P.PORTO ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

IPP – Instituto Politécnico do Porto

LEI/LSIRC

PP – Paradigmas de Programação

2º Semestre

Ficha Prática 6

Sumário
- Métodos
- Sobrecarga de Métodos
- Encapsulamento
- Métodos de acesso
- Modificadores de acesso

Documentação complementar Java:
- Classes
- Objetos
- Informação adicional

Nota: A resolução desta ficha prática pressupõe a utilização da ferramenta git para ajudar na gestão de versões ("backups") do software desenvolvido.
Documentação complementar Git:
- learngitbranching
- gitexplorer

Desenvolva uma aplicação que suporte a gestão de ementas de uma pequena pizzaria. Considere que:

- Cada Ementa contém um conjunto de Pizzas que por sua vez são constituídas por um conjunto de ingredientes (no máximo cada pizza terá 5 ingredientes).
- Para cada Ementa é necessário identificar a sua designação, data de início e data de fim (que não necessita de ser pré-estabelecida inicialmente).
- Cada Pizza é caracterizada pelo seu código, nome, descrição e os tamanhos (Pequena, Média, Grande), número de ingredientes e uma coleção de ingredientes que representam a composição da pizza. O preço da Pizza é estabelecido para cada Menu em função do tamanho da Pizza.
- Para cada Ingrediente é necessário armazenar um código, o nome e a origem (Animal, Vegetal ou Mineral).

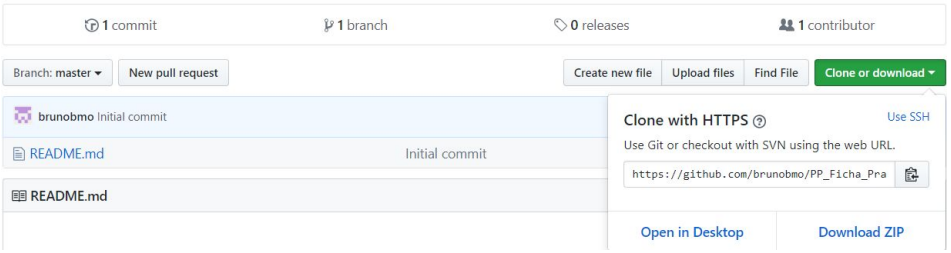
O Git é um sistema de controlo de versões distribuído que auxilia o processo de desenvolvimento de software. Com esta poderosa ferramenta é possível registar o histórico de versões de software a desenvolver. Uma pasta de trabalho do Git é considerada um repositório com um histório completo de todas as revisões realizadas.

O excerto de código para resolução da ficha prática encontra-se alojado num repositório do GitHub. No **repositório** pode encontrar um excerto da resolução do problema apresentado. Para o utilizar devidamente, deve:

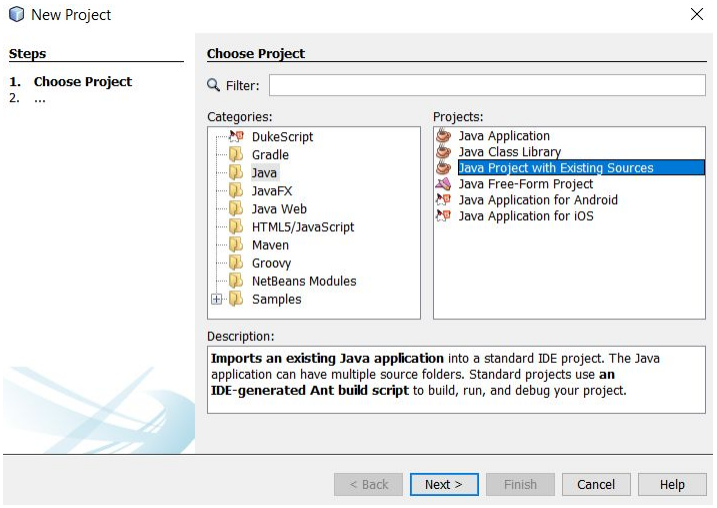
1. Realize o download da ferramenta Git de acordo com o seu sistema operativo.
2. Siga as instruções de instalação sem alterar os parâmetros pré-definidos.

De forma a utilizar o código disponibilizado no seu repositório, deverá:

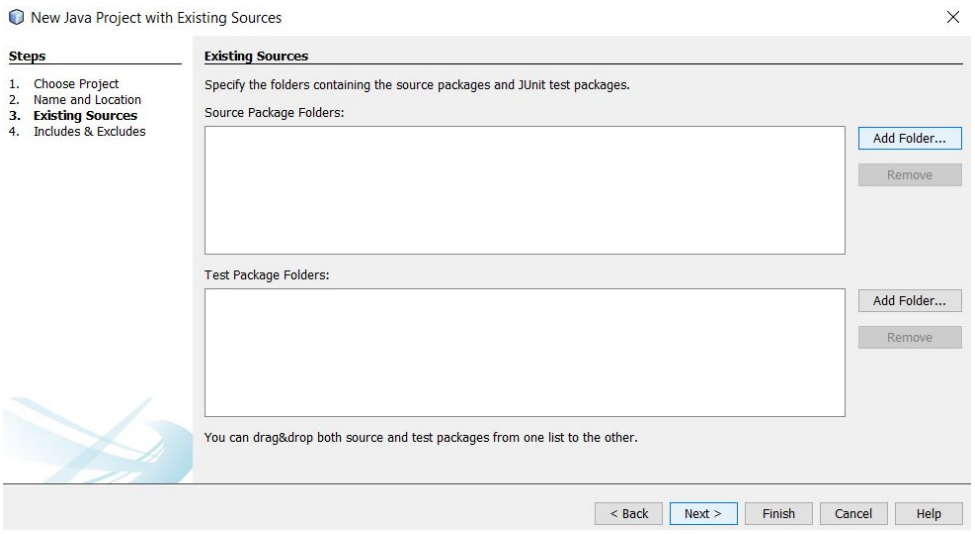
1. Criar uma pasta (na localização que desejar) com o nome: `2019.PP.FP.06`. Clicar com o botão direito do rato na pasta onde criou o projeto (será essa a pasta do seu repositório) e selecionar: `Git Bash Here`. Existem ferramentas gráficas para trabalhar com o Git mas (para já) vamos utilizar a linha de comandos.
2. Na nova janela, deverá:
    a. Digite: `git clone https://github.com/brunobmo/PP_Ficha_Pratica_6.git`. Com este comando realizamos uma cópia do código existente no repositório da ficha prática para o seu repositório local. A hiperligação pode ser obtida na página do git:



b. De seguida crie o projeto Netbeans na pasta criada anteriormente e de acordo com a seguinte configuração:



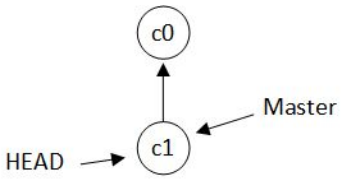
Na janela seguinte:



Clique em `Add Folder` e adicione a pasta `src` do projeto que surgiu após a realização do comando `git clone`.

c. Na pasta do projeto, digite: `git init` (inicializa o repositório local)
d. Ao realizar o clone do repositório fornecido, algumas propriedades já se encontram definidas. É caso dos dois primeiros *commit*. Um *commit* num repositório Git regista uma fotografia (*snapshot*) dos ficheiros que adicionamos anteriormente. É como um copy -> paste, mas nesta caso o Git não copia todo o diretório de cada vez que é realizado um commit. Em cada commit são apenas registadas as mudanças entre uma versão do seu repositório e a seguinte. O Git também mantém um histórico de quando ocorreu cada commit.
e. Para visualizar os commits do seu repositório Git digite: `git log`.

Após a execução dos passos anteriores, é possível representar o estado do seu repositório de acordo com a seguinte imagem:



Master é o nome dado à linha temporal principal do repositório.

HEAD é um nome simbólico para o *commit* atualmente ativo. É essencialmente o *commit* sobre o qual está trabalhar neste momento (c1 significa *commit* 1).

c0 e c1 são apenas referências textuais. Após digitar o comando irá perceber que c0 é o primeiro commit do repositório (com a inclusão de um ficheiro README) e c1 é o commit com o exemplo da ficha prática.

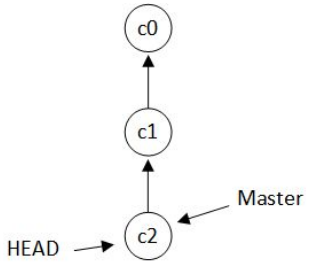
Exercício 1

Com base no código fornecido e num *package* `pp_fp06.PizzaRestaurant`, crie as classes necessárias para responder aos requisitos do problema, considerando que:

- Deve garantir o **encapsulamento** de todas as classes criadas;
- Deve criar **métodos de acesso** necessários para todas as classes criadas;
- Num *package* `pp_fp06.PizzaRestaurant.Enums`, deve criar as **enumerações** para representar o tamanho da `Pizza` (Pequena, Média ou Grande). A "impressão" do tamanho da pizza deve ser apresentada com uma mensagem descritiva (exemplo disponibilizado no repositório de suporte à Ficha Prática);
- Utilize a palavra reservada `this` para se referir a cada variável de instância em todas as classes criadas.

O enriquecimento do exemplo fornecido permeti-lhe realizar a primeira contribuição para o projeto a desenvolver. Está na hora de realizar um "backup" no repositório.

Realize um novo *commit*. Digite: `git commit -m "terceiro commit" -a` (O tipo de mensagens utilizadas têm o propósito de realçar funcionamento do Git. Pode alterar para um textual representativo da contribuição realizada). Digite o comando: `git log` e analise o resultado. É possível representar o estado do seu repositório de acordo com a seguinte imagem:



Agora temos três *commits*. Note que HEAD e Master estão agora direcionados para o novo *commit*.

Exercício 2

Analise o conteúdo da classe `Pizza` e da enumeração `PizzaSize`.

2.1. Crie a classe `Ingredient` de forma a representar a informação sobre ingredientes;

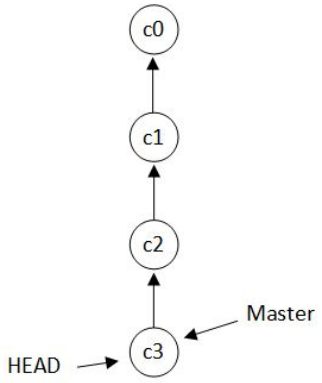
2.2. Documente o código desenvolvido (analise os exemplos javadoc fornecidos nas classes de exemplo).

2.2. Com a classe `Ingredient` devidamente criada e documentada, submeta uma nova versão do seu

projeto no repositório. Realize as seguintes tarefas:

a. Digite: `git log`. O Git indicará que os ficheiros criados para este exercício estão em modo: *untracked*. Isto significa que ao realizarmos o *commit*, os ficheiros não serão adicionados ao repositório.
b. Digite: `git add <nomeficheiro>.java` para cada ficheiro criado na resolução do exercício. Este comando fará com que o Git inclua o ficheiro java desenvolvido na sua árvore de trabalho, de forma a realizar o controlo de versões do ficheiro criado (para além dos que adicionou anteriormente).
c. Digite: `git commit -m "Quarto commit" -a`
d. Analise o estado do seu repositório com o comando: `git log`

O estado do seu repositório deverá ficar como a seguinte representação:



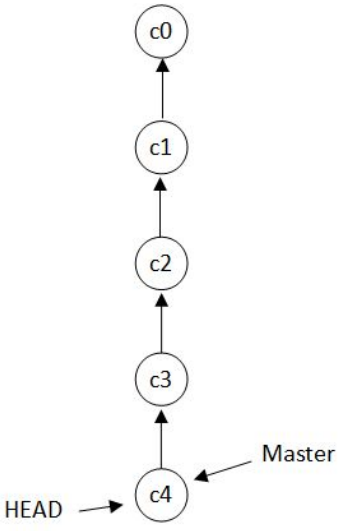
Exercício 3

Altere a classe Pizza de forma a acrescentar um método: `getDescription` que imprima (`System.out`) uma descrição detalhada da `Pizza`, assim como dos seus ingredientes. Exempo: "A Pizza xpto possui 2 ingredientes: Massa alta com...."

Teste as classes criadas através de uma classe `PizzaDemo`. Crie no mínimo, duas pizzas (deverá utilizar uma coleção) com pelo menos três ingredientes cada.

Submeta uma nova versão do seu projeto no repositório de acordo com as instruções indicadas anteriormente.

O estado do seu repositório deverá ficar com a seguinte representação:

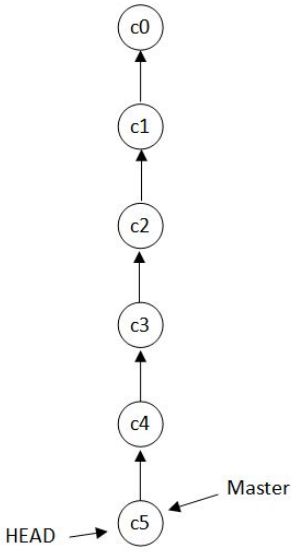


Exercício 4

Altere o método: `getDescription` de forma a que a string representativa da `Pizza` seja retornada no método.

Submeta uma nova versão do seu projeto no repositório de acordo com as instruções indicadas anteriormente.

O estado do seu repositório deverá ficar com a seguinte representação:



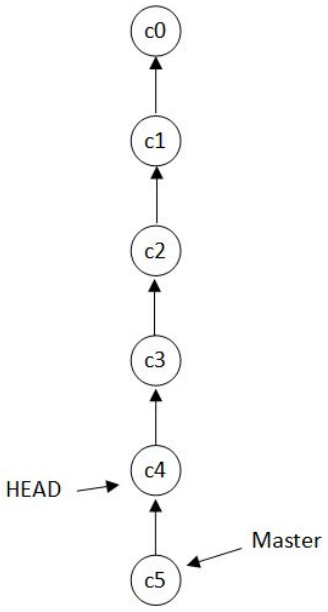
Exercício 5

Compare as alterações que realizou no exercício 4 com a do exercício 3. No entanto, já apagou a primeira versão! Felizmente não é necessário refazer o exercício 3.

Utilizando o Git, vamos voltar ao passado! Para isso:

1. Digite: `git log` para visualizar os *commits* até agora.
2. A versão que imprimia a string encontra-se no commit anterior (c4).
3. Digite: `git checkout master^`. Este comando posiciona o HEAD no commit anterior (é equivalente a colocar `git checkout <hash do commit>`. Desta forma podemos "viajar no tempo".
4. Verifique o seu código!

Estado do seu repositório:



Volte ao presente com o comando: `git checkout master`

Exercício 6

6.1. Na classe `Pizza`, adicione métodos que permitam:

- Associar novos ingredientes a uma `Pizza` até a um máximo de 5. Não deve ser permitido aceder diretamente à variável que representa a coleção dos ingredientes da pizza;
- Remover um ingrediente (identificando o ingrediente pelo seu id) que pertença à coleção de ingredientes;

Teste as alterações na classe `PizzaDemo`.

6.2. Documente o código desenvolvido (analise os exemplos javadoc fornecidos nas classes de exemplo).

6.3. Submeta uma nova versão do seu projeto no repositório de acordo com as instruções indicadas anteriormente.

Exercício 7

7.1. Implemente a classe `Ementa` de forma a que seja possível associar um máximo de 10 `Pizzas`. Para a representação e validação de datas utilize a classe: LocalDateTime.

7.2. Garanta que a `Ementa` só é válida se possuir pelo menos uma `Pizza` vegetariana (a origem de algum dos ingredientes não é Animal)

7.3. Documente o código desenvolvido (analise os exemplos javadoc fornecidos nas classes de exemplo).

7.4. Submeta uma nova versão do seu projeto no repositório de acordo com as instruções indicadas anteriormente.

Exercício 8

8.1. Realize as alterações necessárias de forma a que seja possível armazenar várias `Ementas`. Apenas uma ementa poderá estar ativa num determinado momento (crie um método para o efeito).

8.2. Documente o código desenvolvido (analise os exemplos javadoc fornecidos nas classes de exemplo).

8.3. Submeta uma nova versão do seu projeto no repositório de acordo com as instruções indicadas

anteriormente.

8.4. Com o botão direito do rato na pasta do seu repositório selecione: `Git GUI Here`. Explore a ferramenta.

8.5. Explore os utilitários do Netbeans de suporte ao Git. Botão direito do rato sobre o projeto: