ISAT Job Tracker

Introduction

This document details the source code use and deployment procedures.

Frameworks and Tools

Below details the various tools used by the system in order to modify and/or deploy a new instance of the code. These tools will have to be installed on your system in order to do any changes to the code.

ExtJs

The frontend UI is created using Javascript and a framework called ExtJS. To compile the frontend UI code you will need the Sencha cmd tool installed. This allows the build process to use this tool to build the UI code for you automatically.

https://www.sencha.com/products/extjs/cmd-download/

Click Download with JRE Included and install the tool.

Gradle

Gradle is the actual build tool. It does many tasks related to deployment. I have created a single simple task for full deployments called "fullDeployableBuild" this will build the UI code and compile all of the java code into a single runnable zip file.

It also integrated with the next tool "IntelliJ IDEA" and allows you to click the "bootRun" task which will run the entire application.

I have included the Gradle tool within the source code itself so you do not need to download it independently.

IntelliJ IDEA

IntelliJ IDEA is an integrated development environment (IDE). This is like Visual Studio for Visual Basic. It is a tool that allows you to easily code and create applications written primarily in Java/Javascript while being fully integrated with Gradle.

Download and install this IDE, you can use the free community edition or ultimate if you like.

> https://www.jetbrains.com/idea/download/
> Download and install, configure to have Gradle build plugin enabled during installation; otherwise other defaults are 'ok'

MySQL

MySQL database is used to store all of the database related tables and data. This is not required to be installed on a development instance and is primarily used for production.

I have build the system to be intelligent enough to detect whether you have MySQL and if it is not setup it will run its own 'embedded' database within the application itself called H2. This embedded database is a lightweight version of MySQL.

You do not have to do anything here because you can simply use the provided embedded database that is managed for you automatically.

Source Code

Layout

- Simple-job-tracker
  - gradle
    - wrapper
    - The gradle executable package
  - job-tracker-runner
    - src
      - main
        - java
          - com.job.tracker
            - The Java code
          - db.migration
            - Database Migrations (Do not perform SQL based migration! Always use Java Spring repository initialization)

            - resources
                - company
                    - Resource files for initializing database tables
                - src
                    - ExtJS Javascript Code
                - Resources for the application, configuration properties files
        - build.gradle
    - .gitignore
    - build.gradle
    - gradle.properties
    - gradlew
    - gradlew.bat
    - settings.gradle
    - Build settings, specifically related to the Gradle build process.

Deployment

To run the application in your development environment go to the Gradle Projects tab and click job-tracker-runner:bootRun under applications. If you have made UI changes you will want to run fullDeployableBuild task first to ensure your UI code is ‘compiled’.

If you want to deploy a new production instance then run fullDeployableBuild followed by bootDistZip under application and distribution folders, respectively.

This will create a build directory at the root of the project. Within it there is a distributions folder that contains a zip file of the new code.

Take it and unzip it on the server. Using a text editor open up the bin folder file job-tracker-runner.bat (on Windows) and set

```
DEFAULT_JVM_OPTS=-Duser.timezone=UTC -d64 -server -Xmx4g -Xms4G
-XX:+UseG1GC -XX:MaxGCPauseMillis=200 -XX:ParallelGCThreads=20
-XX:ConcGCThreads=5 -XX:InitiatingHeapOccupancyPercent=70
```

Replace %CMD_LINE_ARGS% with
cuba.reporting.openoffice.path=/Applications/LibreOffice.app/Contents/MacOS

Where the openoffice path points to your openoffice installation folder

The above -Xmx4g -Xms4G defines the Xmx max RAM used and Xms min application RAM used.

An improvement would be to have the build process simply copy these properties over to this file but I didn’t get around to doing that.
Add the job-tracker-runner.bat file to your windows Task Scheduler to run on restart. You can also double click this bat file to run the application.

Production Database

To enable an external production database go to job-tracker-runner.properties within the root source code folder -> job-tracker-runner -> src -> main -> resources -> job-tracker-runner.properties

Then uncomment the first 4 lines by removing the #. These properties define he path to your production database. Create an empty database called job-tracker and you shouldn’t have to change anything if MySql is running on the same server as the application.

Development Setup

1. Install applications required from the Frameworks and Tools section.
2. Open IntelliJ IDEA
3. Click File -> New -> Project from Existing Sources…
4. Find the extracted root folder of the source code. Click Ok
5. Go to Gradle projects tab once it initializes everything.
6. Click the Gradle refresh icon
7. Click the build folder -> build task, once complete click fullDeployableBuild and bootRun.
8. You should have an instance of the simple-job-tracker running on http://localhost:8080

Contact Information

If any further services are needed:

Andrew Popp
ampopp04@gmail.com