Versão de desenvolvimento
Quaisquer falhas favor reportar ao administrador do projeto, através do e-mail **tiagoscd@yahoo.com.br**
http://jthoth.svn.sourceforge.net/viewvc/jthoth/trunk/jthoth-0.0.1-r71.really.r70.jar?revision=71

Generic DAO usando JThoth com Hibernate Annotations

Este código comentado tem por objetivo auxiliar os desenvolvedores a implementar seus aplicativos usando o DAO (*Data Access Object*) genérico fornecido pelo JThoth Framework. Neste exemplo, será utilizada uma base de dados em um servidor MySQL. Apesar do código da base de dados e das anotações do Hibernate se encontrarem disponíveis, eles não serão comentados, pois o objetivo é demonstrar a utilização do *framework*.

Através do JThoth, é possível efetuar as operações básicas CRUD no banco de dados **SEM A NECESSIDADE DE DIGITAR QUAISQUER COMANDOS SQL**.

Banco.sql

```
CREATE DATABASE ex1;
USE ex1;

CREATE TABLE tb_atendentes (
      id INT(10) NOT NULL AUTO_INCREMENT PRIMARY KEY,
      nome VARCHAR(150)
) Engine = InnoDB;
```

hibernate.cfg.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD
3.0//EN" "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
<session-factory>
<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
<property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/ex1</property>
<property name="hibernate.connection.username">USUÁRIO</property>
<property name="hibernate.connection.password">SENHA</property>

<mapping class="Atendente" />
</session-factory>
</hibernate-configuration>
```

Atendente.java

```
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

import java.io.Serializable;

@Entity
@Table(name="tb_atendentes")
public class Atendente implements Serializable {
	private static final long serialVersionUID = 1L;

	@Id
	@GeneratedValue(strategy=GenerationType.AUTO)
	@Column(name="id")
	private int id;

	@Column(name="nome", length=150)
	private String nome;

	public Atendente() {
	}

	public int getId() {
		return id;
	}

	public void setId(int id) {
		this.id = id;
	}

	public String getNome() {
		return nome;
	}

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

AtendenteDAO.java

```
import java.util.List;

import org.jthoth.core.dao.GenericHibernateDAO;

public class AtendenteDAO extends GenericHibernateDAO<Atendente, Integer> {
	@Override
	public void save(Atendente atendente) {
		super.save(atendente);
	}

	@Override
	public void delete(Atendente atendente) {
		super.delete(atendente);
	}

	@Override
	public void update(Atendente atendente) {
		super.update(atendente);
	}

	@Override
	public Integer count() {
		Integer count = 0;

		count = super.count();

		return count;
	}

	@Override
	public Atendente findById(Integer id) {
		Atendente atendente = null;

		atendente = super.findById(id);

		return atendente;
	}

	@Override
	public List<Atendente> findAll() {
		List<Atendente> lista = null;

		lista = super.findAll();

		return lista;
	}
}
```

Main.java

```
import java.util.List;

public class Main {
	public Main() {
		AtendenteDAO dao = new AtendenteDAO();

		Atendente a1 = new Atendente();
		a1.setNome("Tiago");

		Atendente a2 = new Atendente();
		a2.setNome("Priscila");

		Atendente a3 = new Atendente();
		a3.setNome("João");

		dao.beginTransaction(); // inicia uma transação no banco de dados
		dao.save(a1); // salva o atendente a1
		dao.save(a2); // salva o atendente a2
		dao.commitTransaction(); // efetiva a transação no banco de dados

		dao.beginTransaction(); // inicia uma transação no banco de dados
		dao.save(a3); // salva o atendente a3
		dao.delete(dao.findById(1)); // exclui o atendente com id 1
		dao.rollbackTransaction(); // desfaz a transação no banco de dados

		dao.beginTransaction(); // inicia uma transação no banco de dados
		int total = dao.count(); // retorna a contagem de registros
		if (total > 0) {
			/* imprime o nome de todos os atendentes cadastrados */
			for (Atendente a : dao.findAll()) {
				System.out.println(a.getNome());
			}
		}
	}

	public static void main(String[] args) {
		new Main();
	}
}
```