Java WebDeveloper - BRQ

Conteúdo: Java Orientado a Objetos
Data para entrega: 16/05/15

Lista
1

RAFAEL HIROSHI TAGATA

Teoria:

1) Qual a finalidade do método public static void main?

A finalidade do método public static void main, é executar o programa, ou seja, ela é o método que pode chamar outras classes ou métodos para serem testados. Sem o Main, o programa pode ser compilado, mas sua execução só é dada com ela, isto é claro, levando em consideração uma aplicação local, rodando o executável. Pois para desenvolvimento web, o "Main", praticamente não é chamado, devido sua execução ser Cliente/Servidor.

2) O que é um JavaBean? Quais características uma Classe deve possuir para estar em conformidade com o padrão JavaBean?

JavaBean são componentes de software escritos na linguagem de programação Java, onde expõe propriedades, seguindo uma convenção para seu desenvolvimento. São classes utilizadas para encapsular muitos objetos em um único objeto (bean), assim ele pode ser transmitido como único objeto em vez de vários objetos individuais, claro, respeitando as conformidades da Orientação a Objetos e regras de negócio.

Para o JavaBean estar em conformidade com a convenção é necessário:
-Atributos privados
-Construtores (Vazio / Com entrada de argumentos –Sobrecarga)
-Método get e set (encapsulamento)
-Sobrescrita dos métodos da Classe Object (toString, equals, hashCode)

3) Explique a diferença entre os modificadores de acesso private, protected, default e public.

- Private (Acesso somente dentro da própria Classe)
- Default (Acesso dentro da própria Classe ou por Classes que estão no mesmo pacote

- Protected (Acesso por Classes do mesmo pacote ou em pacotes diferentes por meio de herança)
- Public (Acesso Total)

4) O que é um Construtor?

Construtor é a primeira coisa quando chamamos uma classe, ela é responsável por criar o objeto em memória, ou instanciar a classe que foi definida. Ela por padrão do Java, vem vazia. Mesmo que não seja criado um construtor pelo desenvolvedor, ela existe. Podendo ser modificada. Caso o desenvolvedor queira modificar o construtor da classe, mudando sua assinatura. Por convenção, deve-se criar um construtor vazio, para que com isso possa instanciar a classe sem passar argumentos.

Construtores são utilizados para poder mudar a assinatura da classe, podendo ou não ser chamada já com argumentos. Não há limite para a construção de Construtores.

5) Defina encapsulamento. Explique o funcionamento dos métodos set e get.

Encapsulamento é o mecanismo a partir do qual os atributos de uma Classe, são protegidos do acesso externo. Esta proteção baseia-se no uso de modificadores de acesso restritivos para os atributos (private) e na criação de métodos que irão realizar o acesso indireto aos atributos.

Com isso para a manipulação dos atributos por classes externas é dado através de métodos Getters e Setters.

Método Getter, tem o objetivo de "pegar", visualizar o atributo, retornando o valor.

Método Setter, tem o objetivo poder modificar o atributo, enviando valor ao atributo.

6) Defina Sobrecarga de Métodos.

Sobrecarga de Métodos ou Overloading é quando em uma classe, declaramos métodos com o mesmo nome, porem com entrada de argumentos diferentes.

7) Defina Sobrescrita de Métodos

Sobrescrita de Método ou Override ocorre quando uma subclasse sobrepõe métodos da sua superclasse, modificando o comportamento de tais métodos, reprogramando-os na subclasse.

8) Explique a diferença entre os operadores this e super

A diferença de utilizar as palavras reservadas this e super é que a palavra this é utilizada quando se quer fazer uma referência a um objeto ou chamar um construtor da própria classe. Utilizando esta palavra quando queremos evitar a repetição de código,

A palavra reservada super podemos executar o construtor da classe pai dentro da classe filha e ainda escolher qual dos construtores da classe pai queremos executar.

Resumidamente, a palavra this tem relação a própria classe e a palavra super, referência à classe pai.

9) Explique o relacionamento de herança

Herança está relacionado diretamente ao reuso de código, sendo assim, classes mais genéricas e menos especializadas possuem características que podem ser herdadas por classes menos genéricas, porém mais especializadas. Classes filhas herdam características da classe pai. Para a utilização de herança, utilizamos a palavra reservada (extends).

10) Explique o relacionamento de associação

O relacionamento de associação (TER) é o relacionamento de objetos de classes distintas. A navegabilidade é representada através de uma seta nas

extremidades, pois representa o sentido em que informações são disparadas.

Exemplificando:

Uma classe Funcionário TER relacionado outra classe Endereço, a questão de relacionamento é para manter as classes organizadas conforme a Orientação a Objetos, onde cada classe continua mantendo suas próprias características, mas através da associação poder utilizar recursos de outras classes.

11) Explique a diferença entre os tipos primitivos do Java e as Classes Wrappers. Dê exemplos.

Tipos primitivos podemos entender como sendo tipos de informação mais usuais e básicos nos quais são utilizados sem grandes recursos do contrário dos Wrappers, que são classes , onde possuem métodos onde podemos manipular as informações através de seus métodos como o toString.

Exemplos de tipos primitivos: int, double, float, byte, long, boolean etc.

Exemplos de Classes Wrappers: Integer, Double, Float, Byte, Boolean, etc.

12) Qual a importância da Classe Object?

A Classe Object é a principal e mais básica classe do Java. Todas as outras classes têm origem dela, ou seja, todas herdam da classe Object, mesmo que não utilizem a palavra reservada (extends). A importância da classe Object, é que nela possuem métodos que utilizamos no dia a dia do desenvolvimento, não necessitando escreve-las ou podemos modificar-la caso necessário, como Exemplo de métodos da classe Object que podem ser Overread são a toString, hashCode e equals.

13) Explique a finalidade dos métodos equals, toString e hashCode

A finalidade dos métodos equals, toString e hashCode são:

- equals: é verificar se o objeto é igual a outro, seja seu valor ou tipo ou Classe. Diferentemente de (==) onde verifica se a posição de memória é a mesma.

- toString: é retornar a informação como String

- hashCode: é o agrupamento e organização dos objetos.

14) Explique a finalidade do operador instanceof

A finalidade do operador instanceof é verificar se (é um) ou (ser), em outras linguagens como o C# é utilizado a palavra (is).

15) Defina tratamento de Exceções. Explique o uso de try, catch, finally, Exception, throws e throw new

Os tratamentos de exceção:

- Try – é utilizada para indicar um bloco de código que possa ocorrer uma exceção.

- Catch – serve para manipular as exceções, ou seja, para tratar o erro, que possa ocorrer no try

- Finally – Sempre é executado depois do bloco try/catch. O importante é saber que esse bloco sempre será executado, independentemente se passou ou não pelo try/catch.

- Exception: é uma classe que dentro dela possui exceptions mais especificas, como exemplo a RunTimeException, IOException, NullPointerExteption. Ela é filha direta do pai de todas as exceções a Throwable. Utilizamos a Exception, quando não queremos especificar qual tipo de exceção pode ser lançada.

- Throws – é utilizada em classes para avisar/indicar que a classe pode lançar uma exceção. Ou seja, quando sabemos que uma classe pode lançar uma exceção, mas não queremos trata-la imediatamente com o try/catch, fazendo com que quem chame a Classe com o indicativo Throws, que deverá tratar o erro com o try/catch.

- Throw new – Diferentemente do Throws que avisa/indica que uma classe pode lançar uma exceção, o Throw new, realmente lança uma exceção. Quando queremos tratar um erro, que pode ser devido a regra de negócio, por exemplo um tentar sacar dinheiro, com saldo negativo. Podemos lançar com Throw new e ainda podemos criar nossa própria Exception personalizada, indicando uma mensagem como “Saldo insuficiente”, quando uma determinada ação ocorrer.

16) O que são Classes interfaces? Cite características de uma Interface.

A interface é o tipo de programação mais puro do Java, pois programamos o conteúdo dos métodos de uma interface, programamos apenas sua assinatura, ou seja, apenas sua declaração do que deve conter o método. Ficando responsável por implementar seu método quem for utilizar a interface.

As características de uma interface são:

-Todos os métodos de uma interface são implicitamente públicos e abstratos.

-Todos os métodos de uma interface não possuem corpo, apenas assinatura

-Os atributos de uma interface são, por definição, constantes, ou seja, possuem valor final.

-Quando uma Classe implementa uma interface, a Classe deverá fornecer corpo para todos os métodos da interface, exceto se a Classe for abstrata.

-Uma interface pode herdar de todas interfaces.

-Uma classe pode implementar varias interfaces.

17) Defina polimorfismo

Polimorfismo significa "muitas formas", seu conceito é aplicado quando utilizamos o verbo SER entre pelo menos 2 ou mais subclasses, sendo utilizada em interfaces ou Classes abstratas. O objeto deve passar no teste É-UM onde uma Classe mais genérica terá o comportamento definido através de instâncias de Classes mais especificas.

18) Explique as seguintes interfaces de Collections e suas classes:

a. List (ArrayList)
b. Set (HashSet, TreeSet e LinkedHashSet)
c. Map (TreeMap, HashMap, LinkedHashMap)

a) List (ArrayList) – List é uma coleção ordenada, que ao contrário da interface Set, pode conter valores duplicados. Além disso, temos controle total sobre a posição onde se encontra cada elemento de nossa coleção, podendo acessar cada um deles pelo índice.

O ArrayList é uma Coleção que tem seu pai o List, seu comportamento é de um Array, porém com vantagens das quais, do contrário do Array[] convencional,

não precisamos definir o tamanho como ocorre no Array, tornando-o bem mais prático.

b) Set (HashSet, TreeSet e LinkedHashSet) – O Set está diretamente relacionada com a ideia de conjuntos. Assim como um conjunto, as classes que implementam esta interface não podem conter elementos repetidos.

-HashSet: O HashSet é o mais rápido de todos, esta coleção utiliza HashTable e seus elementos não são ordenados, não importa quanto você adicione ou remova, retire, o tempo de execução sempre será o mesmo. E isso é bom para pesquisa de muitos dados, mas a garantia de continuidade dos itens é zero. HashSet é indicada para alta performance sem se importar a ordem com que os elementos estão ordenados.

-A TreeSet tem como característica que implementa o SortedSet que são ordenados automaticamente, ou seja, independente da ordem que for adicionado, eles serão ordenados. Mas a complexidade de adicionar e remover e os contains são maiores.

-LinkedHashSet tem como característica ser o meio termo entre o HashSet e o TreeSet, ou seja, ela proporciona um pouco de performance do HashSet e um pouco do poder de ordenação do TreeSet. Ele também faz uso do HashTable com linked list, ou seja os dados continuam na ordem que são inseridos.

c) Map(TreeMap, HashMap, LinkedHashMap) – O Map tem como característica o armazenamento de pares, chave e valor. As chaves podem ser duplicadas e são utilizadas para localizar um dado de elementos associados.

-HashMap tem como característica de não ter ordenação especifica e permite valores nulos tanto para a chave quanto para os valores armazenados. É utilizado o método put para adicionar valores.

-TreeMap tem como característica, onde a adição e a recuperação dos dados é igual ao HashMap. Com a diferença é que os dados são ordenados pela chave e apenas os valores armazenados podem ser nulos, mas as chaves não.

-LinkedHashMap tem como característica ter a recuperação de dados iguais à do HashMap e do TreeMap, mas a diferença é que os dados no TreeMap

são ordenados pela ordem de adição dos valores e no mapa e assim como o TreeMap apenas os valores armazenados podem ser nulos, mas a chave não.

19) Explique a interface Comparable

A interface Comparable é responsável por fazer comparações de objetos, quando queremos comparar devemos implementar está interface podendo fazer a sobrescrita do método compareTo();

20) O que são tipos genéricos, dê exemplos

Tipos genéricos é uma forma de não precisar definir quando estamos escrevendo um programa, já o deixando tipado. Por Exemplo, os tipos do Java, como String, Double, Integer. Quando estamos definindo uma lista, List<Integer>, já estamos definindo que o tipo dos dados que compõe a List, é inteiro e caso colocássemos uma String, ocorreria um erro. Pois uma String não é um inteiro. Para situações como esta, definimos como Tipo Genérico, fazendo com que quem vá utilizar o método defina seu Tipo de dado que será utilizado.

Podemos definir como tipo Genérico, utilizando por exemplo List<T>, ou qualquer nome, que não seja exclusivo do Java.

Prática:

21) Crie uma Classe JavaBean para a entidade Produto, contendo idProduto, nome, quantidade e preco. Faça para esta Classe:

- Atributos privados
- Sobrecarga de Construtores
- Encapsulamento
- Sobrescrita dos métodos equals, hashCode e toString

Arquivo – lista01_exercicio21

22) Crie uma Classe que faça a leitura dos dados do produto informado pelo Usuario através da API Scanner. Esta Classe deverá ter métodos de leitura para cada

atributo da classe Produto. Faça a validação dos dados do produto, através de uma classe validadora. Demonstre o Main a execução.
Arquivo – lista01_exercicio22

23) Crie um modelo de classes e interface que permita realizar a gravação dos dados da Classe Produto em arquivos TXT e XML, demonstre no Main a execução.
Arquivo – lista01_exercicio23

24) Demonstre no Main o uso de Collections como List, Set e Map

Arquivo – lista02_exercicio24

25) Crie um programa Java que leia os dados de um funcionário informado pelo usuário, contendo idfuncionario, nome e salário. Este programa deverá exibir os dados lidos, e gerar para o funcionário os seguintes cálculos:

a. Desconto de Transporte: 6% do salário
b. Desconto de INSS:
    i. Salario até 1500, desconto de 9%
    ii. Salário de 1500 a 2500, desconto de 10%
    iii. Salário acima de 2500, desconto de 11%
c. Salario liquido (salário - descontos)

Faça com que o sistema imprima os dados do funcionário bem como o valor de cada desconto e ao final seu salário líquido.

Arquivo – lista01_exercicio25