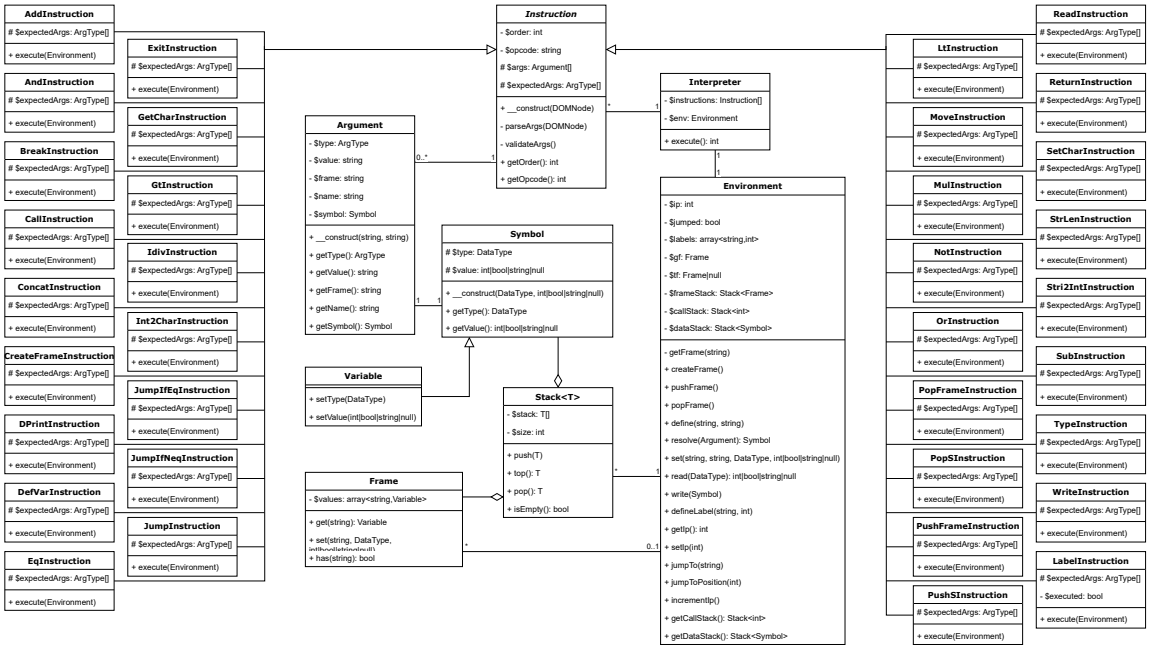
Implementační dokumentace k 2. úloze do IPP 2023/2024

Jméno a příjmení: Milan Vodák
Login: xvodak07

1 Diagram tříd



2 Popis částí

2.1 `Interpreter`

Třída `Interpreter` implementuje metodu `execute()`, která je vstupním bodem celého interpretu. Nejprve provádí zpracování a validaci vstupního XML souboru a z dat vytváří instance instrukcí, které si ukládá do pole. Následně provede jeden průchod programem, při němž spouští pouze instrukce `LABEL` tak, aby před skutečným začátkem interpretace byla definována všechna návěští. Nakonec nastaví čítač instrukcí zpět na začátek a s jeho použitím vykonává jednotlivé instrukce.

2.2 `Environment`

Třída `Environment` popisuje běhové prostředí a aktuální stav interpretu (jinými slovy kontext interpretace). Prostředí obsahuje zásobník rámců, globální a dočasný rámec, zásobník volání i datový zásobník. Navíc uchovává definovaná návěští a číslo aktuálně prováděné instrukce (*instruction pointer*). Třída poskytuje metody pro manipulaci s rámci, proměnnými a návěštími a pro provádění skoků. Prostřednictvím nich komunikují prováděné instrukce s běhovým prostředím.

Metody `jumpTo()` a `jumpToPosition()` umožňují provést skok na návěští, resp. konkrétní pozici úpravou instruction pointeru. Metoda `define()` slouží k definici nových proměnných v konkrétním paměťovém rámci a metoda `defineLabel()` k definici návěští. Prostřednictvím metody `set()` je možné nastavit hodnotu proměnné podle názvu a rámce. Metoda `resolve()` podle daného argumentu vyhledá a vrátí proměnnou, nebo v případě, že argument není proměnná, vrátí odpovídající symbol (viz Sekce 2.7).

2.3 `Frame`

Paměťový rámec sestává z asociativního pole proměnných, identifikovaných jejich názvem, a z metod sloužících k nalezení a nastavení hodnoty.

2.4 `Instruction`

Abstraktní třída `Instruction` je nadtřídou všech instrukcí z instrukční sady a slouží jako jejich vzor. Atributy instrukcí jsou pořadí, operační kód, očekávané typy argumentů a pole obsahující skutečně načtené argumenty. Konstruktor, který je implementovaný pouze v nadtřídě, provádí převod XML uzlu na instanci instrukce a jeho potomků na argumenty. Dále třída obsahuje implementace metod `parseArgs()` a `validateArgs()`, které zajišťují zpracování a validaci těchto argumentů. Nejdůležitější metoda, `execute()`, sloužící k vykonání instrukce, musí být implementována v každé podtřídě.

2.5 Instrukční sada

Každý typ instrukce, podporovaný interpretem, je reprezentován podtřídou třídy `Instruction` (viz Sekce 2.4) ve jmenném prostoru `IPP\Student\Instruction`, přičemž každá konkrétní instrukce implementuje abstraktní metodu `execute()`. Také může redefinovat pole `expectedArgs`, které obsahuje typy argumentů, jež instrukce očekává.

2.6 `Argument`

Argument instrukce má typ vyjádřený výčtovým typem `ArgType` a obsahuje řetězec s hodnotou přečtenou ze vstupu, objekt třídy `Symbol` (viz Sekce 2.7) reprezentující konstantu, návěští nebo typ a atributy `name` a `frame`, které v případě, že argument je proměnná, obsahují řetězec s jejím rámcem, resp. názvem.

2.7 `Symbol`

Symbolem se rozumí konkrétní hodnota v programu – konstanta, návěští nebo typ – případně proměnná reprezentovaná podtřídou. Instance symbolu je vytvořena při instanciaci argumentu, který není proměnnou, nebo při ukládání hodnot na datový zásobník.

Symbol má datový typ určený výčtovým typem `DataType` a hodnotu, na niž se v závislosti na tomto typu nahlíží jako na `int`, `bool`, `string` nebo `null`.

2.8 `Variable`

Proměnná je podtřídou třídy `Symbol` (viz Sekce 2.7) a oproti běžnému symbolu má navíc možnost prostřednictvím metod měnit svůj typ a hodnotu.

2.9 `Stack`

Třída `Stack` je běžnou implementací zásobníku. Je to generická třída typovaná s využitím standardu PHPDoc, díky čemuž může zásobník obsahovat hodnoty libovolného určeného typu a jediná třída je tak použita nejen jako zásobník rámců (typ `Frame`), ale i jako zásobník volání (typ `int`) a datový zásobník (typ `Symbol`).

2.10 Výjimky

Výjimky definované ve jmenném prostoru `IPP\Student\Exception` jsou přímými potomky třídy `IPPException`. Každá z nich odpovídá jinému návratovému kódu a ty, u nichž je to vhodné, navíc přijímají jako parametr instanci instrukce, při jejímž provádění chyba nastala, a zmiňují ji v chybové hlášce. Z důvodu přehlednosti jsou výjimky vynechány v diagramu tříd.