Dokumentace úlohy JSN: JSON2XML v PHP 5 do IPP 2016/2017
Jméno a příjmení: Jakub Frýz
Login: xfryzj01

Úvod

Jednotlivé sekce dokumentace se věnují obsahu souboru, který je v jejím názvu. Při práci na tomto projektu jsem si nedefinoval žádné objektové třídy, neboť jsem uvážil, že pro tento projekt jich není třeba.

Všechny mnou definované funkce:

- `print_help()`
- `process_arguments()`
- `JSON2XML()`
- `process_main()`
- `process_array()`
- `process_other()`
- `validate_name()`
- `simple_array_test()`

Soubory

1 jsn.php

Srdce celého skriptu, neobsahuje definici žádné funkce. Vše pracuje na globální úrovni v tomto pořadí:

1. Zkontroluje se, zda-li byl zadán pouze argument `--help`. Pokud je zadáno více argumentů, skript se ukončí s návratovou hodnotou `1`, jinak se pomocí funkce `print_help()` vypíše nápověda na standardní výstup.
2. V případě, že byly zadány jiné argumenty než `--help`, nadefinují se proměnné `$config`, `$jsn` a `$xml`, které se budou následně používat v některých funkcích, a naimportují se ostatní soubory obsahující funkce.
3. Následně se zpracují zadané argumenty pomocí definované funkce `process_arguments()` do proměnné `$config`, proběhne-li vše jak má.
4. Nyní jsou už argumenty zpracované, tak se naimportuje vstupní JSON. Ten se zadává pomocí argumentu `--output=filename` nebo ze standardního vstupu.
5. Soubor je naimportovaný, lze ho dekódovat. K tomu se použije funkce `json_decode()`. Ta navrátí zpracované pole do proměnné `$jsn`, pokud vše proběhlo v pořádku, jinak `NULL`. `NULL` vratí i pokud zpracovávaný soubor obsahuje pouze `{}` nebo `[]`, což se musí též kontrolovat.
6. Dekódovaný JSON se následně převede na XML. K tomu slouží funkce `JSON2XML()`.
7. Jediné, co zbývá je export. Ten je ovlivněn argumentem `--output=filename`. Je-li zadán, vytvořené XML se uloží do souboru (zde se kontroluje i to, zda-li se soubor vůbec uložil), jinak na standardní výstup.

2 help.php

Obsahuje pouze funkci `print_help()`, jejíž úkol není nic jiného než vypsat nápovědu na standardní výstup.

3 fce.php

Obsahuje funkce `validate_name()` a `simple_array_test()`:

`validate_name($value, $int)` Funkce zkontroluje, zda-li předaný string `$value` je validním názvem pro element XML. K validaci se využije regexu a funkcí `preg_match()` a `preg_replace()`.

`simple_array_test(array $array)` Funkce zkontroluje, zda-li je zadané pole `$array` indexované nebo asociativní (jestli je to objekt nebo jenom pole).

4 arg.php

Obsahuje funkci na zpracování argumentů `process_arguments($arg1)`.

Ta projde pole argumentů `$argv`, zjistí zda-li obsahují hodnotu, zkontroluje jestli vůbec argument má obsahovat hodnotu, popřípadě ukončí skript s chybou. To stejné jsou-li některé argumenty zadané víckrát.

Následně ještě zkontroluje některé podmínky, které musí argumenty splňovat (např. pokud je zadán argument `--start=n`, jestli je zadán i argument `-t` či `--index-items`) a nastaví výchozí hodnoty, pokud nebyly zadané argumentem (např. `--array-name=array-element` nebo `--item-name=item-element`).

5 xml.php

Obsahuje funkce `JSON2XML()`, `process_main()`, `process_array()` a `process_other()`.

`JSON2XML($int)` V této funkci se inicializuje `DOMDocument` do proměnné `$xml`, obalí se kořenovým elementem, pokud byl zadán a určí se, zda-li je na nejvyšší úrovni objekt (asociativní pole) nebo pole (indexované pole). Poté se volá funkce `process_main()` nebo `process_array()`.

`process_main($node, array $array)` Hlavní funkce na procházení objektu (v JSON souboru ohraničeno `{}`, v proměnné `$jsn` je uloženo jako asociativní pole). Pokud obsahuje jiné pole nebo objekt, za použití rekurze se volá tato funkce nebo funkce `process_array()`. Ostatní hodnoty typu `literal`, `string`, `integer` a `real` zpracovává funkce `process_other()`.

`process_array($node, array $array)` Prochází a zpracovává indexované pole `$array` (v JSON je pole označeno `[]`), zohlední zadané argumenty (`-l`, `-s`, `-i` a `--types`) a vloží do XML. Pokud pole obsahuje jiné pole nebo objekt, za použití rekurze se volá tato funkce nebo funkce `process_main()`.

`process_other($node, $key, $value)` Pomocná funkce pro `process_main()`, která zpracovává hodnoty `$value` typu `literal`, `string`, `integer` a `real`, zohlední zadané argumenty (`-l`, `-s`, `-i` a `--types`) a vloží do XML společně s klíčem `$key`.

Připomínky k řešení

- Úkolem `DOMDocument` je vytvořit validní soubor XML, což znamená, že pokud hodnota obsahovala problematické znaky, byly přeloženy i přesto, že nebyl zadán argument `-c`. Tento problém jsem vyřešil tím, že před výpisem souboru na upřesněný výstup jsem si uložil celý XML soubor do `stringu`, zjistil si, jestli byl argument `-c` zadán a pokud nebyl, tak za pomoci funkce `html_entity_decode()` jsem problematické znaky převedl zpět na jejich problematickou verzi.
- Pro zkrácení zápisu jsem se snažil používat ternární operátor (`podmínka ? pravda : nepravda;`).
- Pro validaci názvu elementu byly použity 2 regexy (první, zda-li je název validní a druhý pro nalezení nevalidních znaků)