Journal of Information Systems Education, Vol. 16(2)

Teaching Tip

Hands-On Testing in Visual Basic Courses

Qi Yang
Joe Clifton
Department of Computer Science and Software Engineering
University of Wisconsin at Platteville
Platteville, WI 53818, USA
YangQ@uwplatt.edu Clifton@uwplatt.edu

ABSTRACT

There are some problems with traditional written tests in Visual Basic courses. Well-designed hands-on tests can be used to effectively assess and challenge the students' programming skills. We have found that students welcome the challenge and respond positively to this method of testing.

Keywords: VB.NET, RAD, Traditional Tests, Hands-On Tests

1. TEACHING VB.NET

Our department offers a fairly standard computer science curriculum. We offer two Visual Basic (VB) courses as electives: Programming in VB and Windows Programming. VB.NET is used in both courses. A major goal of these two courses is to prepare the students for Rapid Application Development (RAD). As in any course, an important issue in teaching is how to test the students' programming skills. When the VB courses were first taught, traditional written testing methods were used. However, these did not work well and the students had complaints. There were two main reasons:

1. VB.NET provides a lot of controls; each control has many properties, methods and events. It's very difficult to memorize all the details.
2. The IDE of .NET provides some nice features to help VB programmers to develop programs rapidly without memorizing the syntax details. The Code Editor displays a red wavy line under an incorrect entry so the programmer can recognize and correct it quickly. IntelliSense displays a context-sensitive dropdown listbox with all relevant property and method choices. The programmer can select the desired item without typing. Pointing to an item in the list displays a brief description to aid in decision making. After selecting a method, the parameter list of the method will be displayed to help the programmer complete the statement.

A student can be a very good VB programmer without memorizing details and consequently may not do well on written tests.

2. HANDS-ON TESTING

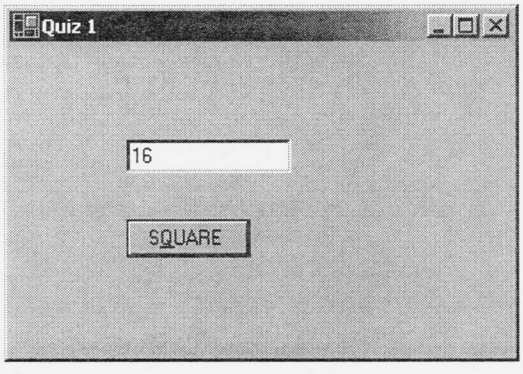
One solution to the above problem is to use hands-on tests instead of written tests. The primary goal of these tests is to ensure that the students can create VB programs proficiently. Traditional computer science and programming concepts are well tested in our required CS1 & CS2 courses. Thus, our VB courses can be more focused. We conduct hands-on tests as follows:

1. The tests are carried out in computer labs, with one programming problem given for each test. Students work individually to create a running solution to the problem. The finished executable is submitted to a write-only instructor "dropbox." The source code is submitted only in special cases.
2. All hands-on tests are done individually, with no discussion allowed. However, the students have open access to all resources they want to use, including textbooks, class notes, and programs they have completed in the class. The entire .NET help system and Internet is also available. One exception is that students are not allowed to use email.
3. If a student is unable to make progress, the instructor can be asked for help; however, the student will lose some points. The penalty is about 10% of the total points on the test each time a student receives help from the instructor.
4. A sample executable program is available to the students so they can see how the program should run.

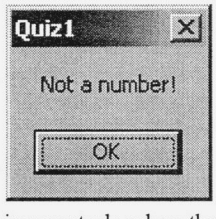
3. TEST DESIGN

A hands-on test should be divided into a sequence of steps such that a student can do it one step at a time and earn points on each step after it's completed. Using VB.NET, programs can be run after completion of any portion, so students don't have to wait until they complete the entire program. This provides a very nice environment for the hands-on tests.

The following program is used in the first quiz for the first VB course. The program has one form with one textbox and one command button. When clicking on the command button, the square of the number in the textbox is displayed



in the same textbox. If the content of the textbox does not represent a number, a message box shown at the right is displayed. The quiz is worth 20 points:

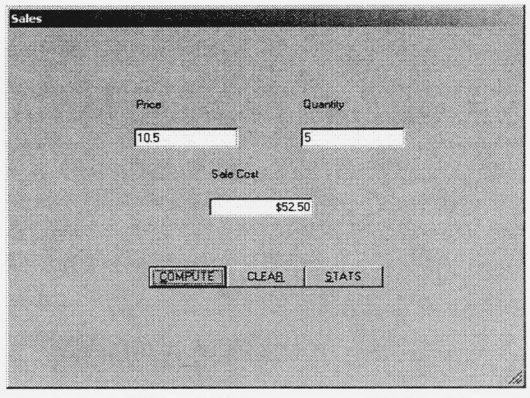


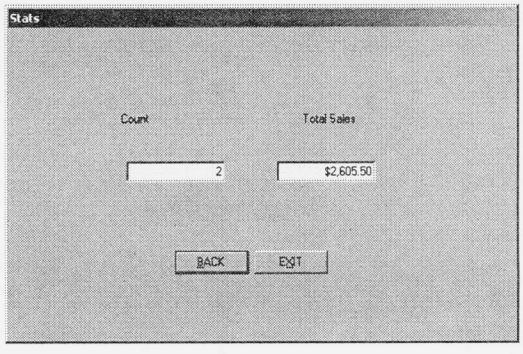
- four points for a form with the correct title and controls
- seven points for properly arranging controls when the user resizes the form
- six points for proper handling of the click event of the command button
- three points for creating an executable file and correctly submitting it

The most difficult part of the quiz is writing the Resize event procedure to arrange the controls dynamically. However, a student can still receive up to 13 points even if they cannot write the Resize procedure.

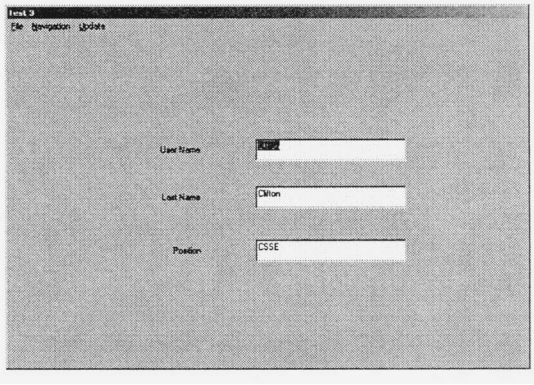
In a later test, the students must create a GUI program with two forms. The program computes sales cost based on price and quantity. It also keeps track of the total sales and a count of the number of purchases. The test is worth 25 points and there are 13 steps. Only four points are for computing and displaying the correct cost value. Three points are for updating the count and total values. The remaining steps are worth one or two points each.

For the final test in the first VB course, the program displays data from a database, one record at a time. The





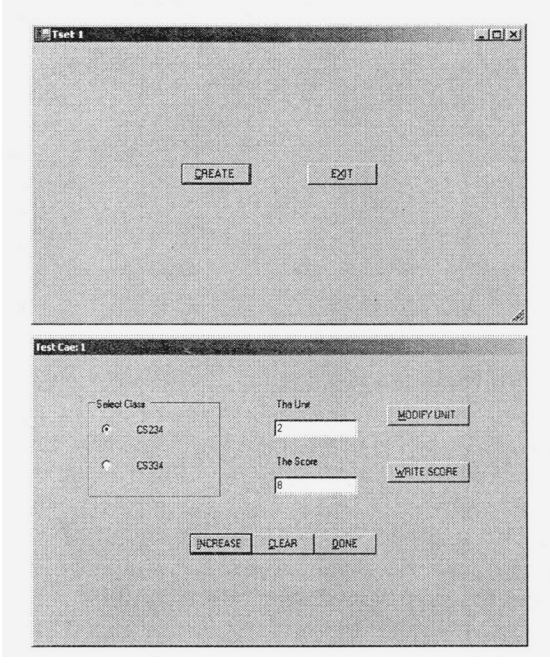
main form has some menus to allow the user to navigate and update the database. The test is worth 35 points, with a focus on database access. Fifteen points are for connecting to the database and displaying records, ten points for the menus, and ten points for the remaining steps. The points are further divided within each category.



In one test of the second VB course, we ask the students to create a program with two forms. The program has one base class and two sub-classes. Clicking on the Create button on the first form will create a new instance of the second form. The second form is used to test the two sub-classes. The interface of the base class is defined as follows:

- Public properties: Score with a range and Unit used to increment Score

- Public method: Inc to increase Score by Unit if the new Score if valid
- Public Event: PerfectScore which is raised when Score reaches the maximum value



The test is worth 50 points. The focus is on classes and inheritance. Since it is possible to create such a GUI program without classes, the students are also required to submit their source code. Seventeen points are for proper construction of the classes. The remaining points are for the front-end program and divided amongst the functionality of the command buttons and radio buttons.

4. GRADING

Most tests can be graded simply by running the executable and verifying the correct functionality for each step. If the tests are well designed, the grading can be done relatively quickly. Contrast this to a written test in the case where a student's solution varies significantly from the instructor's solution. Oftentimes the latter requires the instructor to enter in the solution and test it on the computer to verify correctness.

Students must produce running programs. No credit is given for programs that don't run. Although this puts extra pressure on the students, it does force them to pay close attention in class and work hard to prepare for the tests. In the end, we feel that this makes the students better VB programmers.

Students rarely complain about their scores, since they lose points only when they can't make the program run as required.

The hands-on tests are not graded for documentation and style; however, neither were the written tests. These aspects of programming are graded for on the programming assignments and projects. Time constraints during the test period dictate that students concentrate on the tasks to be performed.

5. FEEDBACK

Many students have expressed their approval and support of the new approach. They agree that hands-on tests assess their programming skills, not their memorization skills. Students rarely complain about their scores, since they lose points only when they can't make the program run as required.

Overall the students receive better grades on hands-on tests versus written tests. This is not to say that the hands-on tests are easier but rather that the focus is better suited to the objectives of the course. Since RAD and proficiency in VB programming are the major objectives, hands-on tests are a more natural assessment. Furthermore, we believe that completion of such tests in itself is evidence that the objectives are being achieved. Being able to create significant programs from scratch within the pressure of a one-hour timeframe is no simple task!

Although not direct evidence of the effectiveness of the hands-on testing approach, one positive point of feedback comes from the VB Programming contest held at the AITP National Collegiate Conference. In 2003 we had a team that finished third out of 43 teams and in 2004 we had a team that finished second out of 42 teams.

When we first tried hands-on testing, students were not allowed to ask any questions of the instructor. A common complaint was that a student who got stuck early lost all subsequent points, even if that material were understood. Allowing the student to ask the instructor questions at the cost a few points was a simple and fair solution that is well accepted. In practice, only a few students ask for help on any given test. We have never had a student ask for help twice on the same test.

Another complaint that is still voiced occasionally is that some students feel too much pressure during hands-on tests. Usually, these students will fail and drop the course. However, the number of such students is very small, and generally smaller than the number of students who typically do poorly on written tests.

6. CONCLUSIONS

Hands-on testing is a very effective approach in our VB courses, where RAD is the main goal. Most students like the approach and enjoy the challenge. They work harder to pass the tests and become better VB programmers.

Grading hands-on tests is easy and fair, and the grades better represent the students' programming skill than that from the written tests.

AUTHOR BIOGRAPHIES

Joe Clifton is a Professor of Computer Science and Software Engineering at the University of Wisconsin - Platteville. He is the Program Coordinator for the Software Engineering program. He earned his BS degree from the University of Wisconsin - Platteville and his Ph.D. from Iowa State University. His research interests include software engineering and object-oriented technology.

Qi Yang is an assistant professor in the Department of Computer Science and Software engineering at the University of Wisconsin - Platteville. He received his Ph.D. from the University of Illinois at Chicago. He has published several journal papers including one in ACM Communications and one in IEEE Transactions on Knowledge and Data Engineering. He also has a number of conference papers, including IEEE International Conference on Data Engineering and ACM Computer Science Conference.

Information Systems & Computing
Academic Professionals

STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 1055-3096