# Aplikace pro řízení paralelního zpracování dat

Uživatelská dokumentace 2017/18

Tomáš Jínek - xjinek00 Jiří Bašta - xbasta03 Pavel Hodoval - xhodov01 David Kozák - xkozak15

# Základní struktura aplikace

Projekt se skládá ze 3 části:

- Common Obsahuje obecné třídy a funkce, které jsou společné pro node a server.
- Server Obsahuje administraci systému s veškerou správou a řízením úkolů a příkazů pro jednotlivé klienty.
- Node Klientská část aplikace, představuje jeden pracovní uzel, na kterém probíhají úkoly, zadané serverem.

# Zprovoznění aplikace

V první řadě je třeba nastavit jednotlivým částem aplikace připojení k databázi.

#### Server

V serverové části aplikace přejdeme do složky src/main/resources/ a zde nastavíme v souboru local.properties údaje pro připojení k MySQL databázi.

V případě změny portu webového serveru (výchozí je 8080) přidáme do souboru application.properties následující řádek:

```
server.port=CISLO
```

Při prvním spuštění webové aplikace je třeba si zaregistrovat účet klasickým způsobem (v sekci Register) a v MySQL databázi tomuto účtu přidat administrátorská práva pomocí SQL příkazu:

```
INSERT INTO `user_roles` (`user_id`, `roles`) VALUES ('1', 'ROLE_ADMIN')
```

#### Node

V klientské části aplikace přejdeme do složky resources stejným způsobem jako výše a nastavíme v application.properties port, na kterém má node naslouchat, připojení k databázi, adresu a port, na které běží webový server, ve výchozím případě je použito:

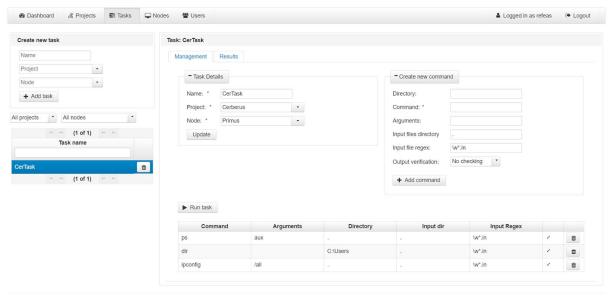
```
cz.vutbr.fit.gja.proj3.node.ServerNotifier.server_url=http://local
hoSt:8080
```

### Přeložení a spuštění aplikace

Pro přeložení aplikace lze využít nástroj maven. Nejdříve je potřeba nainstalovat do lokálního maven repozitáře část Common, což lze vykonat příkazem mvn clean install. Poté je možno stejným příkazem přeložit Server i Node. Po přeložení lze oba programy spustit příkazem mvn spring-boot:run.

## Administrace

Na následujícím obrázku lze vidět grafické rozhraní administrace, konkrétně sekce Tasks. Většina administrace byla vytvořena v obdobném duchu. Levá strana slouží pro rychlé vytvoření nového úkolu, výpis a vyhledávání již vytvořených úkolů. Zbytek představuje detail a editaci rozkliknutého úkolu.



©2018 FIT, Brno University of Technology. All rights reserved. Created by Jifi Bašta, David Kozák, Pavel Hodoval, Tomáš Jínek.

## Komunikace

Komunikaci lze rozdělit do dvou částí. Ta první se týká navázání a udržování spojení mezi serverem a pracovním uzlem. Při startu pracovního uzlu se uzel sám pokusí registrovat na serveru tím, že pošle informace o své existenci na endpoint /hello. Pokud server běží, uloží si tyto informace do své databáze. Pokud by z nějakého důvodu tato registrace selhala, je možné uzel registrovat i ručně v aplikaci v sekci Nodes.

Server pravidelně každých 10 vteřin posílá zprávu na uzel na endpoint /hello, čímž testuje, zda uzel běží. Pokud by některý uzel neodpověděl, označí ho jako neaktivní. Stejně tak pokud uzel dříve označený jako neaktivní odpověděl, je označen jako aktivní.

Druhá část komunikace zahrnuje spuštění tasků na uzlech a předávání výsledků zpět na server. Server pošle na node na endpoint /task informace o tom, jaký task složený z jakých subtasků má být spuštěn. Uzel poté posílá informace o tom, že byl hlavní task zahájen, dílčí task zahájen, dílčí task ukončen (což se opakuje pro každý subtask) a hlavní task ukončen na endpoint /api/task/started, /api/task/subtask/started, /api/task/subtask/started, /api/task/subtask/finished v tomto pořadí. Každých 5 vteřin a také při dokončení subtasku uzel pošle na server výsledky tohoto subtasku na endpoint /api/task/subtask/result.