Persistência

Prevalência de Objetos

Prof. João Eduardo Ferreira
Prof. Osvaldo Kotaro Takai

Última atualização: 09 de setembro de 2004

O que é Prevalência?

- A prevalência de objetos é um conceito que foi desenvolvido por Klaus Wuestefeld e amigos:
  - Todos os objetos são armazenados em memória.
  - Em intervalos regulares (e no momento de fechar a aplicação) os objetos são seriados para o disco; isso é chamado de snapshot.
  - Todas as manipulações de objetos são registrados num log de comandos, o qual é escrito imediatamente em disco.

Fonte: http://www.codeproject.com/cs/database/BambooMigrationKit.asp

Premissas

- A memória RAM é barata e avanços irão surgir:
  - É comum encontrar servidores com vários gigabytes de RAM.
  - RAM Holográfica: 1TB de dados e 1GB por segundo: http://computer.howstuffworks.com/holographic-memory.htm.
  - RAM Magnética: 600GB por polegada^2. MRAMs substituirão as DRAMs: http://computer.howstuffworks.com/mram.htm.
  - RAM de Polímeros: 400.000 CDs, 60.000 DVDs ou 126 anos de música MPG num chip: http://www.thinfilm.se/html/technology.htm.
  - Memória Molecular: http://www.theregister.co.uk/content/archive/7679.html.
  - 2300GB no PC Card: http://www.theregister.co.uk/content/archive/6099.html.

Fonte: http://www.prevayler.org/wiki.jsp?topic=BreakthroughsInMemoryTechnology

Premissas

- Trabalhar com bancos de dados relacionais não é fácil:
  - Persistir dados em sistema orientados a objetos é uma tarefa incrivelmente enfadonha.
  - Se a arquitetura da aplicação não for boa, o código fonte fica horrível, com vários comandos SQL no meio do código Java.
  - Abordagem relacional não se adequou ao projeto de software orientado a objetos; obrigando a transformação entre os modelos orientados a objetos e relacionais, tarefa que é propensa a erros e consome muito tempo.

Fontes: http://www.javafree.com.br/forum/viewtopic.php?t=3922, http://www.preclipse.de/web/index.html

Premissas

- Bancos de dados Orientados a Objetos têm baixo desempenho em aplicações com grande quantidade de dados.
  - Eles certamente requerem menor esforço do programadores em comparação aos bancos de dados relacionais.
  - A implementação tende a ser ainda muito complexa.
- Todos os dados dos objetos de negócio na maioria das aplicações cabem dentro da memória RAM.
- Escrever snapshot para o disco é rápido e dificilmente causa sobrecarga em I/O de disco.

Fonte: http://www.preclipse.de/web/index.html

Premissas

- O log de comandos em disco e os snapshots permitem a recuperação.
  - Se a aplicação cair, por exemplo, por falta de energia, então o snapshot e o log de comandos são usados durante a re-iniciação da aplicação.

O que é Prevayler?

- Prevayler é a implementação em Java do conceito de Prevalência.
- Sua primeira implementação foi disponibilizada em novembro de 2001 como um projeto open-source.
- Atualmente, o Prevayler está na versão 2.02.002: http://sourceforge.net/project/showfiles.php?group_id=36113.

Fonte: http://www.prevayler.org/wiki.jsp

Vantagens

- A prevalência de objetos é muito simples de usar e não sobrecarrega o programador.
- O custo da licenças de SGBDs supera o custo de adquirir memórias RAMs.
- Pode ser implementado em qualquer linguagem OO que serie objetos.
- É muito rápida comparada aos SGBDR e OO.
  - Consultas com Prevayler são:
    - 9.000 vezes mais rápidas do que no Oracle com JDBC e
    - 3.000 vezes mais rápidas do que no MySQL com JDBC.

Fonte: http://www.prevayler.org/wiki.jsp

Quando não usar a Prevalência

- Quando não houver possas que saibam programar.
- Quando não houver quantidade suficiente de memória RAM para conter todos os Objetos de Negócio.
- Quando houver uma restrição forçando o uso de um banco de dados existente.

Questões em discussão

- O que acontece com o garbage collector em sistemas de 64 bits com vários gigabytes?
- É possível usar o Prevayler em aplicações móveis ou em plataformas com restrições de memória?
- Existe prejuízo de desempenho na manipulação de grandes objetos de dados quando o heap ficar severamente fragmentado?
- Não existem ferramentas OLAP OO, ou seja, que rodem em objetos ao invés de tabelas.
- Evolução de esquemas de objetos.

Prevayler

Como funciona?

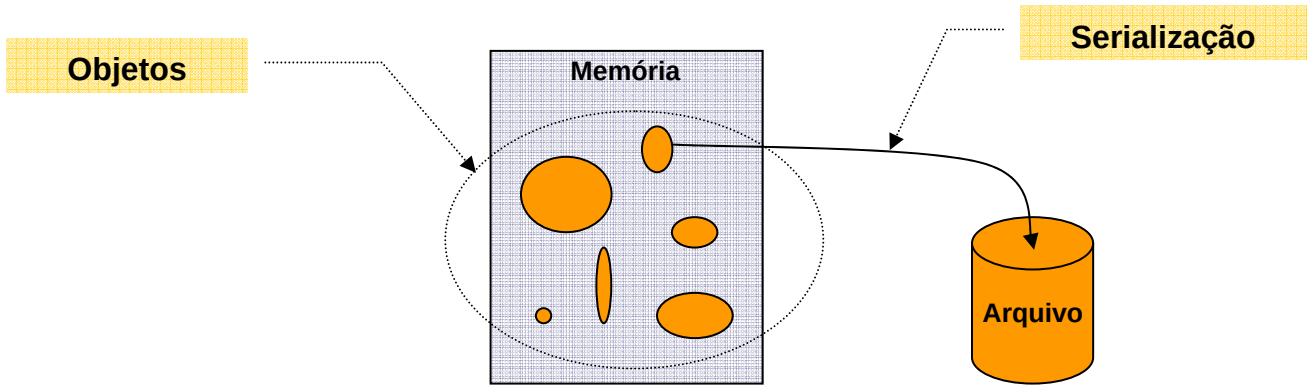
Como funciona?

- Em intervalos regulares (e no momento de fechar a aplicação) todos os objetos em memória são **seriados** para o disco; isso é chamado de **snapshot**.

- Todas as transações em objetos são **seriados** imediatamente em disco.

Serialização de Objetos

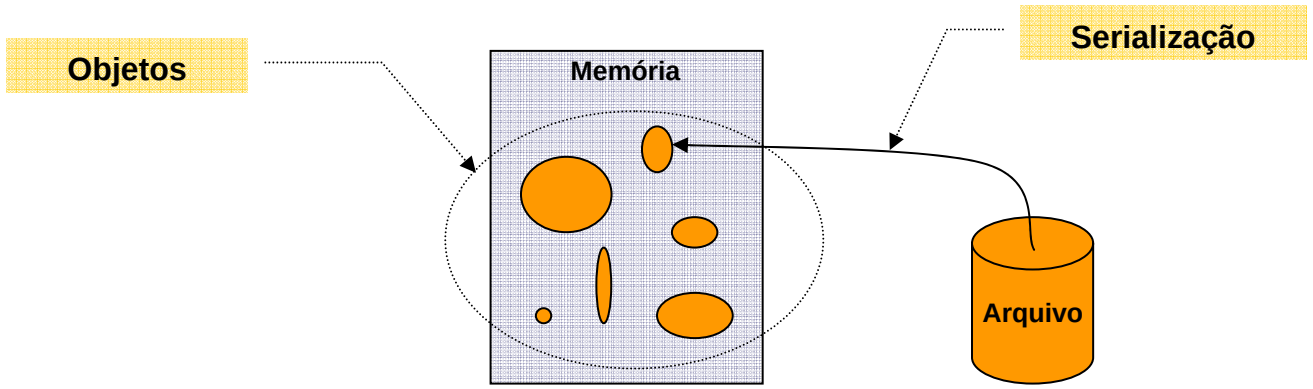
- É o processo de escrever o estado (dados) de um objeto para uma seqüência de bytes.
- A seriação é útil para armazenar o estado de um objeto num repositório persistente, por exemplo arquivo.



http://www.javacaps.com/java_serial.html

Desserialização de Objetos

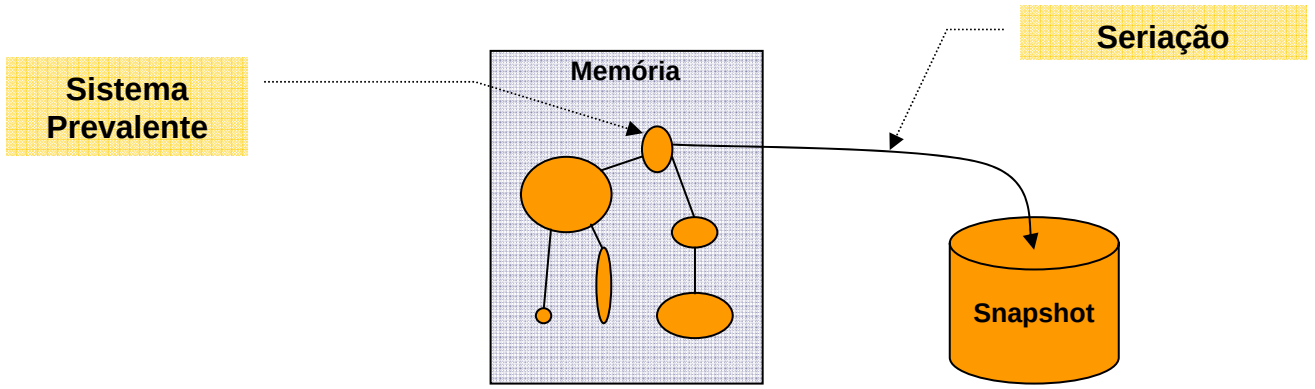
- Uma vez que o estado de um objeto esteja armazenado num arquivo, pode-se usar o processo de “des-seriação” para restaurar o estado do objeto em memória.



http://www.javacaps.com/java_serial.html

Snapshot

- O Snapshot é a cópia em disco dos objetos seriados.
- Em alguns momentos, o Snapshot é criado seriando-se o **objeto raiz** (chamado Sistema Prevalente), fazendo com que todos os objetos que o compõem também sejam seriados.

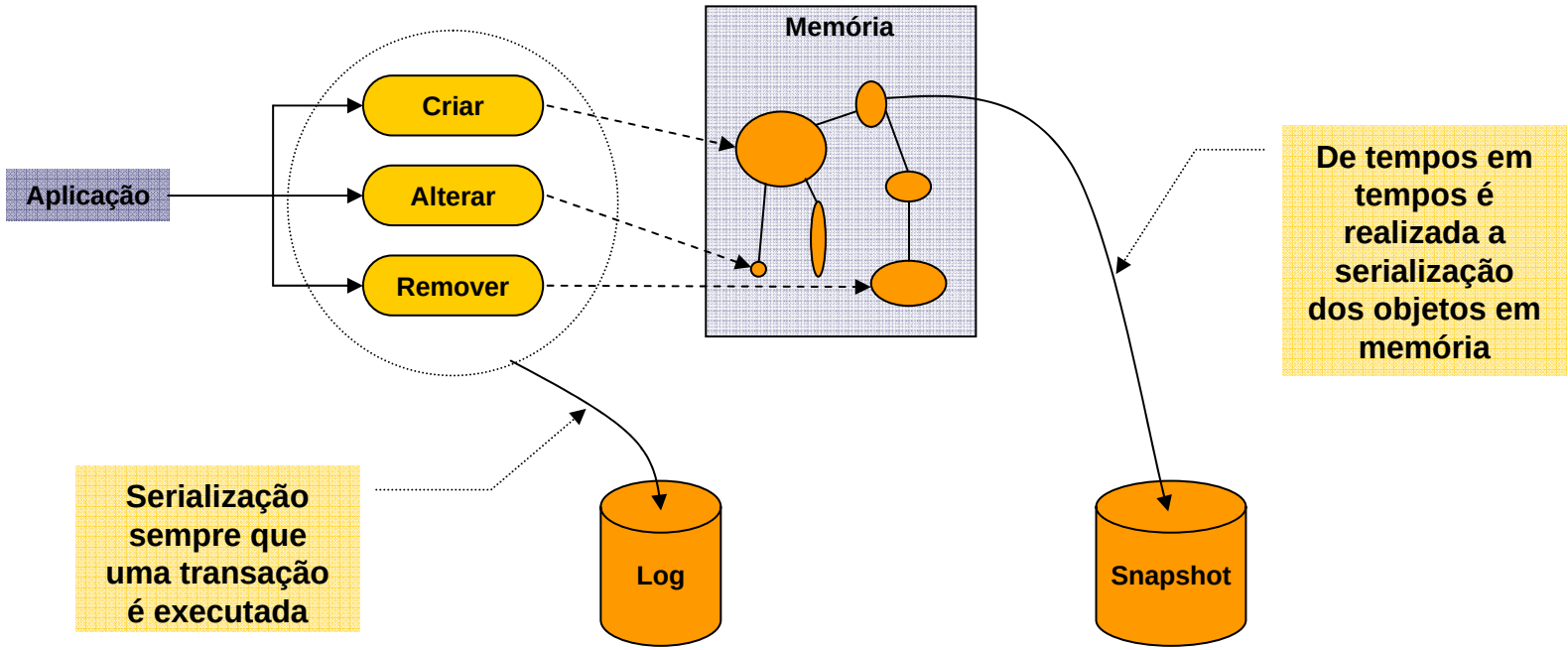


Transações Serializadas

- As transações seriadas são operações realizadas nos objetos em memória e que são armazenadas em arquivos **log** assim que a operação for executada.
- Quando uma aplicação é reiniciada, as seguintes atividades acontecem:
  - “Des-seriação” do último Snapshot (se houver).
  - “Des-seriação” e execução das transações seriadas após o último Snapshot.
- As transações podem ser de:
  - Criação,
  - Alteração e
  - Remoção.

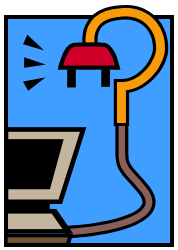
Transações Seriadas

- Aplicação executada pela primeira vez usando o Prevayler:



Transações Seriadas

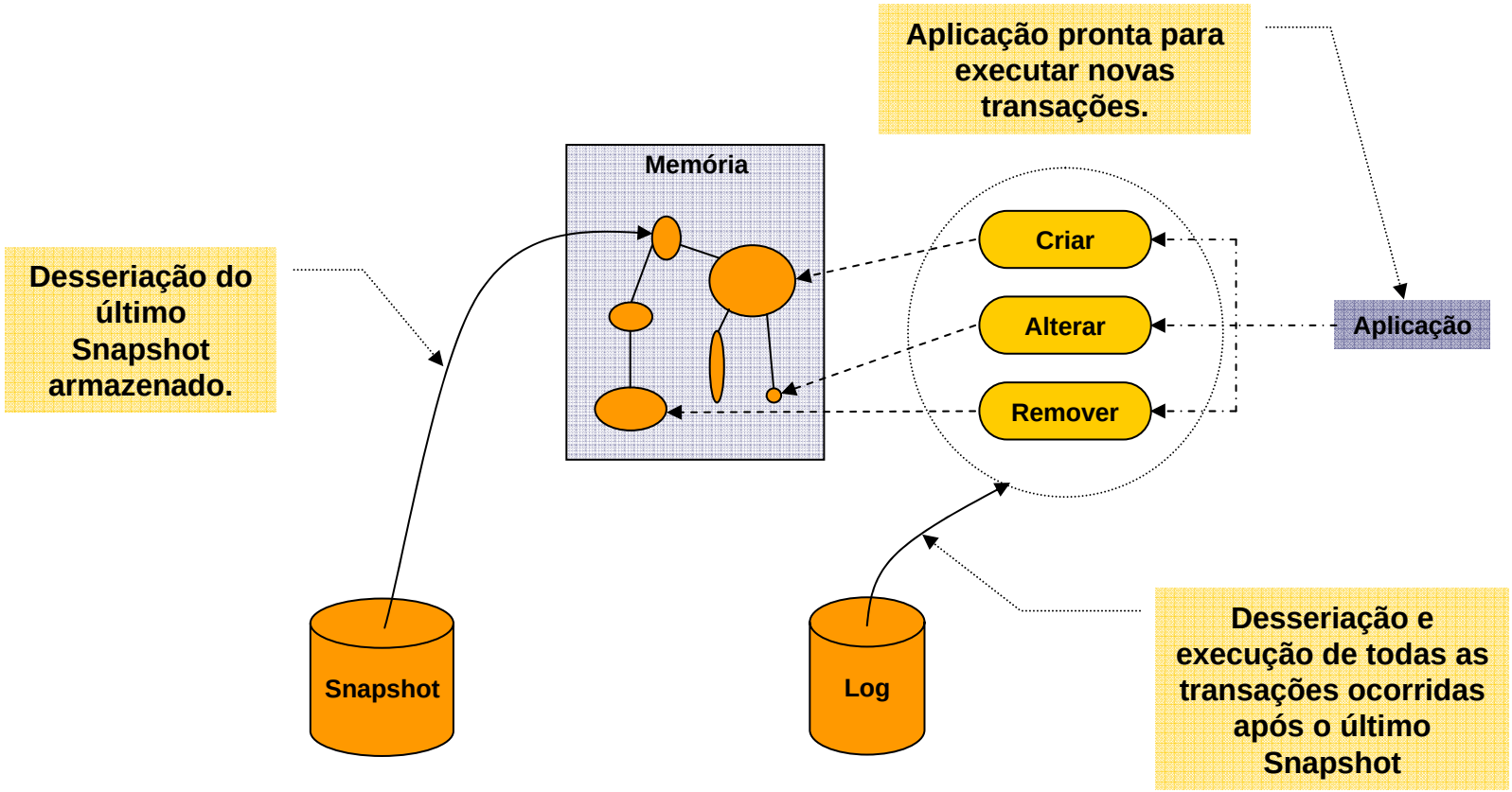
- Imagine que após algum tempo, alguém desligue o computador da aplicação.



- O Prevayler garante que nada será perdido!

Transações Seriadas

- Quando a aplicação é reiniciada usando o Prevayler, os objetos voltam para a memória em seu estado original:



Prevayler

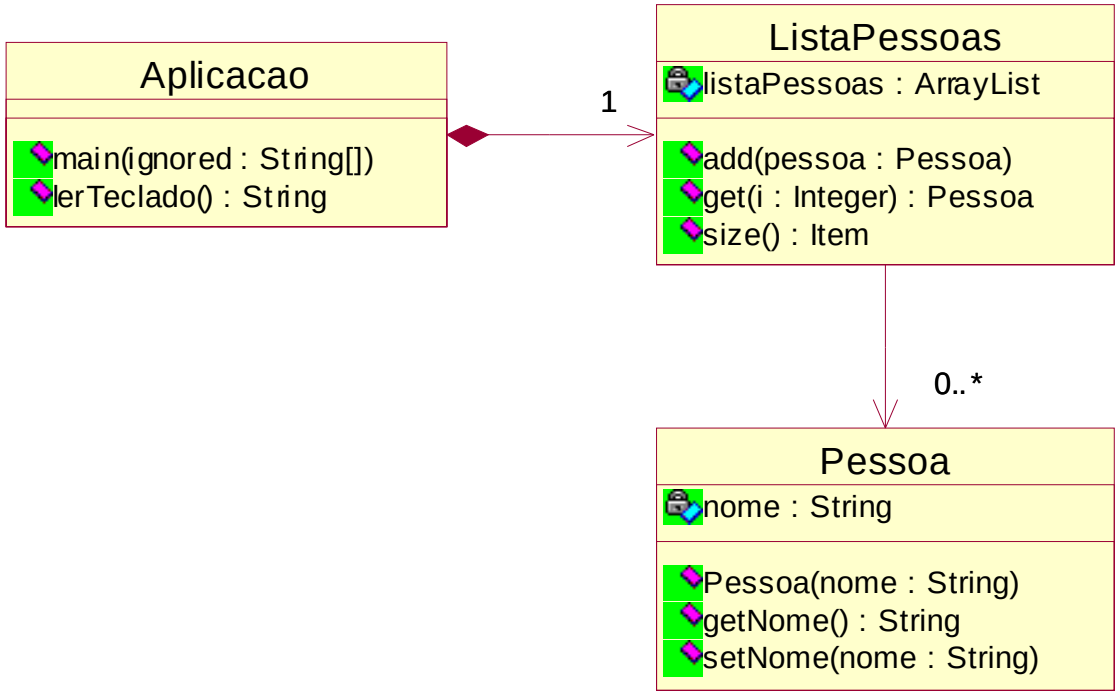
Como usar o Prevayler?

Recursos Utilizados

- Neste tutorial, será utilizado os seguintes recursos:
  - Prevayler: http://sourceforge.net/projects/prevayler.
  - Eclipse: http://www.eclipse.org
  - Plug-in do Eclipse para o Prevayler: http://www.preclipse.de.

Aplicação Exemplo 1

- Será utilizado um exemplo adaptado daquele apresentado no tutorial do **Vitor Fernando Pamplona** cujo objetivo é o de criar uma lista de pessoas:



Aplicação Exemplo 1

- A implementação das classes anteriores em Java sem o Prevayler é:

```
package exemplo1;
import java.util.ArrayList;

public class ListaPessoas {
	private ArrayList listaPessoas = new ArrayList();

	public void add(Pessoa pesssoa) {
		listaPessoas.add(pessoa);
	}

	public Pessoa get(int i) {
		return (Pessoa) listaPessoas.get(i);
	}

	public int size() {
		return listaPessoas.size();
	}
}
```

```
package exemplo1;

public class Pessoa {
	private String nome;

	public Pessoa(String nome) {
		this.nome = nome;
	}

	public String getNome() {
		return nome;
	}

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

Aplicação Exemplo 1

```
public class Main {
	public static void main(String[] ignored) throws Exception {
		ListaPessoas lista = new ListaPessoas();

		System.out.print("Digite o nome da pessoa ou FIM para sair: ");
		String nome = lerTeclado();
		while (!nome.equals("FIM")) {
			lista.add(new Pessoa(nome));
			System.out.println("Pessoa armazenada.\n");
			System.out.print("Digite o nome da pessoa ou FIM para sair: ");
			nome = lerTeclado();
		}
		System.out.println("\n Imprimindo lista de pessoas.\n");
		for (int i = 0; i < lista.size(); i++) {
			System.out.println(lista.get(i).getNome());
		}
	}

	public static String lerTeclado() {
		BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
		try { return reader.readLine(); } catch (IOException e1) { return "FIM"; }
	}
}
```

Aplicação Exemplo 1

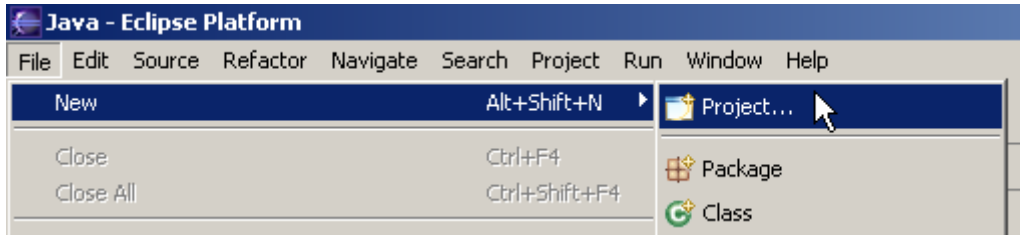
- Este exemplo, embora funcione, não é muito útil pois a lista de pessoas criada não pode ser recuperada após a finalização do programa.
- Para que isso seja possível utilizando o Prevayler, definimos:
  - Sistema Prevalente: ListaPessoas.
  - Classes Seriadas: ListaPessoas e Pessoa.
  - Transação de Criação: PessoaCreateTransaction (uma nova classe).
- Para facilitar o desenvolvimento, o Plug-in declipse será usado.

Passo a Passo

Preclipse

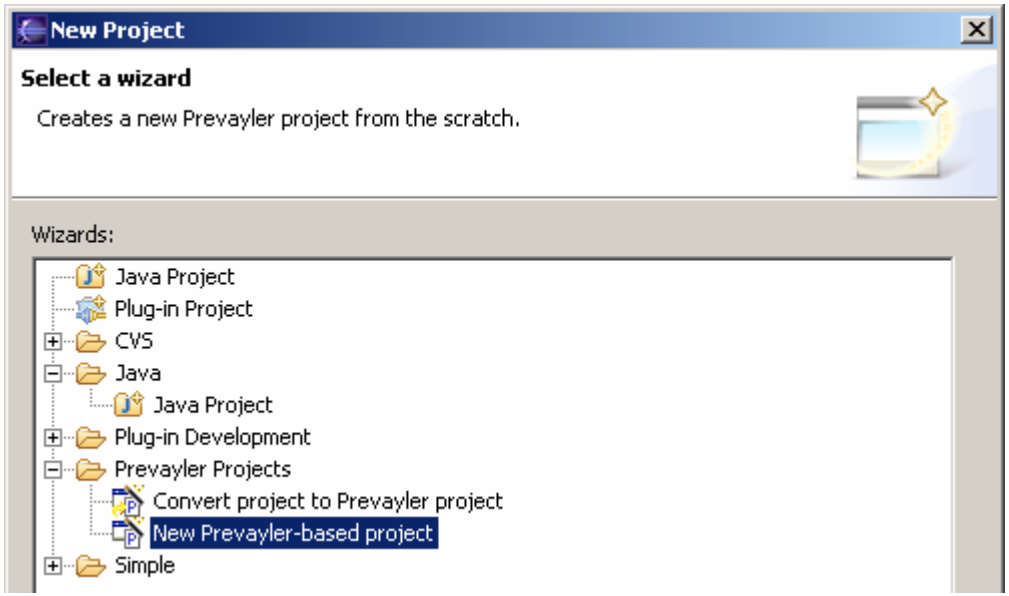
Criando o Projeto

- No Eclipse, com o Plug-in Preclipse devidamente instalado, crie um novo projeto.



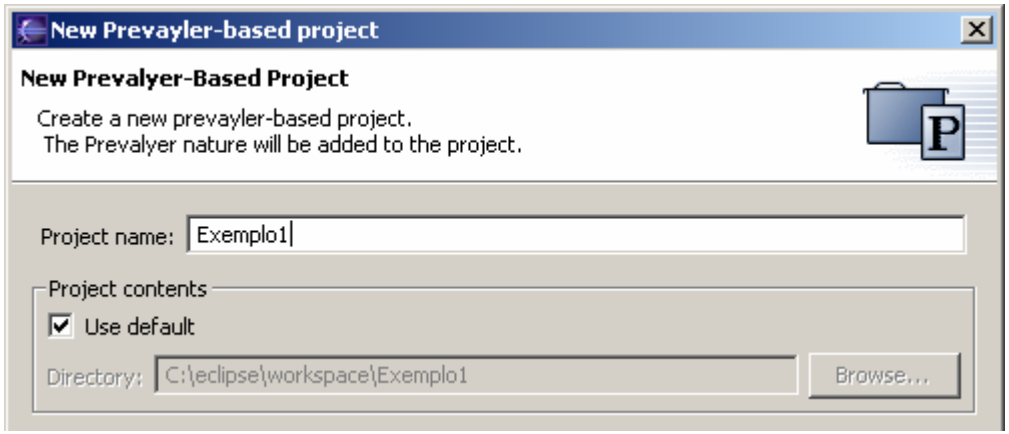
Criando o Projeto

- Selecione New Prevayler-based project e pressione Next.



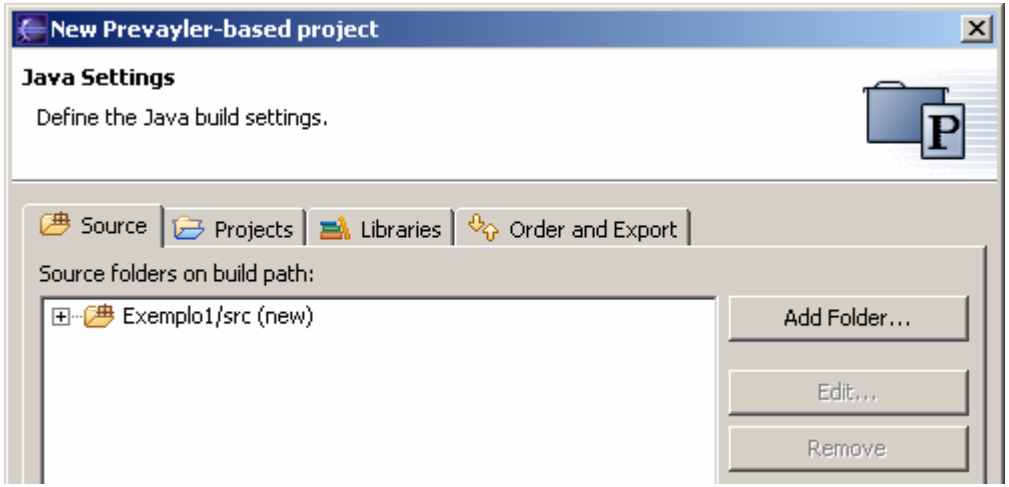
Criando o Projeto

- Digite o nome do projeto; no caso Exemplo1 e pressione Next.



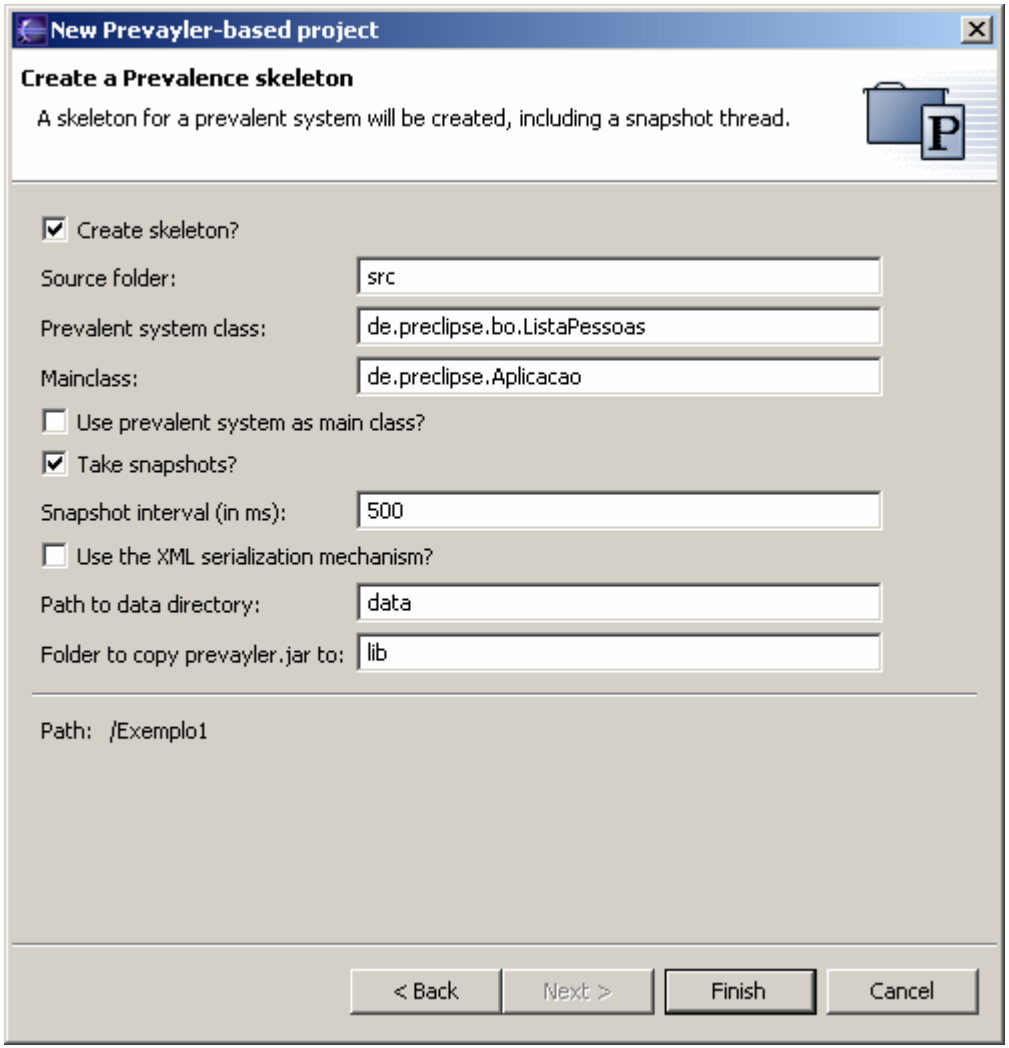
Criando o Projeto

- Nada a fazer nesta tela. Pressione Next.



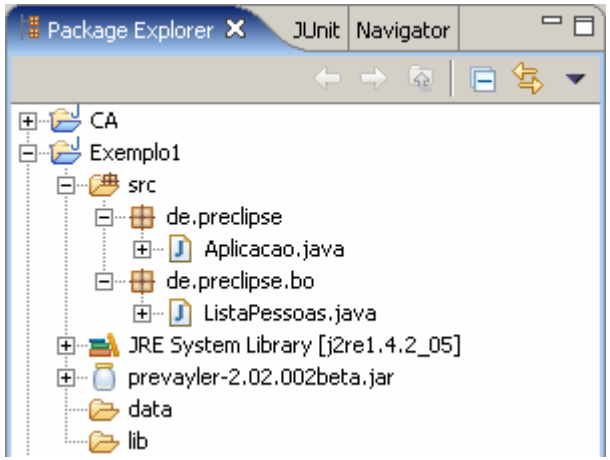
Criando o Projeto

- Na tela apresentada:
  - Selecione a primeira caixa de seleção.
  - No segundo campo digite:
    - **“de.preclipse.bo.ListaPessoas”.**
  - No terceiro campo digite:
    - **“de.preclipse.Aplicacao”**.
  - Manter o restante como está.
  - Pressione Finish.



Criando o Projeto

- O projeto Exemplo1 deverá ter sido criado no Package Explorer:

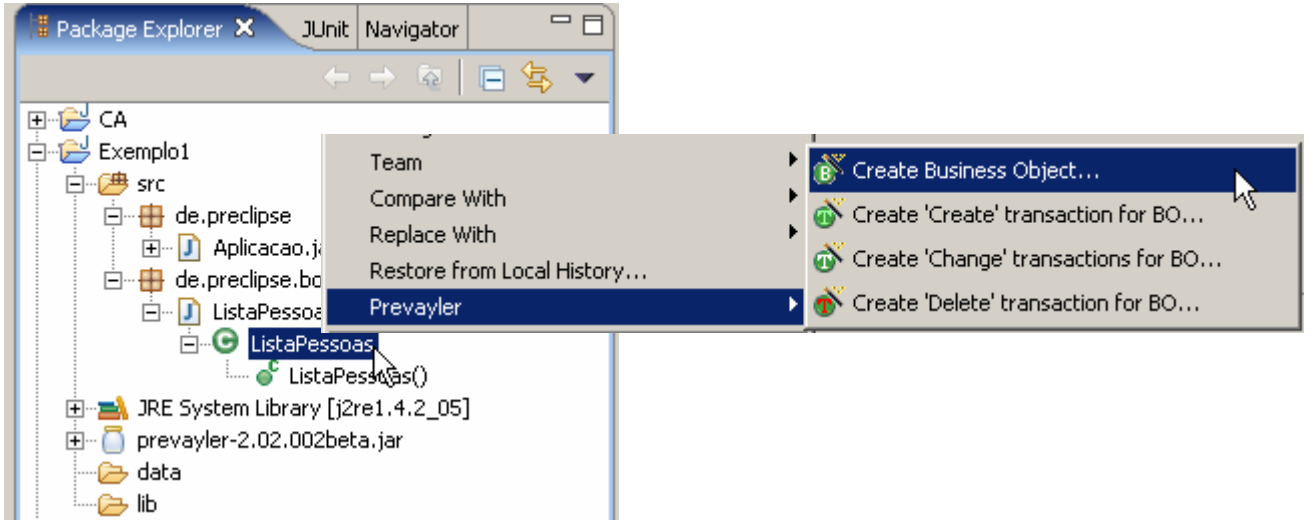


Criando o Projeto

- O Preclipse gerou dois arquivos:
  - **Aplicacao.java**: Este arquivo contém o método main() o qual:
    - Utiliza o padrão de projeto Factory para criar objetos necessários para o Prevayler funcionar, indicando que a classe **ListaPessoas** é o objeto raíz.
    - Insere o código para o Thread que gera um snapshot em intervalos de 500 ms.
  - **ListaPessoas.java**: Contém apenas a estrutura de uma classe serializável para que seja preenchida.

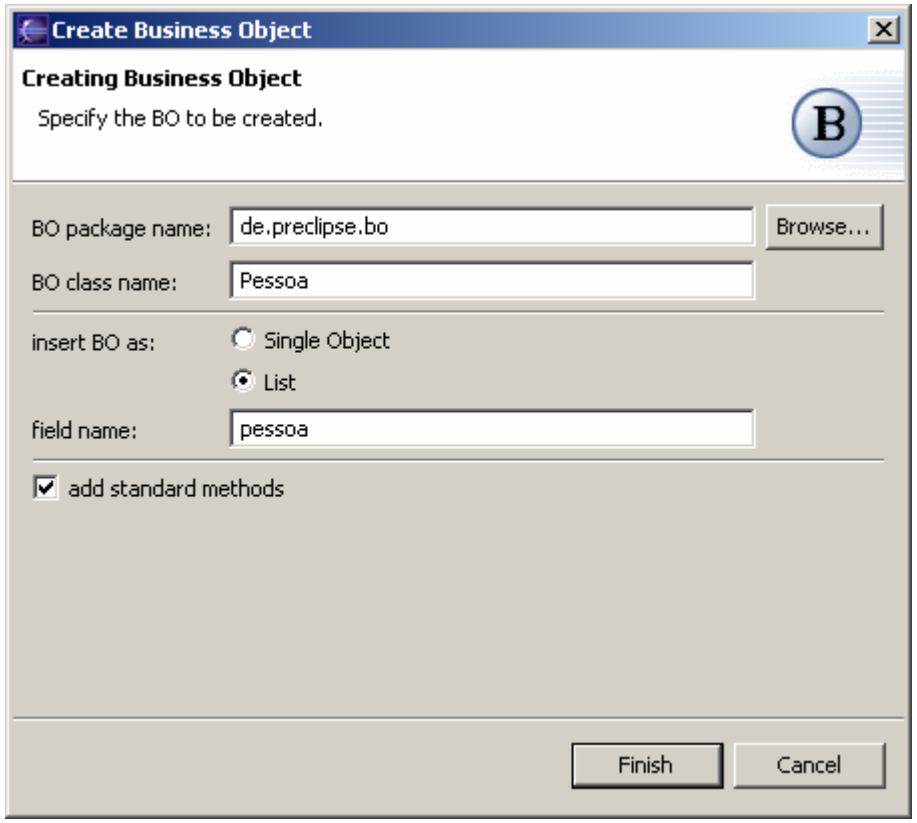
Criando a Classe Pessoa

- A idéia é criar a classe Pessoa como sendo uma classe que compõe a classe ListaPessoas.
- Assim, com o botão direito do mouse sobre a classe ListaPessoas, selecione no menu pop-up a opção “Prevayler/Create Business Object...”



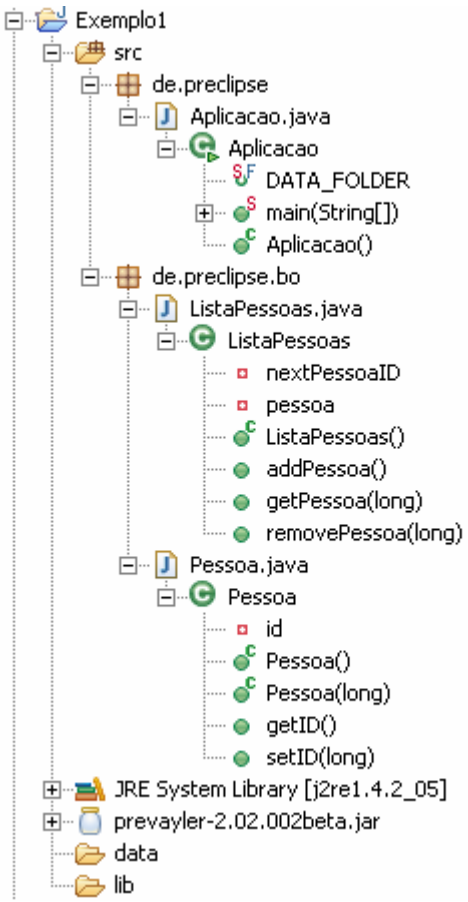
Criando a Classe Pessoa

- No campo “BO class name” digite:
  - Pessoa (com **P** maiúsculo).
- Seleciona a opção List:
  - Isso permite que a classe ListaPessoas mantenha uma lista de objetos Pessoa.
- No campo “field name:” digite:
  - pessoa (com **p** minúsculo)



Criando a Classe Pessoa

- Verifique que em todas as classe criadas pelo preclipse incluiram alguns atribuitos e/ou métodos.



Criando a Classe Pessoa

- Note que a classe Pessoa:
  - Foi criada no pacote que contém os objetos de negócio (BO - Business Objects).
  - Foi criada como sendo uma classe seriável.
  - Contém um atributo ID e métodos get/set correspondentes, necessários para que o sistema de prevalência possa funcionar.
- Note que a classe ListaPessoas também foi modificada:
  - Contém um vetor (arrayList) de pessoas.
  - Um método para adicionar pessoas na lista (addPessoa).
  - Um método para obter pessoas da lista (getPessoa).
  - Um método para remover pessoas da lista (removePessoa).

Modificando a Classe ListaPessoa

- Agora, modifique a classe ListaPessoas para que contenha todos os elementos do projeto original:
  - Acrescente à classe ListaPessoas o seguinte método:

```
public int size() {
    return pessoa.size();
}
```

  - Renomeie
    - addPessoa() para add().
    - getPessoa() para get() e
    - removePessoa para remove().

Modificando a Classe Pessoa

- Modifique agora a classe Pessoa para que contenha todos os elementos do projeto original:
  - Acrescente à classe Pessoa o seguinte atributo:

```
private String nome;
```

  - Acrescente os seguintes métodos:

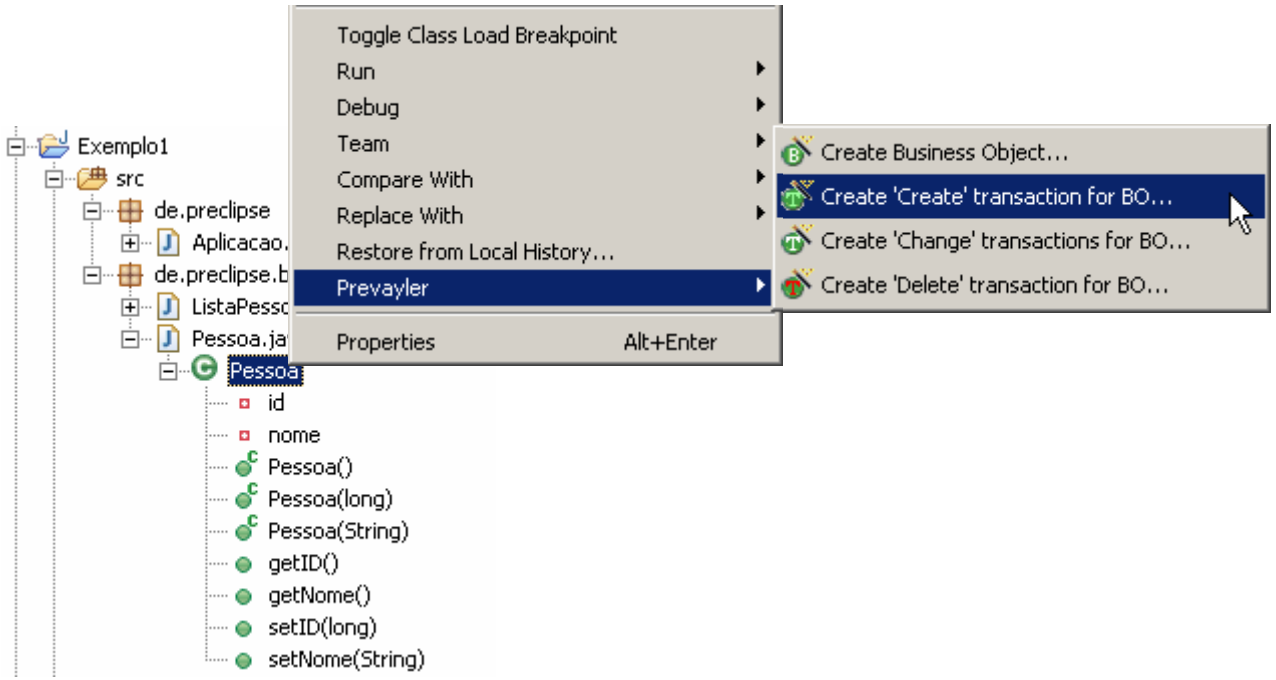
```
public Pessoa(String nome) {
        this.nome = nome;
}
public String getNome() {
        return nome;
}
public void setNome(String nome) {
        this.nome = nome;
}
```

Criando Transações

- Iremos adicionar transações para criar, alterar e remover pessoas da lista de pessoas.
- As transações são necessárias sempre que alterações ocorrerem nos objetos de negócio, tal que o Prevayler possa armazenar tais mudanças no disco e restaurá-las após alguma falha ou reinício da aplicação.
- Note que você não dev e alterar os dados dos objetos de negócio fora de uma transação!
- Nós iremos criar uma transação “create” para Pessoa.

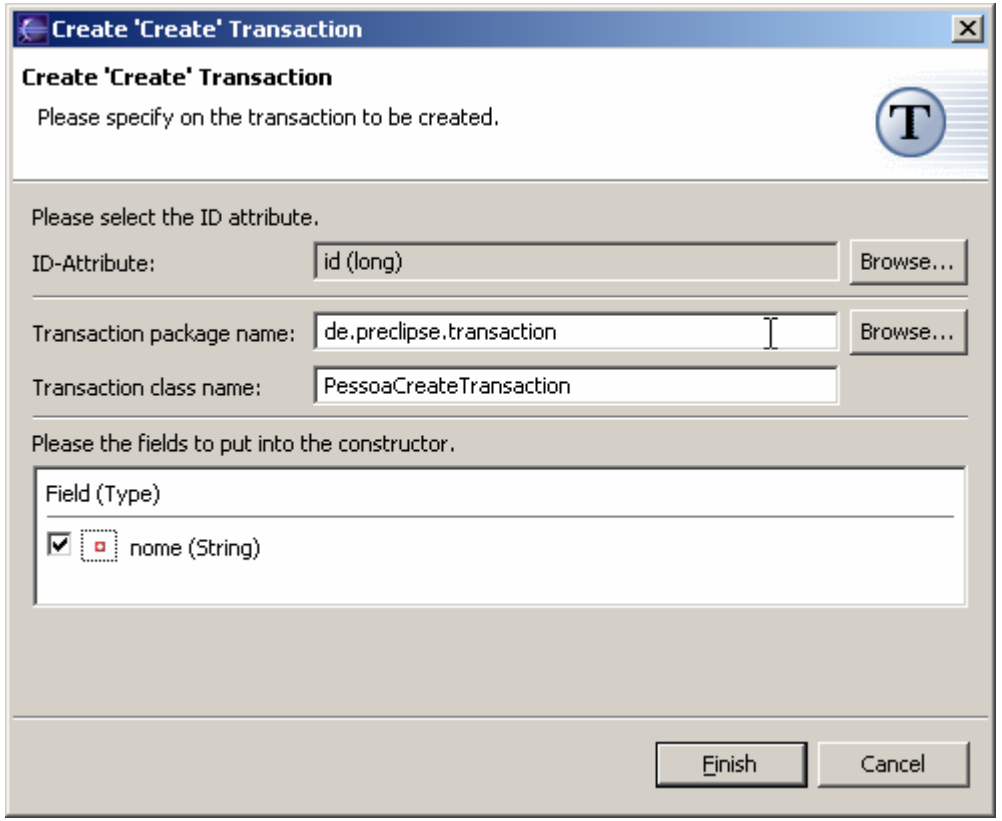
Criando Transações

- Com o botão direito do mouse sobre a classe Pessoa, selecione “Create ‘Create’ transactions for BO...”



Criando Transações

- A caixa de diálogo ao lado irá surgir.
- Desta vez coloque a nova classe:
  - PessoaCreateTransaction
- num novo pacote:
  - de.preclipse.transaction
- como ilustrado na figura ao lado.
- Pressione o botão Finish.



Criando Transações

- Note que, nesta caixa de diálogo, você precisa fornecer um campo contendo o atributo ID. Preclipse irá excluí-lo da lista de campos da nova classe gerada.
- Modifique o método **executeAndQuery** da nova classe para:

```
public Object executeAndQuery(Object prevalentSystem, Date executionTime) throws Exception {
        Pessoa newPessoa = ((ListaPessoas) prevalentSystem).add();
        newPessoa.setNome(nome);
        return newPessoa;
}
```

- Isso fará com que um novo objeto da classe Pessoa seja criado no objeto raíz da classe ListaPessoas.

Modificando a Classe Aplicacao

- Agora, modifique a classe Aplicacao para que contenha todos os elementos do projeto original:
  - Acrescente o método na classe Aplicacao:

```
public static String lerTeclado() {
        BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
        try { return reader.readLine(); } catch (IOException e1) { return "FIM"; }
}
```

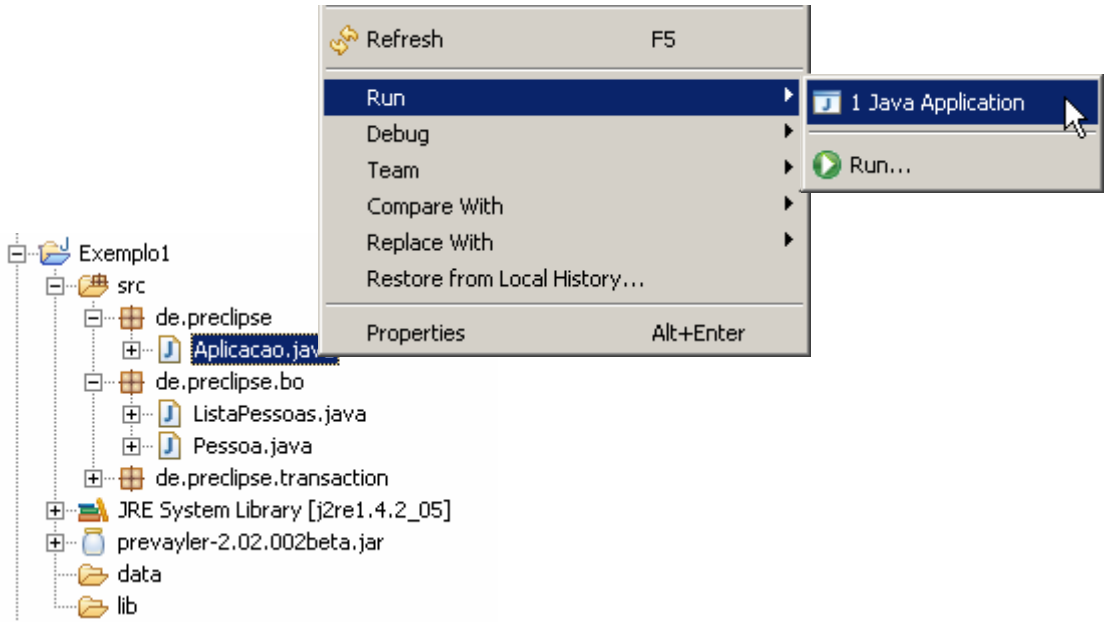
Modificando a Classe Aplicacao

- Inclua no método main, após o comando: snapShotThread.start(), o seguinte código:

```
System.out.print("Digite o nome da pessoa ou FIM para sair: ");
String nome = lerTeclado();
while (!nome.equals("FIM")) {
	try {
		prevayler.execute(new PessoaCreateTransaction(nome));
	} catch (Exception e1) {
		e1.printStackTrace();
	}
	System.out.println("Pessoa armazenada.\n");
	System.out.print("Digite o nome da pessoa ou FIM para sair: ");
	nome = lerTeclado();
}
System.out.println("\n Imprimindo pessoas persistidas.\n");
ListaPessoas lista = ((ListaPessoas) prevayler.prevalentSystem());
for (int i = 1; i <= lista.size(); i++) {
	System.out.println(lista.get(i).getNome());
}
```

Executando a Aplicação

- Clique com o botão direito do mouse sobre a Aplicacao.java e selecione
  - Run / 1 Java Application.



Executando a Aplicação

- Entre com alguns nomes. Por exemplo:
  - João <Enter>
  - Maria <Enter>
  - José <Enter>
  - FIM <Enter>
- Re-execute a aplicação e digite:
  - FIM <Enter>
- Você irá observar que os nomes das pessoas que você digitou será exibido, demonstrando que eles foram persistidos.

FIM