Dokumentace úlohy CHA: C Header Analysis v PHP 5 do IPP 2016/2017
Jméno a příjmení: Jakub Svoboda
Login: xsvobo0z

Zpracování přepínačů z příkazové řádky

Prvním krokem programu je zpracovat případné parametry zadané uživatelem. Do pole `$longops` je proto vložen seznam všech povolených argumentů a s ním je následně volána funkce `getopt` na vyhodnocení argumentů. Parametry jsou následně překontrolovány, například parametr --help nelze použít v kombinaci s jiným parametrem (neplatné argumenty způsobí ukončení programu s návratovým kódem 1), v případě jeho použití se vypíše nápověda a program se ukončí. Pokud jsou parametry validní, je vytvořen nový XML dokument a z parametru --input je nalezen patřičný vstupní soubor nebo adresář se soubory. Ty jsou následně otevřeny pro čtení a jejich obsah je nakopírován do proměnné typu string. V případě že se nepodaří ze souboru číst, je program ukončen s návratovým kódem 2 a je vypsána chybová zpráva na `STDERR`.

Zpracování funkcí

Z textu je za potřebí odstranit nepotřebné části a zbylý dokument nakonec rozdělit na jméno funkce, návratovou hodnotu a parametry. Toho bylo docíleno využitím regulárních výrazů pomocí funkcí `preg_replace()` a `preg_match()`. Prvním krokem je odstranění jednořádkových komentářů a maker. Následně jsou odstraněny symboly nového řádku (pro Windows i Linux) a nahrazeny mezerou. Následovalo odstranění kódu v složených závorkách. Hlavičkové soubory mohou teoreticky obsahovat části kódu, stejně tak u argumentu --input může být zadán soubor s koncovkou .c (zadání to nevylučuje). Protože funkce, která obsahuje kód v závorkách není ukončena středníkem, je místo složených závorek vložen znak zavináče, který se v syntaxi jazyka C nevyskytuje a slouží jako značka pro nalezení konce prototypu funkce. Podobně jsou z textu odstraněny části popisující definice datových typů, struktur, blokových komentářů a výčtů (enum). Výsledný text obsahuje pouze prototypy funkcí.

Funkce jsou z textu rozděleny oddělovači, středníkem nebo zavináčem a vloženy do pole `$funcArray`. Na každé položce pole je provedeno vyhledání návratového typu. Je provedena kontrola parametru --no-inline a pokud je funkce inline, je případně přeskočena. Pro každou funkci je vytvořen nový element v XML dokumentu a jsou u něj nastaveny atributy `file` pro soubor, ze kterého funkce pochází, `name` pro jméno funkce (v případě argumentu --no-duplicates je funkce případně vyřazena), `varargs` je nastaveno na hodnotu „yes“ v případě nalezení sekvence znaků “…“. Jména funkcí a návratové typy jsou z textu odstraněny, stejně tak závorky argumentů. Výsledný řetězec je rozdělen oddělovačem čárka na jednotlivé argumenty.

Cyklus zpracovávající argumenty v poli `$args` prohledává řetězce a vyřazuje argumenty, které jsou pouze void, nebo značí proměnlivý počet argumentů. Pro každý validní argument je vytvořen nový element, kterému je přiřazeno číslo `$number` značící pořadí pro danou funkci a datový typ. Pokud by specifikován parametr --max-param a počet parametrů funkce je vyšší než zadané číslo, je funkce dodatečně z XML souboru odebrána.

Formátování XML souboru

Po zpracování všech argumentů a vytvoření jejich elementů je pomocí funkci `saveXML()` XML soubor převeden do řetězce a zpracován funkcí `format()`. Ta ověří, zda byl zadán parametr --pretty-xml a s jakou hodnou. Pokud byl parametr zadán, ale hodnota nebyla specifikována, nastaví se na čtyři mezery. Řádky s funkcemi jsou poté odsazeny o `n` mezer a řádky s elementy argumentů o `2n` mezer.

Po dokončení formátování je ověřen argument --output, pokud byl zadán, je výstup uložen do nového souboru. Pokud výstupní soubor nebyl specifikován, vypíše se výstup na standardní výstup. V případě, že výstupní

soubor již existuje, je bez dotazování přepsán souborem novým. Pokud nastane problém s právy na zápis, je program ukončen s návratovým kódem 3 a je vypsána chybová zpráva na `STDERR`.