Implemenčná dokumentácia k druhej do IPP 2021/2022
Meno a priezvisko: Igor Hanus
Login: xhanus19

Interpreter

interpret.py

Tvorí hlavnú časť interpretu, z ktorej sa spúšťajú otatné skripty potrebne pre správnu funkčnosť interpretu. Vo funkcii `proces_instruction` iteruje cez list získaných inštrukcii, ktoré su načítané za pomoci knižnice `ElementTree` vo funkciách `load_xml` a `check_header` kde sa kontroluje správnosť načítaného XML. Následne sú volané jednotlivé funkcie zo skriptu `operations.py`. `process_instruction` taktiež aktualizuje zoznamy pre naveštia a indexy operacii v zozname pri skokoch. Argumenty sa spracovávajú pomocou funkcie `parse_arguments` a za správne zoradenie jednotlivých operácii funkcia `sort_instructions`.

operations.py

V tejto časti celého interpretera sa realizujú jednotlivé operácie volané z `interpret.py`. Kazdá operácia ma specifickú funkciu vo forme "Názov operácie"`_op`. V niektorých prípadoch su viaceré operácie zpracovávané v jednej funkcii. Ďalej sa tu používajú pomocné funkcie ako napríklad `replace_escape`, ktorá dostane na vstup reťazec a kontroluje či sa v ňom nenachádza špeciálny znak reprezentujúci ascii hodnotu. V prípade, že nejaký nájde, uloží ho do zoznamu a násldne cez daný reťazec iteruje a nahradzuje jednotlive špeciálné znaky, ich ascii podobou.

FrameHandle.py

Tento skript v sebe obsahuje triedu, ktorá spravuje jednotlivé rámce a operácie nad nimi. Globálny rámec je reprezentovaný ako "slovník", ktorý ma v sebe ulozeny nazov premennej a v dátovom type `tuple` jej typ a hodnotu. Podobým spôsobom je spracovavaý aj dočasný rámec. S týmito hodnotami potom pracujú jednotlivé metódy v danej triede, ktoré riešia vkládanie a vyberanie hodnôt zo zásobniku, získavanie framov podľa ich typov, získavanie premenných a nastavovanie ich hodnôt alebo typov.

Test frame

test.php

Hlavný skript testujúci funkcnosť parseru a interpreteru. Na základe argumentu moze prechádzať testovacie súbory nerekurzívne pomocou `DirectoryIterator` alebo rekurzívne pomocou `RecursiveDirectoryIterator`. Následne z iterátora získa jednotlivé cesty k testovacím suborom a pomocou funkcie `checkAndReplaceMissingFiles` vytvorí chýbajúce testovacie súbory. Následne sa podľa argumentov rozhoduje, ktorý test bude spustený. Test pre parser (`ParserTest`) spustí skript na parser a následne skontorluje vystupné XML pomocou `jexaxml`. Test pre interpret (`interpretTest`) funguje na podobnom princípe ale na porovnávanie používa príkaz `diff`. Tretí test testuje ako interpreter tak aj parser. Teardown testu volá metódy na generovanie HTML zo skriptu `HtmlPrinter.php`.

HtmlPrinter.php

Veľmi jednoduchý skript obsahujúci len HTML kód pre výpis testov. Enum tableType slúži na uľahčenie práce s jednotlivými HTML tabuľkami. Na začiatku testu sa vytvárajú hlavičky tabuľiek testovaných časti a hlavička celého HTML dokumentu. Počas každého vykonaného testu sa pridá nový riadok do tabuľky s potrebnými parametrami, ktoré sa majú zobraziť. Ku koncu testu sa ukončújú jednotlivé tabuľky a taktiež celý HTML dokument. HTML DOkument obsahuje jeden skript, ktorý mení farbu pozadia bunky tabuľky podľa toho či test prešiel úspešne, alebo naopak, neúspešne.