Dokumentace úlohy CHA: C Header Analysis v PHP 5 v PHP5 do IPP 2015/2016
Jméno a příjmení: Marie Faltusová
Login: xfaltu01

Dokumentace k projektu, varianta CHA

1 Úvod

Účelem tohoto projektu je vytvořit skript, který načítá soubory s příponou `.h` a analyzuje jejich obsah. Cílem analýzy jsou deklarace, popř. definice funkcí, které skript zpracoval a vypsal ve formátu XML. Pro tento účel skript využívá funkce a nástroje pro zpracování parametrů příkazové řádky, procházení adresářů, vytváření XML souboru a regulárních výrazů. Využité metody jsou popsány níže.

2 Moduly skriptu

Celý skript je rozdělen na tři logické moduly – zpracování parametrů, funkce skriptu a část vytvářející samotný XML soubor. Každý z nich zpracovává určitou část řešení a jsou popsány v následujících podkapitolách.

2.1. Zpracování parametrů

Skript při svém spuštění akceptuje argumenty zadané do příkazové řádky. Jejich popis a použití je popsáno v nápovědě skriptu, která se spouští parametrem `--help`. Skript si zadané argumenty načte pomocí funkce `getopt` a poté je zpracovává. Zdali byl argument zadán, a případně s jakou hodnotou, zaznamenávají přepínače, které se následně použijí v kódu jako nositelé informací typu `integer` nebo `boolean`.

Pokud nastane chyba, skript vrací chybové hlášení na standardní chybový výstup a příslušnou návratovou hodnotu. Jsou to zejména špatné kombinace argumentů (argument `--help` nesmí být v kombinaci s jiným argumentem), špatné hodnoty některých parametrů (hodnota není číslo, popř. je menší než 0) nebo chybu při otevírání nebo vytváření vstupního a výstupního souboru.

2.2. Sekce funkcí

V rámci zpřehlednění kódu a zjednodušení některých konstrukcí je vytvořeno několik funkcí. První z nich je funkce `file_analysis`. Funkce slouží k načtení a zpracování nalezeného hlavičkového souboru. Jako parametr přijímá nalezenou vstupní cestu k souboru, který následně pomocí regulárních výrazů zpracovává. Jde v prvním kroku o vymazání struktur vynechaných z analýzy: komentáře, makra a řetězce. Následně je použit regulární výraz, který obsahuje vzor pro kompletní definici či deklaraci funkce. Všechny nalezené položky ukládá do pole `functions`, jež je zároveň návratovou hodnotou této funkce. Výjimkou je případ, kdy je zadaný soubor prázdný nebo neobsahuje žádné funkce – pak je návratová hodnota 0.

Další významnou funkcí je `writeXML`. Tato funkce vytváří hlavní část výstupního souboru ve formátu XML – element `function` a `param`. Jako parametry přijímá vytvořený paměťový buffer pro zápis XML formátu, pole funkcí a přepínače argumentů skriptu. Postupně prochází pole nalezených funkcí z daného souboru a pomocí regulárních výrazů je dělí na typ funkce, jméno a parametry. Funkce také provádí kontrolu přepínačů a podle jejich hodnoty upravuje výsledný výpis. Aby bylo zpracování vícenásobných typů parametrů přesnější, je zahrnuto porovnávání rezervovaných slov jazyka C.

2.3. Vytváření XML souboru

Základem této části je rozdělení zápisu podle toho, zdali je argument skriptu vstupní adresa `--input` zadána jako cesta k adresáři nebo konkrétnímu souboru. K prohledávání adresáře a jeho podadresářů je využit nástroj `RecursiveIterator`, který rekurzivně prochází veškeré podsložky zadané cesty a v nich vyhledává soubory s příponou `.h`. Pro každý nalezený soubor se spustí analýza funkcí `file_analysis` a následně zápis nalezených funkcí pomocí funkce `writeXML`. V případě, že je zadána cesta ke konkrétnímu souboru, spouští se analýza pouze jednou a následně probíhá zápis nalezených funkcí.

K vytvoření samotného XML souboru je použit nástroj `XMLWriter`. Pomocí tohoto nástroje skript nejdříve provede otevření paměťového bufferu, do kterého následně probíhá zápis vytvářených elementů. Kořenový je element `functions`. Do něj se následně zanořují elementy `function`, obsahující detailní informace o nalezené funkci, a v něm jsou zanořeny elementy `param`, které jsou seznamem parametrů, které daná funkce

přijímá. Všechny tyto prvky jsou vytvořeny funkcí `startElement`. Po skončení zápisu je otevřen výstupní soubor nebo standardní výstup a zapsaná data v paměti jsou na něj vypsána.

3 Závěr

Skript stojí na funkčnosti regulárních výrazů, které získávají veškerá data z nalezených souborů. Vestavěné funkce `preg_match`, `preg_match_all` a `preg_replace` jsou využity pro zpracování textových dat. Bylo také využito několik dalších nástrojů jazyka PHP - pro práci s adresáři, XML a parametry.

V rámci projektu jsme řešila několik sporných situací. Zejména šlo o zpracování funkce, která nemá žádné parametry. Tyto funkce podle referenčních testů mají dva možné zápisy: typ f (); anebo typ f (void);. Osobně bych druhou variantu zápisu považovala spíše za funkci s právě jedním parametrem bez zadaného jména. Ve skriptu jsem ale tento problém zpracovala podle dostupného referenčního řešení – tedy funkce bez parametrů.

Dále jsem narazila na vlastnost XMLWriteru, kdy v případě, že element `function` nemá žádné parametry, stane se z něj tag nepárový – tedy je ukončen zpětným lomítkem hned v prvním tagu. Nepodařilo se mi tento výpis nijak odstranit a vytvořit párový tag, avšak z logiky zápisu a vlastností těchto značkových jazyků věřím, že se jedná o zápisy ekvivalentní.

Problém jsme také řešila v případě, že není zadán parametr --pretty-xml. XMLWriter automaticky hlavičku odsazuje a vytváří za ní nový řádek. I v případě nastavení hodnot pro odsazení se nevztahovaly nikdy na hlavičku a element functions začínal na novém řádku. Jediný řešením bylo odstranit znak nového řádku dodatečně, před výpisem. I přes tato neelegantní řešení je skript názornou ukázkou přirozené a snadné práce se soubory a řetězci v jazyku PHP.