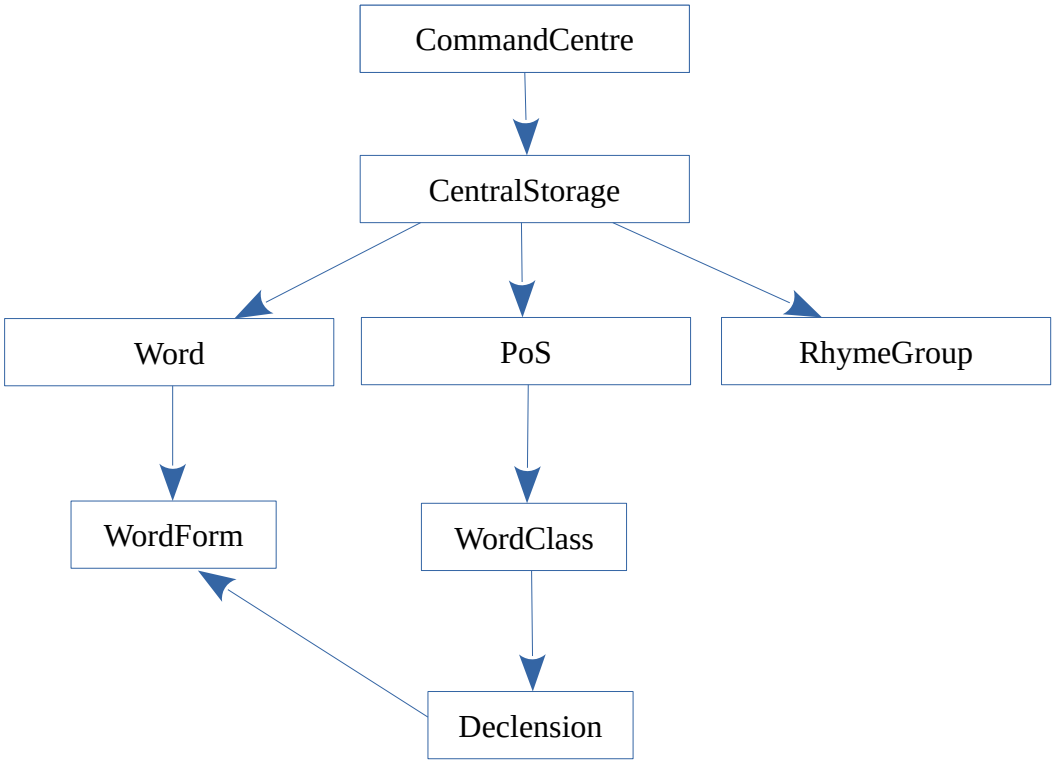
Rosalind's Dictionary - Dokumentace

Popis programu

Slovník Rosalind's Dictionary slouží ke tvorbě konstruovaných jazyků (conlang). Umožňuje uživateli tvořit slovní zásobu conlangu, a zároveň automaticky tvoří různé tvary slov podle pravidel, které uživatel určí. Program se ovládá z příkazového řádku skrze příkazy. Uživatel může skrze program dohledat tvary slov, jejich výslovnost, a s jakými dalšími slovy se rýmují.

Schéma struktur a tříd

Toto schéma naznačuje hierarchii struktur a tříd v aplikaci, detailně popsaných v aplikaci:



Struktury

TransformUnit

Struktura TransformUnit uschovává data, podle nichž se tvoří tvary slov. Obsahuje tyto atributy:

```
string identifier
string regex
string replace
```

Atribut identifier je regex string, podle kterého se určuje, zda se má pravidlo této TransformUnit aplikovat na dané slovo, či ne. Např. TransformUnit, kde identifier je "a$", aplikuje změny na slově "gleira", ale ne na slově "hvils".

Atribut regex je regex string, který určí, jaká část měněného slova má být zaměněna za jiný substring.

Atribut replace určuje substring, který má nahradit část slova určenou atributem regex. Např. TransformUnit, kde regex je "a$" a replace je "u", udělá ze slova "gleira" slovní formu "gleiru".

Příklad:

Vezměme v potaz objekt Declension (popsán níže), který obsahuje tři jednotky TransformUnit:

```
identifier = "[aeioyáéíóúýäöæ]", regex = ".$", replace = "än"
identifier = "[bdðfghjklmnprstþv]", regex = "$", replace = "än"
identifier = "hv", regex = "^h", replace = "g"
```

Slovní forma slova "hvæða" vytvořena tímto objektem bude "gvæðän" - aplikováno je pouze první pravidlo, kde je poslední znak nahrazen písmenem "u", a třetí pravidlo, kde se z "hv" stává "gv".

Ze slova "hvils" pak vznikne "gvilsän" - třetí pravidlo je aplikováno stejně jako u prvního slova, ale místo prvního pravidla je aplikováno pravidlo druhé, kde se "än" přidává na konec slova, aniž by nahrazovalo poslední znak.

Třídy

Word

Třída Word representuje jedno slovo ve slovníku. Lze si ho představit jako kartičku - na jedné straně je slovo v původním jazyce (např. anglicky nebo česky), na druhé straně je slovo v conlangu, spolu s jeho různými formami. Program zachycuje slovní formy pouze v tomto druhém jazyce - nepřirovná například pády conlangu k pádům původního jazyka.

Atributy

Typ	Název	Popis
CentralStorage	central	odkaz na objekt CentralStorage, pod který všechny ostatní objekty spadají.
string	baseForm	základní forma psaného slova
string	basePron	základní forma výslovnosti slova
string	baseRhyme	základní forma vzoru rýmu slova
string	translation	překlad slova do původního jazyka
string	definition	Obsáhlejší definice slova. Může být prázdná. Užitečná, pokud neexistuje žádný adekvátní překlad do původního jazyka.
PoS	partOfSpeech	Slovní druh, pod který slovo spadá.
WordClass	wClass	lexikální třída, pod kterou slovo spadá (např. rod)
List<WordForm>	forms	List slovních forem (utvořených automaticky

Konstruktor

Konstruktor třídy Word vyžaduje tyto argumenty:

Typ	Název	Funkce
string	form	nastavuje atribut baseForm
string	pronunciation	nastavuje atribut basePron
string	rhyme	nastavuje atribut baseRhyme
string	translation	nastavuje atribut translation
string	definition	nastavuje atribut definition
WordClass	wClass	nastavuje atribut wClass. Atribut partOfSpeech je nastaven na wClass.GetPoS() (popsáno níže)
CentralStorage	central	nastavuje atribut central

Konstruktor vytvoří slovní formy na základě existujících objektů Declension obsažených v atributu wClass.

Veřejné metody

void UpdateDeclension
arg: Declension *declension*

Najde svoji slovní formu, která odpovídá danému skloňování *declension*, a aktualizuje ji (tj. znovu derivuje její formu, výslovnost, a vzor rýmu od svého základu skrz modifikace dané v *declension*).

Tuto metodu volá WordClass, když je přidána nebo odebrána TransformUnit od nějaké Declension obsažené v této WordClass.

void NewDeclension
arg: Declension *declension*

Inicializuje novou slovní formu podle svého základu a modifikací *declension*, a přidá ji pod své slovní formy.

Tuto metodu volá WordClass, když je do ní přidána nová Declension.

void RemovedDeclension
arg: Declension *declension*

Najde slovní formu, která odpovídá danému skloňování *declension*, a odebere ji.

Tuto metodu volá WordClass, když je odebrána nějaká její Declension.

void Remove
žádné argumenty

Vymaže všechny své slovní formy.

Tuto metodu volá CentralStorage, když maže slovo.

void ChangeForm / ChangePronunciation / ChangeRhyme
arg: string *newBase*

změní svůj atribut *baseForm* / *basePron* / *baseRhyme*, a propaguje tuto změnu do všech svých slovních forem (WordForm).

Tyto tři metody volá CentralStorage, když je slovo upravováno.

void ChangeTranslation
arg: *newTrans*

změní svůj atribut *translation*.

void ChangeDefinition
arg: *newDef*

změní svůj atribut *definition*.

void ChangeClass

arg: WordClass *newClass*

Postará se o odebrání slova ze seznamů starého slovního druhu (PoS) a staré lexikální třídy (WordClass), změní své atributy *partOfSpeech* a *WordClass* na *newClass.GetPoS()* a *newClass*, a přidá se do jejich seznamů. Následně vymaže všechny své slovní formy, a vytvoří nové, korespondující s novou lexikální třídou.

Tuto metodu volá CentralStorage, když mění lexikální třídu slova.

bool CheckWordForm

arg: string *name*

Vrátí *true*, pokud existuje slovní forma, jejíž psaná forma (WordForm.form) se shoduje s argumentem *name*. Jinak vrátí *false*.

WordForm

Třída WordForm representuje slovní tvar utvořený automaticky podle slova a pravidel skloňování, či upravený uživatelem.

Atributy

Typ	Název	Popis
CentralStorage	central	odkaz na objekt CentralStorage, pod který všechny ostatní objekty spadají.
Word	parent	slovo, jehož jednu formu objekt representuje.
string	form	psaná forma slovní formy.
string	pronunciation	výslovnost slovní formy.
RhymeGroup	rhyme	skupina rýmů, do níž slovní forma zapadá.
Declension	declension	skloňovací pravidlo, podle nějž je slovní forma utvořena.

Konstruktor

Konstruktor třídy WordForm vyžaduje tyto argumenty:

Typ	Název	Funkce
Word	baseWord	nastavuje atribut parent.
Declension	rule	nastavuje atribut declension.
CentralStorage	central	nastavuje atribut central.

Atributy *form* a *pronunciation* jsou derivovány z atribuů baseForm a basePron od slova *parent* a atributů formTransform a pronTransform skloňovacího pravidla *declension*. Atribut *rhyme* je vygenerován obdobně, ale výsledek této morfologické derivace je předán jako argument metodě EvaluateRhyme, která přiřadí slovní formě nějaký objekt typu RhymeGroup (případně ho vytvoří), a tomuto objektu zařadí slovní formu do vlastního listu.

Veřejné metody

void Remove
žádné argumenty

Propaguje informaci o svém vymazání do RhymeGroup, pod kterou spadá.

void ChangeForm
arg: string *newBase*

Derivuje svou psanou formu tak, jako při spuštění konstruktoru, pomocí stringu *newBase* a skloňovacího pravidla *declension*.

void ChangePronunciation
arg: string *newBase*

Změní svou výslovnost obdobně jako v metodě ChangeForm.

void ChangeRhyme
arg: string *newBase*

Vygeneruje nový tvar rýmového vzoru obdobně jako v předchozích dvou metodách, poté tento string předá jako argument metodě EvaluateRhyme, stejně jako při běhu konstruktoru.

void EditForm
arg: string *newForm*

Změní svoji psanou formu přímo na daný string, bez jakékoli morfologické derivace. (Umožňuje nepravidelné tvarosloví).

void EditPronunciation
arg: string *newPron*

Obdobně jako v předchozí metodě změní svoji výslovnost.

void EditRhyme
arg: string *newRhyme*

Předá nový string metodě EvaluateRhyme, která slovní formu zařadí do nové rýmové skupiny.

void Update
žádné argumenty

Zavolá metody ChangeForm, ChangePronunciation, a ChangeRhyme, jako argumenty použije základní tvary rodičovského slova. Tato metoda je zavolána, když je nějaký z těchto atributů rodičovského slova změněn.

Declension

Třída Declension obsahuje informace potřebné ke tvorbě slovních forem. Každý objekt WordClass (popsány níže) má vlastní seznam Declension, které jsou automaticky vytvořeny podle slovního druhu (PoS), pod které WordClass spadá. To znamená, že pod jedním slovním druhem mají všechny lexikální třídy (WordClass) stejný počet Declensions se stejnými jmény, ale s různými pravidly pro tvorbu tvarosloví.

Atributy

Typ	Název	Popis
CentralStorage	central	odkaz na objekt CentralStorage, pod který všechny ostatní objekty spadají.
string	name	název skloňovacího pravidla
List<TransformUnit>	formTransform	seznam transformací psané formy
List<TransformUnit>	pronTransform	seznam transformací výslovnosti
List<TransformUnit>	rhymeTransform	seznam transformací rýmovacího vzoru
WordClass	parent	lexikální třída, pod kterou skloňovací pravidlo spadá.

Konstruktor

Konstruktor třídy Declension vyžaduje tyto argumenty:

Typ	Název	Funkce
string	name	nastavuje atribut name
WordClass	parent	nastavuje atribut parent
CentralStorage	central	nastavuje atribut central

Veřejné metody

void Rename

arg: string *newName*

nastaví atribut *name* na hodnotu obsaženou v *newName*.

AddFormTrans / AddPronTrans / AddRhymeTrans

arg: TransformUnit *transform* NEBO string *identifier*, string *regex*, string *replace*

Těchto šest metod přidá objekt struktury TransformUnit do svého seznamu formTransform, pronTransform, nebo rhymeTransform. Pokud je jako argument dán objekt samotný, přidá ho rovnou, jinak ho vytvoří z daných stringů.

Skrz rodičovskou WordClass a její metodu NotifyOfDeclensionChange (popsanou níže) zavolá u všech slov, které ovlivňuje, metodu UpdateDeclension, která automaticky opraví slovní formy, aby reflektovaly tuto změnu.

RemoveFormTrans / RemovePronTrans / RemoveRhymeTrans

arg: TransformUnit *transform* NEBO string *identifier*, string *regex*, string *replace*

Chová se identicky jako skupina metod k přidávání TransformUnit popsaná výše, jen TransformUnit z listu odebírá a poté propaguje informaci o změně stejně, jako ve výše uvedených metodách.

WordClass

Třída WordClass representuje lexikální třídu, jako například gramatický rod.

Atributy

Typ	Název	Popis
CentralStorage	central	odkaz na objekt CentralStorage, pod který všechny ostatní objekty spadají.
string	name	Název lexikální třídy.
PoS	parent	slovní druh, pod který lexikální třída spadá.
List<Declension>	declensions	Seznam skloňovacích pravidel aplikovaných na slova spadající pod tuto lexikální třídu.
List<Word>	words	Seznam slov spadajících pod tuto lexikální třídu.

Konstruktor

Konstruktor třídy WordClass vyžaduje tyto argumenty:

Typ	Název	Funkce
string	name	nastavuje atribut name
PoS	parent	nastavuje atribut parent
CentralStorage	central	nastavuje atribut central

Veřejné metody

void NotifyOfDeclensionChange

arg: Declension *declension*

U všech slov ze seznamu *words* zavolá metodu UpdateDeclension s argumentem *declension*, která aktualizuje slovní formy tohoto slova.

void AddWordToSelf

arg: Word *word*

Přidá slovo *word* do listu *words*.

void RemoveWordFromSelf

arg: Word *word*

Odebere slovo *word* z listu *words*.

void AddDeclension

arg: string *dName*

Přidá nový objekt typu Declension do svého listu *declensions*. Tento objekt zatím neobsahuje žádné objekty struktury TransformUnit - musí být přidány později.

U všech slov ze seznamu *words* zavolá metodu NewDeclension s nově vytvořeným objektem jako argument, čímž vytvoří novou slovní formu, zatím identickou základní podobě slova, dokud uživatel nepřidá jednotky TransformUnit.

Tuto metodu by měl volat pouze rodičovský PoS, aby nevznikla diskrepance ve skloňování mezi různými lexikálními třídami.

void RemoveDeclension

arg: string *dName*

Odebere objekt Declension ze seznamu *declensions* se jménem shodujícím se s *dName* a propaguje informaci všem slovům v seznamu *words*, kterým se odebere slovní forma korespondující s tímto odebraným skloňovacím pravidlem.

Tuto metodu by měl volat pouze rodičovský PoS, jelikož obsahuje seznam jmen skloňovacích pravidel, shodujících se mezi všemi jeho lexikálními třídami.

void RenameDeclension

arg: string *oldName*, string *newName*

Zavolá metodu Rename s argumentem *newName* u skloňovacího pravidla ze seznamu *declensions*, jehož jméno se shoduje s *oldName*.

Tuto metodu by měl volat pouze rodičovský PoS, aby nevznikla diskrepance ve skloňování mezi různými lexikálními třídami.

bool CheckDeclension

arg: string *name*

Vrátí *true*, pokud v seznamu *declensions* existuje skloňovací pravidlo s názvem shodujícím se s argumentem *name*. Jinak vrátí *false*.

void Rename

arg: string *newName*

Změní atribut *name* na hodnotu argumentu *newName*.

PoS

Třída PoS (zkratka anglického Part of Speech) representuje slovní druh, tedy např. podstatná jména, přídavná jména, slovesa, atd.

Pozor: každé slovo musí spadat pod jeden slovní druh a jednu lexikální třídu pod tímto slovním druhem – každý slovní druh tedy potřebuje ke správnému fungování alespoň jednu lexikální třídu, i když se lingvisticky na třídy nedělí. Proto se při tvorbě slovního druhu automaticky vytvoří lexikální třída s názvem identickým slovnímu druhu.

Atributy

Typ	Název	Popis
CentralStorage	central	odkaz na objekt CentralStorage, pod který všechny ostatní objekty spadají.
string	name	název slovního druhu.
List<WordClass>	wordClasses	seznam lexikálních tříd
List<Word>	words	seznam slov spadajících pod slovní druh
List<string>	declensions	seznam názvů skloňovacích pravidel*

*seznam *declensions* obsahuje pouze názvy místo objektů typu Declension, protože samotné tyto objekty obsahují až lexikální třídy (WordClass), aby mohly slova různých lexikálních tříd mít různá skloňovaní.

Konstruktor

Konstruktor třídy PoS vyžaduje tyto argumenty:

Typ	Název	Funkce
string	name	nastavuje atribut name
CentralStorage	central	nastavuje atribut central

Konstruktor vytvoří jednu lexikální třídu s názvem identickým názvu slovního druhu, protože slovní druh nemůže fungovat bez lexikální třídy. Pokud uživatel přidá vlastní lexikální třídy, může tuto třídu odstranit. Nelze však odstranit třídu, která je ve slovním druhu jediná.

Veřejné metody

void AddWordToSelf

arg: Word *word*

Přidá slovo *word* do seznamu *words*.

void AddWordToSelf

arg: Word *word*

Odebere slovo *word* ze seznamu *words*.

WordClass AddWordClass

arg: string *name*

Vytvoří nový objekt WordClass s názvem *name*, zavolá jeho metodu AddDeclension pro všechny prvky seznamu *declensions*, a přidá ho do svého seznamu wordClasses.

Vrátí nově vytvořenou WordClass.

bool CheckWordClass

arg: string *name*

Vrátí *true* pokud seznam *wordClasses* obsahuje lexikální třídu se jménem shodujícím se s argumentem *name*. Jinak vrátí *false*.

bool CheckDeclension
arg: string *name*

Vrátí *true* pokud seznam *declensions* obsahuje hodnotu argumentu *name*. Jinak vrátí *false*.

void RemoveWordClass
arg: WordClass *wClass*

Odebere lexikální třídu *wClass* ze seznamu *wordClasses*. Pokud je *wClass* poslední prvek tohoto listu, vyhodí výjimku InvalidOperationException.

void AddDeclension
arg: string *name*

Přidá *name* do seznamu *declensions* a u každé WordClass ze seznamu *wordClasses* zavolá AddDeclension s argumentem *name*.

Pokud již *declensions* jednou obsahuje prvek *name*, vyhodí výjimku ItemAlreadyExistsException.

void RemoveDeclension
arg: string *name*

Odebere *name* ze seznamu *declensions* a u každé WordClass ze seznamu *wordClasses* zavolá RemoveDeclension s argumentem *name*.

Pokud se prvek *name* v *declensions* nenachází, vyhodí ItemNotFoundException.

void RenameDeclension
arg: string *oldName*, string *newName*

Vyhodí výjimky za stejných podmínek jako předchozí dvě metody.

Odebere ze seznamu *declensions* prvek *oldName*, a přidá do něj prvek *newName*.

U všech WordClass objektů ze seznamu *wordClasses* zavolá RenameDeclension s argumenty *oldName* a *newName*.

void Rename
arg: string *newName*

Změní atribut *name* na hodnotu argumentu *newName*.

RhymeGroup

Třída RhymeGroup není viditelná uživateli, ale zprostředkovává funkci nacházení rýmů mezi slovními formami.

Atributy

Typ	Název	Popis
CentralStorage	central	odkaz na objekt CentralStorage, pod který všechny ostatní objekty spadají.
string	id	identifikátor, proti kterému se kontrolují slovní formy.
List<WordForm>	rhymes	seznam slovních forem, jejichž rýmovací vzor se shoduje s id.

Konstruktor

Konstruktor třídy RhymeGroup vyžaduje tyto argumenty:

Typ	Název	Funkce
string	id	nastavuje atribut id
CentralStorage	central	nastavuje atribut central

Veřejné metody

Insert

arg: WordForm *form*

Přidá slovní formu *form* do seznamu *rhymes.*

RemoveForm

arg: WordForm *form*

Odebere slovní formu *form* ze seznamu *rhymes*. Pokud je nyní *rhymes* prázdný, smaže se ze seznamu RhymeGroup objektů.

CentralStorage

CentralStorage je třída, která organizuje celý vnitřní běh programu. Vždy existuje jen jedna instance.

Atributy

Typ	Název	Popis
List<Word>	wordList	seznam všech slov
List<PoS>	PoSList	seznam všech slovních druhů
List<RhymeGroup>	RhymeGroupList	seznam všech rýmovacích skupin

Veřejné metody

RhymeGroup AssignRhymeGroup

arg: string *rhymeLiteral*, WordForm *from*

najde rýmovací skupinu, jejíž id se rovná argumentu *rhymeLiteral*. Pokud žádná raková skupina neexistuje, vytvoří ji. Do této skupiny vloží slovní formu *from*, a vrátí ji.

Tuto metodu volá slovní forma, když je generovaná, nebo když se jí změní rýmovací vzor.

void RemoveRhymeGroup

arg: RhymeGroup *group*

Odstraní rýmovací skupinu *group* ze seznamu *RhymeGroupList*.

Word AddWord

arg: Word *word* NEBO string *form*, string *pronunciation*, string *rhyme*, string *translation*, string *definition*, WordClass *wClass* NEBO string *form*, string *pronunciation*, string *rhyme*, string *translation*, WordClass *wClass*

Přidá slovo do seznamu *wordList*. Pokud jsou dány atributy slova místo objektu slova, vytvoří jej. Pokud není dán argument *definition*, v konstruktoru místo něj použije prázdný string.

Zavolá metodu AddWordToSelf na rodičovském slovním druhu (PoS) slova, a na jeho rodičovské lexikální třídě (WordClass). Slovo vrátí.

void RemoveWord

arg: Word *word*

Odstraní slovo *word* ze seznamu *wordList*, na rodičovském slovním druhu (PoS) a lexikální třídě (WordClass) slova zavolá metodu RemoveWordFromSelf se slovem *word* jako argumentem, a na slově *word* samotném zavolá metodu Remove.

void EditWord

arg: Word *word*, string *newForm*, string *newPron*, string *newRhyme*, string *newTrans*, string *newDef*, WordClass *newClass*

Na slově *word* zavolá příslušné metody na změnu všech atribut.

void EditWordForm / EditWordPronunciation / EditWordRhyme / EditWordTranslation / EditWordDefinition / EditWordClass

arg: Word *word* +: string *newForm* NEBO string *newPronunciation* NEBO string *newRhyme* NEBO string *newTranslation* NEBO string *newDefinition* NEBO WordClass *newClass*

Na slově *word* zavolá příslušnou metodu na změnu daného atributu.

Word? GetWord

arg: string *name*

Vrátí první nalezené slovo v listu *wordList*, jehož psaná forma se shoduje s argumentem *name*. Pokud takové slovo nenalezne, vrátí hodnotu *null*.

Word? GetWordByTrans
arg: string *name*

Vrátí první nalezené slovo v listu *wordList*, jehož překlad se shoduje s argumentem *name*. Pokud takové slovo nenalezne, vrátí hodnotu *null*.

bool CheckWord
arg: string *name*

Vrátí *true* pokud se v seznamu *wordList* nachází slovo, jehož psaná forma se shoduje s argumentem *name*. Jinak vrátí false.

bool CheckTrans
arg: string *name*

Vrátí *true* pokud se v seznamu *wordList* nachází slovo, jehož překlad se shoduje s argumentem *name*. Jinak vrátí false.

PoS AddPoS
arg: PoS *partOfSpeech* NEBO string *name*

Pokud je dáno jméno, vytvoří PoS pod tímto jménem. Daný PoS přidá do seznamu *PoSList* a vrátí ho.

void RemovePoS
arg: PoS *partOfSpeech*

Odebere *partOfSpeech* ze seznamu *PoSList*.

void RenamePoS
arg: PoS *partOfSpeech*, string *newName*,

Na *partOfSpeech* zavolá metodu Rename s argumentem *newName*, čímž ho přejmenuje.

bool CheckPartOfSpeech
arg: string *name*

Vrátí *true* pokud se v seznamu *PoSList* nachází slovní druh s názvem *name*. Jinak vrátí *false*.

WordClass CreateWordClass
arg: PoS *where*, string *name*

Ve slovním druhu *where* zavolá metodu AddWordClass s argumentem *name*, čímž vytvoří novou lexikální třídu.

void RemoveWordClass
arg: WordClass *which*

Na rodičovském PoS lexikální třídy *which* zavolá RemoveWordClass s argumentem *which*, čímž ji odstraní.

void RenameWordClass

arg: WordClass *which*, string *newName* NEBO string *PoSName*, string *name*

Na *which* zavolá Rename s argumentem *newName*, čímž ji přejmenuje.

void CreateDeclension

arg: PoS *where*, string *name*

Přidá název skloňovacího pravidla *name* do objektu *where* pomocí jeho metody AddDeclension, která vytvoří objekty typu Declension pro všechny lexikální třídy (WordClass) slovního druhu *where*. Pokud je místo objektu PoS předán název slovního druhu, objekt nejdříve vyhledá.

void RemoveDeclension

arg: PoS *where*, string *name*

Odebere název skloňovacího pravidla *name* ze seznamu ve slovním druhu *where* pomocí jeho metody RemoveDeclension, která zároveň odebere korespondující objekty Declension od lexikálních tříd pod tímto slovním druhem.

void RenameDeclension

arg: PoS *where*, string *name*, string *newName*

Pomocí metody PoS RenameDeclension přejmenuje skloňovací pravidlo *name* na *newName*, což zároveň aktualizuje názvy korespondujících objektů typu Declension v lexikálních třídách.

void AddDeclensionFormTransformUnit / AddDeclensionPronTransformUnit / AddDeclensionRhymeTransformUnit

arg: Declension *dec* +: TransformUnit *unit* NEBO string *id*, string *regex*, string replace

Přidá objekt struktury TransformUnit do seznamu objektu *dec* pomocí jeho metody AddFormTrans, AddPronTrans, respektive AddRhymeTrans.

void RemoveDeclensionFormTransformUnit / RemoveDeclensionPronTransformUnit / RemoveDeclensionRhymeTransformUnit

arg: Declension *dec* +: TransformUnit *unit* NEBO string *id*, string *regex*, string replace

Odebere objekt struktury TransformUnit ze seznamu objektu *dec* pomocí jeho metody RemoveFormTrans, RemovePronTrans, respektive RemoveRhymeTrans.

CommandCentre

CommandCentre je třída, která, podobně jako CentralStorage, má při běhu aplikace pouze jednu instanci. Zpracovává příkazy psané uživatelem, zařizuje odpovědi na příkazovou linku a zachycuje případy, které by mohly vést k nesprávnému běhu aplikace, zastaví je, a napíše uživateli že jsou neplatné. Zařizuje rozhraní mezi uživatelem a aplikací.

Atributy

Typ	Název	Popis
CentralStorage	central	odkaz na objekt CentralStorage, pod který všechny ostatní objekty spadají.
object?	focus	objekt, na který je CentralStorage zaměřen.
Declension?	focusedTrUParent	rodičovské skloňovací pravidlo TransformUnit, na kterou je CentralStorage zaměřen.
string?	focusedTrUSection	popisuje, zda je CentralStorage zaměřen na TransformUnit která ovlivňuje psanou formu, výslovnost, nebo rýmovací vzor. Může nabývat hodnot "f", "p", "r", nebo null.
Dictionary<string, Action<string[]>>	cmdFuncs	slovník, který přiřazuje funkce k jejich uživatelským názvům.

Když je CentralStorage zaměřen na nějaký objekt, umožňuje to některé příkazy, které by jinak nebylo možné spustit. Např. identifikace toho, které slovo chce uživatel vybrat, může být komplikovanější, pokud existuje více slov s identickou psanou formou. Proto se o tento výběr stará příkaz focus, a ostatní příkazy se slovem, místo aby vyžadovaly psanou formu slova, vyžadují, aby slovo bylo zaměřené.

Konstruktor

Konstruktor bere jeden argument CentralStorage, který použije k nastavení atributu central.

Poté nastaví klíče a hodnoty ve slovníku cmdFuncs.

Veřejné metody

bool WaitForCommand

arg: žádné argumenty

Počká, až uživatel napíše na příkazovou řádku příkaz. Ten pak předá metodě CallCommand.

Vrátí hodnotu vrácenou CallCommand. *True* znamená, že program má nadále běžet. *False* znamená, že program má skončit.

bool CallCommand

arg: string *line*

Rozdělí *line* na jednotlivá slova. První slovo použije jako příkaz, zbytek jako jeho argumenty.

Pokud se příkaz rovná stringu "exit", zeptá se uživatele, zda chce opravdu ukončit program. Pokud ano, vrátí *false*. Tento jeden příkaz je hard-coded, pro jednoduchou možnost vrátit hodnotu znamenající, zda má program skončit.

Dále, pokud *cmdFuncs* obsahuje klíč shodující se s příkazem, zavolá funkci uloženou jako hodnotu k tomuto klíči, a jako argumenty jí předá pole stringů utvořené z rozdělení zbytku *line*.

Pokud *cmdFuncs* příkaz jako klíč neobsahuje, informuje na příkazové řádce uživatele.

Vrátí *true*, jelikož se program dostal přes kontrolu, zda chce uživatel program vypnout.

Příkazy

Příkazy mohou vyžadovat různé argumenty podle toho, na který objekt je CommandCentre zaměřeno. Všechny argumenty jsou typu string, jelikož jsou podány uživatelem na příkazové řádce.

Příkazy jsou k nalezení níže, v uživatelské dokumentaci.

Hlavní smyčka programu

Hlavní smyčka je velmi jednoduchá.

Před spuštěním smyčky program připraví nový objekt CommandCentre, který si drží v proměnné *cc*, předá mu nové CentralStorage, a do proměnné *run* uloží hodnotu *true*.

Samotná smyčka je while smyčka, která se opakuje, dokud *run* obsahuje hodnotu *true*. Ve smyčce se na *cc* zavolá metoda WaitForCommand, jejíž výstup se uloží do proměnné *run*. Tato metoda vrací *true*, pokud má program pokračovat, tedy většinu času, a *false*, pokud uživatel potvrdí, že chce z aplikace odejít.

Po skončení smyčky aplikace skončí.

Uživatelská dokumentace

Ovládání aplikace

Aplikace se ovládá z příkazové řádky, pomocí příkazů. Jména příkazů nejsou case-sensitive. Některé příkazy potřebují argumenty, které se píší za příkazy, oddělené mezerou, nikoli čárkou či jinými oddělovači. Po dokončení jednoho příkazu stiskne uživatel klávesu Enter k jeho odeslání.

Některé příkazy využívají mechaniky "zaměření." Program si drží jeden (nebo žádný) objekt, na který je zaměřen, a tento objekt pak může sloužit jako náhrada argumentu u některých příkazů. Jiné příkazy vyžadují, aby byl určitý objekt zaměřen, jinak nemohou být spuštěny. Více detailů lze nalézt v popisu příkazu Focus.

Příkazy

Následuje seznam příkazů a jejich popis.

Focus

Bere 0, 1, nebo 2 argumenty. Následuje seznam různých možností, a jejich funkce (přesné hodnoty jsou psané rovně, proměnné hodnoty v kurzívě, pomlčka již není součástí argumentů).

Pozor, většina těchto podpříkazů vyžaduje, aby byl rodičovský objekt zaměřen (např. pro vybrání lexikální třídy musí uživatel nejdříve vybrat její rodičovský slovní druh).

- (bez argumentů) - vypíše na standardní výstup název objektu, na který je CC zaměřeno, nebo informuje uživatele, že není na nic zaměřeno.

- parent - zaměří se na rodičovský objekt nynějšího objektu (viz diagram na začátku dokumentace) nebo zruší zaměření, pokud by tímto rodičem bylo samotné CentralStorage.
- list - vypíše aktuálně zaměřený objekt, stejně jako focus bez argumentů.
- pos *název* - zaměří se na slovní druh, jehož název se shoduje s argumentem *název*.
- wc *název* - zaměří se na lexikální třídu pod aktuálně zaměřeným slovním druhem, jejíž název se shoduje s argumentem *název*.
- dc *název* - zaměří se na skloňovací pravidlo pod aktuálně vybranou lexikální třídou, jehož název se shoduje s argumentem *název*.
- w *název* - zaměří se na slovo, jehož psaná forma se shoduje s argumentem *název*. Pokud toto pravidlo splňuje více slov, program tyto slova vypíše s jejich ostatními atributy (jako je překlad, slovní druh, atd), očísluje, a dá uživateli vybrat pomocí čísla.
- nt *název* - zaměří se na slovo, jehož překlad se shoduje s argumentem *název*. Stejně jako u předchozího podpříkazu, pokud toto pravidlo splňuje více slov, program tyto slova vypíše s jejich ostatními atributy, očísluje, a dá uživateli vybrat pomocí čísla.
- wf *skloňování* - zaměří se na slovní formu pod aktuálně vybraným slovem, která koresponduje se skloňovacím pravidlem, jehož název se shoduje s argumentem *skloňování*.
- tr *typ* - Typ může být pouze jedno z písmen *f*, *p*, nebo *r*, representující formu, výslovnost, respektive rýmovací vzor. Vypíše dialog, kde očísluje všechny jednotky tohoto typu pod aktuálně vybraným skloňovacím pravidlem, a zaměří se na jednotku TransformUnit, jejíž číslo uživatel vybere. Na rozdíl od ostatních objektů se při vybírání jednotek TransformUnit využívá proměnných focusedTrUParent a focusedTrUSection.

Help

Bez argumentů vypíše seznam příkazů a základní informace o jejich používání.

S názvem příkazu jako argumentem vypíše detailnější popis používání daného příkazu.

Save

Bere 1 argument: název souboru k uložení.

Pokud je název neplatný, zruší ukládání.

Pokud soubor již existuje, zeptá se uživatele, chce-li ho přepsat.

Uloží slovník do souboru typu DAT do složky savefiles.

Load

Bere 1 argument: název souboru k načtení.

Pokud soubor neexistuje, zruší načítání.

Zeptá se uživatele, zda chce přepsat aktuálně načtený slovník slovníkem ze souboru.

Načte soubor.

LsFiles
Žádné argumenty

Vypíše soubory typu DAT ve složce savefiles.

NewFile
Žádné argumenty

Zeptá se uživatele, zda chce vymazat všechny informace v aktuálně načteném slovníku.

Přepíše načtený slovník prázdným slovníkem.

AddPoS
Bere 1 nebo více argumentů.

Pro každý argument vytvoří nový slovní druh s tímto argumentem jako název. Pokud se některé slovní druhy nezdaří vytvořit (např. slovní druhy s jejich názvy již existují), informuje o tom uživatele.

RmPoS
Pokud je CC zaměřeno na nějaký slovní druh, bere 0 nebo více argumentů.

Pokud je dáno 0 argumentů, odstraní slovní druh, na který je zaměřeno.

Pokud je dán alespoň 1 argument, chová se jakoby nebylo zaměřeno na slovní druh.

Pokud není zaměřeno na slovní druh, bere 1 nebo více argumentů.

Pro každý argument odstraní slovní druh s tímto argumentem jako názvem, pokud takový slovní druh nalezne. Opět informuje uživatele o všech neúspěších.

RnmPoS
Bere 2 argumenty (2 nebo 1 pokud je zaměřen nějaký slovní druh)

přejmenuje slovní druh s prvním argumentem jako názvem, na druhý argument.

Pokud je zaměřen slovní druh, může (ale nemusí) substituovat prvnímu argumentu.

LsPoS
Žádné argumenty

Vypíše názvy všech slovních druhů přítomných ve slovníku.

LsPoSWrds
Bere 1 argument (1 nebo 0 pokud je zaměřen slovní druh)

Vypíše všechna slova spadající pod jmenovaný slovní druh.

Pokud je zaměřený slovní druh, může substituovat místo argumentu.

LsWC
Žádné argumenty - potřeba zaměřeného slovního druhu

Vypíše názvy lexikálních tříd přítomných v zaměřeném slovním druhu.

AddWC
Bere 1 nebo více argumentů - potřeba zaměřeného slovního druhu

Pro každý argument přidá do zaměřeného slovního druhu lexikální třídu s tímto názvem, pokud je to možné. Uživatel je informován o všech neúspěších.

RmWC
Bere 1 nebo více argumentů pokud je zaměřený slovní druh, nebo žádné argumenty, pokud je zaměřená lexikální třída.

Pokud je zaměřen slovní druh, odebere z něj všechny lexikální třídy vyjmenované v argumentech, pokud jsou nalezeny.

Pokud je zaměřena lexikální třída, argumenty nejsou povoleny, a odstraněna může být pouze tato třída.

RnmWC
Bere 1 argument pokud je zaměřená lexikální třída, nebo 2 argumenty pokud je zaměřen slovní druh.

Pokud je zaměřená lexikální třída, přejmenuje ji na argument.

Pokud je zaměřen slovní druh, přejmenuje jeho lexikální třídu s názvem prvního argumentu na druhý argument (pokud takovou třídu najde).

LsDc
Žádné argumenty - potřeba zaměřeného slovního druhu nebo lexikální třídy

Vypíše skloňovací pravidla nacházející se v zaměřeném slovním druhu nebo lexikální třídě.

AddDc
Bere 1 nebo více argumentů - potřeba zaměřeného slovního druhu

pro každý argument přidá do slovního druhu skloňovací pravidlo s tímto názvem, pokud se tam již nenachází.

RmDc
Pokud je zaměřený slovní druh, bere 1 nebo více argumentů. Pokud je zaměřené skloňovací pravidlo, nebere žádné argumenty.

Odebere všechna vyjmenovaná skloňovací pravidla ze slovního druhu. Pokud je zaměřené skloňovací pravidlo, odebere se ze slovního druhu pouze toto pravidlo, což znamená, že bude odebráno ze všech lexikálních tříd tohoto slovního druhu.

RnmDc
Pokud je zaměřen slovní druh, bere 2 argumenty. Pokud je zaměřeno skloňovací pravidlo, bere 1 argument.

Přejmenuje skloňovací pravidlo s prvním argumentem za název na druhý argument. Pokud je zaměřeno skloňovací pravidlo, substituuje za první argument. Změna je opět provedena na celém slovním druhu, tedy na všech korespondujících skloňovacích pravidlech ve všech lexikálních třídách.

AddTU
Bere 4 argumenty - potřeba zaměřeného objektu Declension

Přidá jednotku TransformUnit k zaměřenému skloňovacímu pravidlu.

První argument smí být pouze jedno ze tří písmen - f, p, nebo r. Určuje, zda bude jednotka ovlivňovat psanou formu, výslovnost, respektive rýmovací vzor.

Následující tři argumenty nastavují atributy *identifier*, *regex*, a *replace* u nové jednotky TransformUnit.

LsTU
Bere 0 až 1 argument - potřeba zaměřeného objektu Declension.

Vypíše všechny jednotky TransformUnit zaměřeného skloňovacího pravidla.

Pokud je dán argument, musí být jedno z písmen f, p, nebo r. Potom vypíše pouze jednotky TransformUnit ovlivňující psanou formu, výslovnost, respektive rýmovací vzor.

RmTr
Žádné argumenty - potřeba zaměřené jednotky TransformUnit.

Smaže zaměřenou jednotku TransformUnit.

AddW
Bere 6 nebo více argumentů.

Přidá slovo do slovníku.

Argumenty jsou v tomto pořadí:

- Základní psaná forma
- Základní výslovnost
- Základní rýmovací vzor
- Překlad
- Slovní druh
- Lexikální třída

Pokud je příkaz zavolán s více argumenty, všechny argumenty po šestém budou konkatenovány a přidány jako definice.

RmW
Žádné argumenty - potřeba zaměřeného slova

Smaže zaměřené slovo.

EdW
Bere 3 argumenty - potřeba zaměřeného slova

Změní psanou formu, výslovnost, a rýmovací vzor slova.

První argument je nová psaná forma, druhý argument nová výslovnost, a třetí nový rýmovací vzor.

EdWTr
Bere 1 argument - potřeba zaměřeného slova

Změní překlad slova na daný argument.

EdWDef
Bere 1 nebo více argumentů - potřeba zaměřeného slova

Změní definici slova na string konkatenovaný ze všech daných argumentů.

EdWCls
Bere 1 argument - potřeba zaměřeného slova

Umístí slovo do jiné lexikální třídy v rámci jeho slovního druhu. Změnit slovní druh slova není možné.

Argument je jméno lexikální třídy, do které má být slovo umístěno.

LsW
Žádné argumenty

Vypíše všechna slova, včetně jejich výslovnosti, slovního druhu, lexikální třídy, překladu a definice.

Rhm
Žádné argumenty - potřeba zaměřené slovní formy (WordForm)

Vypíše slovní formy, které se rýmují se zaměřenou slovní formou, včetně jejich rodičovského slova, a skloňování, pod které spadají.

LsWfm
Žádné argumenty - potřeba zaměřeného slova

Vypíše slovní formy zaměřeného slova, spolu se skloňováním, pod které spadají.

EdWfm
Bere 3 argumenty - potřeba zaměřené slovní formy

Změní psanou formu, výslovnost a rýmovací vzor zaměřené slovní formy, čímž umožňuje nepravidelné skloňování.

Argumenty reprezentují psanou formu, výslovnost, a rýmovací vzor v tomto pořadí.