# Skript parse.php

Skript parse.php je navržen s částečným využitím konečného automatu, který jsem použil v projektu IFJ v minulém semestru.  
Skript nejprve rozloží vstup na jednotlivé řádky, kde odstraní prázdné řádky a komentáře, aby mohl pracovat se samotným kódem. Poté jednotlivé řádky na rozdělí na samostatná slova, která jsou předána nedokonalému konečnému automatu, který podle názvu instrukce rozpozná a validuje jednotlivé parametry a jejich počet.  
Každý argument jednotlivé instrukce je postupně validován svou vlastní funkcí (funkce jsou okomentované a lze si je prohlédnout v kódu). Validace je řešena buď regex řetězcem nebo rozkládáním řetězce a postupnou verifikací všech možností.  
Vše je nakonec zapsáno pomocí XMLWriteru a pomocí echo posláno do STDOUT.

## Funkce

* argumentsValidation($argv) – Rozhodl jsem se kvůli jednoduchosti vytvořit vstupních parametrů vytvořit si vlastní funkci pro zpracování parametrů a nepoužít getOpt. Funkce je velice jednoduchá a rozpozná, zda byl rozpoznán parametr --help či nebyl zadán žádný. V opačném případě vyhlásí chybu.
* commentsTrim($line) – Odstraní všechny komentáře, které jsou zapsané v kódu.
* headerValidation($line, $header) –Validuje headeru kódu a značí, pokud už byl nalezen
* generateFile($xml) – Generuje XML soubor na STDOUT. Je zde zakomentovaný způsob i zápisu přímo do souboru.
* checkTypes($argumets) – Pomocí regexů zjistí, zda typy jednotlivých symbolů a proměnných jsou správné a zda odpovídají jednotlivé názvy typů.
* varValidator($xml, $split) – Ověřuje přítomnost „@” a zda jména odpovídají regexu.
* symbValidator($xml, $split) – Ověřuje přítomnost „@” a zda jména odpovídají regexu, popřípadě správnost názvu paměťových modelů.
* labelVerify($xml, $split) – Ověření názvu návěští pomocí regexu.

## Použité knihovny a třídy

V této části projektu jsem nepoužil žádnou externí třídu. Hlavní třídu, kterou jsem v projektu použil byl XMLWriter, který využívám na vytvoření XML struktury.