WOJSKOWA AKADEMIA TECHNICZNA IM. JAROSŁAWA DĄBROWSKIEGO W WARSZAWIE

***Sprawozdanie z przedmiotu „Języki formalne i kompilatory”***

**Temat zajęć:** *Scripting*

**Autorzy:** *Jakub Kapusta, Adam Kochański, Mikołaj Bednarek*

**Grupa:** *I6B2S1*

**Prowadzący*:*** *mgr inż. Krzysztof Mierzejewski*

***Opis implementacji zadania***

Zadanie – aplikacja modyfikująca dane, została wykonana w technologii **Java/JavaFX** z dołączonymi silnikami **Nashorn** oraz **Jython.**

**Założenia:**

**Nashorn:**

* Deklaracje zostają zapamiętane między wywołaniami skryptów
* Z powodu słabego typowania w języku JavaScript, konwersja typów dokonywana jest automatycznie. Np. wprowadznie ciągu znaków “49” w pole wartości liczbowej spowoduje wprowadzenie liczby 49 w to pole. Podobnie w przypadku wartości typu boolean. W przypadku wprowadzenia nietypowych znaków np. liter alfabetu łacińskiego. Wprowadzona zostanie liczba 0, czy wartość false w przypadku wartości typu boolean.

**Jython:**

* Deklaracje nie zostają zapamiętywane między wywołaniami
* Podanie do pola wartości o złym typie skutkuje błędem wypisywanym w okienku błędów

Zarówno dla Nashorn, jak i Jython, następuje przekierowanie strumienia wyjścia (std) oraz strumienia błędów (stdr) do specjalnie utworzonych do tego TextArea(zostało to przedstawione w treści sprawozdania dotyczącej widoków).

Kolekcja *entities* została umieszczona w silnikach poprzez zastosowanie następujących poleceń:

* pythonInterpreter.set("entities", entities);
* engine.put("entities", entities);

**Encja:**

Atrybuty:

* Name :String
* Surname :String
* Age :int
* Monthly Salary :double
* Number of children :int
* Employed :boolean
* Married :boolean

W załączonym pliku csv znajdują się encje:



Do wymienionych języków skryptowych dołączone są także przykładowe skrypty.

Zostały one dołączone jako załączniki w osobnych plikach:

* przykladowe.js – Nashorn
* przykładowe.py – Jython

**Nashorn:**

1. Podaj punkty zdolności kredytowej dla klientów

for each(var e in entities) {

var score = 0;

if(e.employed) {

score += 100 - e.age;

if(e.married) {

score += 100;

}

score += e.monthSalary / 10 - 100;

score -= e.numOfChildren \* 10;

}

print(e.name + " " + e.surname + ": " + score);

}

2. Uaktualnij wiek każdego z klientów

for each(var e in entities) {

e.age++

}

**Jython:**

1. Wypisanie informacji o klientach do strumienia błędów

import sys

def emp(entity):

if entity.employed:

return "employed"

return "unemployed"

def mar(entity):

if entity.married:

return "married"

return "single"

for e in entities:

sys.stderr.write(e.name + " " + e.surname + " - age: "+str(e.age)+", "+emp(e)+" "+mar(e)+"\n")

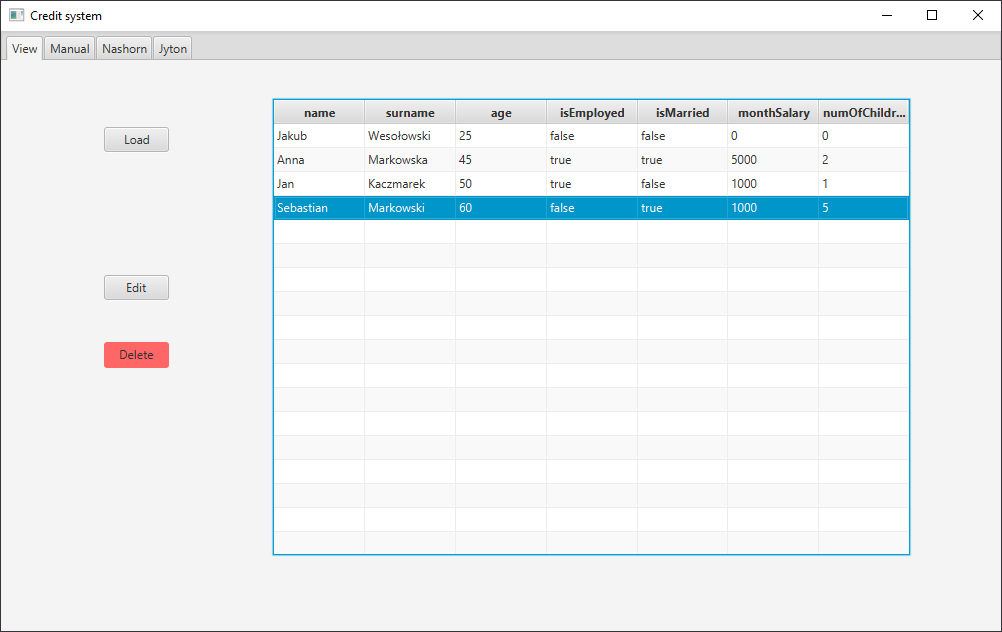
2. Zamień imiona z nazwiskami klientów

for e in entities:

e.name, e.surname = e.surname, e.name

***Widoki:***

**Widok tabeli**

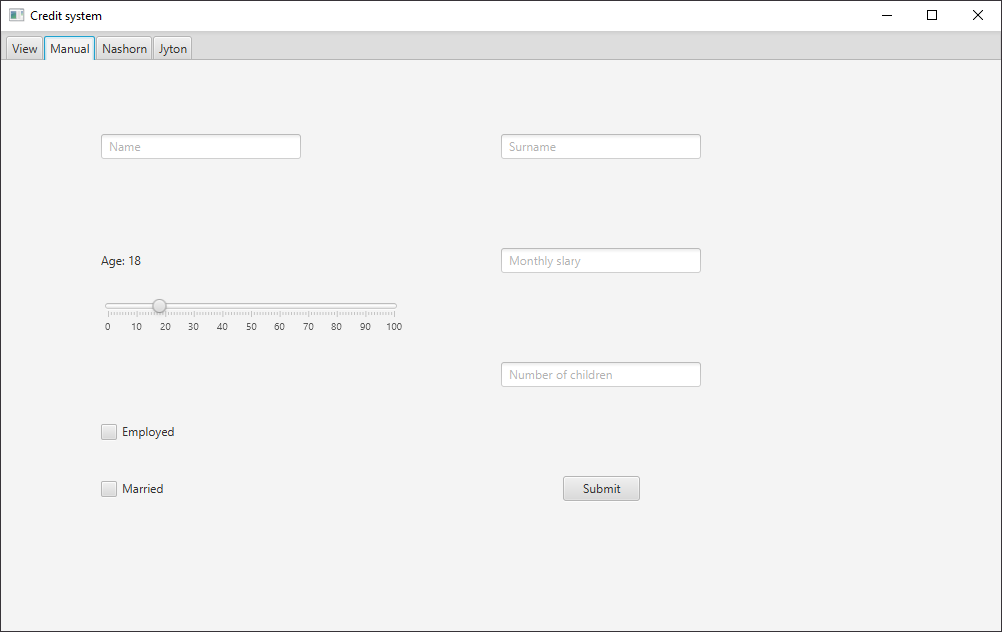


Load – ładowanie pliku csv

Edit – edycja wybranej encji

Delete – usunięcie wybranej encji

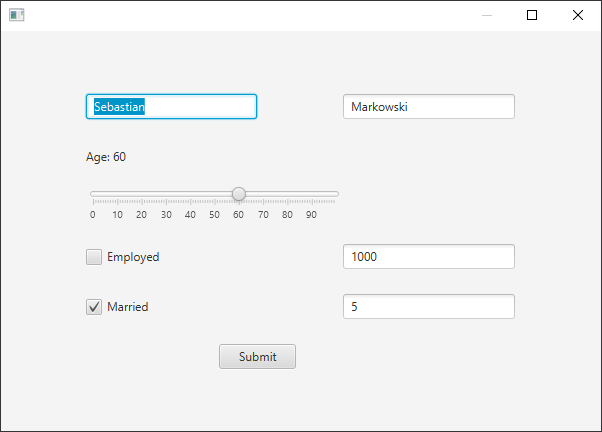
**Widok dodawania encji**



Dane pola oznaczają atrybuty wprowadzanej encji

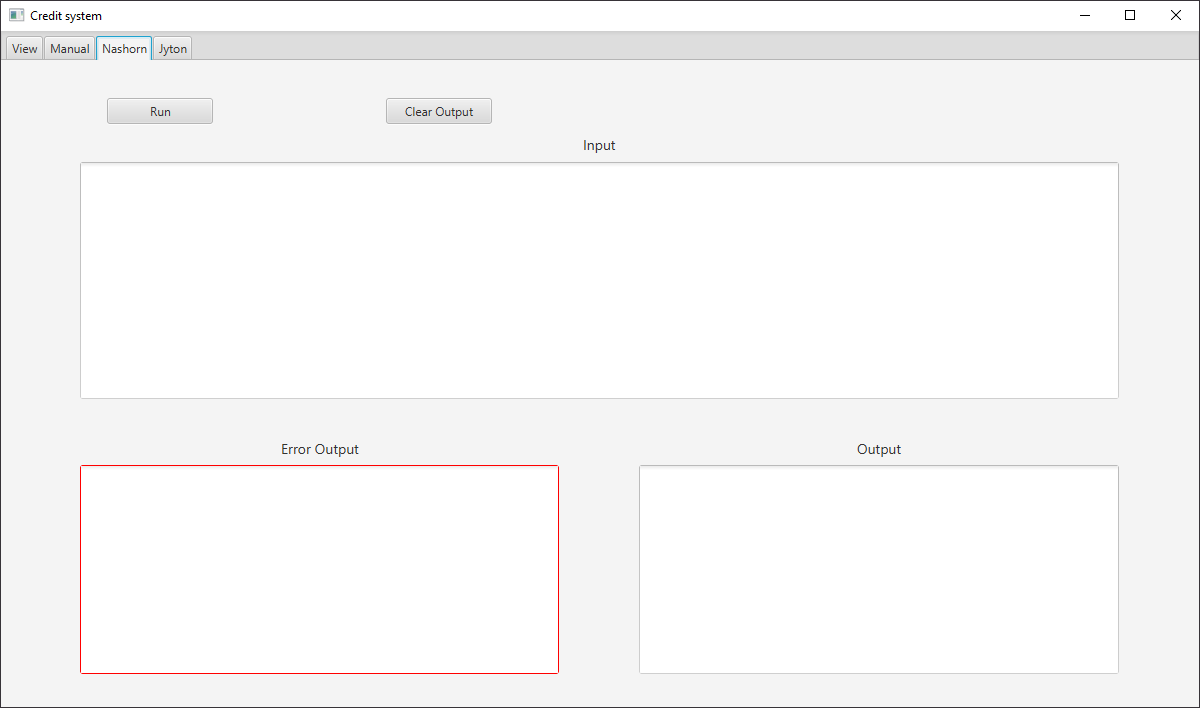
Submit – wprowadzenie encji/ aktualizacja stanu dodawania poprzez wyświetlenie obok napisu z błędem/poprawnym dodaniem

**Widok edytowania encji**



Analogicznie jak w widoku wprowadzania nowych encji. Po pomyślnym wprowadzeniu okienko się zamyka.

**Widok wprowadzania skryptów Nashorn/Jython**



Input – pole do wprowadzania skryptu

Error output – pole w którym wypisywane są błędy

Output- pole wypisywania wyniku działania skryptu

Run – uruchomienie skryptu

Clear Output – wyczyszczenie obydwu pól Output