INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL

Delegação Regional de Lisboa e Vale do Tejo

Centro de Formação Profissional da Amadora

Área de Formação: Ciências Informáticas

Itinerário de Formação: 481039 – Técnico de Informática e Sistemas

Modalidade: Educação e Formação de Adultos

UFCD: UC23 - Programação em Java – avançada

Formador: Rui Almeida

Validado – 29/05/14

Ana Paula Sarandeses

Nome: Luís Caldeira Nº 19 Data:18-05-14

PRA – Reflexão Final do Módulo

UC23

Esta UFCD é a continuação da UC21 e UC22, e serviu para consolidar os meus conhecimentos em Java.

Falámos de "*packages".* Todas as classes que foram estudadas até agora são classes de nível superior, ou seja, estão incluídas apenas em packages, existindo por si próprias e possuindo uma posição particular na hierarquia de classes. Estas classes são definidas em ficheiros de texto específicos que devem possuir o mesmo nome da classe e, ao serem compiladas, é automaticamente gerado pelo compilador de *Java* um ficheiro de código com o mesmo nome, ainda que com outra extensão. Porém, a linguagem Java permite que certas classes possam ser definidas como classes auxiliares à definição de outras, sendo o seu código definido dentro do mesmo ficheiro de definição da classe de alto nível. Estas classes possuem, portanto, definições "aninhadas", ou seja, incluídas nas definições de outras classes de mais alto nível. Designam-se em Java, por tal razão, *inner* classes. As *inner* classes constituem um mecanismo elegante e poderoso de Java, já que

servem de classes auxiliares à definição das classes de alto nível sem que venham a pertencer à hierarquia final de Java, apesar de poderem ser igualmente utilizadas por outras classes para além da que as inclui. É relativamente comum no desenvolvimento de aplicações Java, em especial se forem de grande dimensão, utilizar *inner* classes. A sua importância e utilidade pode ser directamente observada ao analisar-se o código fonte de certas classes de Java. Qualquer grupo de classes pode organizar-se dentro de um "package" porém, evidentemente o mais normal é que se organizem as classes por algum motivo. As classes costumam organizar-se segundo a relação entre elas. Por exemplo, se tivermos um grupo de classes que permitam fazer uma ordenação de elementos num "array" podemos organizá-las deste modo:

Quicksort.class; // Classe para ordenar arrays com o método quicksort

Bolha.class; // Classe para ordenar arrays com o método da bolha

Selecao.class; // Classe para ordenar arrays com o método de seleção

Inserção.class; // Classe para ordenar arrays com o método inserção

/* Podemos englobar estas classes num "package" visto estarem todas relacionadas na sua obrigação. Criaremos para o efeito o "package ordenações" para que se possa acedê-las deste modo: */

Ordenação.quicksort.class; // exemplo acesso a quicksort no package ordenação

*/ igualmente poderíamos ter uma classe de procura de um elemento num array, neste caso repete-se o processo e como exemplo, teríamos o "package tratamento Arrays": tratamento Arrays.ordenacao.Quicksort.class; // exemplo de acesso a quicksort dentro do // package ordenação que por sua vez está // dentro do package tratamento Arrays.

Existem no total 59 packages para organizar o API isto na versão 1.2 de Java.

Realizámos vários exercícios que me ajudaram imenso a entender a linguagem Java. Um dos exercícios que gostei de realizar foi o dos carros.

Se tiver oportunidade de programar em Java numa empresa de renome em Portugal, com certeza que não irei desperdiçar essa oportunidade.

Luís Caldeira