FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

Západočeská univerzita

Fakulta aplikovaných věd

Informatika

Semestrální práce ze ZSWI

Autor:	*Osobní číslo:*
David Markov	
markovd@students.zcu.cz	A18B0262P
Martin Brožek	
brozekm@students.zcu.cz	A18B0180P
Martin Straka	
marstr@students.zcu.cz	A18B0319P

17. května 2020

Obsah

Kapitola 1

Uživatelská dokumentace

1.1 Spouštění jednotkových testů

Ke spuštění jednotkových testů je třeba aplikaci spustit se spouštěcím parametrem –u. Jako druhý parametr lze zadat GUID testovaného filtru. Pokud je druhý parametr ponechán prázdný, dojde ke spuštění všech implementovaných jednotkových testů pro všechny filtry.

Příklad spuštění 1: `SmartTester.exe –u`

Příklad spuštění 2: `SmartTester.exe –u {C0E942B9-3928-4B81-9B43-A347668200BA}`

1.2 Spouštění regresních testů

Ke spuštění regresních tesů je třeba aplikaci spustit se spouštěcím parametrem –r. Jako druhý parametr je nutné aplikaci předat cestu ke konfiguračnímu souboru testovaného scénáře ve složce *scenarios*.

Příklad spuštění: `SmartTester.exe –r ../scenarios/scenario01/config.ini`

Kapitola 2

Technická dokumentace

2.1 Přidání jednotkových testů

Přidání nových jednotkových testů záleží na tom, zda-li jsou přidávány pro nový filtr, nebo pro filtr s již existující testovací třídou. Pokud již existuje testovací třída, pro přidání nového testu je třeba napsat v testovací třídě test ve formě členské funkce, vracející `HRESULT`, která nepřebírá žádné parametry. Tuto funkci je pak třeba zavolat ve funkci `executeSpecificTests()` ve formátu:

```
executeTest(<název testu>, std::bind(&TestovacíTřída::nazevTestu,
this));
```

Pokud testovací třída pro testovaný filtr neexistuje, je třeba tuto testovací třídu vytvořit. Ve složce *src/testers* vytvořte hlavičkový soubor s deklarací třídy. Tato třída musí být odvozena od třídy `GenericUnitTester` a musí implementovat virtuální členskou funkci `executeSpecificTests`. Ve složce *src/testers* dále vytvořte zdrojový soubor s implementacemi jednotlivých členských funkcí. Pro pohodlnost je dobrá v souboru *src/utils/constants.h* doplnit konstanty pro název knihovny, která filtr vytváří a pro jeho `GUID`. Dále je třeba doplnit do konstruktoru třídy `GuidFileMapper` doplnit mapování `GUID` daného filtru na jeho knihovnu ve formátu:

```
guidFileMap.insert(std::pair<GUID, const wchar_t*>(GUID_FILTRU,
NAZEV_KNIHOVNY));
```

V poslední řadě je třeba přidat v konstruktoru třídy `GuidTesterMapper` mapování `GUID` daného filtru na tovární metodu vracející instanci testovací třídy tohoto filtru ve formátu:

```
guidTesterMap[GUID_FILTRU] = std::bind<GenericUnitTester*>
( &GuidTesterMapper::createInstance<NazevTestovaciTridy>, this,
std::placeholders_1, Std::placeholders_2, std::placeholders_3);
```

2.2 Správa regresních testů

Všechny testovací scénáře jsou uloženy ve složce *scenarios*. Každý scénář je třeba mít ve vlastní podsložce, která musí obsahovat konfigurační soubor s názvem *config.ini*, výsledný referenční log s názvem *log.csv* a případný datový *.csv* soubor. V konfiguračním souboru je třeba nastavit název výstupního logu na `Log_File = log.csv` a cestu ke vstupnímu datovému souboru na `Log_File = ../scenarios/nazev_scenare/nazev_souboru.csv`