Dokumentace úlohy XQR: XML Query v Pythonu do IPP 2010/2011
Jméno a příjmení: Martin Knapovský
Login: xknapo02

Toto je dokumentace popisující návrh a implementaci programu XML Query v jazyku python do předmětu „Principy programovacích jazyků a OOP“, který dle zadaného dotazu provádí vyhledání požadovaného elementu v XML souboru.

Parametry programu a jejich význam je dostupný přímo v programu zadáním parametru `--help`, nebo spuštěním programu bez parametrů.

Obecné informace o implementaci

Program se skládá z několika logických celků, jimiž jsou lexikální analyzátor, syntaktický analyzátor, třída obsahující handlery, které jsou použity pro vyhledávání v XML souboru a samotný program, který těchto celků využívá.

Lexikální analyzátor

Lexikální analyzátor je implementací konečného automatu, který je v programu zastoupen funkcemi `get_token()` a `is_keyword()`. Funkce `get_token()` čte zadaný dotaz (funkce je bez parametrů – čte dotaz z globální úrovně) po znacích a rozhoduje tak o typu tokenu, kterými jsou `ALPHA` - token, který obsahuje identifikátor, nebo klíčové slovo, token `NUMBER` pro celé číslo a token reprezentující další znaky dotazovacího jazyka (`( ) < > = . "`). O tom, zda je token `ALPHA` klíčovým slovem, či nikoliv rozhoduje funkce `is_keyword()`, která porovnává identifikátor se známými klíčovými slovy a vrací výsledek tohoto porovnání. Porovnávání lze vypnout pomocí změny hodnoty `keyword_checking` v souboru `xqr.py`.

Syntaktický analyzátor

Je implementován na základě bezkontextové gramatiky uvedené v zadání projektu funkcemi `parse_query()`, `parse_condition()`, `parse_limit()`, `parse_from()`, `parse_order()` a pro svou práci využívá lexikálního analyzátoru. Tyto funkce kontrolují správnost dotazu a převádí ho do vnitřní reprezentace, která je použita pro vyhledávání v XML souboru.

Vyhledávání v XML souboru

Elementy jsou vyhledávány na základě zadaného dotazu pomocí externího modulu `xml.sax`, pro který jsou nadefinovány handlery, které obsluhují signalizaci o počátku souboru, počátku elementu, datech elementu, konci elementu a konci souboru. Těmito handlery jsou:

`startDocument( self )`

Na základě parametrů programu nastavuje hlavičku výsledku vyhledávání. Pokud není zadaný parametr `-n`, pak vkládá řetězec `<?xml version="1.0" encoding="utf-8"?>`, a pokud byl zadán parametr `--root=%s`, pak přidává počateční element, který obaluje výsledek vyhledávání.

`startElement( self, data, attrs )`

Při nalezení počátku jakéhokoliv elementu inkrementuje proměnnou `self._depth`, která uchovává hloubku zanoření elementu a testuje, zda současný element vyhovuje klauzuli FROM dotazu. Při shodě nastaví informaci o tom, že se nacházíme uvnitř prohledávaného elementu a uloží jeho název a zanoření.

Pokud se již uvnitř prohledávaného elementu nacházíme, pak hledáme element vyhovující klauzuli SELECT dotazu. Jakmile jej nalezneme, uložíme jeho hloubku, nastavíme informaci o tom, že byl element vybrán, a pokud navíc dotaz obsahuje podmínku klauzule WHERE, pak nastavíme informaci o tom, že se nacházíme uvnitř podmínky.

Pokud byl prvek vybrán, pak jej přidáváme do dočasného výsledku vyhledávání, a pokud se navíc nacházím uvnitř podmínky, pak testuji, zda testovaný element klauzule WHERE neodpovídá současnému elementu a

pokud ano, pak provádím vyhodnocení podmínky pomocí funkce `_check_condition( self, name, attrs )` a na základě jejího výsledku nastavuji informaci o tom, že podmínka byla splněna.

`characters( self , data )`

Tento handler se volá, pokud jsou v dokumentu nalezena data elementu. Je-li nastavena informace o tom, že byl prvek vybrán, pak data přidáme do dočasného výsledku dotazu a pokud mají být na základě podmínky klauzule WHERE testována data elementu, v němž se nacházíme, pak tato data testujeme pomocí funkce `_check_condition( self, name, attrs )` a nastavujeme informaci o tom, zda byla podmínka splněna či nikoliv.

`endElement( self, name )`

Je volán při nalezení ukončujícího elementu, snižuje se hloubka zanoření.

Je-li název elementu shodný s názvem elementu v klauzuli FROM, nacházíme se současně uvnitř zdrojového elementu a tento ukončující element je ve stejné hloubce jako jeho počáteční element, pak jsme vystoupili z prohledávaného elementu a tuto informaci uchováváme.

Pokud jsme uvnitř prohledávaného elementu a prvek byl vybrán pomocí klauzule SELECT, pak do řetězce s dočasným výběrem přidáváme ukončující element prvku.

Pokud jsem uvnitř prohledávaného elementu a současný ukončující element je element ukončující výběr, pak pokud nebyla zadána podmínka, přidávám dočasný výběr do výběru výsledného, avšak je-li podmínka zadána, pak otestuji, zda v průběhu výběru byla splněna a teprve potom na základě tohoto výsledku přidám dočasný výběr do výběru výsledného.

`endDocument( self )`

Při zadání klauzule LIMIT ořízne výsledek vyhledávání a při zadání parametru `--root=%s` ukončí výsledek výběru zadaným elementem.

Testování podmínek je prováděno pomocí funkce `_check_condition( self, name, attrs )`, která nejprve zjistí, zda porovnává data elementu, nebo jeho atribut a následně zjišťuje podle typu literálu zadaného v podmínce, zda se jedná o porovnání číselné, nebo porovnání řetězců a převádí data elementu, či hodnotu atributu na odpovídající typy. Samotné porovnání je pak prováděno pomocí operátorů `in`, `>`, `<` a == jazyka python.

Pozn.: Pro další informace je zdrojový kód programu podrobně komentovaný.