Dokumentace úlohy CLS: C++ classes v PHP do IPP 2016/2017
Jméno a příjmení: Lukáš Drahník
Login: xdrahn00

Dokumentace úlohy CLS: C++ classes v PHP do IPP 2016/2017
Jméno a příjmení: Lukáš Drahník
Login: xdrahn00

1. Úvod do projektu

1.1. Obsah repozitáře: Repozitář obsahuje zadání, které nemusí již být aktuální (mohly být provedeny menší úpravy), zdrojový kód, testy (některé testy byly upravené a některé zcela nové přidané, oficiálně distribuované testy byly dosti chybné a neměly potřebnou kvalitu pro úspěšné vypracování projektu), dokumentaci , soubor *rozsireni* s uvedenými zapnutými rozšířeními a script pro vytvoření archivu potřebného k odevzdání do informačního systému na FIT v Brně - WIS.

1.1. Požadavky na program: Projekt vyžaduje k běhu PHP verzi 5.6. Kompatibilita s jinými verzemi nebyla vyžadována a tudíž není zajištěna.

1.2. Readme: Obsahuje základní příkazy včetně potřebných příkazů pro zprovoznění repozitáře do spustitelného stavu.

1.3. Vývoj projektu: Vývoj projektu byl provedený postupných navrhováním a implementováním konkrétních modulů dle zadání, které bylo nutné testovat odděleně. K vývoji byl použit distribuuovaný systém správy verzí, Git.

1. Rozdělení na moduly:

1.3. CLS argument parser: Modul starající se o načítání vstupních parametrů pomocí metody *getopt*, parametry jsou v instanci uloženy po spuštění metody *run* a pomocí *public* metod *isOptionSet*, *getOptionValue* je práce s nimi usnadněna. Na modul navazuje zpracování vstupních tříd pomocí CPP parseru, který sám o sobě potřebuje znát mód například i kvůli optimalizacím. Dobrým příkladem je hledání konfliktů, kdy se kupříkladu nakládání s nimi bez zapnutého option *--conflicts* ignoruje.

1.1 CPP struktury a CPP parser: Modul řeší zpracování C++ kódu ze vstupu a jeho následné převedení na struktury. Výstupem modulu jsou C++ objekty například typu třída, metoda, atributy u metody a další, samozřejmě se berou v potaz i jejich návaznosti. V rámci modulu se také dohlíží na patřičnou sémantickou a syntaktickou správnost zápisu programu v jazyce C++ a vyvozování patřičních výjimek a návratových hodnot.

1.2. CLS parser: Základním kamenem modulu je používání *CPP parseru* a tedy navázání a práce s objekty z jazyka C++. V modulu je několikrát použitý rekurzivní sestup, při výpisu na patřičný výstup tak i při generování struktury z objektů typu *XMLElement* o kterém bude více zmíněno níže.

1.3. XML: Pro zápis do souborů byl použit objekt *XMLWritter,* ke kterému byl doprogramován objekt *XMLElement* zajišťující jasné předávání potřebných dat k výpisu v *XML* mezi metodami bez ztráty informací. Tento objekt je sám o sobě velice prostý, ale nezbytně nutný. Zajišťuje například název a párovost elementu, instance potomků a samozřejmě potřebné zachování struktury, tedy pole instancí pod elementů – instance taktéž *XMLElementů*.

2. Propojení modulů:

2.1 Composer: Moduly jsou navržené jako jednotlivé celky. Tato struktura byla zvolena vzhledem k vývoji a možnosti otestovat již hotové moduly. Provázanost kupříkladu rozdělení na jmené bloky řeší balíčkovací systém pro php, Composer. Po stažení projektu je nutné provést příkaz *composer dump-autoload*, který vytvoří linky na soubory specifikované pomocí složek v souboru *composer.json a* vytvoří složku *vendor* uvedenou v *.gitignore*, kde se nachází i veškeré soubory potřebné pro autoload. Ve výsledku to tedy znamená vyhnutí se volání *require, require_once.*

3. Otestování modulů:

Dokumentace úlohy CLS: C++ classes v PHP do IPP 2016/2017
Jméno a příjmení: Lukáš Drahník
Login: xdrahn00

3.1. CLS supplementary tests: Jak již bylo zmíněno, byly použity testy distribuované společně se zadáním rozšířené o obtížnější testy napsaného programu v jazyce C++.
3.2. Is it ok: Skript testující správnost obsahu vygenerovaného archivu k odevzdání.