s vývojovým prostředím. Jde o následující proměnné:

- *errors* – výchozí hodnota: nil

  Tato proměnná označuje seznam s chybovými hláškami a její hodnota je použita při vyhodnocování výrazů v listeneru.

  Hodnota této proměnné může být změněna při spuštění vývojového prostředí, kdy je proveden http request na server vyučujícího. Pokud zde existuje soubor `errors.txt`, hodnota této proměnné je nastavena na jeho obsah. Mělo by jít o seznam dvojic (typ-chyby řetězec).

- *shortcuts* – výchozí hodnota: nil

  Tato proměnná označuje seznam s klávesovými zkratkami (definuje vyučující) a funkcemi, které je obsluhují.

  Hodnota této proměnné může být změněna při spuštění vývojového prostředí, kdy je proveden http request na server vyučujícího. Pokud zde existuje soubor `shortcuts.txt`, hodnota této proměnné je nastavena na jeho obsah. Mělo by jít o seznam seznamů
  ve tvaru `((:gesture-spec znak :control) funkce)`.

- *app-data-path* – výchozí hodnota: cesta k LocalAppData

- *app-data-dir* – výchozí hodnota: cesta k adresáři "LW-IDE", vytvořeném ve složce *app-data-path*

- *current-pos* – výchozí hodnota: výsledek vyhodnocení `(current-pathname)`

4.4 Grafické uživatelské rozhraní

Základní grafické uživatelské rozhraní je popsáno v souborech `listener.lisp`, `editor.lisp` a `lw-ide.lisp`. Kromě toho vývojové prostředí obsahuje i další třídy definované pomocí `define-interface`, a to konkrétně `materials-view` (v `materials.lisp`) a `settings-interface` (v `preferences.lisp`).

4.4.1 ide

Typ: interface

Definované v: `lw-ide.lisp`

Při načtení programu se vytvoří a zobrazí instance této třídy. Předpokládá se, že během práce s programem bude existovat pouze jedna tato instance. Prvky uživatelského rozhraní jsou (shora dolů) menu s akcemi, menu s tlačítky a listener (instance třídy `ide-listener`).

4.4.1.1 Menu

Menu obsahuje některé přednastavené akce, další se doplňují dynamicky (položka commands).

Obsluhu akcí menu zajišťují funkce definované v souboru `actions.lisp`.

- File
  - New (Ctrl + N) - vytvoření nové instance třídy `ide-editor-interface`
  - Open - vybrání souboru, který se následně otevře v nové instanci `ide-editor-interface`
  - Load - vybrání souboru, který je následně načten do aktuálního prostředí
  - Exit - zavření uživatelského rozhraní
- Edit
  - Undo (Ctrl + Z) – zrušení poslední akce
  - Cut (Ctrl + X) – vyjmutí textu
  - Copy (Ctrl + C) – zkopírování textu
  - Paste (Ctrl + V) – vložení textu
- Tools
  - Preferences – otevření okna s možnostmi personalizace vývojového prostředí (volba jazyka, nastavení fontu...)
- Commands
  položky tohoto menu se vytvoří dynamicky po vytvoření instance `ide` na základě obsahu souboru `shortcuts.txt`

4.4.1.2 Tlačítkové menu

Tlačítkové menu se nachází pod textovým menu a obsahuje tlačítko pro zobrazení dostupných učebních materiálů `Materials`. Stažení a zobrazení těchto materiálů zajišťují funkce ze souboru `materials.lisp`.

4.4.1.3 Listener

Instance třídy `ide-listener`.

4.4.2 ide-listener

Typ: editor-pane

Definované v: `listener.lisp`

Jedna instance této třídy se vytvoří po spuštění programu jako součást interface `ide` a předpokládá se, že během práce s programem bude existovat pouze tato instance. Tento listener by měl být funkcemi podobný listeneru z vývojového prostředí Lispworks. Část funkcionality (barevné značení párových závorek, nápověda pomocí klávesy Tab aj.) byla převzata beze změny užitím bufferu s modem `Lisp`, obsluha stisknutí kláves pak byla doprogramována a je obsahem souboru `callbacks.lisp`.

4.4.3 ide-editor-interface

Typ: interface

Definované v: `editor.lisp`

Instance této třídy se vytvoří akcí `New` nebo `Open` z rozhraní `ide`. Předpokládá se, že během práce s programem může existovat více instancí této třídy. Prvky uživatelského rozhraní jsou textové menu, tlačítkové menu a editor (instance třídy `ide-editor`).

4.4.3.1 Menu

Menu obsahuje některé přednastavené akce, další se doplňují dynamicky (položka commands). Obsluhu akcí menu zajišťují funkce definované v souboru `actions.lisp` (stejný soubor jako u třídy `ide`).

Menu editoru obsahuje stejné položky jako menu listeneru, a k tomu některé další:

- File
  - Save (Ctrl + S) – uložení otevřeného souboru
  - Save As – uložení otevřeného souboru pod novým názvem

4.4.3.2 Tlačítkové menu

Tlačítkové menu se nachází přímo pod textovým menu a obsahuje tlačítka dvou akcí:

- Save - uložení otevřeného souboru
- Compile – kompilace textu v editoru, výsledek této kompilace je následně zobrazen v listeneru

4.4.3.3 Editor

Instance třídy `ide-editor`.

4.4.4 ide-editor

Typ: editor-pane

Definované v: `editor.lisp`

Instance této třídy se vytvoří jako součást `ide-editor-interface` a během práce s programem může existovat více instancí této třídy, každá náležející jedné instanci `ide-editor-interface`. Je možné editovat text pouze v editorech otevřených při vytvoření nového souboru (akce `Open` nebo otevření již existujícího souboru (akce `Open`)- editory otevřené při zobrazení výukového souboru s příponou .lisp jsou pouze pro čtení. Tento editor by měl být podobný editoru z vývojového prostředí Lispworks. Část funkcionality (barevné značení párových závorek, nápověda pomocí klávesy Tab aj.) byla podobě jako u třídy `ide-listener` převzata beze změny užitím bufferu s modem `Lisp`.

4.5 Uživatelský adresář

Při spuštění aplikace je ověřeno, zda existuje adresář `*app-data-dir*` — v případě potřeby je vytvořen.

Stažení souboru s daným jménem ze serveru je řešeno funkcí `recursive-download` (v `materials.lisp`).

Pokud bylo stažení souboru úspěšné a soubor daného jména dosud v uživatelském adresáři neexistoval (nebo pokud je stažený soubor novější než soubor u uživatele), tento soubor je zapsán do uživatelského adresáře.

K souborům je následně přistupováno pomocí funkce `get-resource` (v `variables.lisp`) která na základě jména vrací celou cestu k souboru.

4.6 Typy uživatelů

Vývojové prostředí rozlišuje dva typy uživatelů: `:user` a `:admin`. Aktuální hodnota je uložena v `(user-preference "LW-IDE-preferences" "User'')`, výchozí hodnota je `:user`. Typ uživatele lze přepnout pomocí klávesové zkratky Ctrl + B a tato volba ovlivňuje možnosti uživatele (nabídka Preferences).

5 Použití vývojového prostředí

5.1 Instalace a spuštění

Vývojové prostředí lw-ide se otevře spuštěním souboru `lw-ide.exe`. Tento spustitelný soubor vznikl ze zdrojových kódů procesem zvaným *delivery*. Ten je založen na tom, že část paměti Lispworks (tzv. *lisp image*) obsahující funkce a další součásti uživatelského kódu se uloží na disk. Při tomto procesu může uživatel zapínat různé optimalizace, díky kterým je možné *lisp image* zmenšit. Kvůli jednoduchosti nebyla možnost optimalizací využita.

5.2 Práce s vývojovým prostředím - příklady užití

Následující kapitola popisuje práci s vývojovým prostředím z pohledu předpokládaných uživatelů vývojového prostředí — vyučujícího a studenta předmětu Paradigmata programování.

5.2.1 Vyučující

Pracuje se soubory v adresáři umístěném na serveru určeném pro vývojové prostředí (nyní `LW-IDE`). Tento adresář obsahuje:

- Soubor s klávesovými zkratkami (globální) `shortcuts.txt`
- Soubor s chybovými hláškami (globální) `errors.txt`
- Adresář pro každý kurz

Adresář pro každý kurz může obsahovat:

- Soubory s definicemi (verze jazyka)
- Soubory s příklady
- Soubory s knihovnami
- Učební texty a jiné dokumenty

5.2.2 Student

Pracuje se spustitelným souborem. Po spuštění může:

- Vyhodnocovat výrazy v listeneru
- Otevřít existující soubor v editoru, vytvořit nový soubor nebo načíst soubor do aktuálního prostředí
- Uložit otevřený soubor

- Zkompilovat otevřený soubor
- Zobrazit a otevřít dostupný učební text, zdrojové kódy a příklady

Další akce lze doprogramovat a poté zpřístupnit pomocí klávesových zkratek (soubor `shortcuts.txt`, který může být obsažen v adresáři na serveru). Ve vývojovém prostředí jsou zahrnuty tyto dvě výchozí akce:

- Vyhodnocení výrazu z editoru v listeneru
- Zobrazení dokumentace pro dostupné funkce, makra a proměnné

Student dále může změnit nastavení uživatelského prostředí:

- Změnit jazyk (dostupná podmnožina jazyka Common Lisp)
- Změnit velikost a typ fontu

Závěr

V rámci bakalářské práce bylo vytvořeno jednoduché vývojové prostředí pro předmět Paradigmata programování, které lze také jednoduše upravit a použít pro jiné kurzy zaměřené na výuku Common Lispu.

Conclusions

As part of the bachelor's thesis, a simple development environment was created for the course "Programming Paradigms", which can also be easily modified and used for other courses focused on teaching Common Lisp.

A Obsah elektronických dat

Elektronická data odevzdaná v systému katedry informatiky obsahují tyto položky:

`text/`
: Adresář s textem práce ve formátu PDF, vytvořeným s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce. Soubory potřebné pro vytvoření PDF dokumentu textu jsou v ZIP archivu nazvaném `bakalarska-prace.zip`.

`README.txt`
: Textový soubor s informacemi o instalaci a spuštění software vytvořeného v rámci práce.

`lw-ide.zip`
: ZIP archiv se soubory a zdrojovými kódy práce. Po stažení a extrahování obsahuje tyto soubory a adresáře:

  - resources/
    - compile.png
    - icons.bmp
  - actions.lisp
  - callbacks.lisp
  - deliver.lisp
  - docs.lisp
  - editor.lisp
  - listener.lisp
  - load.lisp
  - lw-ide.exe
  - lw-ide.lisp
  - materials.lisp
  - package.lisp
  - preferences.lisp
  - request.lisp
  - utils.lisp
  - variables.lisp

Literatura

[1] Seibel, Peter. *Practical Common Lisp*. Berkeley, Calif.: Apress, 2005. xxv, 499 s. ISBN 9781590592397.

[2] *Common Lisp HyperSpec*. [online]. 2005 [cit. 2023-6-13]. Dostupný z: ⟨http://www.lispworks.com/documentation/HyperSpec/Front/⟩.

[3] LispWorks (ed.). *Editor User Guide* [online]. Version 8.0, 2021 [cit. 2023-6-13]. Dostupný z: ⟨http://www.lispworks.com/documentation/pdf/lw80/editor-u-8-0.pdf⟩.

[4] LispWorks (ed.). *CAPI User Guide and Reference Manual* [online]. Version 8.0, 2021 [cit. 2023-6-13]. Dostupný z: ⟨http://www.lispworks.com/documentation/pdf/lw80/capi-m-8-0.pdf⟩.