UNIVERSIDADE
DE ÉVORA

Engenharia Informática
(2023/2024)

Sistemas Distribuídos

Sistema de Gestão de Artistas de Rua

Pedro Bacalhau - 52043
António Carvalho - 51483

Objetivos

O objetivo do trabalho era implementar as aplicações **Servidor**, **cliente** e **cliente administrador,** implementando também uma base de dados simples e usando uma solução de middleware apresentada nas aulas.

Base dados

A base dados que implementamos apresentam 2 tabelas a tabela **artists** e **donativos** que podem ser criadas da seguinte forma:

```
CREATE TABLE artists (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255),
    type VARCHAR(255),
    local VARCHAR(255),
    performing BOOLEAN,
    approved BOOLEAN
);

CREATE TABLE donatives (
    id SERIAL PRIMARY KEY,
    artistid INTEGER,
    name VARCHAR(255),
    value INTEGER
);
```

Desenvolvimento

Fizemos uso da interface ***Rmi*** para fazer a ligação servidor-cliente. Começámos por criar as tabelas numa base de dados local e de seguida a ligação do servidor com o cliente. A partir do momento que tínhamos esta ligação dada a simplicidade do RMI foi bastante direto a implementação das funções de gestão das bases de dados. Sendo possivel o envio e pedido de informação do lado cliente para a base de dados de seguida tivemos que fazer os pedidos de informações ao utilizador do aplicação do cliente.

Por último foi feito o cliente administrador para permitir gerir a base de dados de forma interativa.

Do lado do cliente normal (**client**) é possível registar um artista, filtrar artistas, e apresentar locais com atuações no momento. Também é possível fazer uma doação e observar as doações recebidas por cada artista.
Do lado do cliente administrador (**clientadmin**) é possível apresentar os artistas por estado e aprovar aqueles que ainda não se encontram aprovados.
O servidor (**server**) tem apenas a inicialização do objeto remoto (**remoteObjectImpl**) da interface (**remoteObject**) e da base de dados. O objeto remoto apenas contém as funções que chamam as funções da classe que gere a base de dados (**BDserver**). Esta por fim tem as funções que permitem consultar, inserir e atualizar campos das bases de dados.
Temos ainda a classe **donative** que trata das doações e a classe **artist** que é o modelo dos artistas na aplicação.

Compilação e Execução

Para compilar e executar o trabalho decidimos proceder da maneira tavez mais habitual e mais básica, pelo terminal diretamente na pasta do trabalho.
Para compilar usámos o comando: *javac -cp resources/postgresql.jar -d build *.java*
para inicializar a interface rmi - *rmiregistry -J-classpath -Jbuild 9000*
para executar o servidor - *java -cp build:resources/postgresql.jar -Djdbc.drivers=org.postgresql.Driver server 9000 localhost bd1 user1 ANTmar02*
para executar o cliente - *java -cp build:resources/postgresql.jar client localhost 9000*

para executar o cliente adminstrador - *java -cp build:resources/postgresql.jar clientadmin localhost 9000*

Conclusão

Este trabalho permitiu-nos aplicar de uma maneira mais complexa algumas das bases aprendidas, mesmo assim não fizemos uso de nenhum utilitário. Não implementámos os bónus de listagem de datas de atuação de artista e da descrição e foto do artista.