

# Challenges in Teaching Java Technology

**Nedzad Mehic**  
**University of Bahrain,**  
**Bahrain**

[nedzad@batelco.com.bh](mailto:nedzad@batelco.com.bh)

**Yasmeen Hasan**  
**KPMG, Bahrain**

[yhasan@kpmg.com.bh](mailto:yhasan@kpmg.com.bh)

## Abstract

*Object-Oriented Programming-OOP is now replacing procedural programming in almost all applications. Software developers prefer using it because of its many advantages. Therefore colleges should be prepared to make changes to their current curriculums to be able to start teaching programming by using an object-oriented language. This paper describes the usage of Java, an object-oriented language, as a first language taught to computer science students. It goes through the reasons why the language fails when used to introduce students to computer programming. As a case study, the paper describes the current situation at the compute science department at the University of Bahrain and outlines the advantages and disadvantages of changing the curriculum in order to include Java as the first language.*

Keywords: Java, Object-Oriented Programming, AWT, JDBC, Java Servlets

## Introduction

Computer science is a fast changing field that has evolved over the years to cover a multitude of topics. The advances in technology, hardware and software have enabled us to work faster, better and more efficient than we did in the past. An evident change can be seen in the evolution of programming languages, which have ranged from 01 commands to the more sophisticated programming techniques that can now include rich elements of graphics, audio, and video.

It is essential for universities and colleges today to educate its students with these latest technologies and adapt to the changes that occur so that the students can be adequately prepared for the future. The benefits of updating curriculums and allowing room for change in the techniques being used to teach will ensure that the students will get the best of what is currently available as well as helping them build their academic knowledge and later on their careers on a sound basis.

Creating this basis means that a lot of time and effort should be put in selecting suitable courses to begin with

---

Material published as part of this proceedings, either on-line or in print, is copyrighted by the author with permission granted to the publisher of Informing Science for this printing. Permission to make digital or paper copy of part or all of these works for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage AND that copies 1) bear this notice in full and 2) give the full citation on the first page. It is permissible to abstract these works so long as credit is given. To copy in all other cases or to republish or to post on a server or to redistribute to lists requires specific permission from the author.

and defining the sequence and importance of the courses to follow. In a typical computer science department, selecting the programming language as an introductory course to students can be difficult, however selecting the right course pays off later on. The problem many college students face with the first courses they take in a selected field is that these courses are not suitable to be introductory courses. They either are too complicated (many students feel that they are not ready to take them) or they are too simple in the sense that the level of what is being taught is below the level of the students who are being taught.

Many subjects such as chemistry, physics, and math have standard introductory courses that don't change since they are sufficient for teaching students the basics that will be used in their other courses. On the other hand, there are other subjects such as computer science that are ever changing and usually don't have one course that could continually be used as an introductory topic. When it comes to areas like this, it is important to establish a curriculum that most students will benefit from. This curriculum should be easily updated and changed especially at the introductory level without affecting other levels of study. Furthermore, in a field like computer science, students need to be exposed to an adequate programming language at the beginning in which they can learn not only programming techniques but also the correct designing methods.

From our teaching experience, we have found that what students are exposed to at the very beginning of their academic studies has the greatest impact on them during their future studies. Students that started off with learning C lan-

## Challenges in Teaching Java

guage at the university level found the transition to other languages much easier than those who started with PASCAL as a first language. In other words, the concepts and methods taught at the beginning, affected the way students would think and analyze problems. The complexity or simplicity of the programming language played a major role in this. At an introductory level, complex languages with too many restricted conditions limited the students thinking by confining it to a certain set of rules. Although this helped in adding more structure, in the long run it did not add the desired value to the student's way of thinking. From this we concluded that we should decide on an introductory course that would have greater benefits on the students and can be easily adapted in the later courses.

After a serious study of the current computer science curriculum at the University of Bahrain, it was suggested that some changes should be made starting with the first programming language that was taught. Before going through the details of what language was suggested to be selected, it is important to take a look at both the structure of the Computer Science Department at the University as well as the current curriculum used.

### Background

The Computer Science Department at the University of Bahrain was established in 1988 following the merger of the Gulf Polytechnic and the Bahrain College for Art, Science, and Education. Prior to this date, the Electrical Engineering Department at the Gulf Polytechnic was offering a B.Sc. in computer science. The Computer Science Department is now part of the College of Science and it offers B.Sc. in computer science for an average of 600 students majoring in computer science. Each semester an average of 50 students graduate and they are seeking jobs in computer companies, government ministries, and in the education sector in Bahrain and in the Gulf area. They work as programmers, Web designers, systems analysts, and system administrators.

### CS Curriculum

The University of Bahrain is on the semester system which means that courses last for a total of sixteen weeks. The computer science curriculum is divided into four years with each year having three semesters including an optional summer semester. The offered programs focus on programming techniques, database management, operating system concepts, networking concepts, and Internet technology among others. With each year, the Department introduces some changes to the curriculum to ensure that students get the latest information on all the advances in technology within the computer science field as well as

preparing students for the future with new and improved courses.

The Computer Science Department at the University of Bahrain had an already existing curriculum that consisted of several computer science and computer information courses. There are several programming languages taught within the program. At the first level, Pascal used to be the first programming language taught to the students followed by a course in C language. Then after, object oriented programming was taught by using C++. Many students expressed that they enjoyed C programming and were able to grasp its concepts much quicker they did with Pascal. One obvious reason behind this was because it was their second language and they already had a background in programming and programming methods. Although both Pascal and C languages have similar concepts, the complications of Pascal came when dealing with pointers as well as the syntax itself. While on the other hand, students found C language to be a refreshing change that offered many advantages. If students were asked to develop a system or write an algorithm using either Pascal or C in other computer courses, it was found that the majority would use C. From this we concluded that if the introductory programming language should be changed, then the obvious selection would be C. However, C is not an object-oriented language and this led us to look for other options.

With the changes in modern programming methods, more and more focus was given to object oriented techniques. As Boch describes it, "object-oriented programming is a method of implementation in which programs are recognized as cooperative collections of objects, each of which represents an instance of some class, and whose classes are all members of a hierarchy of classes united via inheritance relationships" (Boch, 1998). Bertrand Meyer suggests that "if you agree with its [object-oriented designs] goals and techniques, there is no reason to delay bringing it to your students; you should teach it as the first approach to software development" (Meyer, 1993). ACM IEEE 1991 guidelines recommended to include object-orientation in several places in the curriculum and at least one entire course should be based on these methodologies.

Object oriented programming has many benefits such as allowing programmers to reuse objects already created for new programs that will make the time spent on software development less. In addition to this, maintaining and modifying objects is much easier than procedures. As a result, developed systems are more resilient to change with reduced development risk. Many other advantages were seen in using the object-oriented approach, and now almost all software programmers use object orientation instead of procedural programming. Introduction of object-orientated

programming will result in a paradigm shift from the structured analysis, design, and programming methodologies toward the object-oriented paradigm. The object-oriented technology can make the teaching of software much more effective and more attractive for students than before.

Advantages of using object-oriented programming are:

- Covering all phases of software development by a widely uniform methodology.
- Better preparing students to work as professional programmers developing large programs that are readable, reliable, verifiable, reusable, and maintainable (Decker and Hirshfield, 1994).
- Common conceptual framework, common languages and tools (Knudsen and Madsen, 1996).
- Especially important with software engineering courses, analysis, development and implementation.

When learning object-oriented methods and modern languages, students get a richer programming vocabulary. They will spend less time on complicated details of old fashioned languages, they will be aware of the shortcomings of old fashioned languages, and they will be prepared better for the future.

A proper programming language for teaching computer science should be suited for teaching concepts of modern software technology and abstraction techniques. An introductory object-oriented course must then focus on the essential concepts and techniques, and present the students with clear, coherent set of principles. The notation in the programming language must directly support object-oriented principles; it must be a one-to-one correspondence between the language and the method (Meyer, 1993).

However, it is a well-known fact that structured methods are still the state of practice in industry at the present. Industry is calling for integration of object-oriented concepts, languages, and tools in a software curriculum in undergraduate computer science education. We needed a language that had modern programming practices built-in including safe references (as opposed to pointers), garbage collection, and exception handling (Hong, 1998).

Based on our experience, the following suggestions could be considered when introducing object-orientation methods:

- At least one entire course should be based on object-oriented methodologies.
- One solution could be to include object-oriented programming paradigm along side the currently accepted structured programming paradigm.
- Introduce gradual replacement of structural methods

- Early introduction of object-orientation can affect learning and how students program in later years.
- The focus should be on modeling the world the language specific details are less important.

The next challenge that surfaced was the selection of the language. Since C++ was already in use, it was a natural choice, it had all object-oriented features, the language was simple and all books and material were readily available. The main advantages of C++ language are (Nemirovsky, 1998).

- C++ has features of an object-oriented language, abstract data types, inheritance, subtyping, templates and exceptions.
- C++ is a hybrid language
- C++ supports both procedural and object-oriented approaches to programming.
- C++ is available on a wide range of platforms.
- Support material is available for C++.
- C++ is used in industry

However, at the same time Java was emerging as a new language that not only contained OO concepts but was also being used heavily for Internet programming (Kafura, 2000). This gave it an extra edge.

## Introducing Java

Today, Java is a language that it is being announced as a revolutionary step in computer programming. It solves most of traditional stand-alone programming problems. Java, as a powerful programming language was built to be secure, cross-platform, and international. It is being continuously extended (Sun Microsystems Company is making a major release about every nine months) to provide language features and libraries that handle problems that are difficult in traditional programming languages, such as multithreading, database access, network programming and distributed computing (Bowen, 1997).

During the spring of 1997, the Computer Science Department at the University of Bahrain introduced a 300 level elective course in Java programming. This course was mainly designed for students who had a prior knowledge of object oriented concepts and had already taken Pascal, C and C++. The basic outcome from this course was to let students increase their knowledge of Java as an object oriented language, introducing them to many new concepts and methods. In addition to that it was also meant to familiarize students with a language that could later on be used for data structure course (Standish, 1998), Internet or network related programming (Steflik and Prashant, 2000).

## Challenges in Teaching Java

The course itself was designed in two parts. The first part of the course gave an introduction to previously taught object oriented concepts such as inheritance, polymorphism, and encapsulation. The second part explained all features of Java programming, covering classes, Java applet basics, building user interface with Abstract Windowing Toolkit-AWT, Windows and Networking exceptions, handling data through Java Streams, animation and threads, Java Database Connectivity-JDBC, networking both in theory and practice. In each semester following that year, more and more topics were added to the course. Some of the most recent additions that are being covered now include topics such as Swing, Java Beans (Monson-Haefel, 2000) (Roman, 1999), Java Security (Jaworski and Perrone, 2000), Java Servlets (Moss, 1998) (Ayers and Bergsten, 2000), Corba (Vogel and Duddy, 1998), and Java Wallet.

There are around 60 students in two sections of the course per semester each with around 30 students. The Department has a limited number of teaching assistants who handle multiple lab sections and grade the programs submitted by students. The instructor sets the course topics, selects the software tools used in the course, creates and grades course projects, quizzes, the test and final exam. Class time occupies four hours and thirty minutes each week throughout the semester including two, one-hour and fifteen minute lecture sessions each week. There is a two-hour computer lab session each week (except the first week). Lab attendance is compulsory because sometimes part of a lab session is devoted to discussion activities. The students are given programming assignments every week in order to give them more exposure to programming concepts and to the programming language. There is also an optional extra programming assignment that can replace the lowest program grade.

Table 1 gives the course evaluation information.

Lab Assignments	15%
Project	20%
Midterm	25%
Final Exam	40%

**Table 1: Course Evaluation**

The course midterm and the final exam have two parts:

- One hour, closed-book, theoretical part
- Two hour open-book part where students have to solve a practical problem in Sun Microsystems' Java Development Kit-JDK in Microsoft Windows or Solaris SPARC environment.

A student is expected to submit a lab assignment with a short report with high-level overview of the program with

class and method definitions, the data structure and an electronic copy of the program. A course project requirements include a Project Report with appropriate presentation, and submission of an electronic copy of the project program usually on a CD.

Using Java, instructors can cover more material and can expect that students understand it. With C/C++ they had a very hard time getting beyond pointers. Java does not use pointers, only references and it has a standard library that students have on all platforms. It is important to note that runtime errors such as bad subscripts, null pointers, and uninitialized variables cause exceptions that are caught by the language at runtime without causing crashes. Java Strings are much easier to use than char \* arrays. Java garbage-collected storage management is an extremely important feature that makes it a developer-friendly language.

From what was seen, Java had all the elements of a strong programming language that was in many ways easy to implement as well as having many advanced features that made sophisticated programming attainable. Java is the first widely used programming language with language-level support for concurrency (threads) and synchronization (monitors), which considers being a big advantage.

Therefore, it was suggested that this language could be introduced as the first programming language for computer science students at the University. Before implementing this, a study was conducted to find out if it would be the most appropriate language to introduce students to computer science and would it prepare them properly for future studies in different languages. In addition to this, all the problems that were encountered with Java as a programming language were also pointed out. In the next segment, we will firstly describe the benefits of using and Java and then illustrate the weaknesses of the language.

## Advantages of Java

### *Language of the Internet*

Using Java, and more specifically Java applets, programmers are able to design dynamic Web sites, which have many animated effects that could respond to any interaction from the users. This helped in designing better websites that could fit a large range of applications and that had a bigger impact on people who visited them. Bringing a language such as Java to students not only means teaching them a new programming method but it would also introduce them to many designing techniques, the Internet, and network applications. Furthermore, if they were exposed this early on, it would have a better effect on their future programming skills.

## **Built-in libraries**

Java is an object-oriented language, and as mentioned before this means that it deals with objects that can be reused many times in different programs. In addition to this Java has its own set of objects under a built in library. This allows programmers to develop sophisticated programs using complicated algorithms easily. One of Java's strong points is its graphics utility (Ammeraal, 1998). When a certain shape wants to be drawn, all is needed from the programmer is to input the coordinates of the shape to insert it in the desired location. This is done by importing the graphics package from a library and then using the methods under that package to draw that shape. This also applies when inserting images and sound.

Although later on it is important to understand what lies behind each command and how each procedure functions. For first year students this doesn't matter that much. With Java we are able to show students how to design applications properly and how to build programs with the correct techniques and methods in mind, while at the same time shielding them from the complications of the programming language itself. However, at any time one can go through the details behind each package and procedure.

## **Portable**

The main difference between Java and any other languages is that fact that it is platform independent. Code written in the Java programming language will run on any platform, including the large numbers of PCs and Macintosh systems deployed today incorporating Sun Microsystems slogan "Write Once Run Anywhere". Many content developers are moving towards Java technology, using applets written in the Java programming language to create animated and interactive Web applications.

Students can benefit from this feature since it exposes them to more than one type of operating system by allowing them to run their same programs on different platforms. This helps them to understand the different way each operating system works, especially when running programs that have multithreading.

These are the strengths of Java as a programming language. However, the purpose of the study was to select a language suitable as a first language to be taught to computer science students and from what was seen, there were some problems with the language that prevented it from being taught at an introductory level.

## **Problems with Java**

### **Input and Output**

It is important to note that in Java, output can be created simply using `System.out.println()`. However, for input you either have to use `System.in.read()`, or create an instance of a `DataInputStream` or an `InputStreamReader`. In either case, you will have to catch a possible `IOException`, which would involve teaching the students about exceptions in order to do something as simple as reading in a single character (Hong, J. 1998).

Many students would agree that Input and Output is the most difficult part of Java programming. Since Java was mainly designed for Internet programming the developers of the language didn't really focus on the I/O. For this reason, many feel that it is too complicated.

Furthermore, before being able to explain this, students must have an understanding of the concept of streams as well as being able to distinguish between the different types of I/O. Having to deal with several methods such as `FileWriter`, `DataInputStream` and `BufferedWriter` just to name a few is a hindrance. It can be confusing to students who are still at the beginning of the learning process. Therefore, the I/O part is usually to the end of the course, where it could be easier for the students to follow the concepts.

### **Availability of Books**

Although there are many books available on Java, not many of them are written in an introductory manner. Most of these books can be used as reference material to programmers that already know how to program in Java, but they can not be given to students just beginning to learn the language. During all of the semesters that Java was taught at the university students rarely used the available books, they would depend more on the course notes given to them by their lecturer. Although one can find some free of charge online tutorials nothing can actually replace a good textbook.

Furthermore, at an introductory level, students don't just need a book to teach them the concepts of Java programming; they also need a book that will explain the fundamentals of software design using Java. This will help them to develop the needed skills to create good programs using the correct techniques. However, most Java books lack that part, they just jump into the programming details. This can create some misunderstanding of design concepts for the students, which in turn will effect all of their future software developments.

## Challenges in Teaching Java

### Keywords and Syntax

The beauty of C and C++ has always been in the simplicity of the syntax used. There were no complications and students could usually grasp its concepts from the first week. However, Java is a different story. There are many difficulties to Java when it comes to the syntax. The first thing to be noticed is the long keywords that are used. In many cases, students cannot memorize all of them and need reference books when they develop their programs. Without a book, they usually encounter errors that deal with wrong keywords that are usually misspelled. Furthermore, the way that Java was designed makes it difficult to use as an introductory language.

Take for example the first line in a Java application (Hong, 1998):

```
Public static void main (String arg[])
```

For a student to understand this, he/she must first understand all concepts dealing with modifiers, classes, methods, sending parameters and arrays. Since this too complicated to teach first level students all at once, instructors would usually give this command and ask students to use it as it is. Later on each one of these concepts were explained.

### No help available

One way students can learn languages, is to go through the help files that are available. Whenever they have a problem with compilations or an error in one of the keywords, they should be able to solve their problem themselves by finding what it might be. However, Sun's JDK doesn't include any help libraries. Without this many students find it difficult when they are solving their assignments in the lab. In many cases students become frustrated with the continuous error messages that they don't know how to correct.

### Many additions

Finally, a major problem with teaching Java at any level is the subject matter that should be included. There are always new additions to what already seems to be a large amount of topics. Unlike other languages that go through a standard outline for teaching the subject, Java can have many different approaches; the question is which one is the best.

Furthermore, since Java is relatively new, there are always new updates and better versions that are used. Sometimes these versions include new classes and packages that replace the old ones. An example of this would be the Swing class that is replacing the AWT class as the graphics package. These updates means that instructors need to be able to grasp their concepts as soon as they come out, and be

ready to teach them with the beginning of the next semester.

In addition to this, Java is now becoming a strong language that is covering all aspects of programming. Therefore some of the additions are very advanced and cannot be taught at a beginning level. These topics include Java Servlets, which are used for server-side programming; Java beans which feature reusable objects and the Java framework that is used with e-commerce applications.

## Conclusion

The paper discusses a major dilemma in selecting an introductory course that should put emphasis on major concepts, principles and techniques, but will have a positive affect on other courses taught at the computer science department. The introduction of object-oriented courses is a response to increasing demand from industry for object-oriented methods to be included in undergraduate computer science. It has been a major problem for many computer science departments to successfully incorporate object-oriented methods in their curricula. The selected programming language should encourage computer science students to use the same concepts through analysis, design, and implementation of different systems that will overcome limitations of traditional methods of building the systems, which are reusable and more resilient to change.

Although Java is a powerful language that has many advantages over other programming languages, it doesn't seem to be the choice for a first to choose from as a first programming language (Bohnenberger, 2000). The complications that come with it and the difficulties students face are all arguments that show why it can't be used. Students who already know at least one programming language and that understand some levels of software design, would be able to take Java and utilize it as a good language for developing programs, whether they are simple applications, web page applets or have more complicated functions such as connecting to databases. However for students who are new to computer science, it can be very difficult. As a first programming language students need something that has easily understandable concepts and can be used to teach good designing principles as well. The first-year programming syllabus covers the principles of programming, from the concept of a variable up to selection, iteration, files, arrays, and simple pass-by-value functions. It is currently taught through C/C++. It will be a long time before we are able to transform Java into an introductory course. Maybe with the changes to the languages and additions to it, this transformation could happen, but up to then the best thing is to start with C/C++ and then move gradually into

Java once the students have grasped most of the programming concepts.

Currently the Computer Science Department at the University of Bahrain has not taken to teaching Java at an introductory level, but it recognizes the importance of teaching the language at a higher level. In addition to the first Java course that covers applets, applications, networking, databases, graphics and animation, another course is being introduced to cover the more advanced features of Java.

The impact Java has on the education process will be seen in the near future when trained students start entering the market with the knowledge and skills that can be utilized to build many dynamic and creative applications especially for the web. It is also important to note that most of the focus today is given towards e-commerce and e-business applications. Governments, banks, small companies are just a few of the sectors which are trying to move towards transforming their business online, and for that they need a language which will help them with this transformation as well as developers skilled in that language. Java together with XML (Nakhimovsky and Mayers, 1999) (Daconta and Saganich, 2000) have provided the perfect solution to all who wish to join in the e-commerce boom and this gives us yet another reason to teach not only the basic concepts behinds the language but also to provide a complete course which includes all the advanced features that the language can offer.

## References

- Ammeraal, L. (1998). *Computer Graphics for Java Programmers*. John Wiley and Sons.
- Ayers, D., & Bergsten, H. (2000) *Professional Java Server Programming, J2EE Edition*, Wrox Press Ltd.
- Boch, G. (1998). *Object-oriented Analysis, and Design with Applications*. Addison-Wesley.
- Bohnenberger, K. (2000). *Java Pitfalls*. Wiley Computer Publishing.
- Bowen, B. D. (1997). Educators Embrace Java. *ITworld.com, Inc. Magazin*
- Daconta, M., & Al Saganich, A. (2000). *XML Development with Java 2*, SAMS Publishing. Company.
- Decker, R., & Hirshfield, S. (1994). Top-down teaching: Object-orientation programming in CS1, *J SIGCSE Bulletin*, 25(1), 51-55.
- Hong, J. (1998). The Use of Java as an Introductory Programming Language, *ACM Cross-roads Magazine*.
- Jaworski, J., & Perrone, P.J. (2000). *Java Security Handbook*. SAMS Publishing Company.
- Kafura, D. (2000). *Object-Oriented Software Design, and Construction with Java*, Prentice Hall.
- Knudsen, J.L., & Madsen, O.L. (1996). Using object-orientation as a common basis for system development education, *J SIGPLAN Notices*, 31(12), 52-62.
- Meyers, B. (1993). Towards an OO-Curriculum, *Journal. of Object-Oriented Programming*, 585-4.
- Monson-Haefel, R. (2000). *Enterprise Java Beans*, O'Reilly Publishing Company.
- Moss, K. (1998). *Java Servlets*, McGraw-Hill.
- Nakhimovsky, A., & Mayers, T.(1999). *Professional Java XML Programming with Servlets and JSP*, Wrox Press Ltd.
- Nemirovsky, M. (1998). *Is Schrödinger's cat object-oriented?* Taligent, Inc, Accessed Dec. 1998 at <http://www-4.ibm.com/software/developer/library/oobuilding>.
- Roman, E. (1999). *Enterprise Java Beans and the Java 2, Platform Enterprise Edition*, John Wiley and Sons.
- Standish, T.A. (1998). *Data Structures in Java*. Addison-Wesley.
- Steflik, D., & Prashant, S. (2000). *Advanced Java Networking*. Prentice Hall.
- Vogel, A., & Duddy, K. (1998). *Java Programming with Corba*, Second Edition, John Wiley and Sons.

## Biographies

Nedzad Mehic is Associate Professor at the Computer Science Department of the University of Bahrain where he teaches Java and Internet based courses. Dr. Mehic is also Director of the Information Research Center at the Dean-ship of Scientific Research of the University of Bahrain.

Yasmeen Hasan graduated from the University of Bahrain in June 2000 with a degree in Computer Science. As a student she was a lab assistant for the Internet Concepts course and the Java Concepts Course. Her graduation project was on the implementation of Java Servlets. Currently she is working as an IT consultant at KPMG.