Amigo Oculto 1 - Usuários

Uma entidade para representar o usuário

A primeira implementação que faremos neste projeto é a representação de um **Usuário**, que é aquela pessoa que usará o sistema, podendo tanto criar grupos de amigo oculto, quando participar de grupos já criados. Neste nosso projeto, não teremos um usuário administrador geral. Assim, cada usuário será responsável por gerenciar os grupos que criar.

USUÁRIO

A entidade Usuário

Nós usaremos um objeto com os seguintes atributos para representar a entidade Usuário:

- `int idUsuário` - Este primeiro atributo é um valor que identifica cada usuário de forma única. Nesses casos, geralmente trabalhamos com valores numéricos, sequenciais e não significativos, pois esse valor não traz nenhuma informação específica do usuário; é apenas usado para distinguir um usuário do outro. A princípio, o usuário do sistema não deve se preocupar com esse valor. Ele será gerado automaticamente pelo sistema e só será usado nas relações internas entre os seus arquivos.
- `String nome` - Este segundo atributo deve conter o nome completo do usuário. Esse

atributo é que será apresentado em todas as vezes que houver uma listagem de usuários.
- `String email` - Neste atributo, armazenaremos o email do usuário. Esse email será usado tanto para acesso ao sistema (supondo que não existam dois ou mais usuários com o mesmo email), bem como será a forma correta de convidar as pessoas para participarem dos grupos de amigo oculto.
- `String senha` - A senha pessoal de cada usuário será armazenada neste quarto atributo. Ela só será usada para acesso ao sistema.

Por enquanto, esses dados sobre o usuário são suficientes. Observe que as três *strings* serão de tamanho variável, isto é, não estabeleceremos um limite inicial para quantos caracteres cada uma pode conter. Também é importante saber que usaremos a representação Unicode para essas *strings* (mas não temos que fazer nada, pois esse é o padrão).

É claro que você pode ampliar esse conjunto de acordo com sua visão sobre um projeto de amigo oculto. Talvez você queira acrescentar um número de telefone, uma data de aniversário ou outra informação parecida. Sinta-se à vontade para fazer esses acréscimos.

Classe Usuário

Agora você pode criar a classe Usuário. Ela deve ter todos os atributos citados acima, pelo menos um construtor e os métodos de acesso para cada um dos seus atributos. É essencial que sejam implementados pelo menos os métodos `getID()` e `setID()`, além do método `chaveSecundaria()`.

Os dois primeiro métodos são meio óbvios, mas esse método `chaveSecundaria()` nos oferecerá uma forma de identificar o usuário por meio de outro atributo que também deve ser exclusivo. No caso do usuário, usaremos o seu email como esse segundo identificador. Assim, esse método deve apenas retornar o email do usuário.

```
public String chaveSecundaria() {
  return this.Email;
}
```

Representação do usuário como um vetor de bytes

Você também precisará criar dois métodos especiais que serão usados para converter os atributos do usuário em uma sequência simples de bytes (de tamanho de variável) e outro para extrair de volta os atributos de uma sequência de bytes. Esses métodos podem ser chamados de `toByteArray()` e `fromByteArray()`. Os trechos abaixo exemplificam como esses métodos podem ser implementados em Java:

```
public byte[] toByteArray() throws IOException {
  ByteArrayOutputStream dados = new ByteArrayOutputStream();
  DataOutputStream saida = new DataOutputStream( dados );
  saida.writeInt(this.id);
  // Escrever os demais atributos do objeto usando métodos como writeInt(), writeUTF(), wr
```

```
iteFloat(), ...
  return dados.toByteArray();
}
```

```
public void fromByteArray(byte[] bytes) throws IOException {
  ByteArrayInputStream dados = new ByteArrayInputStream(bytes);
  DataInputStream entrada = new DataInputStream(dados);
  this.id = entrada.readInt();
  // Ler os demais atributos do objeto usando métodos como readInt(), readUTF(), readFloat
(), ...
}
```