FAKULTA
APLIKOVANÝCH VĚD
ZÁPADOČESKÉ
UNIVERZITY
V PLZNI

KIV/UUR

Branch Checkout GUI

ParametricPane

2015/2016 Zdeněk Novotný

Obsah

1 Úvod

Tento program byl vytvořen pomocí technologie Java 8[1] a JavaFX[1]. Slouží pro zadávání parametrů a následné spuštění Apache Ant[2] scriptu. Samotný skript může obsahovat libovolné akce vyžadující různá potvrzení, podmínky spouštění a to by měl mít jeho autor na vědomí při vytváření seznamu předávaných parametrů. Názvy předávaných parametrů skriptu musí odpovídat identifikátorům použitým v konfiguračním souboru popsaném v kapitole 3.3. Tyto parametry poté vyžadují lokalizovaný překlad popsaný v kapitole 3.1.

2 Aplikace

2.1 Zavedení a požadavky aplikace

Samotná aplikace nevyžaduje instalaci. Je nutné pouze rozbalit archiv:

```
CheckOutApplication.zip
|- config (složka)
|- lang (složka)
|- CheckOutApplication.jar
|- RUN.bat (Windows)
|- RUN.sh  (Linux)
    Pro případ špatné konfigurace Javy
    nebo spouštění z příkazové řádky.
```

Pro správný běh je nutné mít nainstalovanou verzi **Javy SE 1.8_73** (na této verzi byla kompilována).
Dále je nutné míti správně nastavené prostředí **Apache Ant**.

Kromě tohoto je nutné připravit prostředí pro spouštěný skript, to znamená, pokud spuštěný skript spouští nějaké aplikace, musí tyto aplikace býti v systémových cestách nebo jejich umístění musí být absolutní.

2.2 Užití aplikace

Aplikace se při správném nastavení asociací spouští poklepem na:

```
CheckOutApplication.jar
```

Pokud se aplikace nespustí nebo se otevře v jiném asociovaném editoru. Je možné tuto aplikaci pustit pomocí skriptu:

```
RUN.bat - pro Windows
RUN.sh  - pro Linux
```

Hlavními dvěma prvky aplikace jsou:

- Panel pro zadání parametrů
- Konzole pro výpis aktuálního průběhu

Obě tyto části jsou přepínatelné pomocí „harmoniky“.

2.2.1 Panel parametrů

Panel parametrů je přesněji popsán ve vlastní kapitole 3.1.

2.2.2 Konzole

V této části aplikace se nachází výpisové okno a ve spodní části tlačítka pro kontrolu skriptu. Výpisové okno je jednoduchou implementací prvku zobrazujícího seznam. Tento typ implementace dovoluje „Lazy Load“, čímž se aplikace nezpomaluje a reaguje hbitě na změnu ve výpise do konzole. Pro přichycení posunu, je nutné odrolovat do spodní části konzole.

Celý výpis je možné smazat nebo exportovat do souboru s názvem:

script_log_yyyy-mm-dd_hh-mm-ss.txt

Tento soubor se uloží do kořenového adresáře aplikace.

2.3 Implementace aplikace

Samotná aplikace se skládá ze 3 tříd v nacházejících se v těchto balíčcích:

```
cz.deznekcz.forAimtec.checkout
|- CheckOutApplication
   Statická třída deklarující vzhled a funkčnost aplikace.
cz.deznekcz.forAimtec.checkout.run
|- Consola
|  Třída jejíž instance se chovají jako řádkové konzole.
|- CheckOut
   Statická třída řídící běh samotného skriptu
```

Dále je připojena knihovna myTools[5], které jsou součástí jazykové překlady pomocí **.xlng* souboru nebo souborů **.properties*.

A v neposlední řadě Parametrický panel, jež je nově součástí této knihovny.

2.3.1 CheckOutApplication

Statická třída deklarující vzhled a funkčnost. Pomocí metody `main` této třídy se spouští celá aplikace včetně potřebných připojených knihoven. Dále se inicializuje rozhraní celé aplikace, které je popsáno v kapitole 2.2.

2.3.2 Consola

Třída, jejíž instance se při vložení do rozhraní `JavaFX` jako řádková konzole. Obsah řádků nelze měnit a nelze zkopírovat pouze část.

Instance třídy implementují rozhraní `java.lang.Appendable`, díky níž lze zapisovat jednotlivé řádky do konzole.

Instance třídy mají vlastnost `clear`, kterou lze sledovat pomocí `JavaFX-Beans`.

2.3.3 CheckOut

Statická třída zajišťující kontrolu běhu programu. Po výpis aktuálního stavu je nutné připojit třídy implementující rozhraní `java.lang.Appendable`. Třída je vybavena vlastností, kterou lze naslouchat. Její hodnota je jednou z hodnot výčtového typu:

```
cz.deznekcz.forAimtec.checkout.run.CheckOut.State
|- RUNNING  - proces aktuálně běží
|- STOPPED  - proces aktuálně neběží
|- CHANGING - proces se připravuje k zahájení
              nebo k ukončení
```

Aplikace původně implementuje ruční přerušení, ale protože Ant skript může spouštět nové procesy v rámci jednoho skriptu, je tato funkce nestabilní a může se stát, že se proces neukončí.

3 Parametrický panel

3.1 Popis

Parametrický panel skládá ze dvou sloupců. První sloupec obsahuje lokalizované názvy parametrů přeložené pomocí souboru „*lang_ **_ **.properties*“ nebo „*_ **.xlng*“, přičemž první soubor je nadřazený prvnímu a přepisuje jej. Soubor lokalizace definuje i tooltip (nápověda při posunutí myši nad řádek parametrů).

Například:

```
ParametricPane.text.textParam-label=Textový parametr
ParametricPane.text.textParam-tooltip=Tento parametr slouží pro zadávání...
```

Použitelné typy parametrů jsou dále definovány v kapitole 3.3.
! Při každé změně parametru, je tato změna uložena do souboru „parameter_set.xml“. Tento soubor je dále distribuovatelný mezi stejnými verzemi souboru „parametric_pane.xml“, který je dále popsán v kapitole 3.3.

3.2 Implementace

Pro implementaci panelu do vlastní aplikace je nutné splnit tyto podmínky:

1. SDK Java 1.8_73
2. JavaFX
3. Knihovna myTools

Pro začlenění panelu do aplikace je nutné vytvořit v inicializační sekci uživatelského rozhraní novou instanci třídy:

```
cz.deznekcz.javafx.parametricPane.ParametricPane
```

A tuto instanci vložit jako konečný uzel rozhraní. Dále je možné použít implementaci rodičovské třídy:

```
javafx.scene.layout.BorderPane
```

Kde je zakázáno měnit její střed.

Dále je nutné pro správnou funkčnost přesunu pomocí klávesy **TAB** a kombinace **SHIFT** + **TAB** zaregistrovat předchozí a následující komponentu pro přesuny fokusu. Tím je zhoršena použitelnost při kliknutí myší, může vyvolat posun prvků trochu rozhodit uživatele při výběru hodnot ze seznamu u parametru ***list***. Komponenty se registrují pomocí statických metod:

```
void registerBefore(javafx.scene.Node)
void registerAfter(javafx.scene.Node)
```

Ve třídě:

```
cz.deznekcz.javafx.parametricPane.ParametricTraverser
```

3.3 XML definice

Pro jednodušší práci s definicí parametrů je k dispozici XML schema. Pokud uživatel nemá k dispozici účinný editor na XML (integrovaný např. do Eclipse [3]), je k dispozici kontrola správnosti online na stránkách FreeFormater.com [4]. Vytvoří dialogové okno s chybou.

Dokument začíná hlavičkou:

```
<?xml version="1.0" encoding="UTF-8"?>

<parameters
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="parametric_pane.xsd">
```

kde element ***parameters*** je kořenovým elementem při dekódování XML. Dokument končí uzavřením elementu:

```
</parameters>
```

Do tohoto kořenového elementu je možné dále vkládat elementy popsané dále v kapitolách 3.5 XML atributy a 3.6 XML elementy.

3.4 Dekódování XML

Dekódování XML probíhá pomocí integrovaného nástroj DOM. Pro každý element v souboru „parametric_pane.xml“ jsou 1 ku 1 vytvořeny výčtové konstanty ve třídě: třída Tyto konstanty definují, jaká funkce bude použita k získání tříd reprezentujících dané parametry.

Nebudu zde více rozepisovat zdrojový kód. Důvodem je jeho viditelnost online na stránkách projektu *myTools*[5].

3.5 XML atributy

3.5.1 ATRIBUT: default

Obsahuje hodnotu omezenou typem parametru. Může se jednat o text, číslo nebo logickou hodnotu. Tento typ je specifikován v závorce u jednotlivých parametrů.

3.5.2 ATRIBUT: editable

Tento atribut určuje počáteční stav upravitelnost parametru. Hodnota tohoto atributu je rovna ***true/false***.

3.5.3 ATRIBUT: extension

Tento atribut určuje názvy povolených souborů k vybírání. Momentálně je použit pouze u parametru ***browse_file***.
Příklady:
`extension="*.*"`
`extension="*.java;*.class"`
`extension="run.BAT"` Nynější implementace dovoluje přidělení pouze jednoho filtru a to se všemi povolenými názvy včetně „*****“.

Pro možnost lokalizace názvu filtru je využit celý obsah atributu a klíč pro „***.java;*.class**“ vypadá takto:

```
ParametricPane.extension.*.java;*.class
```

3.5.4 ATRIBUT: format

Jedná se o textový atribut obsahující funkci nebo konstantu.
Například:

```
format="constant ${var1} and $[mod1][mod2]{var2}"
```

Tento formátovací řetězec říká, že bude zřetězen text „**constant** “ včetně mezer a dále bude následovat hodnota parametru ***var1***, text „ **and** “ a poté modifikovaná hodnota parametru ***var2*** s modifikátory ***mod1*** a ***mod2*** v tomto pořadí.
* Mezi povolené modifikátory patří:
- ***lowerCase*** - promění všechna písmenka v malá
- ***upperCase*** - promění všechna písmenka ve velká
- ***numberSplit*** - rozdělí od sebe písmena a číslice podtržítkem, pro jednodušší vysvětlení uvádím pár příkladů pro `format="$dci_[numberSplit][lowerCase]{var}_tst"`:

```
EDC587 -> dci_edc_587_tst
ERR_BY978 -> dci_err_by_978_tst
N_N8_9 -> dci_n_n_8_9_tst
ATEX_123 -> dci_atex_123_tst
N_8_N -> dci_n_8_n_tst
```

3.5.5 ATRIBUT: id

Obsahuje textovou hodnotu vymezující unikátní identifikátor parametru.

3.5.6 ATRIBUT: index

Obsahuje číselnou hodnotu vymezující unikátní pozici parametru v panelu. Toto dovoluje uspořádání větších bloků pro jednodušší uspořádání zaškrtávacích tlačítek (element ***check***).

3.5.7 ATRIBUT: values

Tento atribut určuje seznam proměnných, které se musí shodovat. V této fázi implementace jsou povoleny odkazy pouze na zaškrtávací políčka (element ***check***) a celá podmínka je splněna pouze v případě, že jsou splněny všechny podmínky.
Příklad:

```
values="check1=true;check2=false"
```

Tento příklad říká, že zaškrtávací tlačítko ***check1*** musí být zaškrtnuto a ***check2*** nezaškrtnuto, aby byla splněna celková podmínka.
* V pozdější implementaci bude možné vytvářet složitější struktury povolování, protože budou odstíněny od samotných parametrů do jiné logiky.

3.6 XML elementy

3.6.1 ELEMENT: browse_dir

Tento parametr dovoluje výběr adresáře, jehož celá cesta je následně je poté do tohoto parametru uložena absolutně.

Dále tento parametr musí obsahovat atributy ***id*** a ***index***.

3.6.2 ELEMENT: browse_file

Tento parametr dovoluje výběr souboru, jehož celá cesta je následně je poté do tohoto parametru uložena absolutně.

Soubory lze filtrovat dle obsahu elementu ***extension***

Dále tento parametr musí obsahovat atributy ***id*** a ***index***.

3.6.3 ELEMENT: editable

Tento párový element není parametrem, ale slouží pro možnost dynamického použití parametrů. Do tohoto stromu lze vkládat libovolné parametry nebo další podskupiny, které mají tuto vlastnost (v této fázi vývoje pouze element ***text***). Pro tento element je nutné specifikovat počáteční hodnotu pomocí atributu ***default***(boolean) a hodnotovou funkci v atributu ***values***.

3.6.4 ELEMENT: enabled

Tento párový element není parametrem, ale slouží pro možnost dynamického použití parametrů. Do tohoto stromu lze vkládat libovolné parametry nebo další podskupiny. Pro tento element je nutné specifikovat počáteční hodnotu pomocí atributu ***default***(boolean) a hodnotovou funkci v atributu ***values***.

3.6.5 ELEMENT: check

Tento parametr uchovává hodnotu ***true***/***false***, musí obsahovat atributy ***id*** a ***index***. Na tento parametr lze vázat uzamykání a upravitelnost parametrů nebo dalších zaškrtávacích polí. V době psaní tohoto dokumentu nejsou implementovány jiné než součinové podmínky pro elementy ***editable*** a ***enabled***.

3.6.6 ELEMENT: list

Tento parametr musí obsahovat atributy ***id*** a ***index***. Slouží pro výběr hodnoty ze seznamu. Element pro tento typ parametru je párový a mezi jeho značky se vkládají další elementy ***value***, které svým textovým obsahem symbolizují položky výběru tohoto seznamu.

3.6.7 ELEMENT: logic

Tento párový element není parametrem, ale slouží jako filtr parametrů pro výsledný výpis. Pokud je parametr logický, slouží pouze jako pomocný při vyplňování a ukládají se pouze do souboru s aktuálním nastavením parametrů.

Do těla tohoto párového elementu je možné přidat libovolný z ostatních (další elementy ***logic*** uvnitř tohoto, jsou již nadbytečné).

3.6.8 ELEMENT: password

Tento parametr musí obsahovat atributy ***id*** a ***index***. Slouží jako textové pole se skrytými znaky před zraky kolemjdoucího. Implementace aktuálně nedovoluje možnost hodnotu tohoto pole neukládat a prozatím bude vždy uložena v souboru s aktuálními hodnotami.

3.6.9 ELEMENT: text

Tento parametr musí obsahovat atributy ***id*** a ***index***. Slouží jako vstupní pole, pokud je hodnota atributu ***editable*** rovna ***true***, v opačném pouze zobrazuje text, který se dynamicky mění závisle na funkci obsažené v atributu ***format***. Tato funkce nemusí obsahovat parametry, v tomto případě je použita jako výchozí hodnota.

3.6.10 ELEMENT: value

Pro tuto verzi implementace nejsou k dispozici další rozšiřující informace, než které jsou obsaženy v popisu elementu ***list***.

V pozdější implementaci bude k dispozici možnost lokalizovat tyto hodnoty a nebo budou moci obsahovat formátovací funkci jako parametry ***text***.

4 Závěr

Se svou prací jsem spokojen. Myslím, že účel, pro který byla stvořena, dokáže splnit. Část této práce (Parametrický panel) je možné využít i při dalším programování obdobných nástrojů, které mají za úkol vytvářet jednodušší formuláře.

Tento panel je dále připraven se rozšiřovat o další typy parametrů a provázanost pomocí funkcí.

4.1 Testování

Aplikace byla během implementace testována konzultantem cílové skupiny a automatickým testem na provázanost parametrů. Bohužel toto testování nedokázalo eliminovat všechny chyby a je možné, že při špatné konfiguraci XML 3.3 se nemusí načíst tato konečná aplikace správně. V tomto případě se objeví dialogové okno s chybou.

Reference

[1] [Java] Webové stránky technologie Java 8: http://docs.oracle.com/javase/8

[2] [Ant] Webové stránky technologie Ant: http://ant.apache.org/

[3] [Eclipse.org] Webové stránky IDE Eclipse: http://www.eclipse.org/

[4] [FreeFormater.com] Webové stránky s užitečnými nástroji pro úpravu a kontrolu XML a schémat: http://www.freeformatter.com/xml-validator-xsd.html

[5] [DeznekCZ] Zdeněk Novotný: *myTools* [online]. [cit. 2016-05-19]. Dostupné z: https://github.com/DeznekCZ/myTools