**FIAP – Faculdade de Informática e Administração Paulista**

**Desenvolvimento De Aplicações Java - Soa E Internet Das Coisas**

**MATEUS ARTUR VIEIRA SANTOS**

**VICTOR LUIZ DE LIMA PEREIRA MONTEIRO**

**TRABALHO FINAL**

**São Paulo**

**2018**

**MATEUS ARTUR VIEIRA SANTOS**

**VICTOR LUIZ DE LIMA PEREIRA MONTEIRO**

**TRABALHO FINAL**

Trabalho Final apresentado a Disciplina de Fundamentos da Tecnologia Java e UML no Curso de **30SCJ - Desenvolvimento De Aplicações Java - Soa E Internet Das Coisas** da FIAP – Faculdade de Informática e Administração Paulista, sob a orientação do Professor: Samuel Lima.

Coordenação: Albert da Costa Xavier

**São Paulo**

**2018**

**ÍNDICE**

[**1.**](#_lbcpnpl2vvo6) **Explicação do Cenário da aplicação 3**

**2. Apis utilizadas. 3**

**3**[**. Instalação da Aplicação**](#_lbcpnpl2vvo6) **4**

**4**[**. Manual**](#_2fi7n98448ak) **& Evidencia de testes 9**

**4**[**.**](#_oz1yj9mj4fdo) **UML 14**

1. **Explicação do cenário da aplicação.**

A aplicação foi construída pensando no cenário de uma loja onde vende-se diversos tipos de produtos,onde o cliente seleciona os produtos que quer comprar, escolhe também sua forma de pagamento (Crédito ou Débito), caso a escolha seja Débito, o cliente tem 5% de desconto na sua compra. Após concluir a compra, geramos uma nota fiscal para o cliente, mostrando os produtos comprados. e também as informações pessoais do cliente.

**2. Apis utilizadas**

Em nossa aplicação utilizamos diversas Apis, como as **Lambdas** para facilitar a criação de um objeto Runnable, que com ele conseguimos iniciar uma **Thread**.

Também utilizamos a API de **IO**, para criar nossos arquivos de **NotaFiscalDebito** e **NotaFiscalCredito.**

Utilizamos também as **Streams** para percorrer nossas **Collections** facilmente, como por exemplo nossa lista de Itens, que nela também utilizamos o **Method References** para exibir seus dados de uma forma mais rápida(Utilizada na classe NotaFiscal).

Também utilizamos o Enum para demonstrar o tipo de cartão que o cliente pode utilizar.Utilizamos a API **LocalDate** de Joda Time para criar a data de nascimento do cliente também quando nossas notas fiscais foram geradas(Utilizada nas classes TestePagamentoDebito,TestePagamentoCredito,NotaFiscal e Cliente).

Um recurso utilizado também foi os métodos **.mapToDouble()** e **.sum()** da API Stream, onde nos deu a facilidade de recuperar o valor total dos produtos comprados.(Utilizada na classe Cliente e também NotaFiscal).

Também utilizamos o operador **Ternário** para substituir a condição IF/Else (Classe Cliente e NotaFiscal).

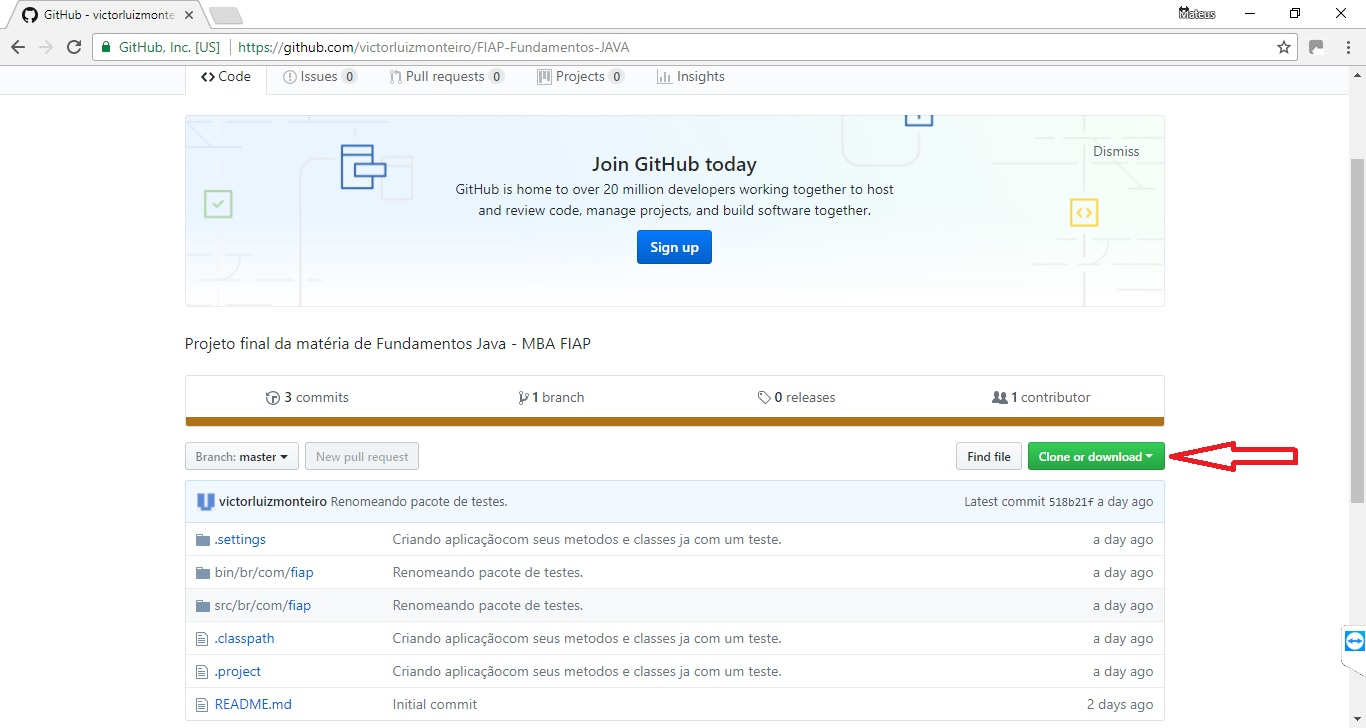
**3.** **Instalação da Aplicação**

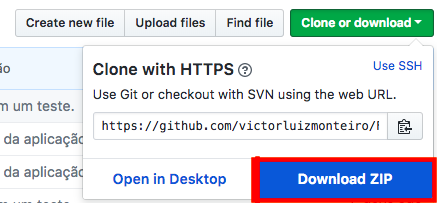
Abaixo vamos demonstrar como instalar a aplicação:

Passo 1: Acesse o link abaixo para baixar a aplicação.

* <https://github.com/victorluizmonteiro/FIAP-Fundamentos-JAVA>

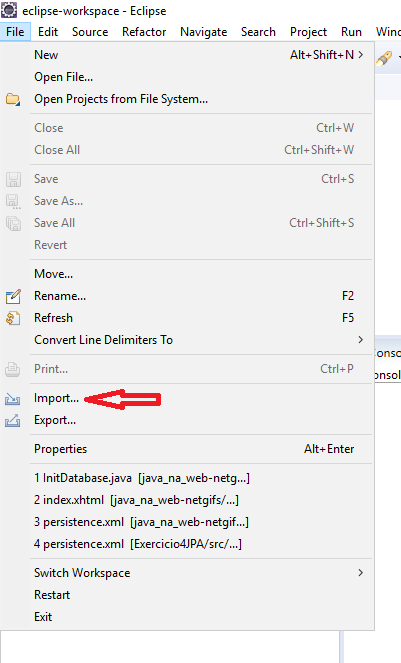
Passo 2: Fazer o Download da Aplicação no formato utilizando a opção Download ZIP



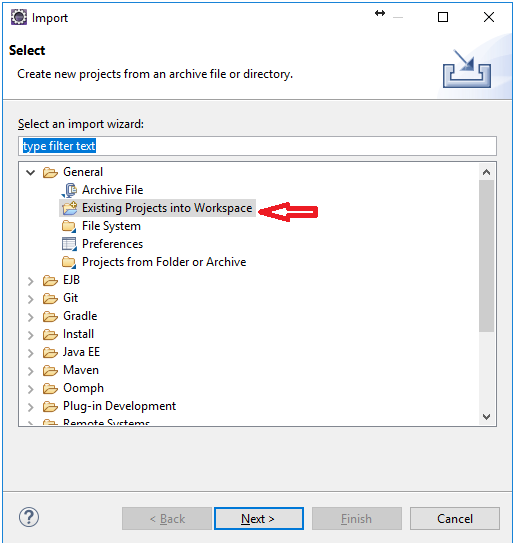


Passo 3: Com o Eclipse aberto, seguir a sequência de instruções para instalar a aplicação.

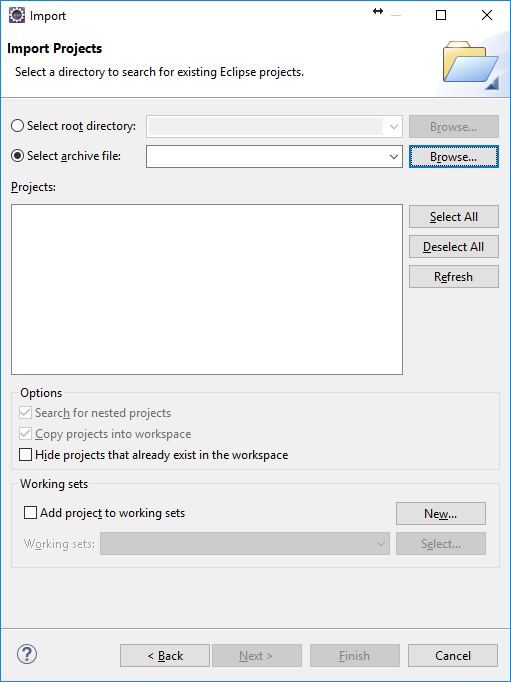
Passo 3.1 Clicar em File -> Import

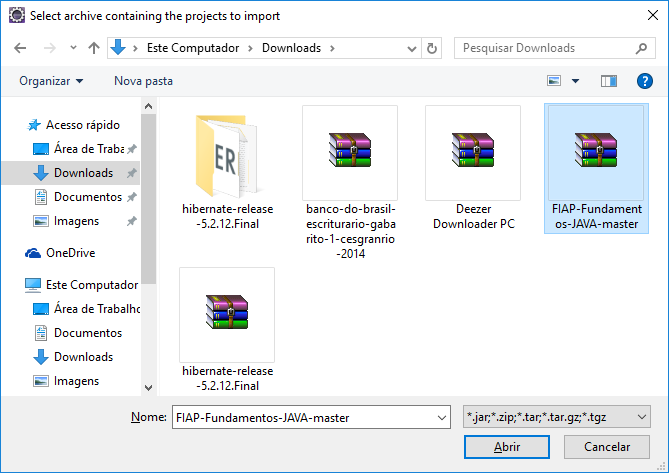


Passo 3.2 na Janela a seguir Clicar em **Existing Projects into Workspace**

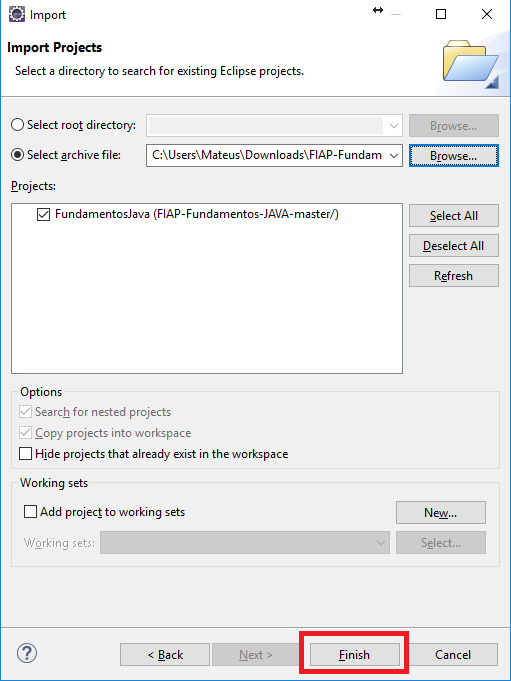


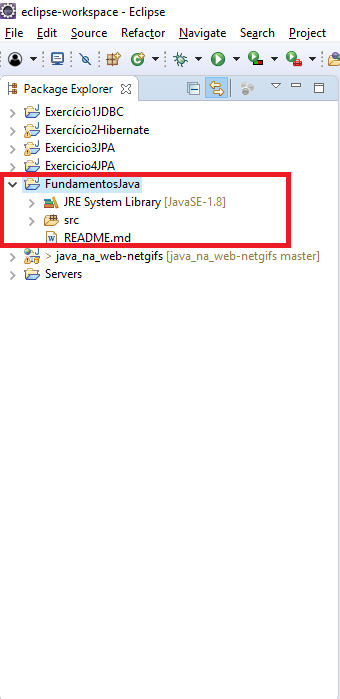
Passo 3.3 Selecionar a opção **Select archive file** e clicar em **Browse** para procurar o arquivo .zip baixado.





Passo 3.4 Clicar em **Finish.**





A aplicação foi instalada com sucesso!

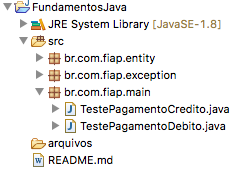
# **4.** **Manual**

Depois de seguir os passo a passo da demonstração anterior de como instalar a aplicação, abaixo vamos demonstrar como executar a aplicação:

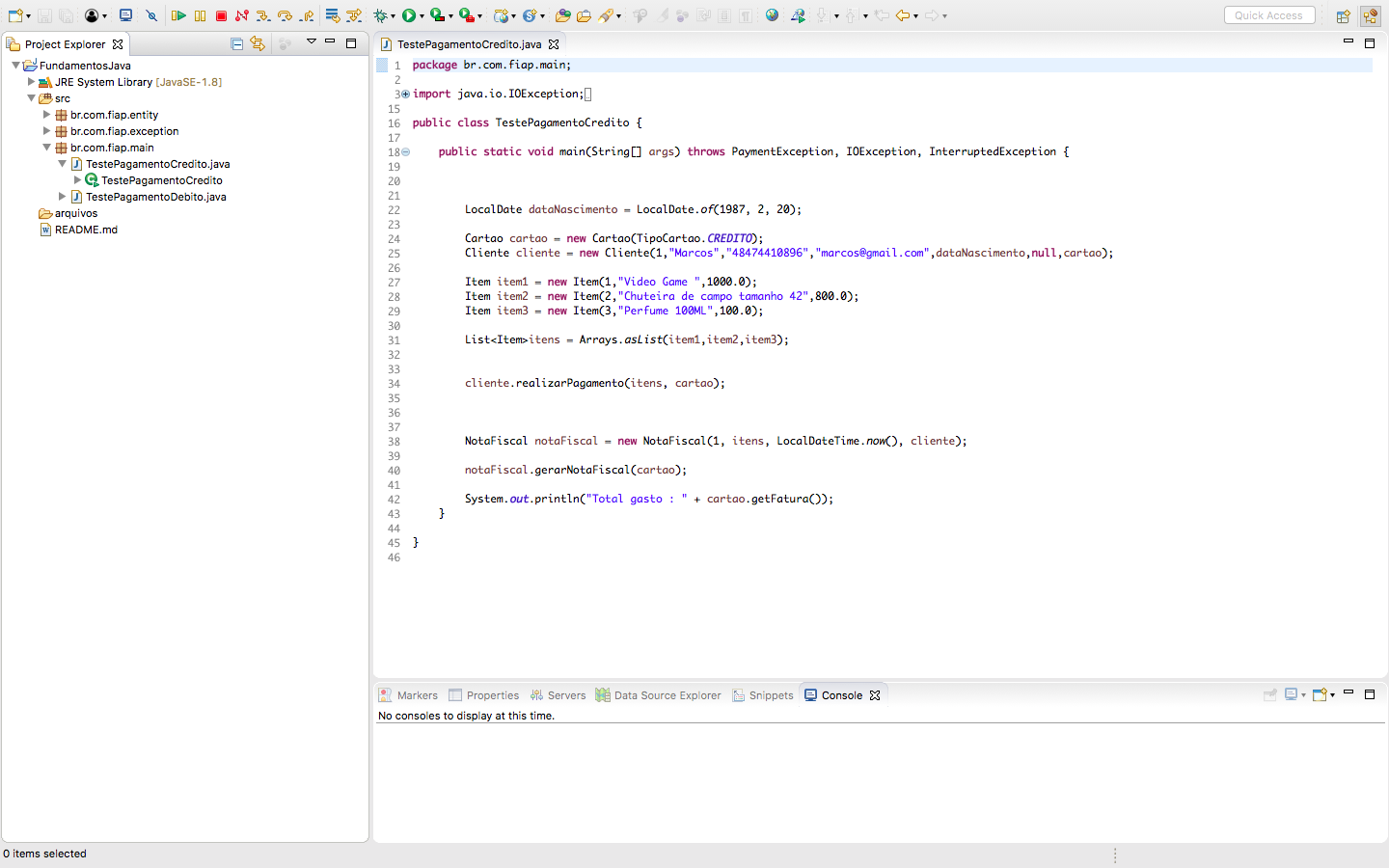
OBS:. Nossa aplicação está separada em duas classes de testes,TestePagamentoCredito e TestePagamentoDebito, iremos mostrar os testes e execução de cada uma delas.

Cada uma delas gera um arquivo texto que estará dentro da pasta arquivos, onde teremos uma ilustração de uma Nota Fiscal, os arquivos gerados são NotaFiscalCredito.txt e NotaFiscalDebito.txt

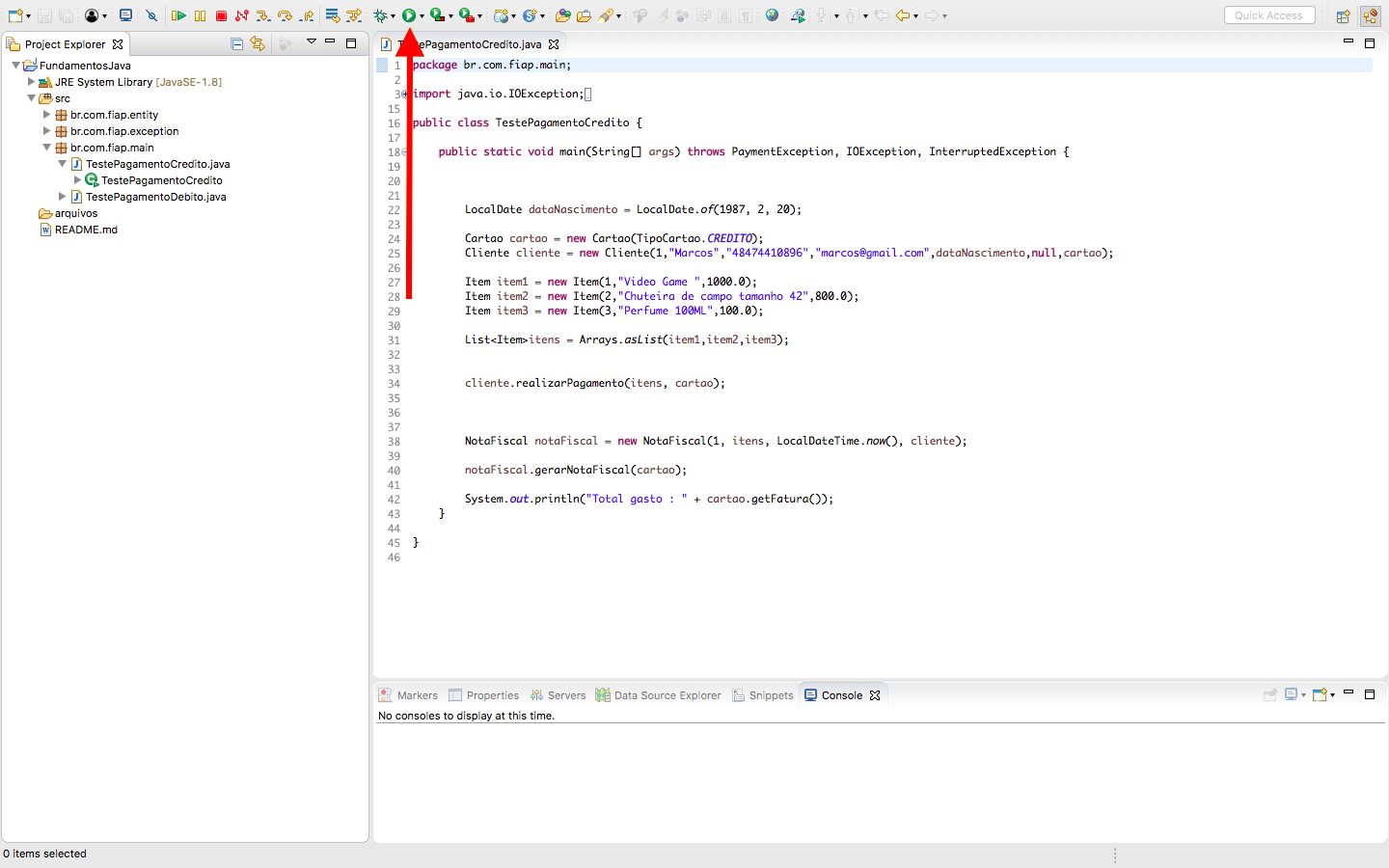
Passo 1: Abra o pacote: br.com.fiap.main



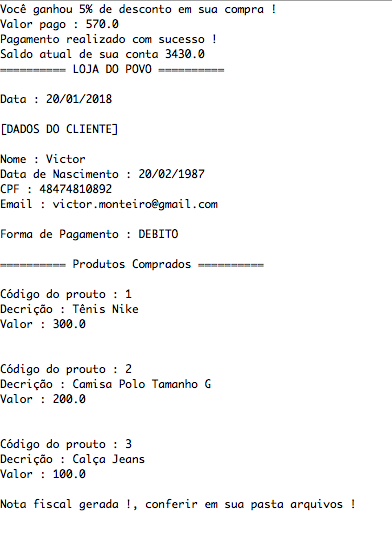
Passo 2: De um duplo clique na classe TestePagamentoDebito.

****

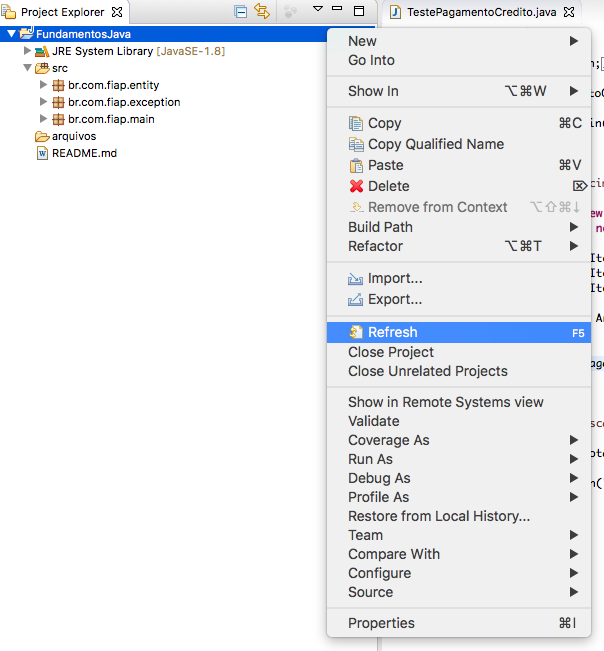
Passo 3: Execute a aplicação conforme a evidência abaixo.

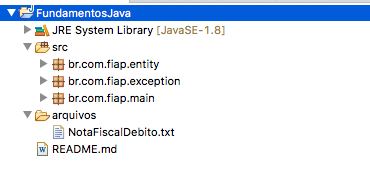
****

Passo 4: Após executar a classe TestePagamentoDebito temos a seguinte saída no console :

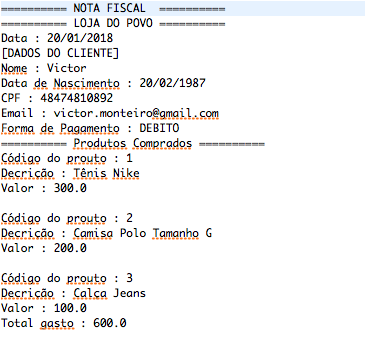
****

**Passo 5 : Após rodar a classe de testes,automaticamente nossa aplicação criou um arquivo texto na pasta arquivos, criada na raiz do projeto, para que a mesma seja atualizada devemos clicar com o botão direito em cima do projeto e clicar em Refresh .**

**Passo 6 : Com isso, temos nosso arquivo Criado :**

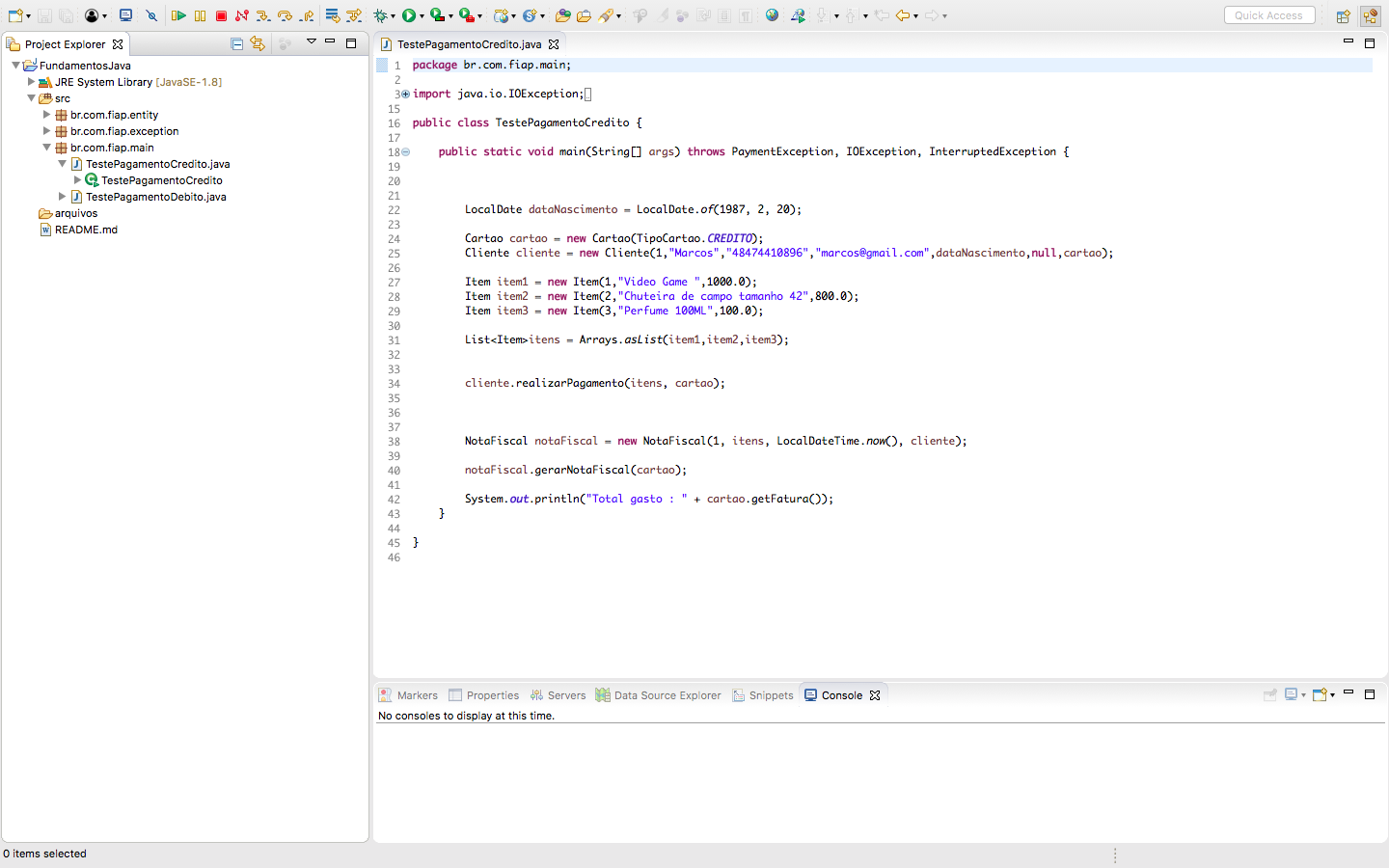
****

**Passo 7 : Dar duplo clique em cima do nosso arquivo NotaFiscalDebito.txt, teremos o seguinte resultado :**

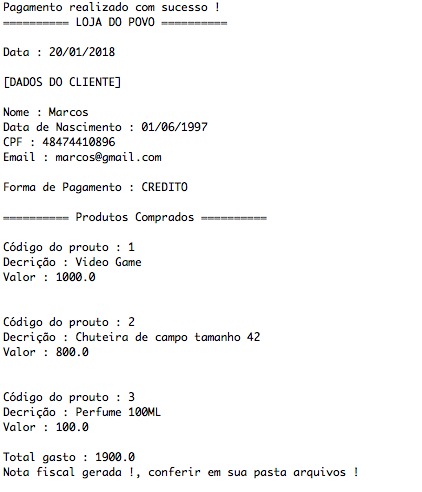
****

**Para executar a classe TestePagamentoCredito, favor seguir os passo 2,3,5 e 6 , abaixo será mostrado as entradas e saidas do teste da classe.**

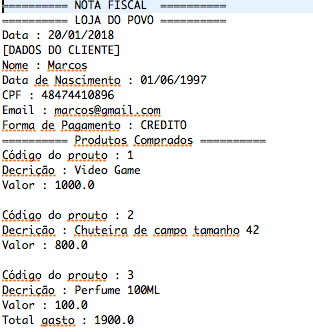
# **Entrada de dados da classe TestePagamentoCredito :**



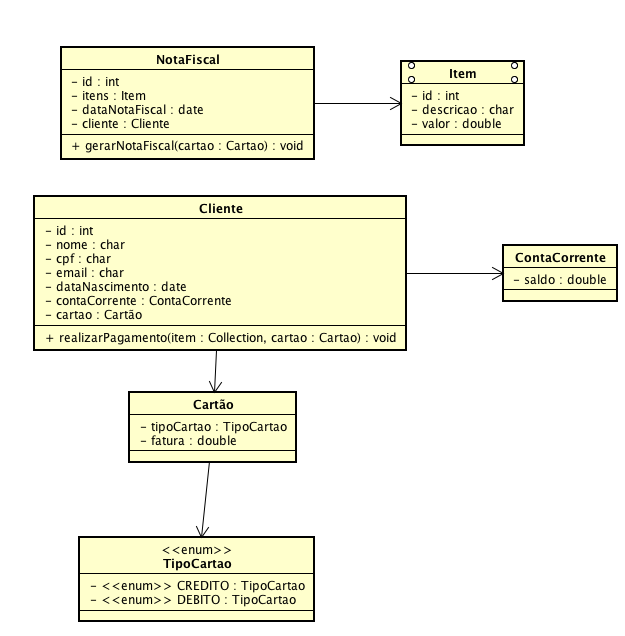
**Saída esperada :**

****

**Arquivo NotaFiscalCredito.txt**

****

# **5.** **UML**

****