

DB4O – Um BD com estilo!

Italo Moreira Campelo Maia

Este artigo visa mostrar como criar um mini aplicativo com acesso a banco de dados utilizando o DB4Objects.

Java e os bancos de dados

Seja em uma aplicação desktop ou em uma aplicação web, a não ser que ela seja muito simples, você irá em algum momento necessitar persistir dados. Devido à complexidade ou/e importância da integridade dos dados persistidos, diversas ferramentas de persistência foram criadas, inclusive sistemas especializados na persistência e obtenção de dados, os **bancos de dados**.

A tarefa de persistência de dados em java, a até pouco, era considerada uma atividade sacal, devido à grande necessidade de codificação, muitas vezes repetitiva, exigida para se executar pequenas tarefas no banco de dados. Problemas como o paradigma entre linguagens orientadas a objetos(OO) e bancos de dados relacionais têm sido uma grande preocupação dos desenvolvedores; ferramentas como o Hibernate tentam tornar esse paradigma menos visível, entretanto a necessidade de codificação de xml ou do uso extensivo de annotations não chega a ser uma solução ideal(embora caracterize um grande avanço).

Neste artigo, mostrarei como se criar um pequeno aplicativo desktop de cadastro usando o DB4Objects, que não somente é um framework que facilita o processo de persistência, como também é o database em si.

Espero que o artigo esteja do agrado de todos.

O que você já precisa saber:

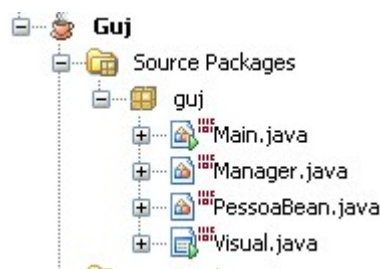
Para usufruir integralmente do conteúdo mostrado neste artigo, é importante que o leitor possua uma boa base de java 5.0, design patterns, e algum conhecimento sobre bancos de dados.

Conselhos!

Na confecção deste projeto, utilizei-me da IDE Netbeans 5.0 beta 2, e aconselho enfaticamente que o leitor deste artigo possua a mesma instalada. Não é um pré-requisito, entretanto, o Matisse será de grande valia na construção da GUI.

Vamos começar?

Como a proposta deste artigo é construir um aplicativo bastante simples, só teremos quatro classes envolvidas no processo, a nossa classe Manager, que irá fazer toda a validação, acesso e persistência dos nossos dados, uma classe Visual, que será a GUI com o usuário, o PessoaBean e a nossa classe Main, que irá inicializar o aplicativo. Seu projeto deve ficar parecido com este:



Também assegure-se de ter a biblioteca db4o-5.0 adicionada ao seu projeto. Ela pode ser conseguida no site do desenvolvedor: www.db4o.com.

Atente!

Em uma aplicação real, você teria que atentar para toda uma abordagem MVC e modelagem especial, garantindo a legibilidade do código e sua extensibilidade.

Construindo nossa aplicação

Nossa classe Manager deve possuir os seguintes métodos: open(String bank), que irá estabelecer a comunicação com o nosso banco de dados; close(), que irá fechar a conexão com o nosso banco de dados; get(PessoaBean pessoa), que será o bean com o qual iremos trabalhar, e del() que será responsável por deletar o nosso bean.

Eis o código do nosso Manager.java:

```
package guj;

import guj.bean.PessoaBean;
import java.util.List;
import java.util.ArrayList;

import com.db4o.Db4o;
import com.db4o.ObjectContainer;
import com.db4o.ext.ExtObjectContainer;
import com.db4o.ObjectSet;

/**
 *
 * @author Italo
 */
public class manager{
    private ExtObjectContainer db;
    private String location;

    /** Recebe como argumento a localização do banco. Uma
     * entrada válida seria 'db.yap'
     */
    public manager(String dbLocation) {
        location = dbLocation;
        open();
    }
    /** Busca uma pessoa no banco e deleta os resultados*/
    public void del( FuncionarioBean bean ){

        ObjectSet obj;
        obj = db.get(bean);
        while( obj.hasNext() )
            db.delete( obj.next() );
    }
    /** Retorna o nosso bean*/
    public List get( PessoaBean bean ){

        ObjectSet obj;
        obj = db.get(bean);
        System.out.println( "Tamanho do resultado: "+ obj.size() );

        List<PessoaBean> list = new ArrayList<PessoaBean>();

        while ( obj.hasNext() )
            list.add((PessoaBean)obj.next());

        return list;
    }
    /** É importante que o bean passado como argumento seja
     * copiado em outro bean, pois o db4objects mantém uma referência
     * aos beans nele adicionados.
     */
    public void store( FuncionarioBean bean ){
        PessoaBean pb = new PessoaBean();
        pb.setNome(bean.getNome());
        pb.setEnd(bean.getEnd());
        db.set(pb);
    }
}
/** Abre a conexão com o banco de dados
```

```
Note também que mesmo o db criando um ObjectContainer
o que realmente é usado pela aplicação é o ExtObjectContainer
que é uma instância do ObjectContainer só que mais completa e
apropriada para produção. */
public void open(){
    ObjectContainer db = Db4o.openFile(location);
    this.db = db.ext();
}

public void close(){
    if(!this.db.isClosed())
        db.close();
}
}
```

Quando open() é chamado, um arquivo igual ao argumento passado ao manager é criado no diretório raiz(onde está seu .jar). Este arquivo será onde nossos dados serão persistidos. Note que você não precisa informar ao banco a forma do seu bean, antes de passá-lo para persistência. Agora vejamos o nosso bean:

```
package guj;

/**
 *
 * @author Italo
 */
public class PessoaBean {

    private String nome;
    private String end;

    /** Creates a new instance of PessoaBean */
    public PessoaBean() {
    }

    public String getNome() {
        return nome;
    }

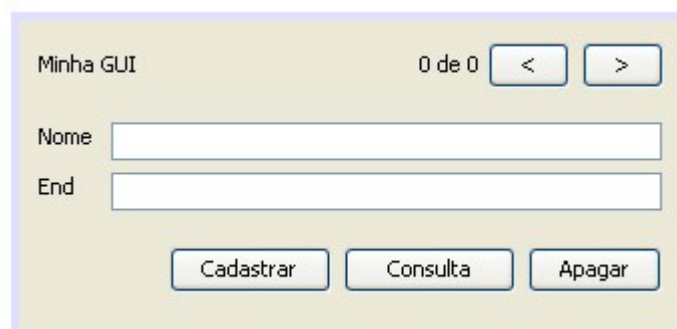
    public void setNome(String nome) {
        this.nome = nome;
    }

    public String getEnd() {
        return end;
    }

    public void setEnd(String end) {
        this.end = end;
    }
}
```

Note que nosso bean, também é extremamente simples, não precisando estender absolutamente nada. Com apenas estas duas classes, você acaba de criar a camada de persistência do seu projeto. Para finalizar, crie sua gui, contendo os campos do seu bean, e adeque o seu método main da classe Main para que ele inicie a gui e receba como argumento o seu manager.

Aqui está uma sugestão de GUI:



Minha GUI 0 de 0 < >

Nome

End

Cadastrar Consulta Apagar

Conclusão

Neste artigo foi apresentado como é fácil e rápido se construir um aplicativo com acesso a banco de dados utilizando o db4o, que junto a sua licença gpl o torna um grande atrativo para desenvolvedores que gostam de agilidade e poder em sua codificação. Em caso de dúvidas ou sugestões, mande-me um email, obrigado.

Italo Moreira Campelo Maia(italo.maia@gmail.com.br) é atualmente programador da Cagece, Ceará.