Encapsulation

Technology, Information Technology

Encapsulation Object-oriented programming has four key concepts: ion, encapsulation, inheritance and polymorphism. Encapsulation is the technique used by a programmer to control the extent to which classes hides their implementation details and internal data from other classes. Hiding implementation details allows programmers to rework the implementation details of say a method within a class without changing the code that calls the altered method (Sierra & Bates, 2008). This way, encapsulation protects the data and code of one module from being randomly accessed by other modules. For this reason, information hiding and/or data hiding are terms that are often used to denote encapsulation. Communication between modules therefore can only take place through public programming interfaces, which in this case will be that particular code’s APIs (Application Programming Interfaces). The main reason why encapsulation is key concept of object-oriented programming is that it gives code increased / better maintainability, flexibility and extensibility. This is of critical importance in the development of code libraries. The designers of Java libraries or libraries for any other programming languages want their code to behave the same wherever they are used universally. They would also like to have the freedom to optimize their code without affecting the correctness of other any other code using them. Encapsulation gives programmers the freedom to make changes and improvements to their modules with the certainty that the client code will not be affected by those changes (Eckel, 2003). The most popular facility used for encapsulation in Java is use of access control mechanism which specifies the accessibility of members, interfaces and classes. The accessibility of a member, interface or class is determined in two ways. The first way is the default or package access which comes inherently from the location of where the declaration of a member is done. The second way is the use of any of the following three access modifiers on the declaration statement: public, protected and private (Bloch, 2008). The levels of access control in increasing accessibility is as follows: private access, package or default access (which has no keyword), protected access and public access. Package access applies to any class, member or interface that is declared without specifying an access modifier. Non-nested or top-level classes and interfaces can only use either public access or package / default access modifiers. Instance variables, methods, nested interfaces and nested classes could use any of the following access levels: (1) public access, where the member is accessible from anywhere; (2) protected access, where – subject to a few restrictions - members are accessible from subclasses of the class where they have been declared and from any class in the package they have been declared in; (3) package / default access, where the member is accessible from classes within that package; and (4) private access where the member can only be accessed from within the top-level class it is declared (Bloch, 2008; Eck, 2011; Sierra & Bates, 2005). The example below shows the concept of encapsulation of data by use of access specifiers and accessor methods. class Student { private int age; private String registrationNo; public void setAge(int newAge){ this. age = newAge; } public int getAge(){ return age; } public void setRegistrationNo(int regNo) { this. registrationNo = regNo; } public String getRegistrationNo(){ return registrationNo; } }// end Student Suppose another class, say, Teacher wants to access the instance variables age and registrationNo, in the example above it will not be able to do so directly because they are marked private. To access these member variables, class Teacher will have to use the accessor methods setAge(), getAge(), setRegistrationNo(), and getRegistrationNo(). In this case the accessor methods could be described as class Student’s API. Java differs with other programming languages when it comes to encapsulation on two fronts. Firstly, Java has three access specifiers (private, protected and public) whereas other languages such as C# have five (private, internal, protected internal, protected and public). Most importantly though is that in Java, the rule of thumb is to ensure that members or classes are made as inaccessible / private as possible and to expose only those methods that you want the client programmer to use (Bloch, 2008; Eckel, 2003). In other languages such as C, the rule of thumb is to keep entities as accessible as possible without restriction. References Bloch, J. (2008). Effective Java (2nd ed.). Upper Saddle River, NJ: Addison-Wesley. Eck, D. J. (2011). Introduction to Programming Using Java, Version 6. 0. Retrieved from http://math. hws. edu/javanotes/ Eckel, B. (2003). Thinking in Java (4th ed.). New York: Prentice Hall. Sierra, K., & Bates, B. (2005). Head First Java (2nd ed.). Sebastopol, CA: O’Reilly Media, Inc. Sierra, K., & Bates, B. (2008). SCJP Sun Certified Programmer for Java 6 Study Guide Exam (310-065). New York: McGraw-Hill.