

# Programação com Objectos/Projecto de Programação com Objectos/Enunciado do Projecto de 2020-2021

From Wiki\*\*3

< Programação com Objectos | Projecto de Programação com Objectos

AVISOS - Avaliação em Época Normal	[Expand]
------------------------------------	----------

Material de Uso Obrigatório	[Expand]
-----------------------------	----------

--

## ÉPOCA NORMAL

O objectivo do projecto é desenvolver uma aplicação de gestão para uma empresa de distribuição. A empresa concede prémios de fidelização aos bons clientes (baseando-se no volume de compras). A funcionalidade da aplicação inclui, entre outras acções: registar/manipular dados de produtos para venda, registar/manipular dados de clientes, registar/manipular dados de fornecedores de produtos para venda, registar/manipular transacções de venda e encomenda e fazer pesquisas várias sobre a informação armazenada.

Neste texto, o tipo **negrito** indica um literal (i.e., é exactamente como apresentado); o símbolo \_ indica um espaço; e o tipo *itálico* indica uma parte variável.

## Empresa, Produtos, Clientes, Fornecedores e Transacções

Os produtos, clientes e fornecedores possuem uma chave única (cadeia de caracteres, não havendo distinção entre maiúsculas e minúsculas). As transacções (vendas e encomendas) possuem uma chave única inteira.

O saldo inicial da empresa é 0 (zero).

## Propriedades e Funcionalidade dos Produtos

Todos os produtos têm um fornecedor, um preço (um número inteiro positivo), um valor crítico (utilizado na gestão de existências) e o valor das existências.

A empresa vende caixas e contentores de vários tipos. Outros tipos de produtos diversos para venda são disponibilizados à empresa por fornecedores.

As caixas são produtos que têm um volume relativamente pequeno e servem para guardar um número relativamente pequeno de itens, por forma a protegê-los durante o seu transporte. As caixas têm um dado nível de serviço: normal, aéreo, expresso e em mão.

Os contentores são produtos que têm um volume muito maior que as caixas, permitindo guardar uma maior quantidade de itens. Têm os mesmos 4 níveis de serviço das caixas. Adicionalmente, os contentores têm 4 níveis de qualidade de serviço: B4, C4, C5, DL.

Cada livro é caracterizado pelo respectivo título, autor e ISBN (cadeias de caracteres).

Poderão ser definidos novos tipos de produtos (e.g. discos ou CDs/DVDs), que terão propriedades específicas.

## Propriedades e Funcionalidade dos Clientes

Os clientes fazem compras que podem ser pagas mais tarde.

Cada cliente tem um nome (cadeia de caracteres) e uma morada (cadeia de caracteres). Associada a cada cliente existe ainda informação relativa às suas compras.

Um cliente tem ainda um estatuto (e.g., cliente "Elite" -- ver abaixo), o qual tem algum impacto na sua relação com a empresa.

Dado um cliente, é possível aceder ao historial das suas transacções.

## Propriedades e Funcionalidade dos Fornecedores

Os fornecedores respondem a encomendas por parte da empresa.

Cada fornecedor tem um nome (cadeia de caracteres) e uma morada (cadeia de caracteres).

Dado um fornecedor, é possível aceder ao seu historial de transacções. É ainda possível activar/desactivar a realização de transacções com um fornecedor. Um fornecedor inactivo não pode realizar transacções.

## Propriedades e Funcionalidade das Transacções

Existem dois tipos de transacção: vendas e encomendas. As transacções são identificadas por um número (inteiro), atribuído de forma automática pela aplicação. Este identificador começa em 0 (zero), sendo incrementado quando se regista uma nova transacção. A sequência de identificadores é partilhada por todas as transacções (vendas e encomendas).

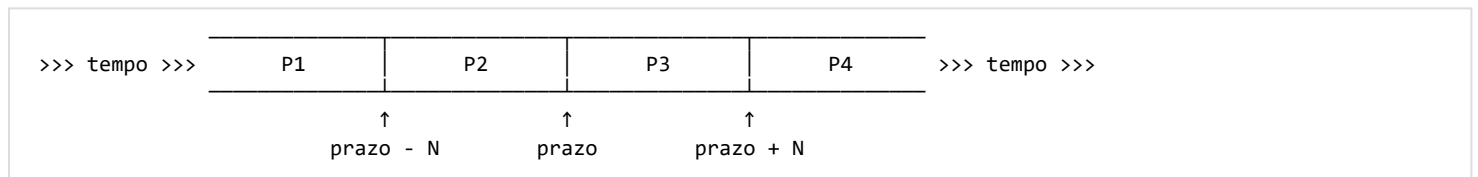
Uma encomenda está associada a um fornecedor e envolve uma ou mais unidades de um ou mais produtos fornecidos pelo fornecedor em causa. O custo unitário de cada produto corresponde ao custo do produto em causa. A encomenda guarda o seu custo total, para que futuras alterações no preço de um produto não alterem o valor que foi pago pela encomenda. Quando se faz uma encomenda deve considerar-se que a encomenda foi paga imediatamente e que as existências da empresa foram actualizadas considerando os produtos encomendados.

Uma venda envolve uma ou mais unidades de um único produto. Cada venda tem uma data limite de pagamento: primeiro realiza-se a venda e só depois é que se procede ao seu pagamento.

O preço a pagar por um cliente depende do tempo que demora a realizar o pagamento. São considerados os seguintes períodos (**N** é 5 para caixas, 8 para contentores e 3 para livros):

- **P1** - até **N** dias antes do limite de pagamento ( $\text{data\_limite\_de\_pagamento} - \text{data\_actual} \geq N$ ).
- **P2** - até à data limite ( $0 \leq \text{data\_limite\_de\_pagamento} - \text{data\_actual} < N$ ).
- **P3** - até **N** dias depois da data limite ( $0 < \text{data\_actual} - \text{data\_limite\_de\_pagamento} \leq N$ ).
- **P4** - após **N** dias depois da data limite ( $\text{data\_actual} - \text{data\_limite\_de\_pagamento} > N$ ).

Intervalos ao longo do tempo:



## Notificações

Quando se regista um novo produto, os clientes devem ser colocados como entidades interessadas em receber notificações sobre eventos a ele associados. Em qualquer momento, um cliente pode activar ou desactivar as notificações relativas a um produto. Os eventos a considerar são os seguintes: (i) quando o produto passa de stock 0 (zero) para outro valor (positivo); (ii) quando um produto fica mais barato. As notificações são compostas pelo identificador do produto e pela descrição da notificação: **NEW**, para novas existências de produtos, mas não quando se regista um novo produto; e **BARGAIN**, para descidas de preços. As notificações são registadas nos clientes que as recebem.

A entrega de notificações deve ser flexível e prever vários meios de entrega, e.g., correio postal, SMS, email, entre outras. O meio de entrega por omissão corresponde a registar a notificação na aplicação.

## Contabilização de Pontos (Clientes)

As multas e os descontos aplicam-se apenas no pagamento de transacções de clientes (vendas).

Existem três classificações distintas de clientes: **Nomal**, **Selection** e **Elite**. A classificação de um cliente tem impacto nas multas, descontos e prazos de pagamento a aplicar no pagamento de uma transacção (venda).

Quando um cliente paga uma transacção dentro do prazo, acumula um número de pontos correspondente a 10 vezes o valor pago, se o pagamento for realizado dentro do prazo. Não há contabilização de pontos em pagamentos atrasados. A verificação do atraso é realizada quando se realiza o pagamento de uma transacção.

Os clientes passam ao nível **Selection** se acumularem mais de 2000 pontos. Os clientes passam ao nível **Elite** se acumularem mais de 25000 pontos.

Se um cliente se atrasa no pagamento da transacções, é despromovido: um cliente **Elite** passa a **Selection** se o pagamento ocorrer com um atraso de pagamento superior a 15 dias (perde 75% dos pontos acumulados); um cliente **Selection** passa a **Normal** se o pagamento ocorrer com um atraso de pagamento superior a 2 dias (perde 90% dos pontos acumulados).

As multas e os descontos dependem do estatuto do cliente e dos prazos associados à venda e ao estatuto.

	P1		P2		P3		P4	
	Multa	Desconto	Multa	Desconto	Multa	Desconto	Multa	Desconto
Normal	0	10%	0	0	5% diários	0	10% diários	0
Selection	0	10%	0	≥ 2 dias antes da data limite: 5%; depois, sem desconto	> 1 dia depois da data limite: 2% diários (0, caso contrário)	0	5% diários	0
Elite	0	10%	0	10%	0	5%	0	0

## Data

A data é representada por um número inteiro e tem inicialmente o valor 0 (zero). A data pode começar com outro valor se se recuperar o estado da empresa a partir de um suporte persistente.

Os avanços de data são valores inteiros que representam o número de dias.


## Funcionalidade da aplicação


A aplicação permite manter informação sobre as entidades do modelo. Possui ainda a capacidade de preservar o seu estado (não é possível manter várias versões do estado da aplicação em simultâneo).

A base de dados com os conceitos pré-definidos é carregada no início da aplicação.

É possível saber os saldos da empresa (diferencial entre vendas e compras). Existe um saldo disponível, correspondente à diferença entre as vendas realmente pagas e as encomendas, e um saldo contabilístico, correspondente à diferença entre o valor contabilístico das vendas (pagas ou não e considerando descontos/penalizações à data da consulta de saldo) e as encomendas.

Deve ser possível efectuar pesquisas sujeitas a vários critérios e sobre as diferentes entidades geridas pela empresa.

- 

Note-se que não é necessário/desejável implementar de raiz a aplicação: já existem classes que representam e definem a interface geral da funcionalidade do *core* da aplicação, tal como é visível pelos comandos da aplicação.
- 

A interface geral do *core* já está parcialmente implementada na classe **woo.Storefront** e outras fornecidas (cujos nomes devem ser mantidos), devendo ser adaptadas onde necessário. É ainda necessário criar e implementar as restantes classes que suportam a operação da aplicação.

## Serialização

É possível guardar e recuperar o estado actual da aplicação, preservando toda a informação relevante, descrita acima.

## Funcionalidade Associada a Entidades do Domínio

A seguinte funcionalidade sobre produtos deve ser suportada pela aplicação: (i) visualizar um produto; (ii) registar um novo produto; (iii) alterar o preço de um produto.

A seguinte funcionalidade sobre clientes deve ser suportada pela aplicação: (i) visualizar um ou mais clientes; (ii) registar um novo cliente; (iii) activar/desactivar notificações relativas a produtos; (iv) consultar o histórico de transacções realizadas por um cliente.

A seguinte funcionalidade sobre fornecedores deve ser suportada pela aplicação: (i) visualizar um fornecedor; (ii) registar um novo fornecedor; (iii) Permitir/inibir um fornecedor; (iv) consultar o histórico de transacções realizados.

A seguinte funcionalidade sobre transacções deve ser suportada: (i) visualizar uma transacção; (ii) registar uma nova venda; (iii) registar uma nova encomenda; (iv) pagar uma transacção.

## Requisitos de Desenho

Devem ser possíveis extensões ou alterações de funcionalidade com impacto mínimo no código já produzido para a aplicação. O objectivo é aumentar a flexibilidade da aplicação relativamente ao suporte de novas funções. Assim, deve ser possível:

- Adicionar novos tipos de produtos;
- Definir novas entidades que desejem ser notificadas da alteração do estado dos produtos;
- Adicionar novos modos de entrega de mensagens (notificações);
- Adicionar novas políticas de recompensa de clientes;
- Adicionar novas formas de consulta.

Embora na especificação actual não seja possível remover entidades, a inclusão desta funcionalidade deve ser prevista, por forma a minimizar o impacto da sua futura inclusão.

## Interacção com o utilizador

Descreve-se nesta secção a **funcionalidade máxima** da interface com o utilizador. Em geral, os comandos pedem toda a informação antes de procederem à sua validação (excepto onde indicado). Todos os menus têm automaticamente a opção **Sair** (fecha o menu).

As operações de pedido e apresentação de informação ao utilizador **devem** realizar-se através dos objectos *form* e *display*, respectivamente, presentes em cada comando. As mensagens são produzidas pelos métodos das bibliotecas de suporte (**po-uuilib** e **woo-app**). As mensagens não podem ser usadas no núcleo da aplicação (**woo-core**). Além disso, não podem ser definidas novas. Potenciais omissões devem ser esclarecidas antes de qualquer implementação.

As excepções usadas na interacção, excepto se indicado, são subclasses de **pt.tecnico.po.ui.DialogException**, são lançadas pelos comandos e tratadas por **pt.tecnico.po.ui.Menu**. Outras excepções não devem substituir as fornecidas nos casos descritos.

A apresentação de listas (Fornecedores, Transacções, etc.) faz-se por ordem crescente da respectiva chave: dependendo dos casos, a ordem pode ser numérica ou lexicográfica (UTF-8), não havendo distinção entre maiúsculas e minúsculas.



Note-se que o programa principal e os comandos e menus, a seguir descritos, já estão parcialmente implementados nas *packages* **woo.app**, **woo.app.main**, **woo.app.products**, **woo.app.clients**, **woo.app.suppliers**, **woo.app.transactions**, **woo.app.lookups**. Estas classes são de uso obrigatório e estão disponíveis no CVS (módulo **woo-app**).

## Menu Principal

As acções deste menu permitem gerir a salvaguarda do estado da aplicação, abrir submenus e aceder a alguma informação global. A lista completa é a seguinte: Abrir, Guardar, Mostrar data actual, Avançar data actual, Gestão de Produtos, Gestão de Clientes, Gestão de Fornecedores, Gestão de Transacções, Consultas, Mostrar Saldo Global.

As etiquetas das opções deste menu estão definidas na classe **woo.app.main.Label**. Todos os métodos correspondentes às mensagens de diálogo para este menu estão definidos na classe **woo.app.main.Message**.



Estes comandos já estão implementados nas classes da *package* **woo.app.main** (disponível no CVS), respectivamente: **DoOpen**, **DoSave**, **DoDisplayDate**, **DoAdvanceDate**, **DoOpenMenuProducts**, **DoOpenMenuClients**, **DoOpenMenuSuppliers**, **DoOpenMenuTransactions**, **DoOpenMenuLookups**, **DoShowGlobalBalance**.

## Salvaguarda do estado actual da aplicação

Inicialmente, a aplicação está vazia ou tem apenas informação sobre as entidades que foram carregados no arranque (via ficheiro textual).

O conteúdo da aplicação (toda a informação actualmente em memória) pode ser guardado para posterior recuperação (via serialização Java: **java.io.Serializable**). Na leitura e escrita do estado da aplicação, devem ser tratadas as excepções associadas. A funcionalidade é a seguinte:

- **Abrir** -- Carrega os dados de uma sessão anterior a partir de um ficheiro previamente guardado (ficando este ficheiro associado à aplicação, para futuras operações de salvaguarda). Pede-se o nome do ficheiro a abrir (**openFile()**). Caso ocorra um problema na abertura ou processamento do ficheiro, deve ser lançada a excepção **FileOpenFailedException**. A execução bem sucedida desta opção substitui toda a informação da aplicação.
- **Guardar** -- Guarda o estado actual da aplicação no ficheiro associado. Se não existir associação, pede-se o nome do ficheiro a utilizar, ficando a ele associado. Esta interacção realiza-se através do método **newSaveAs()**. Não é executada nenhuma acção se não existirem alterações desde a última salvaguarda.

Note-se que a opção **Abrir** não permite a leitura de ficheiros de texto (estes apenas são utilizados na inicialização da aplicação).

A opção **Sair** nunca implica a salvaguarda do estado da aplicação, mesmo que existam alterações.

## Mostrar data actual

A data actual do sistema é apresentada através da mensagem **currentDate()**.

## Avançar data actual

O número de dias a avançar é pedido através de **requestDaysToAdvance()**. O valor indicado deve ser positivo. Caso contrário, deve ser lançada a excepção **InvalidDateException**.

## Gestão e consulta de dados da aplicação

- **Menu de Gestão de Produtos** -- Abre o menu de gestão de produtos.
- **Menu de Gestão de Clientes** -- Abre o menu de gestão de clientes.
- **Menu de Gestão de Fornecedores** -- Abre o menu de gestão de fornecedores.
- **Menu de Gestão de Transacções** -- Abre o menu de gestão de transacções.
- **Menu de Consultas** -- Abre o menu de consultas (pesquisas).

## Mostrar saldo global

Esta opção apresenta os valores (inteiros) correspondentes aos saldos disponível e contabilístico da empresa. Embora internamente o valor dos saldos esteja representado em vírgula flutuante, a apresentação é arredondada ao inteiro mais próximo.

A apresentação faz-se através da mensagem **currentBalance()**.

## Menu de Gestão de Produtos

Este menu permite efectuar operações sobre a base de dados de produtos. A lista completa é a seguinte: Visualizar todos os produtos, Registar caixa, Registar contentor, Registar livro, Alterar preço de produto.

As etiquetas das opções deste menu estão definidas na classe **woo.app.products.Label**. Todos os métodos correspondentes às mensagens de diálogo para este menu estão definidos na classe **woo.app.products.Message**.

Sempre que é pedido o identificador de um produto (**requestProductKey()**) e o produto não existir, é lançada a excepção **UnknownProductKeyException** (excepto no processo de registo). No processo de registo, caso o identificador do produto indicado já exista, deve ser lançada a excepção **DuplicateProductKeyException**. Na ocorrência de excepções (estas ou outras), as operações não têm efeito.



Estes comandos já estão implementados nas classes da *package* **woo.app.products** (disponível no CVS), respectivamente: **DoShowAllProducts**, **DoRegisterProductBox**, **DoRegisterProductContainer**, **DoRegisterProductBook**, **DoChangePrice**.

## Visualizar todos os produtos

O formato de apresentação de cada tipo de produto é o seguinte:

```
BOX|idProduto|idFornecedor|preço|valor-crítico|stock-actual|tipo-de-serviço
CONTAINER|idProduto|idFornecedor|preço|valor-crítico|stock-actual|tipo-de-serviço|nível-de-serviço
BOOK|idProduto|idFornecedor|preço|valor-crítico|stock-actual|título|autor|isbn
```

## Registar caixa

O sistema pede o identificador que ficará associado ao produto, o preço (**requestPrice()**), o valor crítico (**requestStockCriticalLevel()**) e o identificador do fornecedor (**requestSupplierKey()**) e o tipo de serviço de transporte associado (**requestServiceType()**).

Se o fornecedor não existir, deve ser lançada a excepção **UnknownSupplierKeyException**. Se a resposta para o tipo de serviço não for **NORMAL**, **AIR**, **EXPRESS** ou **PERSONAL**, deve ser lançada a excepção **UnknownServiceTypeException**.

Imediatamente após o registo, o número de existências é zero.

## Registar contentor

São pedidas as mesmas informações que para o registo de caixas e verificadas as mesmas condições. Além daquelas informações, é ainda pedida a qualidade de serviço (**requestServiceLevel()**). Se a resposta não for **B4**, **C4**, **C5** ou **DL**, deve ser lançada a excepção **UnknownServiceLevelException**.

Imediatamente após o registo, o número de existências é zero.

## Registar livro

O sistema pede o identificador único, o título (**requestBookTitle()**), o autor (**requestBookAuthor()**), o ISBN (**requestISBN()**), o preço (**requestPrice()**), o valor crítico (**requestStockCriticalLevel()**) e o identificador do fornecedor (**requestSupplierKey()**). Se o fornecedor não existir, deve ser lançada a excepção **UnknownSupplierKeyException**, tal como para os outros produtos.

Imediatamente após o registo, o número de existências é zero.

## Alterar preço de produto

O sistema pede o identificador do produto e o novo preço (**requestPrice()**).

Tal como mencionado acima, os clientes podem ser notificados quando o preço de um produto varia.

Caso ocorra um erro durante a alteração do preço de um produto existente, a operação falha silenciosamente.

## Menu de Gestão de Clientes

Este menu permite efectuar operações sobre a base de dados de clientes. A lista completa é a seguinte: Mostrar cliente, Mostrar todos os clientes, Registar cliente, Activar/desactivar notificações de um produto, Mostrar transacções do cliente.

As etiquetas das opções deste menu estão definidas na classe **woo.app.clients.Label**. Todos os métodos correspondentes às mensagens de diálogo para este menu estão definidos na classe **woo.app.clients.Message**.

Sempre que for pedido o identificador de um cliente (**requestClientKey()**) e o cliente não existir, é lançada a excepção **UnknownClientKeyException** (excepto no processo de registo). No processo de registo, caso o identificador indicado já exista, deve ser lançada a excepção **DuplicateClientKeyException**.



Estes comandos já estão implementados nas classes da *package* **woo.app.clients** (disponível no CVS), respectivamente:

**DoShowClient**, **DoShowAllClients**, **DoRegisterClient**, **DoToggleProductNotifications**, **DoShowClientTransactions**.

## Mostrar cliente

É pedido o identificador do cliente e apresentada a sua informação. O formato de apresentação é o seguinte (o valor das compras efectuadas refere-se ao momento da compra; o valor das compras pagas refere-se ao valor realmente pago):

```
id|nome|endereço|estatuto|valor-compras-efectuadas|valor-compras-pagas
```

O estatuto corresponde a **NORMAL**, **SELECTION**, **ELITE**, conforme o caso.

Após esta linha, são apresentadas as notificações do cliente (modo de entrega por omissão), pela ordem em que foram enviadas pela aplicação.

```
tipo-de-notificação|idProduto|preço-do-produto
```

Após esta visualização, considera-se que o cliente fica sem notificações registadas.

## Mostrar todos os clientes

O formato de apresentação é como para clientes individuais (opção anterior), mas não se apresentam as notificações dos clientes.

## Registar cliente

São pedidos o identificador do cliente, o nome (**requestClientName()**) (cadeia de caracteres) e o endereço do cliente (**requestClientAddress()**) (cadeia de caracteres) e regista-se o novo cliente. Quando um cliente é registado, aceita notificações relativas a todos os produtos.

Se já existir um cliente com o mesmo identificador, deve ser lançada a excepção **DuplicateClientKeyException**, não se realizando o registo.

## Activar/desactivar notificações de um produto

São pedidos o identificador do cliente e o identificador do produto (**requestProductKey()**). Se as notificações relativas ao produto estavam activas, passam a estar inactivas, e vice-versa. É apresentada na saída o resultado da operação: **notificationsOn()** ou **notificationsOff()**.

## Mostrar transacções do cliente

É pedido o identificador do cliente e apresentadas todas as transacções por ele realizadas. O formato de apresentação é como descrito abaixo (visualização de transacções de venda).

## Menu de Gestão de Fornecedores

Este menu apresenta as operações disponíveis sobre fornecedores e assuntos relacionados. A lista completa é a seguinte: Mostrar fornecedores, Registar fornecedor, Permitir/Inibir transacções, Mostrar transacções do fornecedor.

As etiquetas das opções deste menu estão definidas na classe **woo.app.suppliers.Label**. Todos os métodos correspondentes às mensagens de diálogo para este menu estão definidos na classe **woo.app.suppliers.Message**.

Sempre que é pedido o identificador da fornecedor (**requestSupplierKey()**), é lançada a excepção **UnknownSupplierKeyException**, se o fornecedor indicado não existir (excepto no processo de registo). No processo de registo, caso o identificador indicado já exista, deve ser lançada a excepção **DuplicateSupplierKeyException**.



Estes comandos já estão implementados nas classes da *package* **woo.app.suppliers** (disponível no CVS), respectivamente:

**DoShowSuppliers**, **DoRegisterSupplier**, **DoToggleTransactions**, **DoShowSupplierTransactions**.

## Mostrar fornecedores

O formato de apresentação é o seguinte:

```
id|nome|endereço|activo?
```

Os valores para o campo *activo?* são **yes()** ou **no()**.

## Registar fornecedor

São pedidos o identificador do fornecedor, o nome (**requestSupplierName()**) (cadeia de caracteres) e o endereço (**requestSupplierAddress()**) (cadeia de caracteres), registando-se o novo fornecedor. Quando um fornecedor é registado, fica no estado activo.

Se já existir um fornecedor com o mesmo identificador, deve ser lançada a excepção **DuplicateSupplierKeyException**, não se realizando a operação.

## Permitir/Inibir transacções

É pedido o identificador do fornecedor. Se as transacções estavam activas para esse fornecedor, passam a estar inactivas, e vice-versa. É apresentada na saída o resultado da operação: **transactionsOn()** ou **transactionsOff()**.

## Mostrar transacções do fornecedor

É pedido o identificador do fornecedor e apresentas todas as transacções por ele realizadas. O formato de apresentação é como descrito abaixo (visualização de transacções de encomenda).

## Menu de Gestão de Transacções

Este menu apresenta as operações relacionadas com transacções. A lista completa é a seguinte: Visualizar, Registar Venda, Registar Encomenda, Pagar.

As etiquetas das opções deste menu estão definidas na classe **woo.app.transactions.Label**. Todos os métodos correspondentes às mensagens de diálogo para este menu estão definidos na classe **woo.app.transactions.Message**.

Sempre que é pedido o identificador do fornecedor (**requestSupplierKey()**), é lançada a excepção

**UnknownSupplierKeyException**, se o fornecedor indicado não existir. Sempre que é pedido o identificador do cliente (**requestClientKey()**), é lançada a excepção **UnknownClientKeyException**, se o cliente indicado não existir. Sempre que é pedido o identificador de produto (**requestProductKey()**), é lançada a excepção **UnknownProductKeyException**, se o produto indicado não existir. Sempre que é pedido o identificador da transacção (**requestTransactionKey()**), é lançada a excepção **UnknownTransactionKeyException**, se a transacção indicada não existir.



Estes comandos já estão implementados nas classes da *package* **woo.app.transactions** (disponível no CVS), respectivamente: **DoShowTransaction**, **DoRegisterSaleTransaction**, **DoRegisterOrderTransaction**, **DoPay**.

## Visualizar

O sistema pede o identificador da transacção a visualizar.

Nas apresentações, o campo *valor-base* é o valor da transacção sem multas/descontos.

Se a transacção for respeitante a uma venda a um cliente, apresenta-se com o seguinte formato:

```
id|idCliente|idProduto|quantidade|valor-base|valor-a-pagamento|data-Limite|data-pagamento
```

O campo *valor-a-pagamento* corresponde ao valor que será realmente pago (considerando possíveis multas/descontos à data da visualização). A data de pagamento (e o separador correspondente) só é apresentada se a venda tiver sido paga.

Se a transacção corresponder a uma encomenda a um fornecedor, apresenta-se com um cabeçalho (linha inicial):

```
id|idFornecedor|valor-base|data-pagamento
```

A linha inicial é seguida por várias linhas, cada uma com a descrição de cada produto incluído na transacção (por ordem de inserção na encomenda):

```
idProduto|quantidade
```

## Registar Venda

Para registar uma venda, é pedido o identificador do cliente, a data limite para o pagamento (**requestPaymentDeadline()**), o identificador do produto a vender e a respectiva quantidade (**requestAmount()**). Se a quantidade for superior às existências actuais, deve ser lançada a excepção **UnavailableProductException** (não se realiza a venda).

A actualização dos produtos da empresa tem lugar logo após o registo da venda, ou seja, considera-se que a venda é instantânea.

## Registar Encomenda

Para registar uma encomenda, é pedido o identificador do fornecedor. De seguida, é pedido o identificador do produto a encomendar e a quantidade a encomendar (**requestAmount()**). Estas duas perguntas são repetidas para outros produtos enquanto a resposta a **requestMore()** (pergunta feita depois de cada quantidade) for afirmativa (leitura de um booleano).



Se o fornecedor indicado estiver inibido de efectuar transacções, deve ser lançada a excepção

**UnauthorizedSupplierException** e o comando não tem efeito. Se o identificador de um produto não corresponder ao fornecedor indicado, deve ser lançada a excepção **WrongSupplierException** e o comando não tem efeito.

A actualização de existências dos produtos da empresa tem lugar logo após o registo da encomenda, ou seja, considera-se que a encomenda é instantânea. A actualização do saldo da empresa também é assumida como instantânea, i.e., assume-se que a encomenda é paga a pronto.

## Pagar

Apenas é possível pagar vandas. Tentativas de pagamento de encomendas não produzem nenhum resultado.

É pedido o identificador da venda a pagar. Se a venda já tiver sido paga, não é realizada nenhuma acção.

## Menu de Consultas

Este menu apresenta as operações relacionadas com consultas. A lista completa é a seguinte: Mostrar produtos com preço abaixo de limite, Mostrar facturas pagas por cliente.

As etiquetas das opções deste menu estão definidas na classe **woo.app.lookups.Label**. Todos os métodos correspondentes às mensagens de diálogo para este menu estão definidos na classe **woo.app.lookups.Message**.

Sempre que é pedido o identificador do cliente (**requestClientKey()**), é lançada a excepção **UnknownClientKeyException**, se o cliente indicado não existir. Sempre que é pedido o identificador de produto (**requestProductKey()**), é lançada a excepção **UnknownProductKeyException**, se o produto indicado não existir.

A apresentação de resultados é como indicado nos casos já descritos de apresentação das várias entidades.



Estes comandos já estão parcialmente implementados nas classes da *package* **woo.app.lookups** (disponível no CVS), respectivamente: **DoLookupProductsUnderTopPrice**, **DoLookupPaymentsByClient**.

### Mostrar produtos com preço abaixo de limite

Pede-se o valor limite pretendido (**requestPriceLimit()**) e apresentam-se todos os produtos disponíveis na empresa cujo preço é inferior ao preço indicado. É apresentado um produto por linha.

### Mostrar facturas pagas por cliente

Pede-se o identificador do cliente e apresentam-se as transacções do cliente que já estão pagas (uma transacção por linha).

## Leitura de Dados a Partir de Ficheiros Textuais

Além das opções de manipulação de ficheiros descritas no menu principal, é possível iniciar a aplicação com um ficheiro de texto especificado pela propriedade Java **import**.

As várias entidades têm os formatos descritos abaixo. Assume-se que os títulos não podem conter o carácter | e que o preço é um número inteiro (sugere-se a utilização do método **String.split** para o processamento preliminar destas linhas). Não existem entradas mal-formadas.

Cada linha tem uma descrição distinta mas que segue os seguintes formatos.

```
SUPPLIER|id|nome|endereço
CLIENT|id|nome|endereço
```

```
BOX|id|tipo-de-serviço|id-fornecedor|preço|valor-crítico|exemplares
CONTAINER|id|tipo-de-serviço|nível-de-serviço|id-fornecedor|preço|valor-crítico|exemplares
```

```
BOOK|id|título|autor|isbn|id-fornecedor|preço|valor-crítico|exemplares
```

As definições de fornecedores e de clientes precedem sempre as dos produtos.

Um exemplo de conteúdo do ficheiro inicial é como se segue:

A codificação dos ficheiros a ler é garantidamente UTF-8.

```
SUPPLIER|S1|Toshiba|Tokyo, Japan
SUPPLIER|W2|Papellaria Fernandes|Oeiras, Portugal
SUPPLIER|P1|Publicações Europa-América|Lisboa, Portugal
SUPPLIER|P3|O'Reilly|Köln, Germany
CLIENT|R2|Jorge Figueiredo|Lisboa, Portugal
CLIENT|E4|Filomena Figueiredo|Lisboa, Portugal
CLIENT|ER|Abdul Figueiredo|Casablanca, Morocco
CLIENT|O9|Hellen Figueiredo|San Francisco, CA, USA
CLIENT|H2S04|John Figueiredo|Wellington, New Zealand
CLIENT|H20|Rohit Figueiredo|New Delhi, India
BOX|C6H5OH|NORMAL|W2|2|20|100
BOX|H2|AIR|W2|4|20|100
BOX|O3|EXPRESS|W2|8|20|100
BOX|CO2|PERSONAL|W2|16|20|0
CONTAINER|M4|NORMAL|B4|W2|2|20|100
CONTAINER|M2|AIR|C4|W2|4|20|100
CONTAINER|M5|EXPRESS|B4|W2|8|20|100
CONTAINER|M3|PERSONAL|DL|W2|16|20|0
BOOK|B1256|Os Lusíadas|Luís de Camões|1234567890|P1|58|2|5
BOOK|B9854|Head First Java|Sierra & Bates|9876543210|P3|75|2|5
BOOK|B1937|How to fix almost everything|Satoshi Yamada|1928374650|S1|5|2|5
```

💡 Note-se que o programa nunca produz ficheiros com este formato.

## Execução dos Programas e Testes Automáticos

Usando os ficheiros **test.import**, **test.in** e **test.out**, é possível verificar automaticamente o resultado correcto do programa. Note-se que é necessária a definição apropriada da variável **CLASSPATH** (ou da opção equivalente **-cp** do comando **java**), para localizar as classes do programa, incluindo a que contém o método correspondente ao ponto de entrada da aplicação (**woo.app.App.main**). As propriedades são tratadas automaticamente pelo código de apoio.

```
java -Dimport=test.import -Din=test.in -Dout=test.outhyp woo.app.App
```

Assumindo que aqueles ficheiros estão no directório onde é dado o comando de execução, o programa produz o ficheiro de saída **test.outhyp**. Em caso de sucesso, os ficheiros das saídas esperada (**test.out**) e obtida (**test.outhyp**) devem ser iguais. A comparação pode ser feita com o comando:

```
diff -b test.out test.outhyp
```

Este comando não deve produzir qualquer resultado quando os ficheiros são iguais. Note-se, contudo, que este teste não garante o correcto funcionamento do código desenvolvido, apenas verificando alguns aspectos da sua funcionalidade.

## Notas de Implementação

Tal como indicado acima, algumas classes fornecidas como material de apoio, são de uso obrigatório e não podem ser alteradas. Outras dessas classes são de uso obrigatório e têm de ser alteradas.

A serialização Java usa as classes da *package* **java.io**, em particular, a interface **java.io.Serializable** e as classes de leitura **java.io.ObjectInputStream** e escrita **java.io.ObjectOutputStream** (entre outras).

Categories:    **Ensino**   **PO**   **Projecto de PO**