2009

An investigation into student reactions towards rad versus traditional programming environments for novice developers

Pansy Colkers
Edith Cowan University

Edith Cowan University

Copyright Warning

AN INVESTIGATION INTO STUDENT REACTIONS TOWARDS RAD VERSUS TRADITIONAL PROGRAMMING ENVIRONMENTS FOR NOVICE DEVELOPERS

By: Pansy Colkers

Student ID: 10093283

Honours Thesis

Semester 2, 2009

Supervisor: Dr Justin Brown

USE OF THESIS

The Use of Thesis statement is not included in this version of the thesis.

ABSTRACT

The traditional approach to programming using text editors is widely used in many institutions to teach introductory programming. These types of traditional programming environments provide fundamental programming concepts for learning, especially in the context of novice developers.

In recent years, teaching institutions have seen a trend towards the introduction of visual "drag-and-drop" rapid application development (RAD) environments for teaching novice programmers. These environments capture student interest in programming by allowing the construction of workable programs within a short time frame based on minimal pre-existing coding knowledge. However, some have argued that these visual RAD environments might not be suitable for providing fundamental programming concepts and syntax to novice developers.

This research examines student perceptions towards visual RAD environments in comparison to traditional environments for learning programming for novice developers, mainly focusing on the novice developer's "first" programming environment. To gather student reactions towards these programming environments, surveys, interviews and workshops were conducted with novice, intermediate and expert level student programmers. The results indicate that while visual RAD environments managed to capture the majority of the participants' interest, the traditional approach was largely accepted as the most appropriate "first" environment for novice developers. Another finding from this research is the participants' perceptions of the key aspects of learning programming, which also formed part of the deciding factors for the "first" environment. Understanding the underlying concepts, syntax and logic of the program seem to be the most important aspects followed by interest level and the ability to build workable programs quickly. The majority of participants perceived that traditional programming environments could help novice developers with understanding underlying concepts and syntax better than visual RAD environments. Although visual RAD environments do not require a traditional programming environment at the early stage of programming, the latter would become necessary as the program grows and more complex functions are required. Overall, the visual RAD environment was still the preferred environment for development despite the lack of pedagogical benefits compared with traditional environments.

COPYRIGHT AND ACCESS DECLARATION

I certify that this thesis does not, to the best of my knowledge and belief:

(i) *Incorporate without acknowledgment any material previously submitted for a degree or diploma in any institution of higher degree or diploma in any institution of higher education;*

(ii) *Contain any material previously published or written by another person except where due reference is made in the text of this thesis; or*

(iii) *Contain any defamatory material.*

(iv) *Contain any data that has not been collected in a manner consistent with ethics approval.*

The Ethics Committee may refer any incidents involving requests for ethics approval after data collection to the relevant Faculty for action.

Signed .....

Date 13/02/2010

Table of Contents

Table of Figures

Table of Tables

1. Introduction

"Programming is a cognitively challenging task and training novices can be a challenging undertaking" (Raadt, 2008, p. 19).

Programming is not an easy subject, especially for students new to the field. According to Teague and Roe (2008), the failure rate for introductory programming courses has been consistently high over the past five years compared with that of other courses such as database systems and professional studies. The enrolment and retention of computing students has also decreased in recent years (Clear et al., 2008), though of course there are likely to be other factors affecting the failure rates and dropout rates beyond just the content difficulty. Though many studies and different approaches to teaching programming have been conducted in order to improve the quality of the introductory programming courses and quantity of qualified programmers, little research exists in the literature that addresses the question of what type of programming environment should be introduced first to novice programmers. This research examines two types of programming environments for novice programmers for web application development, a traditional programming environment and a visual rapid application development (RAD) environment.

Programming environments are tools that assist programmers with creating and editing software applications and they can have major impacts on the ease and effectiveness of learning programming languages (Vogts, Calitz & Greyling, 2008). Traditional approaches for introductory programming courses, featuring console-based programming exercises with traditional programming environments, have been challenged in terms of their relevancy within the modern programming industry given the difficulty level and motivation factors in comparison to visual programming environments (Schaub, 2009).

Many commercial and open-source visual programming environments are available in today's market and widely used in many institutions to assist in teaching programming languages. These programming environments are referred to as visual RAD environments and most of the application development tasks can be completed with "drag and drop" actions. The term visual RAD is used in this thesis to describe a

programming process where a majority of the development takes place using drag and drop components that are integrated using a visual interface. Most visual RAD environments allow for different levels of abstraction, from looking at a component visually to exposing its functionality via a textual interface. This is seen as different to purely textual development systems, which provide little or no visual representation of objects and their functionality. Although visual RAD allows for ease of implementation and rapidity to some extent, the complex features and hidden programming principles make it unclear as to whether it is a suitable first environment for novice programmers (Pears et al., 2007; Schaub, 2009). Conversely, traditional programming methods, also known as hand-coding or textual-programming, provide the flexibility and knowledge of programming concepts that visual RAD might not be able to provide (Wong, 2006). It may be that the syntactical nature and non-visualisation of the traditional programming environment make it difficult for novice programmers to write a complete and error-free application (Chainini & Yamada, 1998). This research aims to examine some of these issues by investigating student perceptions of using visual RAD environments in comparison to traditional environments in learning programming.

1.1. Background to the Study

The approach to application development is changing in the information and communications technology (ICT) industry, and companies rely on rapid and robust application development environments to hasten the design and implementation of software systems (Agarwal, Prasad, Tanniru, & Lynch, 2000). It is unsurprising that many universities adopt similar environments to teach programming languages to students because in most universities it is the relevance of technology used in the industry, rather than the pedagogical benefits of learning, that drive such decisions (Mannila & Raadt, 2006; Pears et al., 2007; Raadt, Watson & Toleman, 2002, 2003). Learning programming is often cognitively challenging, complex and requires knowledge and skill in execution (Vogts et al., 2008; Weir, Vilner, Jos, & Nordstr, 2005). Programming environments are necessary for programmers to write, compile and execute applications and perhaps have a significant impact on the process of learning programming for novice programmers.

Textual programming, also known as traditional programming, is widely used in various institutions in teaching introductory programming courses according to the study carried out by Raadt et al, (2002). Many professional programmers prefer traditional programming methods over RAD tools for the reason that it provides high levels of fidelity (Kyrnin, n.d), that is, the ability to control and manipulate every aspect of the program's execution and function. Programmers have full control of the application and they get exactly the result for which they code (Agarwal et al., 2000). Traditional pathways of learning programming may also have some influence on the preference for programming environments. A typical learning pathway for a developer may start with traditional programming before progressing to the RAD tools at the later stage of the learning phase (Schaub, 2009). Traditional programming environments focus on teaching a programming language, whereas RAD tools focus on using programming to implement an algorithm (Calloni & Bagert, 1994; Schaub). Having to incorporate the syntax and logic to create a functional application is challenging for introductory programming students. The frustration for novice programmers usually lies in the syntax errors where a small little dot, '.', can make a big difference to running a program successfully, and in the difficulty of locating and correcting the faulty logic (Chainini & Yamada, 1998) while still trying to learn the logic. To minimise the effort required to produce the working program, different types of RAD tools are created and exploited in today's programming field.

To increase the rapidity of application construction, programming environments with pre-built functionalities and visual presentation of coding and processes have been developed. These environments, or tools, are referred to in this thesis as visual RAD tools and their functionality and component capability over the past two decades has improved. However, their extensive sets of features and concepts make them challenging to adapt to, or make effective use of, not only for novice programmers but also professional programmers (Agarwal et al., 2000; Pears et al., 2007). There are mixed reactions in both the industry and teaching institutions to the feature sets of visual RAD tools. Being able to show the prototype to the customer within a short period of time is one of the major advantages that visual RAD tools can provide to companies (Agarwal et al.). Kaneshige (2009) however, argues that visual RAD tools are not as easy to use as they are claimed to be. Figuring out where and why an error occurs within a visual RAD environment can require the knowledge of a seasoned developer. The pre-built components and functions make visual RAD tools valuable and increase the

expectations of customers and management but somewhat limit the scope of what a programmer can do to provide the customised functionality that software consumers may demand (Agarwal et al.; Peter, 2009). In addition, visual RAD tools are often considered to be "anti-quality" due to the trade-off between speed and quality. For some, visual RAD is considered "Rough and Dirty" (Howard, 2002, p. 27). In terms of the pedagogy of programming language, some instructors believe that the use of visual RAD tools hinders or masks the basic principles of programming (Raadt et al., 2002). The novice programmer can build a functional application almost at the first try without the knowledge of syntax and rules of the programming language underlying the actual environment (Goldweber, Bergin, Lister & McNally, 2006).

Calloni and Bagert (1994), Calloni, Bagert and Haiduk (1997) and Cilliers, Calitz and Greyling (2005) have experimented with the use of visual RAD tools in introductory programming courses. These attempts have been successful, leading to a significant increase in students' grades, but it was not clear if this approach helped novices to become real programmers or whether it was limited to just an improvement in the final grades. Another undetermined factor from these studies was that they have not yet defined which programming environment should come first. This research mainly focuses on the students' perceptions of these programming environments in the web application development environment and their reactions to the learning sequence.

1.2. Purpose and Rationale of the Study

The main purpose of this research is to examine the impact and selection of programming environments on the teaching and learning of programming languages for novice programmers in the area of web application development. This study aims to improve the learning experience of programming by discovering the student point of view on different approaches and the impact of the sequence of programming environments on the novice developer. This research also focuses on the preferences regarding the first environment of students when defining the learning pathway of a novice web developer.

1.3. Definitions of the Terms

For the purpose of this study, the following definitions are used:

Visual rapid application development (RAD): RAD is a visual, drag and drop programming environment for application developments. It is often considered to be a "codeless" environment with visual representations of functionality without the user needing to physically write the program code. As stated previously, in the context of this thesis visual RAD is any environment that provides a visual representation of coding objects and how they interact with other objects. Microsoft's Visual Studio and the NetBeans environment could be considered prime examples of visual RAD development systems, although they can also be programmed using a textual interface.

Traditional programming: The development of applications by hand coding or writing in textual syntax using a text-based, non-visual interface. A traditional environment is considered to be one where there is no visual representation of objects of any kind. Developing an application in vi or any other text editor system would be considered a traditional approach.

PHP (Pre-Hypertext Processor): PHP is a widely used, general purpose scripting language that is especially suited to web development and can be embedded into HTML.

ASP.Net: (Active Server Pages): ASP.Net is a server-side script engine for dynamically generated web pages run within the Microsoft .Net Framework.

ICT: Information and communication technology.

Visual Studio: Visual Studio is a multi-purpose development environment for all types of applications, including web-based systems. Visual Studio places an emphasis on visual development but also allows the developer to switch to a code-based environment.

Sandstone universities: Sandstone universities are tertiary education institutions in Australia that were established before the 1950s (Ashenden & Milligan, 1999).

Workshop: A workshop in this research is referred to as a classroom or online-based learning area where students perform practical programming exercises.

1.4. Statement of Research Questions

The primary research question of the study is:

"What is the student reaction to visual RAD versus traditional programming environments for novice programmers in a web application development context?"

Three supporting questions were defined in order to address the outcomes of the primary research question.

As the focus on GUI-based applications in the programming industry has increased, visual programming environments are becoming more popular in first-year introductory programming curricula. The first supporting question examines the impact of choosing visual RAD as the "first-environment":

"Should visual RAD environments be taught as the 'first environment' to novice programmers?"

It is apparent in traditional programming environments that the majority of functionality has to be built from the "ground-up", and novice programmers have to learn everything from the syntax to the structures and principles of programming. In visual RAD environments, the main focus is on the knowledge of how to use the pre-built components. This leads to the second supporting question:

"Does visual RAD require pre-existing traditional programming knowledge?"

The third supporting question aims to examine the preferences of the students for different programming environments:

"Which is the preferred programming environment among novice developers?"

1.5. Significance of the Study

This research focuses on the areas that are important for the future design of introductory programming courses, an issue that remains relatively undeveloped in the literature. These areas include the visual RAD environment as the first programming experience, pre-existing programming knowledge for visual RAD and the attitudes of novice programmers towards different programming environments in web application development. Little research has been carried out on the impact of visual RAD tools in introductory programming courses for novice programmers in the context of web application development. This research also focuses on another underdeveloped area of the literature, that being the importance of the teaching sequence in traditional versus visual RAD development environments.

While this thesis focuses on web applications development, the results can largely be generalised for other types of software development where a choice needs to be made between a traditional or visual method of development.

2. Literature Review

"Computer technologies are no longer seen as intellectual products and tools for only a small community of specialists, but as useful tools for masses" (Pham, 1996, p. 149).

Over the past four decades, computer programming as part of computer science has evolved significantly with the development of new programming languages and tools to facilitate the ease of development and the learning of programming. Still, many novice programmers have difficulty learning programming as indicated by the increasing failure rates in introductory programming courses (Bergin & Reilly, 2005; Clear et al., 2008; Teague & Roe, 2008). According to Kolling and Rosenberg (1996), programming environments contribute more towards learning programming for novice developers than the programming languages themselves. This literature will examine the role of programming environments in learning programming, specifically examining the issues of the traditional programming environment and the visual RAD environment as well as the role both of these play in the education of novice programmers.

This chapter is structured in four sections: traditional programming, rapid application development, teaching programming, and learning styles and motivation.

2.1. Traditional Programming

"Traditional computer science courses focus on highly technical aspects of computing, and aim to provide students with fundamental knowledge on the inner working of computer systems, and the design and development of algorithms and software" (Pham, 1996, p. 150).

Programming in textual format using text editors predominantly emphasises the programming concepts and makes the programming process transparent to the programmer (Wong, 2006). This type of environment has been used widely in computer science courses and is still being used in the majority of current programming courses (Raadt et al., 2002, 2003; Raadt, Watson & Toleman, 2004; Vogts et al., 2008).

Programming in text-based formats without the help of visual editors has been the traditional way of developing computer programs since the mid 1960s when the earlier programming languages, such as FORTRAN and PASCAL, were first introduced (Kolling & Rosenberg, 1996; Wexelblat, 1981). This traditional style of programming is also referred to as hand-coding, textual or text-based programming in various forms of the literature (Calloni & Bagert, 1994; Calloni et al., 1997; Chainini & Yamada, 1998; Wong, 2006). Traditional programming environments are primarily designed towards developing the procedural programming techniques as they were first introduced for such paradigms. Traditional environments have not changed much since they were first introduced and still typically involve stand-alone tools such as an editor, compiler, debugger and runtime environment (Kolling & Rosenberg). The developer writes the source program in the text-editor, uses the compiler to transform it into machine language and uses the runtime environment to view the results of the program.

As the program is to be written in the textual format using a stand-alone text-editor, traditional programming environments require the programmer to be aware of all the syntax and commands available for the specific programming language. Most, if not all, programming languages allow the programmer to use traditional programming methods to develop applications, regardless of whether they provide a visual interface or not. Many universities are using traditional methods to teach novice programmers languages such as C and Java (Raadt et al., 2002, 2003, 2004). According to Raadt et al. (2002, 2003, 2004), universities try to avoid the use of programming environment-specific languages and tend to use the traditional programming environments, which include text-editors and command-line compilers, whenever possible. The main reasons for this are the provision of distinct steps in the programming process and lower costs compared with RAD environments, which can be expensive to deploy on a per-user basis (Raadt et al., 2002, 2003, 2004).

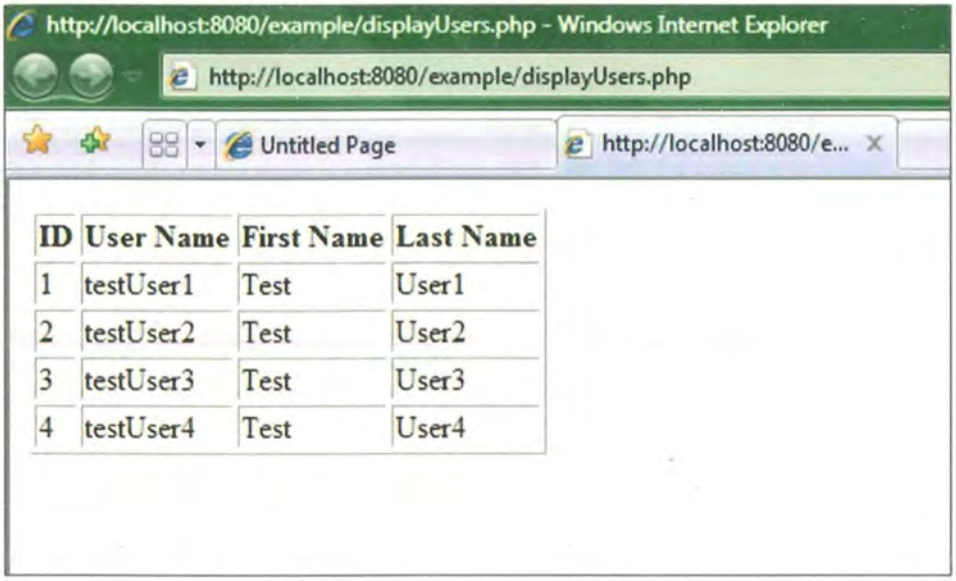
Developing an application in a traditional programming environment involves requirement planning and specification determination from how the output should appear to what tasks the final program should perform. Once the requirements are set, it is up to the programmer to develop the required application. The programmer has complete control over the development process and there should not be any environment-specific limitations imposed on how the application might look or how it should function. In other words, the programming environment should be as flexible as

possible so that the developer can achieve exactly what is needed in the final application. Another major benefit of using traditional programming is the fact that it is more compact when it comes to on-screen representation compared with the visual development tools, especially for functions using mathematical expressions and deeply embedded loop statements (Wong, 2006). It is easier to maintain, modify or enhance the existing program using hand-coding, certainly at the business logic level of the application. As discussed previously, Raadt et al.(2002) have highlighted the fact that traditional programming emphasises the steps in the programming process, such as loops, conditions, variable declarations, parameter passing and database connection strings, unlike visual RAD environments where these processes are hidden in the abstracted purpose that the pre-built objects and components represent (Chainini & Yamada, 1998; Schaub, 2009). These basic skills are important for novice programmers learning a programming language, and most of the time these programming concepts can be applied across different programming languages. Programming using a traditional method has high levels of fidelity, as there are fewer or no "generic" components compared with programming in RAD. Every component typically has to be built from the "ground up".

Even though traditional programming offers much flexibility, it can only be accomplished through in-depth knowledge of both syntax and logic. Therefore, it can be difficult for novice programmers to develop an application of significant scope using traditional methods early in their learning cycle. "Textual programming languages contain a number of syntactic design features that help slips to occur or make them hard to find once they have occurred" (Green & Petre, 1996, p. 23). One of the major contributors to software failure is the fact that software is intangible (Sommerville, 2007) and that little or no visual feedback in traditional programming environments can make it difficult for novice programmers to develop and debug an application. The result can only be seen after the program has been compiled and run. It could be said that having environments that lack more user-friendly features, such as code auto-complete and dynamic error checking, is not a bad thing for novice developers. Novice developers need to learn how to write sections of code, execute them and try to fix problematic code by correctly interpreting error messages. This process of trial and error, while leading to angst and frustration, can teach lessons that will remain with the developer for the rest of their career.

Figure 2-1 and Figure 2-2 provide examples of two simple PHP applications that display data from a MySql database, written using traditional programming methods. The first application retrieves the user information from the database and displays the result in a table (Figure 2-1). The developer needs to know all the syntax and logic to connect to the database, and it has more lines of code compared with the same process carried out in a visual RAD environment (discussed in the next section). Conversely, if the programmer wishes to enhance the program to display in the same table a list of books borrowed by the user, as in the second example (Figure 2-2), this would be a more complicated and somewhat less intuitive task in a visual RAD environment. In a traditional environment, it is just a matter of adding a few extra lines of codes, as shown in Figure 2-2.

Displaying a list of users from database



ID	User Name	First Name	Last Name
1	testUser1	Test	User1
2	testUser2	Test	User2
3	testUser3	Test	User3
4	testUser4	Test	User4

Traditional Programming in PHP

```
<?php
$conn = mysql_connect("server1", "root", "") or die('Database connection fail.');
mysql_select_db("examples", $conn);

echo "<table border='1'>";
echo "<tr><th>ID</th><th>User Name</th><th>First Name</th><th>Last Name</th></tr>";

$result = mysql_query("select * FROM userAccounts");
while($row = mysql_fetch_array($result))
{
    echo "<tr><td>" . $row['id'] . "</td><td>" . $row['userName'] . "</td><td>";
    echo $row['firstName'] . "</td><td>" . $row['lastName'] . "</td></tr>";
}
echo "</table>";
?>
```

MySQL Database

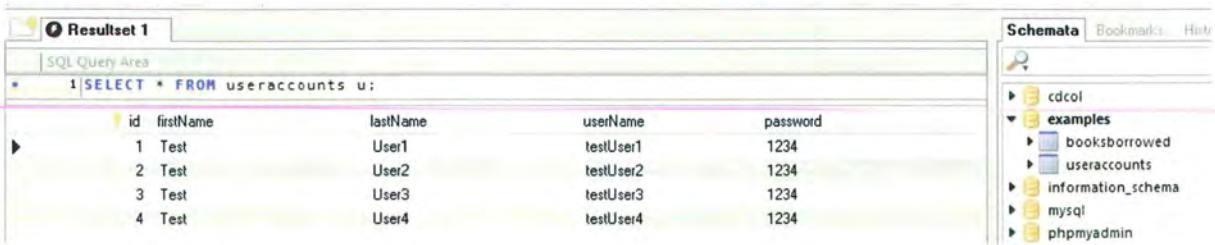


Figure 2-1: PHP example to display data from database

Displaying users and books from database

http://localhost:8080/example/displayUsersWithBooks.php

http://localhost:8080/example/displayUsersWith...

ID	User Name	First Name	Last Name	Books		
				Book Name	Author	Due Date
1	testUser1	Test	User1	Book A	Author A	09/10/2009
				Book E	Author E	05/05/2009
2	testUser2	Test	User2	Book Name	Author	Due Date
				Book B	Author B	02/07/2008
				Book F	Author F	04/04/2009
3	testUser3	Test	User3	Book Name	Author	Due Date
				Book C	Author C	04/05/2008
				Book G	Author G	13/04/2009
4	testUser4	Test	User4	Book Name	Author	Due Date
				Book D	Author D	03/05/2009

Traditional Programming in PHP

```
<?php
$conn = mysql_connect("localhost", "root", "") or die('Database connection fail.');
mysql_select_db("examples", $conn);

echo "<table border='1' cellspacing='0' cellpadding='3'>";
echo "<tr><th>ID</th><th>User Name</th><th>First Name</th><th>Last Name</th><th>Books</th></tr>";

$result = mysql_query("select * FROM userAccounts");
while($row = mysql_fetch_array($result))
{
    $id = $row['id'];
    echo "<tr><td>" . $id . "</td><td>" . $row['userName'] . "</td><td>" ;
    echo $row['firstName'] . "</td><td>" . $row['lastName'] . "</td>";
    $bookResult = mysql_query("select * FROM booksborrowed where userID=". $id . "");
    echo "<td><table border='1' cellspacing='0' cellpadding='1'>";
    echo "<tr><th>Book Name</th><th>Author</th><th>Due Date</th></tr>";
    while($bookRow = mysql_fetch_array($bookResult))
    {
    echo "<tr><td>" . $bookRow['bookName'] . "</td><td>" . $bookRow['author'] . "</td><td>" ;
    echo $bookRow['duedate'] . "</td></tr>";
    }
    echo "</table></td></tr>";
}
echo "</table>";
?>
```

MySQL Database

Resultset 1

SQL Query Area

```
1 SELECT * FROM booksborrowed b:
```

id	bookName	author	duedate	userid
1	Book A	Author A	09/10/2009	1
2	Book B	Author B	02/07/2008	2
3	Book C	Author C	04/05/2008	3

Schemata | Bookmarks

- cdcol
- examples
  - booksborrowed
  - useraccounts
- information_schema
- mysql

Figure 2-2: PHP example to display data from database with nested table

2.2. Visual Rapid Application Development

RAD can be defined in various ways. According to Martin (1991), there are four aspects of RAD: the tools, methodology, people and management. The characteristics of RAD tools include capability for planning, data and process modelling, code generation,

testing and debugging. The methodology includes requirements planning, user design and construction and production deployment where requirements planning and user design can sometimes be consolidated into one cycle. Agarwal et al. (2000) describe it as a software development methodology similar to the spiral, iterative model. Agarwal et al. (2000, p. 177) also state that RAD tools can also be classified as "a class of tools that allow for speedy object development, graphical user interfaces, and reusable codes for client/ server applications. The tools enable the methodology and circumscribe what is accomplished during a development project".

Object-oriented, event-driven, visual programming languages have formed part of changes in the application development trend towards higher level programming languages, which have led to the concept of visual RAD (Agarwal et al., 2000; Honchell & Robertson, 1996; Kolling & Rosenberg, 1996). Most of the programming tasks in visual RAD are accomplished using the drag-and-drop icons or menu-driven interfaces. This leads to the definition of visual RAD environments as iconic programming or visual programming systems (Calloni & Bagert, 1994; Calloni et al., 1997; Cilliers et al., 2005; Ichikawa & Hirakawa, 1990). Some visual RAD tools provide the syntactically correct source code of the intended programming language of the given algorithm, while others mask the underlying code and present only controls and their developer editable properties.

A wide range of visual RAD tools is available in the market to cater for different programming languages and types of applications. The variety of tools ranges from planning and modelling to programming, testing and debugging. Rational Rose is an example of a visual RAD planning/modelling tool. The Rational Rose programmer can draw the architecture/design of the system by simply dragging and dropping icons to generate the structural source code from the model drawn. Other software like Objecteering, Eclipse UML2 Tools and Modelio are also available, and they provide similar functionality to Rational Rose. Programming support tools, such as Borland, Eclipse, BlueJ and NetBeans, provide some visual programming environments for Java alongside traditional interfaces. Many of the visual RAD tools are language dependent but some, for example Microsoft's Visual Studio, provide the environment for more than one programming language (though often the runtime environment is operating-system dependent). Some of the popular web application development tools that provide visual RAD techniques include Visual Studio, Dreamweaver and Adobe's ColdFusion.

Microworlds, such as Karel J Robot, Alice, Jeroo, Robocode and PigWorld, provide the visual environment for programmers to visualise the complete program state throughout program execution (Goldweber et al., 2006; Pears et al., 2007).

Honchell and Robertson (1996) have highlighted that the widespread availability of pre-written, object-oriented visual RAD software modules saves the cost of training, troubleshooting and maintenance of applications and promotes the use of reusable components across standardised applications. It also makes it possible to share the workload among programmers, as it is able to provide the standardised look and feel of the application by making use of the existing templates and objects. Sharing of workload and easy drag and drop features should, in theory, result in reduced development time for medium to large applications. The Multi-User Programming Pedagogy for Enhancing Traditional Study (MUPPETS) is a visual object-oriented system design aimed at introducing students to building complex 3D applications visually, while also giving them console access to the underlying classes and code (Egert, Bierre, Phelps & Ventura, 2006). The feedback from the system is very visual and is provided immediately as the developer programs it. Testing and debugging can occur while still in the visual interface. The major benefit of visual RAD tools is that almost anyone can program the application, as little or no syntax knowledge is required to create a simple application (Rode, 2004). The developer needs only to know what "function" they need to perform and then locate the required pre-built components and drag them into the interface for application-specific customisation.

These do-it-yourself visual RAD tools promise application development without requiring "real" developers, an issue that has raised concerns about the future of programmers and programming in the ICT industry (Kaneshige, 2009). Goldweber et al. (2006) provide an example of the Alice Programming Environment, an interactive 3D microworld designed to facilitate the learning of computer programming by large portions of the general population. It is easy enough for the student developer to successfully program in Alice almost immediately. A basic program would be unlikely to fail because there is no syntax to master and the programmer can only select legal choices/statements/commands through the iconic interface. Even though the program might not fail syntactically, it might not work as intended or solve the problem at hand. The generalised plug and play components offered in visual RAD environments may not function exactly as the application requires them to, which can cause major

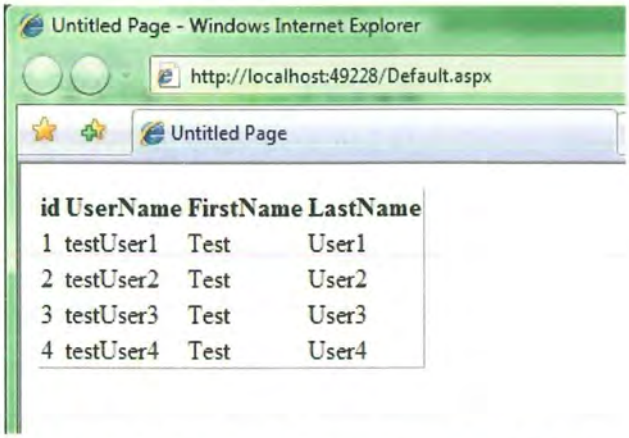
challenges to design applications that fit a given requirement (Peter, 2009). Levels of fidelity for problem solving could be lower with visual RAD tools compared with traditional programming methods. Visual RAD objects are often developed to be "flexible" and "generic" to suit as many scenarios as possible, which in turn can cause the problem of them not being able to provide the outcome that a developer wants but rather the closest approximation that the RAD tool can deliver. The level of programming in visual RAD environments tends to be more limited and most commonly centres on creating forms and manipulating databases (Wong, 2006). In particular, developing forms interfaces for managing database content is an extremely code-intensive practice and is one of the reasons that this thesis has focused on the web application development aspect of coding.

From the list of risks involved in using RAD identified by Agarwal et al. (2000), one of the key drawbacks is the unrealistic expectations of management regarding how quickly systems can be constructed using such approaches. "With ICT budgets being squeezed, along with the growing dysfunctional relationship between ICT staff and managers ... the promise of cheap 'codeless' development that sidesteps ICT resonates loudly with business people" (Kaneshige, 2009, p. 43). In fact, developing applications using visual RAD has many limitations and sometimes a small but complex customisation involving underlying business logic on a form submission, or when displaying data from data sources, could take longer than writing the whole program using traditional methods. In some cases, the extensive set of concepts and features provided in visual RAD tools makes it hard to learn and is often problematic for novice programmers trying to understand the provided functionality (Pears et al., 2007). Interface complexity and the sheer number of options and permutations can make visual RAD tools difficult to use for novices beyond just the drag and drop level.

The following example describes the steps involved in programming a simple Asp.Net application in Microsoft's Visual Studio environment (Figure 2-3). Compared with the example using a traditional method (Figure 2-1), the example in Figure 2-3 displays the same result in four simple steps with no Asp.Net syntax knowledge required. However, to display the nested table, such as that in Figure 2-2, programming in Visual Studio is much more complicated and requires the developer to know the detailed functionality of each component in order to customise the look and feel. Asp.Net applications require

many complicated steps to accomplish the same task that can be done with 15 lines of codes in PHP (Appendix A).

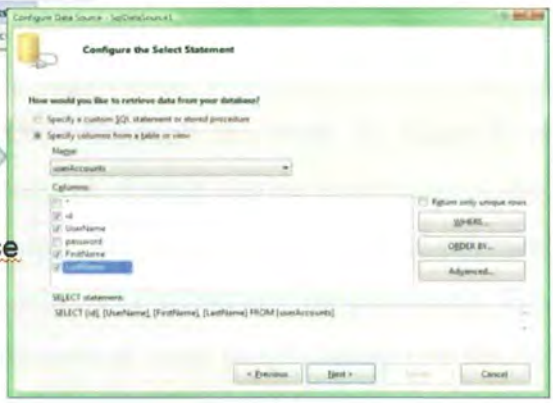
Displaying users from database



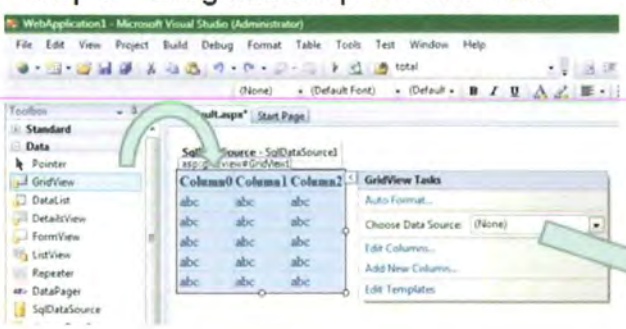
Steps in RAD Tool (Visual Studio 2008)

Step 1 – Drag and drop the SQL DataSource

Step 2 – Configure the SQL DataSource



Step 3 – Drag and Drop the Grid View



Step 4 – Link to the SQL DataSource

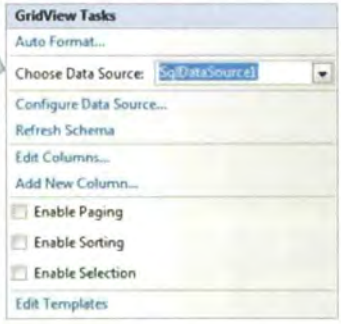


Figure 2-3: Asp.Net example to display data from database using Visual Studio

2.3. Teaching Programming

It is apparent from the literature that computer programming is a difficult subject and failure rates are consistently higher than for other topics (Lahtinen, AlaMutka & HannuMatti, 2005). Teague and Roe's (2008) research indicates that introductory programming units have had high failure rates over the past five years compared with their counterparts such as database systems and system architectures. Numerous studies have been conducted to identify the cause of the high failure rates in introductory programming courses and to make the comprehension of programming easier for novice programmers. Such studies include revising the current programming curricula, programming exercises, programming language and programming environments used to teach the novice developers (Giordano & Carlisle, 2006; Goldweber et al., 2006; Mannila & Raadt, 2006; McIver & Conway, 1996; Milne & Rowe, 2002; Pears et al., 2007; Raadt et al., 2002, 2004; Schulte & Bennedsen, 2006).

Introductory programming courses were first designed to teach the three main aspects of programming: problem solving, describing algorithmic solutions to a problem and verifying the algorithm (Gries, 1974, 2006). Although the work by Gries is now outdated from a technical standpoint, the concept of what novice programmers should learn has not changed since 1974. Over the last three decades, a great deal of research has been conducted on the different approaches to introductory programming. Figure 2-4 represents a traditional pathway for developers in many programming courses. Most developers start with learning traditional programming in their introductory courses, after which they learn visual RAD programming techniques and use traditional and/or visual RAD tools as required.

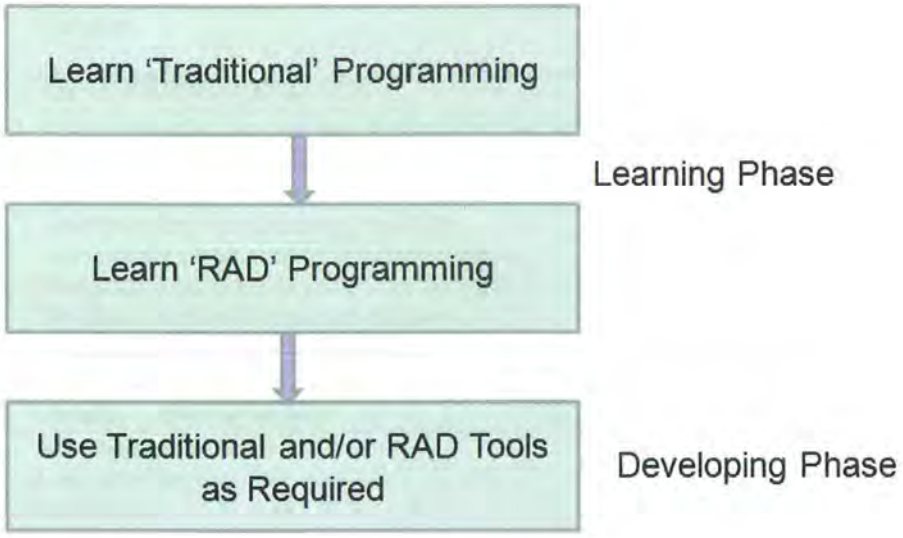


Figure 2-4: Traditional pathway for developers

Syntax and logic are the two main focus areas of teaching programming to novice programmers. Even though learning the syntactical structure of one programming language is challenging, Robins, Rountree and Rountree (2003), Weir et al. (2005) and Lahtinen et al. (2005) have noted that problem solving, designing, planning and program construction are the areas where novice programmers have the greatest learning difficulties. Many researchers also argue that teaching object-oriented paradigm at the introductory level is much more difficult than teaching a procedural approach at the same level (Kinnunen & Malmi, 2008; Weir et al.), which could in turn cause issues with visual RAD environments, which tend to be largely object-oriented (OO) based. Table 2-1 provides the results of a survey of both student and teacher perceptions towards programming course content across six universities where teachers had experience teaching one or two programming courses (Lahtinen et al.). According to the study, students perceived dividing functionality into procedures, understanding programming structures and finding bugs in their own programs to be among the main difficulties they faced in learning programming, whereas the instructors thought the programming development environment was the most challenging aspect. Recursions, pointers, references and error handling are the three most difficult concepts for students. In terms of learning programming, most instructors felt that practical sessions helped students learn issues about programming, whereas most students preferred working alone on programming coursework. Both teachers and students in the survey agreed that programming examples are the best materials for learning programming.

Question	Code	Students			Teachers		
		N	Avg	Std	N	Avg	Std
THE COURSE CONTENTS							
What kind of issues you feel difficult in learning programming?							
Using program development environment	I1	553	2,43	0,99	33	2,61	0,90
Gaining access to computers/networks	I2	536	2,11	0,95	32	1,97	0,78
Understanding programming structures	I3	556	2,92	1,02	33	3,27	0,67
Learning the programming language syntax	I4	555	2,75	1,01	33	2,70	0,73
Designing a program to solve a certain task	I5	555	3,12	0,98	33	3,97	0,73
Dividing functionality into procedures	I6	543	3,10	1,09	31	4,06	0,63
Finding bugs from my own program	I7	549	3,28	1,03	33	3,91	0,77
Which programming concepts have been difficult for you to learn?							
Variables (lifetime, scope)	C1	541	2,10	0,97	34	2,41	0,70
Selection structures	C2	552	1,98	0,90	34	2,38	0,70
Loop structures	C3	551	2,09	0,97	34	2,79	0,91
Recursion	C4	512	3,22	1,03	31	4,06	0,96
Arrays	C5	526	2,79	1,15	33	3,24	0,71
Pointers, references	C6	518	3,59	1,04	32	4,44	0,56
Parameters	C7	513	2,60	1,09	32	3,47	0,76
Structured data types	C8	496	2,90	1,03	31	3,45	0,81
Abstract data types	C9	499	3,02	1,10	31	4,06	0,81
Input/output handling	C10	519	2,96	1,04	32	3,75	0,88
Error handling	C11	481	3,33	1,01	32	4,13	0,79
Using language libraries	C12	465	3,04	1,09	32	3,88	0,71
LEARNING AND TEACHING PROGRAMMING							
When do you feel that you learn issues about programming?							
In lectures	S1	543	3,01	1,01	33	3,21	1,02
In exercise sessions in small groups	S2	510	3,44	1,10	32	3,84	0,99
In practical sessions	S3	514	3,77	1,03	31	4,35	0,75
While studying alone	S4	546	3,79	1,06	31	3,42	0,72
While working alone on programming coursework	S5	539	3,98	1,09	33	4,00	0,79
What kind of materials have helped/would help you in learning programming?							
Programming course book	M1	515	3,35	1,03	33	3,30	0,88
Lecture notes/copies of transparencies	M2	539	3,39	1,05	34	3,47	0,71
Exercise questions and answers	M3	523	3,33	1,07	34	3,62	1,02
Example programs	M4	551	4,19	0,86	34	4,24	0,65
Still pictures of programming structures	M5	490	3,15	1,00	30	3,70	0,75
Interactive visualizations	M6	315	3,33	1,03	27	4,07	0,87

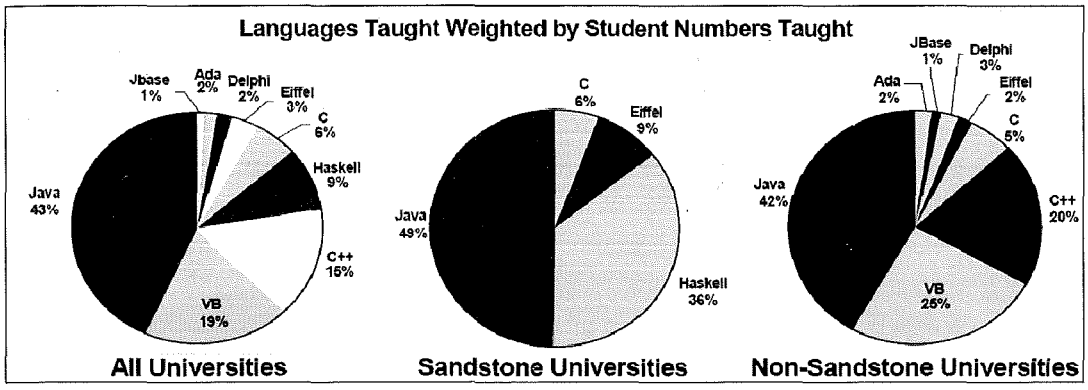
(Lahtinen et al., 2005)

Table 2-1: Results on sections course content and learning and teaching

Novice programmers tend to approach programming "line-by-line" and often learn in a "context specific" (Lahtinen et al., 2005) style, where they find out how to perform a certain function as it is required. This suggests a targeted, rather than holistic, approach to learning programming for some novice developers. Lister et al. (2004) have conducted a study on the reading and tracing skills of novice programmers and the results agree with those of Lahtinen et al., that most novice programmers use the "line-

by-line" approach to developing a program. Lahtinen et al., also highlight that novice programmers often experience difficulty in combining the syntax and semantics of individual statements into larger, valid programs. Therefore, it is important to combine the concept knowledge and programming structures in the learning process so that novice developers can go beyond algorithmic programming into true applications development.

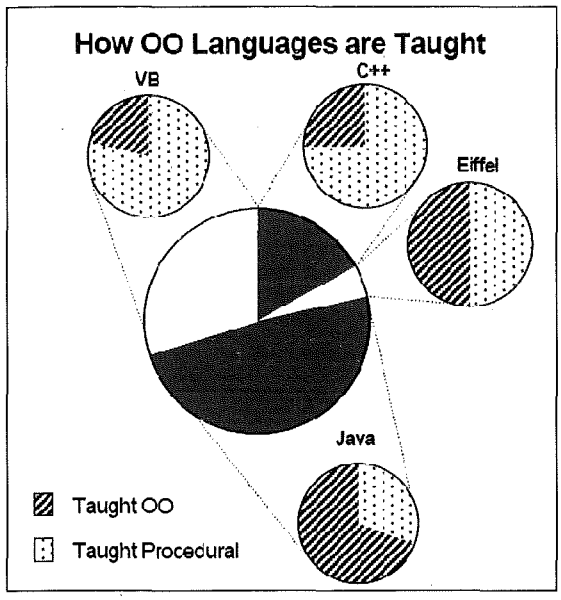
Given the increased popularity of visual RAD tools over the past decade, many institutions are looking at ways to change the existing learning pathway and introduce visual RAD at the introductory level. Goldweber et al. (2006) recommended using visual programming environments, such as robotics systems or 3D microworlds, to teach novice programmers. Haden (2006) emphasises the use of game programming to teach traditional programming skills. António José Mendes aims to achieve success with the students of a CS1 course with the use of visual environments such as BlueJ and Karnel J. and Robot (Weir et al., 2005). Lahtinen et al. (2005) argue that whatever the approach, it is important that students learn the basic structure of programming, such as loops, variables, recursion and parameter passing, at some point. These programming processes are largely hidden behind the layers of abstraction in visual RAD tools. This has raised concerns that novice programmers are not able to learn these basic processes if RAD tools are introduced at the introductory level (Raadt et al., 2002, 2003, 2004). To this end, the literature suggests that some middle ground must be found between the core coding skills of the traditional approach and the use of RAD environments and methods in the application development community.



(Raadt et al., 2002)

Figure 2-5: Languages taught in Australian universities weighted by student numbers

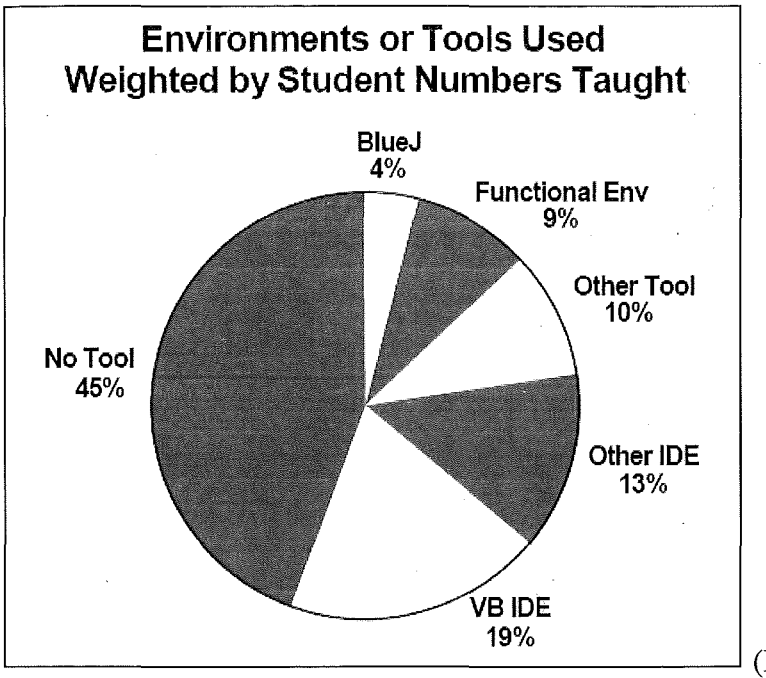
Many languages have been taught in introductory programming courses in different institutions over the past four decades, with Java, C, and C++ on top of the list of languages taught today (Pears et al., 2007). According to the research by Dale (2005), Raadt et al. (2002), Schulte and Bennedsen (2006), many universities around the world use Java for their introductory programming courses and VB and C++ are the second most common languages. It is worth noting that there is a clear distinction between the languages taught by sandstone universities and non-sandstone universities. Non-commercial languages, such as Eiffel and Haskell, are taught mostly in sandstone universities (Figure 2-5). The difference in language choices between sandstone and non-sandstone universities could be due to the fact that sandstone universities focus on the pedagogical benefits of language in choosing the programming language, while non-sandstone universities focus on the industry relevance, marketability and student demand for a language (Raadt et al.). In general, most of the universities across the world make the language choice based on "the factors such as faculty preference, industry relevance, technical aspects of the language, and the availability of useful tools and materials"(Pears et al., 2007, p. 207).



(Raadt et al., 2002)

Figure 2-6: How OO languages are taught in Australian universities

It is interesting to note that though 86% of the above-mentioned languages are object-oriented programming (OOP) languages, less than half are taught using an objects early approach except for Java. Figure 2-6 shows 70% of Java instructors are using the object-oriented approach, but the rest are asking their students to ignore the class declaration in Java until later (Raadt et al., 2002). This contradicts the findings of Dale (2005), where more than 50% of Java instructors teach the step-wise approach rather than an object-oriented approach (OO). Procedural programming provides a learning environment that emphasises the basic programming concepts, such as looping, iteration, recursion and variable declarations, which are important for novice programmers (Lahtinen et al., 2005). On the other hand, OO focuses on abstraction ability, inheritance and polymorphism. Even though many universities emphasise the importance of abstraction in OO, Or-Bach and Lavy (2004) point out in their study that students are struggling with high levels of abstraction. Only 4 out of 46 students could provide the highest level of abstraction for a given algorithm, while the majority could only provide a very low level of abstraction. This finding would seem to indicate that there may be merit in teaching programming fundamentals rather than programming fundamentals and advanced concepts at the same time, which may rule out a mixture of traditional and visual RAD in the same unit of learning.



(Raadt et al., 2002)

Figure 2-7: Teaching tools and environments

"Programming at all levels of experience need to work within environments which give them access to the tools which they must use to accomplish their tasks" (Pears et al., 2007, p. 210).

Raadt et al. (2002) highlight in their research that the majority of instructors choose to use simple text editors and command-line compilers for their introductory courses, the main reason being that teaching complex environments takes up most of the valuable time to teach the tool rather than the programming language itself. Other than simple editors, Figure 2-7 shows the other types of tools being used at some universities in introductory programming courses. Despite the fact that many RAD tools are widely available on the market, text editors and command-line-compilers are often preferred if the language permits. The main reasons for this, according to Raadt et al. (2002, p. 334), are "cost for students, time required to familiarise students with the environments, and the blurring of distinct steps in the programming process". Pears et al. (2007) also point out that many of the RAD tools are designed to fit professionals' needs and tend to be too complicated and confusing for the novice programmer. The extensive set of concepts and features, errors and warning messages may be hard for the novice programmer to understand.

Calloni et al. (1997), following on from the research carried out in 1994 (Calloni & Bagert, 1994), state otherwise, arguing that using the iconic approach to teach first year programming is more effective compared with the normal text editor, hand-coding approach. Based on the comparison of teaching the integrated BACCII++ and C++ to C++ only to first-year students, they found that integrating both tools and language to teach the first-year students resulted in better marks in their final exam and in the overall course (Calloni et al.). Similar results were produced in the research carried out by Cilliers et al. (2005), where B# and Delphi IDE were compared. However, these studies did not describe the level of difficulty of the assessments and workshops and thus were not able to conclude if visual RAD helps novices to be better programmers.

The reason that Raadt et al's (2002) research results vary from those of Cilliers et al. (2005) and Calloni et al. (1997) could be that the former is based on perceptions from the teachers' points of view and is generalised for many different programming languages. On the other hand, the research of Cilliers et al. and Calloni et al. focuses on the outcome of the students' exam results. These results are based on specific programming languages and students have the knowledge of both language and environment, a factor that might contribute to such positive outcomes. Essentially, it may be that Calloni's research better describes the ability of students to pass an exam rather than exit as practical programmers.

2.4. Learning Styles and Motivation

According to Jenkins (2002), learning style and motivation are two possibilities that make programming difficult for novice programmers to learn.

Classifications of learning styles vary across different researchers and their context of research. In general, there are four dimensions of learning styles: sensing-intuitive, visual-verbal, active-reflective and sequential-global (Papaeconomou, Zijlema & Ingwersen, 2008; Parvez & Blank, 2007). Bohlen and Ferratt (1993) have used Kolb's Learning Style Model to determine a learner's predominant learning style for end-user learning of computer software. Galpin, Sanders and Chen (2007) have used the same method to study the impact on the method of teaching in a South African university.

Kolb's Learning Style Model includes four types of learners: convergers, assimilators, divergers and accommodators (Bohlen & Ferratt, 1993; Galpin et al., 2007). Convergers and assimilators prefer abstract conceptualisation, which is highly relevant to learning visual RAD environments. Divergers and accommodators have strengths opposite to those of convergers and assimilators; they prefer concrete experience. For these types of learners, it is perhaps the effectiveness of the learning materials and not the type of programming environment that has a primary influence. However, given that the majority of existing learning materials for programming, such as those in textbooks and on the web, are based largely on traditional environments, this environment may be better suited to these types of learners.

"Motivation of students is a key issue if they are to learn"(Jenkins, 2001, p. 53).

Three factors of motivation by Entwisle, as quoted by Jenkins (2001), include extrinsic: the desire to perform in order to attain rewards; intrinsic: deriving from an interest in the subject and achievement: that is, competitiveness. Many different factors influence student motivation in learning programming, such as prior experience, programming environment, exercises, teachers, peers and requirements for job or study, for example (Bergin & Reilly, 2005; Jenkins; Mamone, 1992; Walter, Forssell, Barron & Martin, 2007). Research by (Halland & Malan, 2003) indicates that teachers of introductory programming courses perceive that visual RAD environments are fun for the students and that this could keep them interested in programming. Students can see more reward more quickly with visual RAD environments, whereas in a traditional environment there is lots of conceptual groundwork to cover before students can see a given result. With traditional text-based environments, it can be a daunting task to write hundreds of lines of code before the application can be fully, or even partially, visualised.

Learning styles are raised here as they are often discussed in the same context and literature as that dealing with motivation. Given that this thesis examines student perceptions of different approaches to programming, motivation is considered an important area of the literature to address. The analysis chapters will show that motivation is indeed very important to novice developers and that the different approaches to programming do influence student motivations.

3. Research Method and Design

Collection of data to support the given topic is the most important part of a research project. Choosing appropriate research methods and design is a crucial part of gathering reliable and useful data in order to address research questions. Understanding the research methodology for a given piece of research helps a great deal in choosing the actual research methods and design. Research methodology answers the question of how the research should be conducted and research methods include the tools to collect the data, such as interviews and focus groups (Dawson, 2006). A combination of different methods, both qualitative and quantitative, was used to conduct this research. This combination of methods, also known as triangulation of a multi-method approach, provides for a more complete dataset to gather data required for the research. Dawson believes that a triangulation of research approaches is a good way of conducting research, as it allows the researcher to work against the weaknesses of two or more methodologies by allowing one set of findings to complement those of the others.

3.1. Research Methods

3.1.1. Selection Process

There are a number of ways to gather student reactions to the use of visual RAD versus traditional programming environments for novice programmers. One common way of approaching this would be by having two introductory programming classes, one of which would be taught using a visual RAD environment only and the other, using a traditional programming environment only. Comparing the results of the students from these two classes over a teaching period would provide a good understanding of the comparative effectiveness of the different programming environments. This experimental approach has been used in numerous studies, for example, those of Calloni and Bagert, Calloni et.al. and Cilliers et al. (Calloni & Bagert, 1994; Calloni et al., 1997; Cilliers et al., 2005). Due to the time frame of this research, however, this approach was deemed too lengthy. Other effective methods, such as observation, questionnaires and interview approaches, were considered for this research. These methods, when used in conjunction with one another, were envisaged as suitable for providing useful data regarding student responses to the proposed research questions

detailed earlier. Figure 3-1 shows the overall relationship between the methods that were adopted for use in this research.

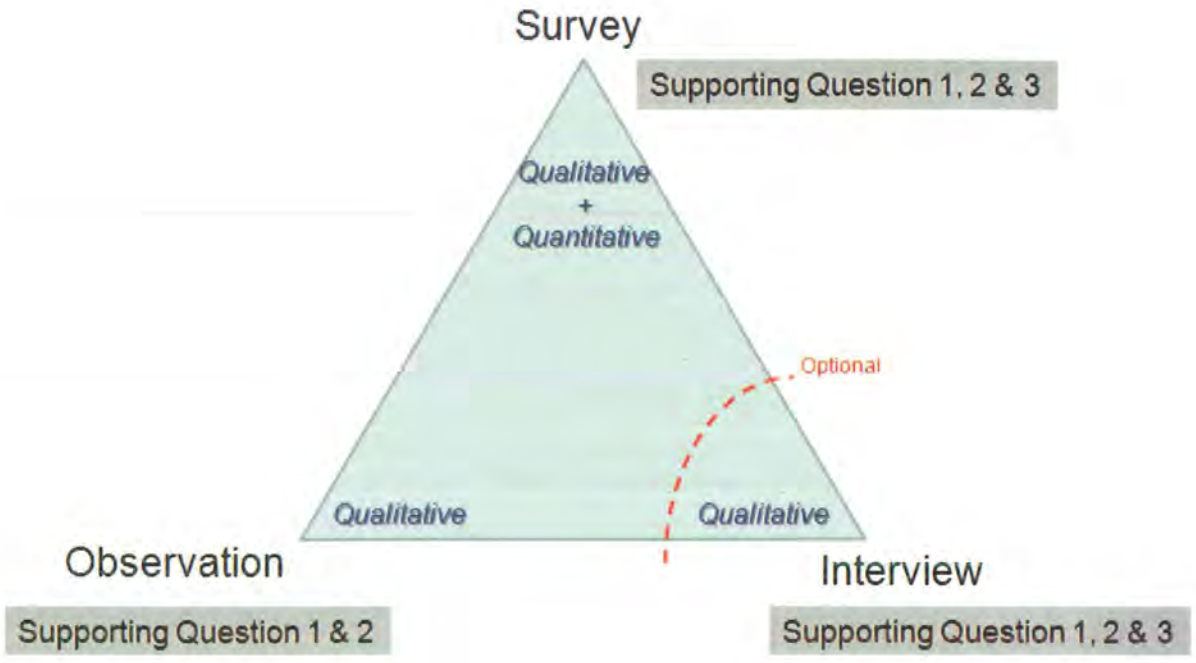


Figure 3-1: Method triangulation

3.1.2. Survey Method

The survey research method is a very common method of collecting research data according to Babbie (2000). A typical piece of survey research would include administering a standardised questionnaire to a target audience. It enables the researcher to capture a broad picture of the experiences and views of a target population with little or no personal contact with them (Clough & Nutbrown, 2002). For this research, the survey method was used to investigate student attitudes towards the visual RAD environments versus traditional programming environments. There are three basic types of survey questions: closed-ended, open-ended and a combination of both (Dawson, 2006). Closed-ended questions usually follow a set format and produce numerical results suited to statistic analysis. Open-ended questions provide qualitative data relevant to the study and though more holistic in nature, can also be statistically analysed in some situations (Dawson). Both these types of questions were used heavily throughout this research and its survey instruments. The advantage of this mixed question approach is that the results can be presented in the form of statistics and tables,

and at the same time it overcomes the limitation of the possible dead-end answers that a respondent may give when providing responses to open-ended questions (McNeill & Chapman, 2005). To acquire the maximum number of participants, the surveys were conducted online and anonymously. Pre-workshop surveys and post-workshop surveys were conducted to gather the students' perceptions of their programming experience and thoughts on visual RAD versus traditional programming environments. The quantitative and qualitative data gathered from the survey method helped to answer the majority of the research questions in this research.

3.1.3. Interview Method

According to Newman and Benz (1998), quoted from Patton (1990), a research interview can be characterised as a strategy to find out from people things that cannot be observed directly. Through probes, follow-up questions and non-verbal cues, data collected can be enhanced through the interview process (Newman & Benz). The limitation of this method depends on the interviewer's ability to execute these tasks, and the validity of the data collected could be influenced by the subjective bias of the interviewer that affects their interpretation of the data (Newman & Benz). Interviews can be in the form of structured, unstructured or semi-structured questions (Dawson, 2006; Newman & Benz). Structured interviews collect standardised data from all the participants and unstructured interviews are used to gain holistic understanding of the interviewees' points of view with respect to a given situation (Dawson; Newman & Benz). Semi-structured interviews allow interviewers to gather data that can be compared and contrasted with other interviews. The validity of the data collected through interviews can be enhanced by using the structured or semi-structured approaches. This research used the semi-structured approach to collect the students' perceptions towards visual RAD and traditional programming environments.

3.1.4. Observational Method

According to Newman and Benz (1998), the observational method is the most frequent data-collection method used for qualitative research. Clough and Nutbrown (2002), quoted from Cohen, Mannion and Morrison (2000), state that observational data allows the researcher to look at the "live" data from "live" situations. There are two types of

observational methods, participant observation and direct, or non-participant, observation (Dawson, 2006; Newman & Benz). Participant observation involves the researcher as a member of the studied group and the researcher is "involved in the lives of the people being observed"(Dawson, 2006, p. 33). Direct, or non-participant observation, involves looking at the interaction of subjects in a given situation. Non-participant observation is a better approach in terms of the validity of the data gathered compared with participant observation according to Newman and Benz. This research gathered data using a non-participant observational method where the researcher observed the reaction of students to a given programming environment within a computer lab and via online workshops based on the click-stream events and in-class observations. In-class observations provided the basic information on the questions asked and problems encountered during workshops. The click-stream was captured for all of the students' mouse-clicks on the back and next navigations and help links to determine the time taken to complete the tasks and how they approached different problems at different levels.

Essentially, this research was based on a multi-method approach, with the hope of capturing the core data using the survey methodology and providing triangulation of those results by comparing them with results gained via the interview and participant observation methods. As the following chapters will show, this methodology was only partially successful due to technical and participant issues.

3.2. Research Design

An online workshop and three lab-based workshops were conducted to observe students' reactions towards visual RAD and traditional programming environments. Pre- and post-questionnaires delivered during these workshops (Appendix B and C) and interviews (Appendix D) supported the data gathered from the observations. All the information gathered was anonymous and logs (Appendix E) of user actions, timestamps and survey responses were all stored in a backend database. The triangulation of these different methods provided sufficient and useful data for this research in answering the main research question: *"What is the student reaction to visual RAD versus traditional programming environments for novice programmers in a web application development context?"*

Figure 3-2 provides the high-level view of the research design and implementation for this research.

Figure 3-2: Research design and delivery

3.2.1. Participant Recruitment

Participants were recruited from within the university computer science school in which this study was conducted. This school of computer science had approximately 2000 students enrolled across numerous degrees and will be referred to in this thesis as "the school". To recruit the participants for the online and lab workshops, a notice was posted on the school's current students' homepage, which is visible to all students within the school (Appendix F). This was followed up by visits to some of the first-year programming classes to inform the students of these workshops personally and increase their awareness of them. The aim was to recruit approximately 20 participants for these workshops, although the final number was 15 participants for the lab sessions and 49 participants for the online session. The online session was designed as a fall-back approach if the in-class participation was considered too low. After the first in-class only session was run, a second recruitment posting was made to the school's website,

aimed specifically at online participants only. Of both the in-class and online groups, a total of 29 participants completed both pre- and post-surveys and the rest of the participants just completed some parts of the pre-survey. Although six students registered to participate in a follow-up interview, only one participant turned up for the interview session.

3.2.2. Survey Delivery

At the start of the workshop, consent to participate in the research was gathered (Appendix G) before an online survey (Appendix B) was conducted to gather the demographic information of the participants, including age, gender and their current course of study. This survey also included the programming and learning experience of the participants, such as their previous experience with languages and environments, their perceptions of their own "level", i.e. novice through to expert, and their existing preferences for languages or environments. After the survey, step-by-step coding exercises on traditional programming and visual RAD environments were conducted, whereby in-class participants developed a small, functional web application and online participants watched a series of videos on building the same web application using the same instructional materials. A post-survey (Appendix C) was carried out to gather students' perceptions towards visual RAD versus traditional environments based on the completed programming exercises. This survey also addressed any problems experienced by students during the exercises, student confidence in developing further web applications using each environment and the changes in preferences identified in the pre-exercise survey. The survey data provided the majority of data that formed the analysis for addressing all three supporting questions.

Although the pre- and post-surveys provided much useful information on students' perceptions and provided both qualitative and quantitative data, it appeared to cause survey fatigue with many of the participants due to having to answer so many questions. Many of the online participants left after the pre-exercise survey without even completing the programming exercises. However, overall, the survey data provided enough quality data upon which to base the outcomes of this research. Unfortunately, due to time constraints arising from the development of the surveys, workshop materials and the system that managed the integration and delivery of both, a pilot study was not

conducted. In hindsight, some of the survey fatigue and timing issues might have been corrected had there been time for an initial "dry run" of the research design.

3.2.3. Coding Exercises

Coding exercises used the PHP scripting language with the EditPlus text editing tool as the traditional programming environment (Appendix H) and Microsoft Visual Studio 2008 with ASP.Net programming environment for the visual RAD exercises (Appendix I). The choice of these environments was largely influenced by the web application programming environment within the school's curriculum as well as the availability of both with minimal setup requirements. Both of these environments were pre-installed in the school's labs. It was assumed for this research that student perceptions and reactions could be generalised for traditional programming environments and visual RAD environments based on their use of EditPlus and Visual Studio 2008. In a larger study within the context of a longer time frame, a truer comparison of "what makes a visual RAD environment" could be carried out before undertaking a comparison with different types of environments.

Participants were given three database-driven web development exercises for each of the environments. The sequence of environments differed between the participants. It was designed in such a way that half of the participants started with the traditional programming environment followed by the RAD environment; the other half of the participants were provided with the environments in the reverse order. The first exercise provided the step-by-step instructions to connect to Microsoft's Access database and display the data from a database using the PHP language. An identical exercise was conducted in the visual RAD environment using drag-and-drop rather than code techniques. After the basic level, participants were asked to enhance the first exercise to allow editing and deleting of the records from the table. The third exercise involved inserting new records into the database.

Participants were allowed to navigate through the exercises with "Next" and "Back" buttons, and some hyperlinks to online resources from Google search were also provided. All the click events were recorded against generated user ids to keep track of the time taken for each exercise and to observe each participant's action on each step.

The observation data from coding exercises provided some useful information to support the data seen in the survey responses, although the overall outcome of this participant observation had less impact than had been hoped for. This was particularly true in terms of the entirely online participants, who tended to skip sections of the video instructions and jump ahead to later sections.

Some technical issues with Microsoft Visual Studio setup for the in-class workshops were encountered on the day of data collection. Although 1.5 hours was allocated for each of the in-class sessions, it was not enough and resulted in a change to the format of delivery to a more a more instructor-led rather than student-led approach.

3.2.4. Interviews

Participants were asked to provide their email addresses if they wish to participate in the follow-up interview to gather more detailed qualitative data. An email was sent out to those participants who provided one with the time and place for the one-on-one interview scheduled a week after the lab workshops. As stated, only one participant took part in an interview with just a single comment from that interview contributing to the final thesis.

4. Data Analysis

The focus of this chapter is the analysis of the data gathered from the pre- and post-exercise surveys collected from three in-class workshops and the online session. Participants in the in-class workshops were asked to fill in these surveys before and after they had attempted two sets of programming exercises, while the online participants were presented with the similar surveys before and after the video demonstration of the same sets of exercises. The click-stream events that were also captured during the sessions for each participant were initially intended to be used to identify the behavioural trend between traditional exercises and visual RAD exercises. Due to the time constraints during the lab sessions, the exercises were demonstrated on screen, and almost 90% of the in-class participants followed the on-screen instructions rather than doing the work on their own. Therefore, the behavioural trend is not as conclusive as was originally hoped.

4.1. Pre-exercise Survey

This section examines the responses to the pre-exercise survey questions from both in-class and online participants. The pre-exercise survey consists of four sub-sections: demographics, programming experience, perception of visual RAD versus traditional programming environments and learning experience. The same set of instruments was given to all participants. The main purpose of this first survey was to gain an understanding of each participant's current level of expertise and experience with the programming environments and to gather their reaction to these environments.

4.1.1. Demographics

A total of 64 students from the school participated in this research, 16 of whom were drawn from the in-class sessions and 48 from the online session. Only 29 of the 64 participants completed the exercises and participated in the post-survey. The data from the participants who did not complete the exercises was not analysed in this chapter.

Of the 29 participants who had completed the exercises, 90% were male and the rest were female. The majority of the participants (31%) were aged between 18 and 21, followed by 22 to 25 (24%), with another 24% being over the age of 30 (Table 4-1). A small percentage were either under 18 years of age or in the 26-29 years of age range.

Table 4-1: Q2. Which of the following age groups do you fall into?

	< 18 years	18 – 21	22 – 25	26 – 29	30+ years
Lab	0	5	5	1	1
Online	3	4	2	2	6

While this research was aimed at the novice programmers, it also included those with intermediate and expert levels in order to gain a maximum number of respondents and to examine the differences in perceptions between participants with various levels of expertise. A majority of the students who participated in this research (31%) were first-year undergraduate students, while 24% were in their second year and the rest were third-year and post-graduate students. A total of 55% of the participants were studying for a Bachelor of Computer Science qualification, while a further 17% were enrolled in a Bachelor of Information Technology course. The rest of the participants were drawn from post-graduate studies, honours studies and other courses. A total of 76% of the participants were studying on campus, while 7% were studying online and 17% were studying in mixed mode. Regardless of their current mode of study, 79% preferred to study on campus while the other 21% preferred the mixed mode. No participant selected online as the preferred mode of study. This could be because most of the participants preferred using the step-by-step instructions or following the lecturer's on-screen examples while performing the programming tasks (Table 4-11).

4.1.2. Programming Experience

In this section, participants were asked about their level of experience in programming environments and tools. Of the participants, 80% indicated that they had experience in at least one programming language and 66% of participants had done programming before becoming a university student. Only 41% of the participants had experience in web application development. Despite their previous experience in programming, 58%

of those who had done programming before coming to university rated themselves as novices.

Many of the participants had used Java, C and C++ programming languages, and some had experience in web application development languages such as HTML, ASP, PHP and ASP.Net. Visual RAD environments, such as NetBeans and Visual Studio, were the most common programming environments overall among the participants. Although some of the participants had answered that they had learned traditional environments first, they did not list any traditional environment as the programming tool that they had used before. This is probably because some programming units or learning materials made use of the code view of visual RAD environments to teach the basic programming principles; thus the participants had trouble distinguishing between the two.

When the participants were asked to rate their level of experience as a programmer, 52% rated themselves as novice; 41% as intermediate and 7% as expert. Table 4-2 indicates that while 45% of the first-year participants rated themselves as novices, the other 55% thought that they had an intermediate level of programming experience. On the other hand, the majority of the second-year students (6 out of 7) thought they were at the novice level. Looking back at the number of programming units completed for those second-year participants, many of them had done at least two to three programming units during their course of study. The third-year students who rated themselves as novice programmers had completed three to five programming units. One of them commented later in the survey that he was not interested in programming and the other indicated that programming was not the focus of his current course and his previous programming experience was out of date. These could be the reasons for rating themselves as novices.

Table 4-2: Q5. How many units have you completed in your course so far?

	Some or all of the first-year units	Some first and second-year units	First, second and some of the third-year units	Other (honours and postgrad studies)	<Blanks>
Novice	5	6	2	0	2
Intermediate/Expert	6	1	5	2	0

4.1.3. Visual RAD Versus Traditional Programming Environments

In this section, students were asked about their perceptions of the visual RAD environment versus traditional programming environment based on their previous experience. Overall, a majority agreed that visual RAD environments are easier and that they would prefer to use the visual RAD environment. However, in terms of the first environment for novice programmers, a traditional programming environment was the preferred choice among the participants.

Table 4-3 shows that a majority (55%) of the novice programmers did not have any previous experience in the visual RAD environment. Perhaps they had not done any programming at all or had just started with a traditional programming environment.

Table 4-3: Q14. Have you ever programmed in a visual rapid application development environment before (such as Microsoft's Visual Studio)?

	Yes	No
Novice	6	9
Intermediate/Expert	10	4

Table 4-4 shows that a majority (59%) of the participants agreed that, based on their experience, the visual RAD environment made programming easier. Most of the participants liked visual RAD due to the ability to find and fix errors quickly and the ability to design the user interface easily. Many of the participants who had answered "Neutral" mentioned that they had never used the visual RAD environment before. Only one participant strongly disagreed with the statement based on the reason that visual RAD could add unnecessary code that creates application "bloat". Overall, it seems that both novice and intermediate/expert participants had similar views on this.

Table 4-4: Q15. From my experience, I feel that visual RAD tools make programming easier

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
Novice	4	5	6	0	0
Intermediate/Expert	5	3	5	0	1

Table 4-5 shows that while novice developers were equally split between the perception of their ability to understand visual RAD features and functions, many of the intermediate and expert level developers disagreed that visual RAD features are hard to understand. This relates to the second supporting research question: "*Does RAD require pre-existing traditional knowledge?*" Some of the comments from participants indicated that the developers need to have a basic understanding of programming and what the tools can do before they start to program, and only that knowledge would help them to understand the RAD features and functions more readily. This could be one of the reasons that 55% of the participants preferred a traditional environment as the first environment, as indicated in Table 4-9. In addition to this, 66% of the participants also felt that traditional programming helped them understand the programming processes, such as conditions, loops and variable declarations, better (Table 4-6). They felt that traditional programming environments give the programmer a better understanding of the language overall and the workings of the processes, which gives the programmer greater control when it comes to error debugging.

Table 4-5: Q16. I feel that visual RAD features and functions can (or look to) be hard to understand

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
Novice	0	4	7	4	0
Intermediate/Expert	0	2	5	7	0

Table 4-6: Q17. I feel that traditional programming environments help me understand the programming processes better (e.g. variable declaration, condition, loops, recursion)

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
Novice	5	4	5	1	0
Intermediate/Expert	7	3	1	2	0

Table 4-7 shows that a majority of the intermediate/expert level developers (64% of intermediate/expert developers) found learning syntax in a traditional environment difficult, whereas many of the novices (40% of them) indicated otherwise. This appears to contradict their responses to the statement "I find programming of any kind difficult to learn", as shown in Table 4-12. Again, this could be related to the finding that many of the novice programmers (47% of them) were not interested in programming, which reduced their engagement with the learning process.

Table 4-7: Q18. I feel that learning syntax in traditional programming is (or looks) difficult

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
Novice	1	3	5	6	0
Intermediate/Expert	3	6	1	3	1

More than 70% of the participants had learnt a traditional programming environment first and 10% of the participants had learnt both environments around the same time (

Table 4-8). Only one participant indicated that he had learnt a visual RAD environment first. This participant indicated that he had done programming before becoming a university student, where he was most likely exposed to a visual RAD environment. He seemed to favour the visual RAD environment in general, as most of his answers on preference between the traditional and visual RAD environments indicated the latter. Apart from this participant, overall responses indicated traditional environments are the

preferred pathway among the developers. Table 4-9 indicates that 55% of the participants preferred to learn the traditional environment first. This relates to the first supporting question: "*Should RAD be taught as the 'first environment'?*" It seems that most of the participants felt traditional environments would be a better "first environment" for novices compared with visual RAD environments. However, when it comes to web application development, many of the participants preferred to use visual RAD environments, as shown in Table 4-10. This could be largely due to the visual RAD environment providing an easier and faster development system for the user interface. The integration with HTML and the need for state management in user-specific environments for web applications could also be easily handled in visual RAD environments with built-in functionality, unlike traditional environments where these tasks are tedious and time-consuming to code manually.

Table 4-8: Q20. I have learnt

	Traditional first	RAD first	Same time	Neither
Novice	10	0	1	4
Intermediate	11	1	2	0

Table 4-9: Q21. Which environment would you prefer to learn first as a novice programmer?

	Traditional	RAD	No preference
Novice	8	4	3
Intermediate	8	4	2

Table 4-10: Q19. If I were asked to program a web application, I think I would prefer to use

	Traditional	RAD	Not sure
Novice	2	4	9
Intermediate	4	7	3

Participants who preferred the traditional environment to be the first environment for novice programmers felt that novices should know the basics of programming, which are usually hidden in visual RAD tools, and once the developer knows how to program in a traditional environment, it would be easier to program in visual RAD environments. However, those who preferred a visual RAD environment to be the first environment argued that being able to write a workable program faster in visual RAD could maintain the interest of novices in programming and that it was easier to grasp the concepts for starting out. With visual RAD programming environments, working programs could more easily be put together, which increased the confidence and enthusiasm of learners (Halland & Malan, 2003).

4.1.4. Learning Experience

In this section, participants were asked about their experience in learning programming. Table 4-11 shows that 41% of participants preferred step-by-step instructions while doing programming exercises and 24% preferred to follow the lecturer's on-screen examples. The other 34% preferred to work on their own or to use textbook and online resources.

Table 4-11: Q22. When doing programming exercises, I prefer

	Step-by-step	Lecturer's example	To work on my own	textbook and online resources
Novice	8	4	2	1
Intermediate/Expert	4	3	5	2

Table 4-12 shows that while 47% of the novices agreed that programming is difficult to learn, another 33% thought it was not difficult. A majority of the intermediate and expert participants (71%) found that programming is not difficult. The difference in these reactions seems to be impacted somewhat by the interest in programming in general. Some of the participants who found it hard to learn commented that they are not interested in programming and it is hard to understand the logic-driven side of

programming, such as arrays, buffers and loops. Participants who felt that programming is not difficult found it interesting and rewarding to learn and easy to apply the knowledge across different programming languages, as it follows a defined set of rules apart from slight differences in syntax.

Table 4-12: Q23. I find programming of any kind difficult to learn

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
Novice	3	4	3	3	2
Intermediate/Expert	0	3	1	8	2

Although the majority of the participants (72%) expected to learn a number of different environments during their studies, they (55%) preferred to use the same environment for all programming tasks (Table 4-13 and Table 4-14). Some comments indicated that they found it better to be proficient in one environment rather than knowing many environments, yet not mastering any of them.

Table 4-13: Q24. I expect to be able to program in a number of different environments over the duration of my studies

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
Novice	3	6	3	3	0
Intermediate/Expert	5	7	1	1	0

Table 4-14: Q25. Where possible, I would always like to use the same environment for all programming tasks.

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
Novice	3	3	5	3	1
Intermediate/Expert	2	8	2	1	1

Table 4-15 shows 52% of the participants agreed that the first environment is the most important. One interesting comment from the participant who disagreed with the statement was that the most commonly used environment for the development is more important than the first environment. Another participant argued that it is neither the environment nor the language that is most important, but rather it was the fundamentals that he had learnt, such as variables and iterations, that were most important. However, many of the participants agreed that the first environment set the foundation for understanding programming and could be the motivation for the developer to continue programming.

Table 4-15: Q26. From my experience, the first environment learned is still the most important

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
Novice	4	5	4	2	0
Intermediate/Expert	1	5	3	5	0

Table 4-16 indicates that a majority of the novices (53% of novices) were not sure if they would be involved in programming in their future career, but a majority of intermediate and expert participants seemed confident that they would be doing programming of some sort in their future career.

Table 4-16: Q27. In my future career, I expect to

	Do programming	Program when I have to	Program as career	I am not sure
Novice	2	4	1	8
Intermediate/Expert	5	3	5	1

In summary, participants' perceptions based on their programming background before the programming exercises indicated that while the traditional programming

environment should be taught as the first environment for the novices, the visual RAD environment is still the preferred choice for web application development among all levels of expertise. It seems that while programming processes and basic programming principles are important for novices, the ability to create the user interfaces easily and quickly in visual RAD environment influenced the overall preference.

4.2. Post-exercise Survey

The post-exercise survey consists of three sections, each primarily focusing on one of the research questions discussed in Chapter One. Participants went directly to the post-exercise survey upon the completion of watching/doing the programming exercises described in Chapter Three.

4.2.1. Section One

This section focuses on the first supporting question "*Should visual RAD environments be taught as the 'first environment' to novice programmers?*" It also looked at the efficiency and pedagogical benefits of a visual RAD environment versus traditional programming environments.

Table 4-17 indicated that 45% of the participants, the majority of whom were novices, had never used the visual RAD environment before. This is consistent with, and reinforces, a similar question asked in the pre-survey (Table 4-3).

Table 4-17: Q1. Is this the first time you have used (seen the use of) a visual RAD environment (certainly for building a working application)?

	Yes	No
Novice	8	7
Intermediate/Expert	5	9

Overall, more than 65% of the participants agreed that the visual RAD environment is quicker and easier than traditional programming environments (Table 4-18 and Table

4-19), primarily because the pre-built components in visual RAD provide a head start in creating an application. One participant mentioned that even though the visual RAD environment is easier compared with a traditional environment, it is necessary to understand the underlying code generated by the visual RAD components.

Table 4-18: Q2. Based on the (video) exercises, I feel that programming in

	RAD is quicker than traditional	traditional is quicker than RAD	Equally quick
Novice	10	4	1
Intermediate/Expert	10	2	2

Table 4-19: Q3. Based on the (video) exercises, I feel that programming in

	RAD is easier than traditional	Traditional is easier than RAD	Equally easy
Novice	9	4	2
Intermediate/ Expert	10	2	2

Table 4-20 shows that overall, 34% of participants felt that they would not be able to write loops, variable and condition statements if they had started with a visual RAD development, while 28% of them thought they would. The decision is much clearer on the issue of the traditional environment, where 55% felt they would have a better understanding of writing loops, variables and conditions if they had started with a traditional environment (Table 4-21). Only 10% disagreed with this statement. This could be correlated to a question in the pre-survey (Table 4-6), where 66% thought traditional programming environments helped them understand the programming process better. One of the novice programmers commented that he had started with the traditional programming environment, which gave him a solid foundation for the flow control, and he felt that the focus on a visual RAD environment was more on the form components rather than control structures. A majority (68%) of the participants agreed that they learnt more about programming syntax and concepts using traditional environments (Table 4-22).

Table 4-20: Q4. I feel that I would be able to write loops, variables and condition statements if I had started with visual RAD development

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
Novice	0	5	5	5	0
Intermediate/Expert	0	3	6	5	0

Table 4-21: Q5. I feel that I have or would have a deeper understanding of being able to write loops, variables and condition statements if I had started with traditional development

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Novice	4	6	5	0	0
Intermediate/Expert	4	2	5	2	1

Table 4-22: Q10. I feel that I learn more about actual programming syntax and concepts using

	RAD	Traditional	Equally
Novice	4	7	4
Intermediate/Expert	1	10	3

Table 4-23 demonstrates that while a high percentage (40%) of the novices felt that they would feel confident in using traditional environments, intermediate and expert participants thought otherwise. Reason given by one of the novices was that the traditional environment was the only environment he had learnt so far and therefore he would feel more confident using it. Intermediate participants felt that the visual RAD environment was much more user friendly and could help to guide novices to create a working web application.

Table 4-23: Q6. In web application development, I feel confident as a novice developer to use

	RAD	Traditional	Equally
Novice	5	6	4
Intermediate/Expert	9	1	4

Table 4-24 shows that 66% of the participants believed they had enough technical experience to use a visual RAD environment for actual development. One of the participants again mentioned that visual RAD made the implementation easier; however, it was his traditional knowledge that helped him debug the errors in a visual RAD environment.

Table 4-24: Q9. I feel that I have enough technical experience to use a visual RAD environment for actual development as presented in the (video) exercises

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
Novice	2	7	4	2	0
Intermediate/Expert	3	7	2	2	0

Table 4-25 shows that more than 70% agreed that the first environment has a significant impact on learning programming. It serves as the first impression of the programming experience for novice programmers. A positive experience with the first environment could maintain the interest of the novice developer in programming. The first environment could also act as the foundation for subsequent learning. It is important that novice programmers understand the basic concepts of programming early in the learning stage (Raadt, 2008).

Table 4-25: Q7. I feel that the first environment has a significant impact on learning programming

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
Novice	5	8	1	0	1
Intermediate/Expert	3	5	2	3	1

While 47% of the novice participants thought traditional environments should be introduced first to the novice programmer in web application development, 40% thought that visual RAD should be first introduced (Table 4-26). Intermediate and expert participants were equally divided between visual RAD and traditional environments as a first environment. Their opinions seem to have changed after working on or watching the programming exercises. In the pre-exercise survey, the traditional environment was the preferred choice as the first environment by far. However, it could also be that this time the question was more specifically expressed towards the first environment for web application development. When asked about the preferred environment for web application development in general in the pre-exercise survey, visual RAD was the preferred choice compared to a traditional environment.

Table 4-26: Q8. Which programming environment do you think should be introduced first to novice programmers in web application development?

	RAD	Traditional	Does not matter
Novice	6	7	2
Intermediate/Expert	6	6	2

Visual dialogs, step-by-step wizards, integration of different components into one environment and the ability to create a program by drag-and-drop without worrying about code were some of the many aspects of visual RAD environments that participants felt would help novice developers learn programming. However, many of the participants also agreed that visual RAD environments could be quite confusing for

novices with feature overload and a significant part of the programming concepts hidden behind the pre-built functions. Conversely, in traditional environments these are apparent to the developers, even though functionality needs to be built from the "ground up". It seems that while developers understood the importance of the basic programming principles and concepts for novice programmers, the drag-and drop approach of a visual RAD environment provided for quicker and easier development of a web application. Overall, a slightly higher percentage of participants (45%) felt that traditional environments were still the more suitable choice to be the first environment in web application development compared with visual RAD environments.

4.2.2. Section Two

This section focuses on the second supporting question: "*Does RAD require pre-existing traditional programming knowledge?*"

Table 4-27 indicates that many (46%) of the novices felt they would need more programming experience to use visual RAD effectively. This could be because many of them had never used the visual RAD or they had very minimal programming experience in general. However, many of them indicated in Section One (Table 4-24) that they had enough technical experience to use the visual RAD as presented in the workshops. This could be because the exercises were guided step-by-step and they were fairly simple and basic programming tasks. However, the majority (53%) of the novices believed that they would require the traditional programming knowledge to program successfully in a visual RAD environment (Table 4-28).

Table 4-27: Q16. I feel that I would need more programming experience to use visual RAD environments effectively

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree	<blank>
Novice	2	5	4	2	0	2
Intermediate/Expert	1	4	3	4	1	1

Table 4-28: Q17. I feel that I would be able to program successfully in a visual RAD environment without traditional programming knowledge

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree	<blank>
Novice	0	1	3	7	1	3
Intermediate/Expert	0	6	2	3	2	1

It is interesting to note that while a majority (53%) of the novice programmers felt that previous programming experience is necessary when using a visual RAD environment, intermediate and expert participants (43%) seemed to think otherwise (Table 4-29). However, many agreed in their comments that novice developers could program a basic application in visual RAD without previous experience, but that they would definitely require programming knowledge in order to implement more complex, higher-level applications.

Table 4-29: Q18. Given the nature of visual development in RAD, I feel that previous programming experience is not necessary

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree	<blank>
Novice	0	2	3	6	2	2
Intermediate/Expert	0	6	4	3	0	1

The opinion of intermediate and expert participants on what knowledge was required to program in a visual RAD environment differed from that of the novice developers. As indicated in Table 4-30 and Table 4-31, intermediate and expert participants (43%) did not seem to care about the underlying code that makes the visual RAD work; rather, they (50%) felt that it was sufficient to program using a visual RAD environment as long as a developer knows what components to use and when. However, one of the intermediate participants argued that it is important to know the underlying code to overcome the limitations of the visual RAD environment, otherwise a programmer would be limited to only the functionality of the pre-built components that the environment could offer.

Table 4-30: Q19. As a novice programmer, I feel that it is sufficient to program using a visual RAD environment as long as I know what components to use and when

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree	<blank>
Novice	0	5	1	5	2	2
Intermediate/Expert	2	5	3	2	1	1

Table 4-31: Q20. I feel that it is not important to fully understand the underlying code that makes the visual RAD components work

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree	<blank>
Novice	0	1	2	5	5	2
Intermediate/Expert	1	5	3	2	2	1

As indicated in Table 4-32, a majority (52%) of the participants felt that being able to build a workable program is the most important aspect in learning programming regardless of the environment, mainly because that is what is required to produce an end result. Visual RAD environments are superior in this respect compared with traditional environments, where it requires great effort to cover conceptual groundwork to build a program that is workable. In a visual RAD environment, if the correct components are placed together correctly, a workable application, or part of one, should result. More participants (59%) agreed that learning syntax first is the most important aspect of becoming a programmer (Table 4-33). This reinforces the previous finding that traditional environments should be the first environment for novices. Many participants agreed that to become a "real" programmer, it is important to start with the very basics of programming. One participant commented that the program might be workable but it could be "clunky and inefficient" if the programmer did not understand the underlying programming language that drove it.

Table 4-32: Q21. I feel that being able to build a workable program is the most important aspect of learning programming, regardless of the environment

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree	<blank>
Novice	3	6	3	1	0	2
Intermediate/Expert	1	5	3	3	1	

Table 4-33: Q22. I feel that learning programming syntax first is the most important aspect of becoming a programmer

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree	<blank>
Novice	6	4	2	1	0	2
Intermediate	4	3	3	1	1	

On the whole, 62% agreed that the ability to learn new environments quickly is more important than the type of environment it is (Table 4-34). They felt that it is essential for a programmer to be versatile in the different programming environments that they may came across in the workplace. This is probably the reason that a majority of participants expected to learn different programming environments during their course of study.

Table 4-34: Q23. Regardless of traditional or visual RAD methods of web programming, I feel that being able to learn any new environment quickly is more important than which type of environment it is

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree	<blank>
Novice	1	9	1	2	0	2
Intermediate/Expert	2	6	5	0	0	

In terms of learning environments, 38% of participants agreed that visual RAD environments are more suited to self-learning in terms of web application development (Table 4-35). This could be due to some of the functionality in a visual RAD

environment being based on step-by-step wizards, which can make it easier for novice developers to work on their own. Traditional environments on the other hand, are perceived to be more suited to classroom-based learning (Table 4-36) due to their "blank canvas" starting point. The basic concepts, theories and syntax in traditional programming probably require more guidance from an instructor compared with visual RAD environments.

Table 4-35: Q24. Which environment do you feel is appropriate for novice developers for self-learning in the web application development context?

	RAD	Traditional	Equally	<blank>
Novice	6	6	1	2
Intermediate/Expert	5	3	5	

Table 4-36: Q25. Which environment do you feel is appropriate for novice programmers for classroom-based learning in web application development context?

	RAD	Traditional	Equally	<blank>
Novice	4	7	1	3
Intermediate	4	5	4	

Overall, many of the participants felt that basic applications developed in visual RAD environments, as presented in the workshops, would not require pre-existing traditional programming knowledge. However, to be able to implement better functional and customised, or more complex and larger applications, it would necessary to have underlying traditional knowledge.

4.2.3. Section Three

This section focuses on the third supporting question: "*Which is the preferred programming environment among novice developers?*" Many of these questions are

related to the programming exercises that participants watched or completed during their workshop sessions.

The participants were presented with three programming exercises for each of the programming environments. The first exercise involved conducting a basic search from a database, developing a table based on the keyword entered and displaying the results in an HTML table. The second exercise involved editing and deleting the records in the table, and the third exercise involved adding new records. Table 4-37 through Table 4-39 indicate that a majority of the participants preferred the visual RAD environment for these exercises. Even though it is quite evident that in some exercises a traditional environment seemed easier and in other exercises visual RAD seemed easier, most of the participants gave the same preference for all three questions. Only three participants changed their answers from one question in the pre-survey to the subsequent question in the post-survey.

Table 4-37: Q28. Which environment do you prefer for "Search" (based on the video exercises)?

	RAD	Traditional	Both	<blank>
Novice	6	3	3	3
Intermediate	7	4	0	3

Table 4-38: Q29. Which environment do you prefer for "Edit/ Delete" (based on the video exercises)?

	RAD	Traditional	Both	<blank>
Novice	5	4	3	3
Intermediate/Expert	6	3	1	4

Table 4-39: Q30. Which environment do you prefer for "Insert" (based on the video exercises)?

	RAD	Traditional	Both	<blank>
Novice	4	5	3	3
Intermediate/Expert	7	4	0	3

Participants from the in-class sessions were given an optional challenge exercise in both the visual RAD and traditional environments. Many of them did not attempt the exercises due to the time constraint discussed earlier. Only four managed to complete the challenge in the visual RAD environment and five completed the challenge using the traditional environment (Table 4-41 and Table 4-42). It should be noted that though these participants indicated that they completed the challenge exercises, they did not upload the code in the optional upload area provided to them.

Table 4-40: Q31. Did you manage to complete the challenge exercise using visual RAD environment?

	Yes	No	<blank>
Novice	2	4	1
Intermediate/Expert	2	3	0

Table 4-41: Q32. Did you manage to complete the challenge exercise using traditional environment?

	Yes	No	<blank>
Novice	3	3	1
Intermediate/Expert	2	3	0

Instead of the challenge exercises, online participants were asked if they thought they would be able to code the example applications in these environments. Table 4-42 and Table 4-43 showed that 35% felt they could code in a visual RAD environment and 47% thought they could do so in a traditional environment.

Table 4-42: Q31. Based on the video exercises, do you think you could code the example application in a visual RAD environment?

	Yes	No	<blank>
Novice	3	3	2
Intermediate/Expert	3	2	4

Table 4-43: Q32. Based on the video exercises, do you think you could code the example application in a traditional environment?

	Yes	No	<blank>
Novice	3	3	2
Intermediate	5	1	3

Overall, 31% of participants preferred the visual RAD environment based on the programming exercises and 24% preferred the traditional environment.

Table 4-44 shows that novices are equally divided between the visual RAD and traditional environments, while intermediate and expert participants seem to prefer the visual RAD environment. Many of the participant preferences did not change much from the pre-survey when they were asked about their preferred environment for web application development. Only one participant who selected the traditional environment as the preferred choice in the pre-survey changed to the visual RAD environment as the overall preference for the given programming exercises.

Table 4-44: Q33. Overall, based on these (video) exercises, I would prefer

	RAD	Traditional	Both	<blank>
Novice	4	4	4	3
Intermediate/Experts	5	3	3	3

Even though the majority of participants preferred the visual RAD over traditional environment, 41% (mostly novices) would use the traditional environment if they had to further develop the exercises (Table 4-45). This might be based on the same reason that they had only learnt a traditional environment so far in their courses and would not feel confident enough to program in a visual RAD environment.

Table 4-45: Q34. If I had to further develop these exercises (with extra functions), I would use

	RAD	Traditional	Both	<blank>
Novice	4	8	0	3
Intermediate/Expert	6	4	0	4

Table 4-46 indicates that most of the participants (34%) found that the traditional exercises were easier to understand, primarily because they had prior knowledge in the traditional programming. It could also be that because in the exercises, the participants were primarily required to copy and paste code to a file, while in the visual RAD environment they actually had multiple steps to follow in detail to achieve a similar result. For example, the search function in the traditional environment required four selections of code to be copied and pasted into a single .php file, while in the visual RAD environment it meant dragging a datasource control, aligning this with the correct database and database table, dragging a gridview control, then linking the two together, after which a textbox control had to be integrated with the datasource search method. While these steps were all done visually, it could be that the cognitive load required to read and apply the correct sequence was higher than that of copy/paste.

Table 4-46: Q35. Which set of exercises do you feel is easier to understand?

	RAD	Traditional	Both	<blank>
Novice	3	5	4	3
Intermediate/Expert	5	5	1	4

Table 4-47 shows that 41% of the participants agreed that teaching and learning materials are more important than the type of programming environments. This correlates with question Q37 (

Table 4-48), where 38% of participants agreed that availability of useful resources influenced their reaction to these programming environments. Overall, a large percentage of participants (34%) agreed that the traditional environment has the most useful online resources (Table 4-49). This is probably because with visual RAD environments, it is hard to find examples where the actual visual interface is used as the basis for instruction. A large number of textbooks and online tutorials might purport to demonstrate step-by-step instructions in, say, Microsoft Visual Studio, but may have all the examples presented in code-behind mode only.

Table 4-47: Q36. I feel that the teaching and learning materials are more important than the type of programming environments

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree	<blank>
Novice	1	4	6	1	0	3
Intermediate/Expert	1	6	4	0	0	

Table 4-48: Q37. I feel that availability of useful resources (textbooks or websites) influenced my reaction to visual RAD versus traditional programming environments

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree	<blank>
Novice	1	4	6	1	0	3
Intermediate	1	5	4	0	1	

Table 4-49: Q38. Which environment did you feel had the most useful online (web-based) resources (such as tutorials/code examples)?

	RAD	Traditional	Both	<blank>
Novice	2	5	5	3
Intermediate/Expert	2	5	3	

Even though these exercises did not require the participants to set up and configure their environments, 48% of participants agreed that this could affect their reaction to the programming environment (Table 4-50). In most cases, visual RAD was deemed as having an advantage over a traditional environment in this area because most of the visual RAD environments are also integrated development environments and everything can be installed from a single package. In traditional environments, the compiler, database, editor and debugging tools are often separate items that need to be configured together.

Table 4-50: Q39. I feel that setup and configuration issues (of the environment) could affect my reaction to RAD versus traditional programming environments

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree	<blank>
Novice	1	8	2	1	0	3
Intermediate/Expert	1	4	3	2	1	

The quantitative analysis within this chapter indicates that the visual RAD environment was the preferred choice for all of the programming exercises and overall web application development, despite better availability of resources in traditional environments. In terms of pedagogical benefits, participants found that traditional environments are still the most appropriate environment novice developers. The following chapter will provide further context to the quantitative results presented in this chapter.

5. Discussion

This chapter discusses the findings from this study based on the three main focuses of this research: the traditional environment, the visual RAD environment and the learning sequence of these programming environments.

The findings confirm the current trend in programming courses, which is starting the learning phase with a traditional programming environment first followed by visual RAD in later stages. This sequence still appears to be the preferred pathway for novice developers in learning programming for web application development even though the intermediate and expert developers were equally split on the preferred choice.

Based on the qualitative data from participants, the following factors have been identified as important aspects of learning programming:

1. Understanding programming syntax and concepts
2. Understanding underlying logic of the program
3. Ability to enhance further
4. Ability to build workable program
5. Interest.

5.1. Understanding Programming Syntax and Concepts

A majority of the participants indicated in the survey responses that understanding the programming concept is the most important aspect of learning programming. Basic programming concepts learnt in introductory programming courses includes loops, variable declaration, recursion, conditions and objects. Understanding how and why the program behaves in certain ways is the foundation to understanding more complex and difficult programs. Gries (1974, 2006) argues that understanding programming concepts at the introductory programming level is the main focus of traditional programming courses and he believes that it should still be the case. Although many participants in this study agreed that they would be able to use these programming concepts if they had started with a visual RAD environment, more of them felt that a traditional environment helped them understand these concepts more fully. Open-ended survey responses indicate that it is because the traditional environment forced them to manually program "from scratch", that they could absorb the knowledge better. The fear of hidden

programming concepts is always a challenge in introducing visual RAD in introductory programming courses (Schaub, 2009; Tew, McCracken, & Guzdial, 2005). It is argued in Tew et al. that these programming syntax and concepts could have a remarkable impact on the retention of novice programming students, and the focus should be more on inviting and retaining students, Although, as indicated in Chapter Two, learning the programming concept and syntax is generally perceived to be one of the most difficult aspects of learning programming (Lahtinen et al., 2005).

5.2. Understanding Underlying Logic of the Program

Understanding the underlying logic of a program can be derived through different approaches. With traditional environments, this could be done by reading the lines of code and trying to understand how the program works. With visual RAD, it could be done by identifying the links between different controls and experimenting with their settings until something works. A majority of participants agreed that it requires a great deal of understanding of how the components work in a visual RAD environment to make them work correctly. Mannila (2006) argues that there is very little correlation between the ability to write code and read the code. Although students may understand the programming syntax and concepts, it does not mean that they would be able to understand what the program does and the logic behind it. Some participants indicated that this is the most important aspect in learning programming, i.e. to be able to develop or modify existing application code and structures.

5.3. Ability to Enhance Further

Most of the participants agreed that the ability to enhance the existing program is one of the key aspects in learning programming. Software is rarely a one-off project, as it requires new functions and enhancements as user requirements evolve (Sommerville, 2007). The ability to enhance further relies heavily on the ability to read and understand the underlying programming logic. Many of the participants agreed that traditional programming environments give the programmer complete control and do not limit the

ability to enhance further. A visual RAD tool, on the other hand, can be limited by the controls the environment provides.

5.4. Ability to Build a Workable Program

Most participants felt that the ability to build a workable program quickly is one of the most important aspects in learning programming. With traditional programming environments, it can take hundreds or thousands of lines of code to create even a small workable and functional program. With visual RAD, this can quickly be done through simple dragging and dropping of icons given that the appropriate controls are available. As indicated in the research carried out by Halland and Malan (2003), teachers and students found it more rewarding and interesting to work in visual RAD because of this. It was argued by some participants that although the program might be workable, the program might not function as expected if the student developer does not understand the underlying structure and logic. However, others argued that it is the workable result that the clients want in the programming industry. As is apparent in the literature, it is this aspect of visual RAD programming that makes it popular among non-programmers (Kaneshige, 2009; Rode, 2004).

5.5. Interest

Many participants found the traditional programming environment non-motivating and daunting because it requires numerous lines of code to achieve even tiny outcomes and is difficult to debug along the way. Tew et al. (2005) provided a strong argument that interest and motivation should be the first priority in the selection of environment in introductory programming course design. Visual RAD provides an environment that allows for the building of workable programs quickly and keeps the programmer motivated to further develop the program. Many of the participants in this study indicated that they would use the visual RAD tool for future developments in web application developments even though many of them had no prior experience in using these tools. Findings from this thesis and the associated literature may be indicative of

reasons why researchers are trying to introduce visual RAD environments into introductory programming courses (Chainini & Yamada, 1998; Goldweber et al., 2006; Halland & Malan, 2003; Seals, 2005).

5.6. Traditional Programming Environment

The traditional programming environment was perceived as a good learning tool for novice programmers. A majority of novice and intermediate participants felt that traditional programming environments provide better pedagogical benefits compared with visual development. Developing the program "from scratch" helped them understand the concepts and flow of the application and gave them confidence to enhance the program further.

Table 5-1 shows the comments from some participants regarding the benefits of traditional programming in learning programming processes such as variable declarations, loops and recursions.

"If there are any errors or something isn't displaying right you have the ability to find the problem and correct it." **Participant #3: Lab, Male, 22 – 25, BITHons, expert, Completed**
"They give the programmer complete control." **Participant #10: Lab, Male, 18 – 21, BCompSc, intermediate, Completed**
"Have a better understanding of the workings of the processes." **Participant #18: Online, Male, 30+, BSc(CompSc), intermediate, Completed**
"Working with the skeleton I guess you get a feel of what is happening behind the scenes, so when it comes time to figuring out what might be wrong with a program you have a head start because you know what is interacting with what." **Participant #25: Online, Male, 26 – 29, BCompSc, novice, Completed**
"I did start with traditional development, and it did give me a solid grasp of flow control." **Participant #11: Lab, Male, 22 – 25, BCompSc, novice, Completed**
"An understanding of the logic behind programming does help." **Participant #22: Online, Male, 18 – 21, BCompSc, intermediate, Completed**

"I think knowing the processes behind the visual interface helps get a better understanding of how the program actually is functioning."
Participant #24: Online, Male, 22 – 25, CompSc/CreatMusTech, novice, Completed
"Since I would need to write more code in a non-RAD environment, I guess that I would achieve a better understanding of loops, variables and condition statements."
Participant #28: Online, Male, 30+, MCompSc, intermediate, Completed

Table 5-1: Students comments on benefits of a traditional environment for programming processes

The above participant comments indicate that being able to see the code in traditional programming environments helps students to see the underlying logic and design of the application that they are building. Although the textual interface makes it harder to visualise the application and debug the syntactical errors, many agreed that it is easier to debug the logical error in traditional programming environment compared with visual RAD. This could be because traditional programming environments are primarily designed for more linear programming, especially for the exercises provided to participants in this research (Kolling & Rosenberg, 1996). Although Mannila (2006) argues that novice developers have problems reading code and understanding logic, most of the time the business logic is in close proximity to where it is required within a traditionally developed application. In other words, calculation code is typically near input code within a procedurally written application. Visual RAD tools on the other hand, have very abstract interfaces and the code of the components are largely hidden, or the code that influences and controls one object may be discretely separated from another control upon which it is dependent (Calloni & Bagert, 1994). Debugging the logical error requires understanding of what the controls do, how they work, how they interact and in what context they sit.

Although three of the participants (one novice and two intermediate level developers) had largely negative responses towards the traditional environment during the pre-survey, some of the responses changed during the post-survey. One of these participants indicated early in the pre-survey that:

"No point using traditional, waste of time. Same thing done with visual RAD, better

convenience, development atmosphere and gives you more control over your project without wasting time."
Participant #14: Online, Male, 18 – 21, BInfoTech, novice, Completed

However, this participant's answer changed during the post survey and he indicated that it would indeed require traditional programming knowledge to program in a visual RAD environment. His answers were probably influenced by the tone of the questions rather than the programming exercises, as the click stream indicated that this participant had spent very little time on the programming exercises. Nonetheless, all three participants' overall preferences still leaned towards the visual RAD rather than traditional programming environment.

Most participants agreed that traditional environments fulfil the first three aspects listed earlier in this chapter (understanding programming syntax and concepts, understanding underlying logic of the program and ability to enhance further). However, it is also evident that traditional environments are somehow more difficult to learn because of the syntax and having to build the program "from scratch". It could take hours to build a workable program in a traditional environment. These factors could portray the impression to novice developers that programming is difficult. While it is important to learn the basics of programming at an early stage, it is also important to keep the interest and enthusiasm of the new programmers in order for them to continue with programming. Bergin and Reilly (2005) found in their research that motivation has a huge impact on performance in learning programming. The findings in this thesis indicate that participants seem to be more enthusiastic in developing web applications with visual RAD rather than traditional environments. Table 5-2 illustrates some negative reactions from participants in regards to a traditional environment.

"Not easy to picture layout."
Participant #7: Lab, Male, 18 – 21, CompSc, novice, Completed
"No error checking or correction—makes it difficult to learn and even harder to keep learning."
Participant #9: Lab, Male, 18 – 21, BCompSc, novice, Completed
"The necessity to learn a complex and exacting syntax."
Participant #11: Lab, Male, 22 – 25, BCompSc, novice, Completed

"...can be hard to initially understand basics and fundamentals."
Participant #15: Online, Female, < 18, BCompSc, novice, Completed
"It can be difficult to know how the program will look when working with just text. And know just what libraries already exist for use."
Participant #24: Online, Male, 22 – 25, CompSc/CreatMusTech, novice, Completed
"In more complex applications—such as a web application that deals with HTML code and another language—a novice programmer may struggle to absorb and properly understand all of this new information at once."
Participant #15: Online, Female, < 18, BCompSc, novice, Completed
"All the syntax can be daunting."
Participant #26: Online, Male, 18 – 21, BCompSc, intermediate, Completed

Table 5-2: Students comments on difficulties of a traditional programming environment

As the comments indicate, many novice participants found the traditional programming complex and difficult to learn, mainly because of the syntax and the non-visualisation of the application. Results indicate that although intermediate and expert participants felt that programming was not difficult, they felt that it is difficult to learn syntax in a traditional programming environment. The traditional programming environments often lack basic help in writing programs, such as syntax checking and displaying of available objects and methods, which makes it harder, especially for the novice programmers, to build a workable program in a short time frame. During the in-class workshops, the code samples for traditional programming exercises were provided to the participants. Even though they were only required to copy and paste the code from the instructions to the text editor, some of the participants encountered syntax problems such as missing a ';' or '}'. It took them a while to figure out the problem, or in some cases, they resorted to restarting the code from the beginning.

In terms of the learning resources available, a traditional programming environment has some advantages over visual RAD environments. Writing a program in traditional programming environments is limited only by the programming language, unlike a visual RAD environment, where it is constrained by both language and the environment/software. There are more tutorials and code samples available online for

traditional programming environments. Even though participants agreed with this perception, they indicated that regardless of the availability of training resources, a majority still preferred the visual RAD environment overall.

While a traditional environment appears to be the more suitable environment for novice developers in terms of learning programming, its ability to boost the interest and motivation of the new programmers seems far behind that of visual RAD environments.

5.7. Visual RAD Environment

The responses to visual RAD environments were largely positive. A majority of participants liked the rapidity of building a workable program and the ability to visualise the application even before it is completely built. Even though many of the participants indicated that they had never used the visual RAD environment before, it still gave them an impression that it would be better suited to web application development compared with traditional programming environments.

Table 5-3 details some of the comments from participants in regards to the ease and rapidity of programming in visual RAD environments.

"Find and fix common errors quickly, due to better error output." *"Usually pretty self-explanatory interfaces."* **Participant #9: Lab, Male, 18 – 21, BCompSc, novice, Completed**
"RAD tools make component-based programming easier, especially when form design is involved." *"I don't think tools like Visual Studio are hard to use."* **Participant #11: Lab, Male, 22 – 25, BCompSc, novice, Completed**
"Spares the trouble of writing code to create the user interface, which saves a lot of time in a working environment under strict time constraints." *"It's fairly simple if you have a basic understanding of programming languages and what you're doing."* **Participant #19: Online, Male, 30+, BInfoTech, intermediate, Completed**
"Shows where errors are, easier to navigate and integratable with other tools." **Participant #24: Online, Male, 22 – 25, CompSc/CreatMusTech, novice, Completed**

"Visual RAD tools make interface design much easier: trying to make a GUI in Java using JFrames, and coding each element on the screen, is a nightmare. RAD environments can also take care of some very 'fiddly' aspects of programming, e.g. database connectivity."
Participant #27: Online, Male, 18 – 21, CompSc, expert, Completed
"Palettes with drop on form capabilities and abilities to change properties, code events and procedures with ease."
Participant #29: Online, Male, 30+, BSc, intermediate, Completed

Table 5-3: Ease and rapidity of programming in visual RAD environment

Based on the comments, it seems evident that the drag-and-drop feature in visual RAD environments makes it quicker to implement an applications graphical user interface. Not having to write code "from scratch" gives the programmer a head start in the application development process. However, the downfall is that it can be very unclear to the programmer what is happening behind these drag-and-drop components. This could cause problems if customisation of the standard component is required. In such a situation, more extensive knowledge of programming processes would be necessary in order to understand how things work. The survey responses indicate that a majority of the participants felt that they would not be able to write loops, conditions and variable declarations if they had started with visual RAD.

"I think it's not good to highly depend on RAD environment for novice programmer."
Participant #5: Lab, Male, 18 – 21, BInfoTech, intermediate, Completed
"Can get confusing."
Participant #7: Lab, Male, 18 – 21, CompSc, novice, Completed
"Working with components so much doesn't help you when you have to code the whole thing yourself."
Participant #11: Lab, Male, 22 – 25, BCompSc, novice, Completed
"Logic errors. RAD seems to be more suited to lazy developers and to encourage a lack of pseudocode development"
Participant #12: Lab, Male, 22 – 25, BCompSc, novice, Completed
"...overload of toolbars/icons"
Participant #13: Online, Male, 26 – 29, IT, novice, Completed

"Not having an understanding of the logic behind the application."
Participant #22: Online, Male, 18 – 21, BCompSc, intermediate, Completed
"That a lot of the programming is hidden behind prebuilt functions and buttons."
Participant #24: Online, Male, 22 – 25, CompSc/CreatMusTech, novice, Completed
"As in question 11, the drag and drop doesn't teach novices the basic structure of a program, which makes it hard to go from a RAD to a traditional environment"
Participant #22: Online, Male, 18 – 21, BCompSc, intermediate, Completed

Table 5-4: Downfalls of visual RAD environment

Table 5-4 shows the disadvantages of visual RAD environments for novice developers as perceived by the participants. Most indicated that hidden coding logic in visual RAD is the major downfall, as a novice developer would require a knowledge of basic programming in order to progress further as a developer. During the in-class workshops, though step-by-step instructions were provided, many of the participants struggled with the configuration of the components and where to make the changes, mainly because of the interface complexity and a lack of understanding of how the components work. Visual RAD environments provide numerous features and components and each component has numerous properties. Trying to understand all these can be quite daunting to a novice developer. Although understanding the toolbars and icons could come with experience in using the environment, understanding the logic behind the components and their functionality requires knowledge of fundamental programming concepts.

5.8. Learning Sequence

One main focus of this research was to investigate the impact and selection of first programming environments. Survey results show that most of the participants learnt traditional programming environments as their first environment. Only one participant (Participant #6: Lab, Male, 18 – 21, BCompSc, intermediate, Completed) indicated that he learnt visual RAD before a traditional environment. As shown in Chapter Four, this participant appeared to have learnt a visual RAD environment before becoming a university student. He still preferred the first environment for almost all of the aspects mentioned covered within the survey. Apart from this participant, there were mixed

reactions between novices and intermediate/expert participants on the important aspects of learning programming.

A majority of the participants agreed that the first environment has a major impact on learning programming. It sets an important foundation for the whole learning process of programming. Table 5-5 shows some comments from participants in regards to the importance of the first environment.

"A bad IDE can put you off a language for good."
Participant #9: Lab, Male, 18 – 21, BCompSc, novice, Completed
"The first environment you are exposed to always relates to every environment you are exposed to subsequently."
Participant #12: Lab, Male, 22 – 25, BCompSc, novice, Completed
"It forms the foundation of your thought processes to the field."
Participant #24: Online, Male, 22 – 25, CompSc/CreatMusTech, novice, Completed
"What you learn first will often leave you thinking that that was the 'right' way of doing things, and anything you learn after must be 'wrong', because it's different."
Participant #26: Online, Male, 18 – 21, BCompSc, intermediate, Completed

Table 5-5: Importance of first environment

Programming is generally accepted as a challenging subject within the literature. However, the findings from this research indicate that intermediate and expert developers thought otherwise. This is likely to be because they feel proficient at programming and, at this point in time, it seems easy to them. Open-ended responses revealed that many of these intermediate and expert developers enjoy programming and they had a solid amount of programming experience upon which to base their confidence in their own programming capability. On the other hand, many of the novices indicated that they found programming difficult. Although not many of these participants provided the reason for this, some responses indicated that they were very new to programming and did not have enough confidence or were simply not interested in programming.

As mentioned previously, there are many important aspects of learning programming that need to be considered when selecting the first programming environment. Many of the participants considered the ability to understand programming concepts and syntax as the most important aspects of learning programming, and almost 60% agreed that they learned more with traditional programming environments. For traditional programming, as most of the programs have to be hand-coded, it is necessary to know the programming processes before starting to code the application. In a visual RAD environment, code competency it is not required until a later stage when the developer needs to further enhance pre-existing features. Almost 50% of the participants agreed that being able to build a workable program is the most important aspect of environment selection. Most of the comments indicated that, at the end of the day, the output is all that matters, especially for the novices, to keep them motivated and interested in programming.

Table 5-6: Importance of ability to understand programming concepts shows the comments from participants in regards to the important aspects of programming.

"...understanding the languages concepts and syntax is more important." **Participant #3: Lab, Male, 22 – 25, BITHons, expert, Completed**
"Learning programming concepts first helps more." **Participant #15: Online, Female, < 18, BCompSc, novice, Completed**
"You may be able to create a workable program, but that program might be clunky and inefficient because you never took the time to learn a more detailed understanding of the language." **Participant #24: Online, Male, 22 – 25, CompSc/CreatMusTech, novice, Completed**
"The most important part of learning is to understand concepts and how things work." **Participant #27: Online, Male, 18 – 21, CompSc, expert, Completed**
"Programming LOGIC is far more important to learn—syntax can be easily learned once programming logic and techniques are properly under." *"Visual RAD environments tend to hide information. For example: using a wizard to retrieve records from a database table as opposed to writing the code to establish the connection and execute SQL statements. The wizard does not help learning in this instance."* *You can build a workable program using a drag-and-drop technique in a Visual RAD environment with almost no programming knowledge. Therefore you are not so much a programmer—you are someone who just knows how to use a specific tool.* **Participant #28: Online, Male, 30+, MCompSc, intermediate, Completed**

"No matter which environment you use, you still need to know the basics of programming syntax and concepts. RAD would be easier to learn but also easier to skip over vital programming concepts." *"Learning the basics is always the best way to go... and it all starts at the syntax level as far as I can tell."* *"A workable program is all the client wants aint it?"*
Participant #29: Online, Male, 30+, BSc, intermediate, Completed

Table 5-6: Importance of ability to understand programming concepts

One of the interesting findings from this research is that in all three questions, with regards to pre-existing knowledge for visual RAD environments, many of the participants agreed that a visual RAD environment requires the knowledge of a traditional programming environment, even though some of them preferred to have visual RAD as the first environment. It might be that a developer could successfully work in visual RAD as a novice, but in order to fully make use of the features provided in the environment, they might require more comprehensive knowledge of programming. Participants indicated that their previous experience in traditional programming helped them understand more of the visual RAD environment.

The visual RAD environment was by far the preferred environment for web application development among the participants. However, there are differing opinions on the first environment for web application versus the first environment for other types of applications. Participant #10 (Lab, Male, 18 – 21, BCompSc, intermediate, Completed) said during the interview, "I would use RAD for more interface-oriented applications like web applications but traditional for more logic oriented applications like games." During the pre-survey, when asked about the preferred first environment for novice developers in general sense, 55% preferred a traditional environment and only 28% preferred visual RAD. However, when asked about the preferred first environment for novice developers for web application development during the post-survey, the figures changed noticeably. A larger number of the participants (45%) preferred a traditional environment compared to visual RAD. However, overall, it is still preferable to have learned a traditional environment first before the visual RAD for the novice developers, mainly because of their current experience with the traditional environment as well as the pedagogical benefits of it.

6. Conclusion

The goal of this study was to examine the impact and selection of visual RAD environments for novice developers in learning web application development. This chapter will summarise the findings and discussion in the context of the three supporting questions and primary research question.

6.1. Visual RAD as First Environment

Supporting question # 1: *"Should visual RAD environments be taught as the 'first environment' to novice programmers?"*

The results of this research provided very positive feedback towards the visual RAD environment from novice, intermediate and expert programmers. Most of them found programming in visual RAD motivating and exciting, as they were able to see the results quickly. This finding correlates well with the previous literature from Halland and Malan (2003). Fast development and visualisation of the application in visual RAD tools help students build applications easily and almost error-free (given application complexity). However, the participants were concerned that visual RAD might not be suitable as a first programming environment for novice developers, mainly due to the hidden programming concepts. This issue of visual RAD has been raised by previous researchers in the introductory programming field, as discussed in the literature review. One of the main advantages of a visual RAD environment, the reduction of code cutting and related syntax complexity (Dann, Cooper, & Pausch, 2005), becomes the major disadvantage in selecting visual RAD in programming courses. Many participants believed that a traditional programming environment is required to develop core programming skills. This relates to a similar study conducted within the Alice programming environment for game development by Sykes (2007), where he concluded this to be one of the major disadvantages of the visual programming environment. However, visual RAD was described as a rewarding and enjoyable environment by many of the participants. According to the literature, visual RAD is believed to help in attracting novice developers' interest in programming (Haden, 2006; Seals, 2005; Sykes,). Another solution, as explored by other researchers in order to achieve both benefits, is to introduce both environments at the same level (Calloni et al., 1997;

Cilliers et al., 2005). Although it showed solid results in student performance in exams, one must take into consideration the amount of time spent to introduce each of the environments to novice programmers to a usable level. Introducing both at the same time could confuse students in understanding the basics of these environments. So when it comes down to selecting visual RAD as a first environment, it is a matter of a trade-off between pedagogical benefits and capturing student interest to enhance the success rates of the introductory programming courses.

If based on the participants' reactions from this research alone, visual RAD environments should not be taught as a first environment to novice programmers, although it perhaps should be introduced as early as possible thereafter.

6.2. Traditional Programming Knowledge Experience for Visual RAD

Supporting question # 2: *"Does visual RAD require pre-existing traditional programming knowledge?"*

Although a few problems were encountered by participants at first in familiarising themselves with the components and functions provided by the visual RAD environment, many of them (based on the observation in labs) managed to solve the programming problems presented in the workshops. The responses also indicated that many of the novice participants felt confident in using the visual RAD and believed they had enough technical experience for the given exercises. Many of the visual RAD environments, especially in terms of web development, are designed to require little or no programming knowledge to build simple data-driven applications (Goldweber et al., 2006; Kaneshige, 2009; Rode, 2004). However this can only be accomplished with a thorough understanding of the components and functions provided in the given visual RAD tool. Based on observation during the in-class workshops, the first problem encountered in the visual RAD exercises for many of the participants was locating the correct component to use. Although step-by-step instructions were given, a slight difference in the display of toolbox and property dialogs from sample screenshots could easily confuse them. This was mainly because they did not have the in-depth understanding of what each component was used for and how the basic structure of the visual RAD tool worked. Although this problem was later reduced as they continued with the exercises, a majority of the novice participants indicated that they would not

feel confident in using the visual RAD to enhance further on the programming exercises, especially without examples to follow. The responses showed that it is important to fully understand the components in visual RAD environments and that traditional programming knowledge was necessary, especially if they were to enhance the program beyond the base visual RAD capabilities. For basic functional web applications this is not necessary, but the limitations of visual RAD would become apparent when the project grew bigger and more complex functionality was required. Kaneshige argues in his article that "real"/proficient programmers are still required to develop and maintain complex applications. The graphical drag and drop only approach of the visual RAD environment seems to be unrealistic in "real-world" problems, which facilitated many of the RAD tools to implement both graphical and code views (Peter, 2009; Sykes, 2007; Wong, 2006).

Developing basic and generic functional web applications might not require traditional programming experience, but for real-world applications and larger enterprise solutions, traditional programming experience is a must.

6.3. Preferred Programming Environment

Supporting question # 3: *"Which is the preferred programming environment among novice developers?"*

Participant preferences in programming environments differed according to the aspect of the programming. Many of the responses indicated that visual RAD was the preferred environment for overall web development due to the easier integration with GUI components and the convenience of visualising forms without the need to code for hours. The ability to build workable programs quickly, along with interactive and interesting ways of developing programs, also contributed to this. Nevertheless, participants believed that the preference would be different for different types of programs. Based on the interview response from a novice developer, a traditional environment is preferred for game programming. This is in contrast to the previous literature, where visual RAD was believed to be a better environment to introduce game programming for novice developers (Dann et al., 2005; Goldweber et al., 2006; Haden, 2006; Sykes, 2007; Walter et al., 2007). Overall, responses indicated that visual RAD was preferred more for GUI-based applications. However, in terms of learning,

traditional environments provided better learning of fundamental concepts and the syntax of programming (Halland & Malan, 2003; Raadt et al., 2002, 2003). It was the preferred "first environment" for all types of programming tasks mainly for that reason. Most of the intermediate and expert level participants believed that the traditional environment is the better environment for learning the basics of programming.

Based on the context of this research being web application development, visual RAD was the preferred programming environment to use among both novice and expert level participants, but traditional was the preferred first environment to learn.

6.4. Student Reaction to Visual RAD versus Traditional Programming Environments

The primary research question of the study was: *"What is the student reaction to visual RAD versus traditional programming environments for novice programmers in a web application development context?"*

Student reaction to visual versus traditional programming environments was very positive and indicated that visual RAD environments have an important role to play in terms of the learning experience of novice developers. While it seems evident that novice developers still feel the need to learn traditional programming environments first, it appears that in the long term, a majority of the participants in this study see themselves as developing in visual environments. The ability to use pre-built controls, or objects that visualise complex interface features, along with the rapidity of development and prototyping, are seen as the key benefits of visual environments. Participants also experienced a higher level of motivation when using the visual environment presented in this study, as they went from a "blank slate" to functional web applications in a matter of minutes. Experiencing such progress so quickly seems to be an important factor for novice programmers, who can quickly become exasperated when working with more traditional environments, which have significant learning overheads in terms of integrating code to generate both client and server-side functionality. In would seem that web applications in particular are well disposed to visual environments, as the messy integration of HTML, server-side code and database connectivity is handled in stand-alone, pre-configured objects.

6.5. Limitations of Research

As stated in Chapter Three, this research was generalised for different programming environments based on the experiment using one programming language and one tool for each of the programming environments. It is possible that participant reactions might vary if different languages or tools were used. However, due to the time and resource limitations it was not possible to increase the scope of the experiment or the number of environments used. While this study employed different levels of students with different programming expertise, the sample size could be considered quite small to allow for generalisation of the results. The large number of survey questions did somewhat offset the small number of participants; however, this led to the issue of survey fatigue, which subsequently impacted the number of fully completed surveys. Many of the findings relied heavily on participants' prior knowledge and experience and on only three programming exercises, which could have had significant impact on participants' reactions, especially for visual RAD, as many of the novice programmers were not exposed to the visual RAD environment previously. Better understanding of the impact and selection of these programming environments could be further developed. However, this study has produced some interesting findings and could easily be expanded into a larger study.

6.6. Recommendations for Further Research

Visual RAD environments were found to provide a positive environment for programming web applications, although many of the participants had not used the environment before participating in this research. It would be more appropriate to conduct the workshops over a defined period of an introductory programming course with exposure to different types of visual RAD programming environments for more in-depth perspectives on these environments. It would also be better to monitor student performance in subsequent programming courses as the impact of the first environment is felt.

7. Reference List

Agarwal, R., Prasad, J., Tanniru, M., & Lynch, J. (2000). Risks of rapid application development. *Communications of the ACM, 43*(11), 177-188.

Ashenden, D., & Milligan, S. (1999). *The good universities guide: Universities, TAFE and private colleges in 2000.* Subiaco: The Australian; Hobsons Australia Pty Ltd.

Babbie, E. R. (2000). *The practice of social research* (9th ed.). Belmont, CA: Wadsworth/Thompson Learning.

Bergin, S., & Reilly, R. (2005). *The influence of motivation and comfort-level on learning to program.* Paper presented at the 17th Annual Workshop of the Psychology of Programming Interest Group, University of Sussex, Brighton, United Kingdom.

Bohlen, G. A., & Ferratt, T. W. (1993). The effect of learning style and method of instruction on the achievement, efficiency and satisfaction of end-users learning computer software. *Proceedings of the 1993 Conference on Computer Personnel Research* (pp. 273-283), St Louis, Missouri, United States.

Calloni, B. A., & Bagert, D. J. (1994). Iconic programming in BACCII vs. textual programming: which is a better learning environment? *SIGCSE Bull., 26*(1), 188-192.

Calloni, B. A., Bagert, D. J., & Haiduk, H. P. (1997). Iconic programming proves effective for teaching the first year programming sequence. *Proceedings of the Twenty-eighth SIGCSE Technical Symposium on Computer Science Education* (pp. 262-266), San Jose, California, United States.

Chainini, D. S., & Yamada, E. M. (1998). United States Patent No. 5760788.

Cilliers, C., Calitz, A., & Greyling, J. (2005). The effect of integrating an iconic programming notation into CS1. *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education* (pp. 108-112), Caparica, Portugal.

Clear, T., Edwards, J., Lister, R., Simon, B., Thompson, E., & Whalley, J. (2008). The teaching of novice computer programmers: bringing the scholarly-research approach to Australia. *Proceedings of the Tenth Conference on Australasian Computing Education - Volume 78* (pp. 63-68), Wollongong, NSW, Australia.

Clough, P., & Nutbrown, C. (2002). *A student's guide to methodology*: SAGE Publications, London, United Kingdom.

Cohen, L., Mannion, L., & Morrison, K. (2000). *Research methods in education* (5th ed.). London: RoutledgeFlamer.

Dale, N. (2005). Content and emphasis in CS1. *SIGCSE Bulletin, 37*(4), 69-73.

Dann, W., Cooper, S., & Pausch, R. (2005). *Learning to program with Alice*. New York: Prentice Hall.

Dawson, D. C. (2006). *A practical guide to research methods: A user-friendly manual for mastering research techniques and projects* (2nd ed.). Oxford, United Kingdom: How to Books Ltd.

Egert, C., Bierre, K., Phelps, A., & Ventura, P. (2006). *Hello, M.U.P.P.E.T.S.: using a 3D collaborative virtual environment to motivate fundamental object-oriented learning*. Paper presented at the Companion to the 21st ACM SIGPLAN Symposium on Object-oriented Programming Systems, Languages, and Applications, Portland, Oregon, USA.

Galpin, V. C., Sanders, I. D., & Chen, P.-y. (2007). Learning styles and personality types of computer science students at a South African university. *Proceedings of the 12th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education* (pp. 201-205), Dundee, Scotland.

Giordano, J. C., & Carlisle, M. (2006). Toward a more effective visualization tool to teach novice programmers. *Proceedings of the 7th Conference on Information Technology Education* (pp. 115-122), Minneapolis, Minnesota, USA.

Goldweber, M., Bergin, J., Lister, R., & McNally, M. (2006). *A comparison of different approaches to the introductory programming course*. Paper presented at the Australasian Computing Education Conference, Hobart, Tasmania, Australia.

Green, T., & Petre, M. (1996). Usability analysis of visual programming environments: a 'cognitive dimensions' framework. *Journal of Visual Languages and Computing*.

Gries, D. (1974). What should we teach in an introductory programming course? *SIGCSE Bulletin, 6*(1), 81-89.

Gries, D. (2006). What have we not learned about teaching programming? *Computer, 39*(10), 81-82.

Haden, P. (2006). *The incredible rainbow spitting chicken: teaching traditional programming skills through games programming*. Paper presented at the Australasian Computing Education.

Halland, K., & Malan, K. (2003). Reflections by teachers learning to program. *Proceedings of the 2003 Annual Research Conference of the South African*

Institute of Computer Scientists and Information Technologists on Enablement Through Technology (pp. 165-172), Port Elizabeth, South Africa.

Honchell, J. W., & Robertson, T. L. (1996). *Is the role of applied programming languages changing?* Paper presented at the 26th Annual Conference on Frontiers in Education, 1996. FIE '96, Utah, United States.

Howard, A. (2002). Rapid application development: rough and dirty or value-for-money engineering? *Communications of ACM, 45*(10), 27-29.

Ichikawa, T., & Hirakawa, M. (1990). Iconic programming: where to go? *Software, IEEE, 7*(6), 63-68.

Jenkins, T. (2001). The motivation of students of programming. *Proceedings of the 6th Annual Conference on Innovation and Technology in Computer Science Education* (pp. 53-56), Canterbury, United Kingdom.

Jenkins, T. (2002). *On the difficulty of learning to program.* Paper presented at the 3rd Annual Conference of the LTSN Centre for Information & Computer Science.

Kaneshige, T. (2009). A future without programming. *Information Age, April/May 2009,* 42-44.

Kinnunen, P., & Malmi, L. (2008). CS minors in a CS1 course. *Proceedings of the Fourth International Workshop on Computing Education Research* (pp. 79-90), Sydney, Australia.

Kolling, M., & Rosenberg, J. (1996). An object-oriented program development environment for the first programming course. *Proceedings of the Twenty-seventh SIGCSE Technical Symposium on Computer Science Education* (pp. 83-87), Philadelphia, Pennsylvania, United States.

Kyrnin, J. (n.d). WYSIWYG vs. hand coding, the great debate. *Journal.* Retrieved March 5, 2009, from About.com: http://webdesign.about.com/cs/htmleditors/a/aa021400a.htm

Lahtinen, E., AlaMutka, K., & HannuMatti, J. (2005). A study of the difficulties of novice programmers. *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education* (pp.14-18), Ballarat, Victoria, Australia.

Lister, R., Adams, E. S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M. (2004). *A multi-national study of reading and tracing skills in novice programmers.* Paper presented at the Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education, Leeds, United Kingdom.

Mamone, S. (1992). Empirical study of motivation in a entry level programming course. ACM *SIGPLAN Noices, 27*(3), 54-60.

Mannila, L. (2006). Progress reports and novices' understanding of program code. *Proceedings of the 6th Baltic Sea Conference on Computing Education Research: Koli Calling 2006* (pp. 27-31), Uppsala, Sweden.

Mannila, L., & Raadt, M. d. (2006). An objective comparison of languages for teaching introductory programming. *Proceedings of the 6th Baltic Sea Conference on Computing Education Research: Koli Calling 2006* (pp. 32-37), Uppsala, Sweden.

Martin, J. (1991). *Rapid application development.* New York: Macmillan Publishing Company.

McIver, L., & Conway, D. (1996). *Seven deadly sins of introductory programming language design.* Paper presented at the International Conference on Software Engineering: Education and Practice (SE:EP '96), Victoria, Australia.

McNeill, P., & Chapman, S. (2005). *Research methods* (3rd ed.). Abingdon, Oxfordshire; New York: Routledge.

Milne, I., & Rowe, G. (2002). Difficulties in learning and teaching programming: Views of students and tutors. *Education and Information Technologies, 2*(Volume 7, Number 1/March, 2002), 55-66.

Newman, I., & Benz, C. R. (1998). *Qualitative-quantitative research methodology: Exploring the interactive continuum*: Southern Illinois University Press, Carbondale, Illinois, United States.

Or-Bach, R., & Lavy, I. (2004). Cognitive activities of abstraction in object orientation: an empirical study. *SIGCSE Bulletin, 36*(2), 82-86.

Papaeconomou, C., Zijlema, A. F., & Ingwersen, P. (2008). Searchers' relevance judgments and criteria in evaluating web pages in a learning style perspective. *Proceedings of the Second International Symposium on Information Interaction in Context* (pp. 123-132), London, United Kingdom.

Parvez, S. M., & Blank, G. D. (2007). A pedagogical framework to integrate learning style into intelligent tutoring systems. *Journal of Computing in Small Colleges, 22*(3), 183-189.

Patton, M. Q. (1990). *Qualitative evaluation and research methods* (2nd ed.). Newbury Park, CA: Sage Publications.

Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J. (2007). *A survey of literature on the teaching of introductory programming.* Paper

presented at the Working Group Reports on ITiCSE on Innovation and Technology in Computer Science Education, Dundee, Scotland.

Peter, K. (2009). The value of visual design in software development. *Interactions, 16*(1), 66-68.

Pham, B. (1996). The changing curriculum of computing and information technology in Australia. *Proceedings of the 2nd Australasian Conference on Computer Science Education* (pp. 149-154), Melbourne, Australia

Raadt, M. D. (2008). *Teaching programming strategies explicitly to novice programmers.* Unpublished doctoral dissertation, University of Southern Queensland, Queensland, Australia.

Raadt, M. D., Watson, R., & Toleman, M. (2002). Language trends in introductory programming courses. *Proceedings of the Informing Science + IT Education Conference* (pp. 329-337), Cork, Ireland.

Raadt, M. D., Watson, R., & Toleman, M. (2003). Language tug-of-war: industry demand and academic choice. *Proceedings of the fifth Australasian conference on Computing education - Volume 20* (pp. 137-142), Adelaide, Australia.

Raadt, M. D., Watson, R., & Toleman, M. (2004). Introductory programming: what's happening today and will there be any students to teach tomorrow? *Proceedings of the sixth conference on Australasian computing education - Volume 30* (pp. 277-282), Dunedin, New Zealand.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education, 13*(2), 137 - 172.

Rode, J. (2004). *Nonprogrammer web application development.* Paper presented at the CHI '04 Extended Abstracts on Human Factors in Computing Systems, Vienna, Austria.

Schaub, S. (2009). Teaching CS1 with web applications and test-driven development. *SIGCSE Bulletin., 41*(2), 113-117.

Schulte, C., & Bennedsen, J. (2006). What do teachers teach in introductory programming? *Proceedings of the Second International Workshop on Computing Education Research* (pp. 17-28), Canterbury, United Kingdom.

Seals, C. (2005). Visual programming for novice programmer teachers. *Proceedings of the 2005 Conference on Diversity in Computing* (pp. 26-27), Albuquerque, New Mexico, USA.

Sommerville, I. (2007). *Software Engineering* (8th ed.). England: Pearson Education Limited.

Sykes, E. R. (2007). Determining the effectiveness of the 3D Alice programming environment at the computer science I level. *Journal of Educational Computing Research 36*(2), 223 - 244.

Teague, D., & Roe, P. (2008). *Collaborative learning – towards a solution for novice programmers*. Paper presented at the Tenth Australasian Computing Education Conference (ACE2008), Wollongong, Australia.

Tew, A. E., McCracken, W. M., & Guzdial, M. (2005). Impact of alternative introductory courses on programming concept understanding. *Proceedings of the First International Workshop on Computing Education Research* (pp. 25-35), Seattle, Washington, USA.

Vogts, D., Calitz, A., & Greyling, J. (2008). Comparison of the effects of professional and pedagogical program development environments on novice programmers. *Proceedings of the 2008 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on IT Research in Developing Countries: Riding the Wave of Technology* (pp. 286-295), Wilderness, South Africa.

Walter, S. E., Forssell, K., Barron, B., & Martin, C. (2007). *Continuing motivation for game design*. Paper presented at the CHI '07 Extended Abstracts on Human Factors in Computing Systems, San Jose, California, USA.

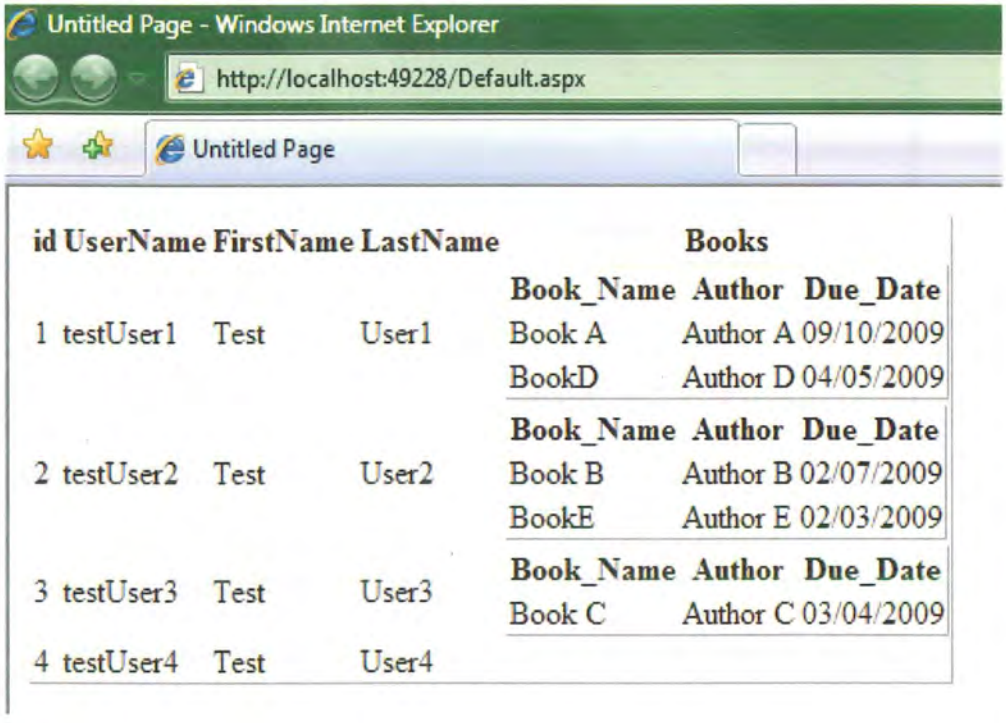
Weir, G. R. S., Vilner, T., Jos, A., & Nordstr, M. M. (2005). Difficulties teaching Java in CS1 and how we aim to solve them. *Proceedings of the 10th annual SIGCSE Conference on Innovation and Technology in Computer Science Education* (pp. 344-345), Monte de Caparica, Portugal.

Wexelblat, R. L. (Ed.). (1981). *History of programming languages*. Pennsylvania: Academic Press.

Wong, W. (2006). Graphical and text-based programming: complementary, not competitive. *Journal*. Retrieved February 26, 2009, from Electronic Design: http://electronicdesign.com/Articles/Index.cfm?AD=1&ArticleID=13241

Appendix A: Programming Example Using RAD

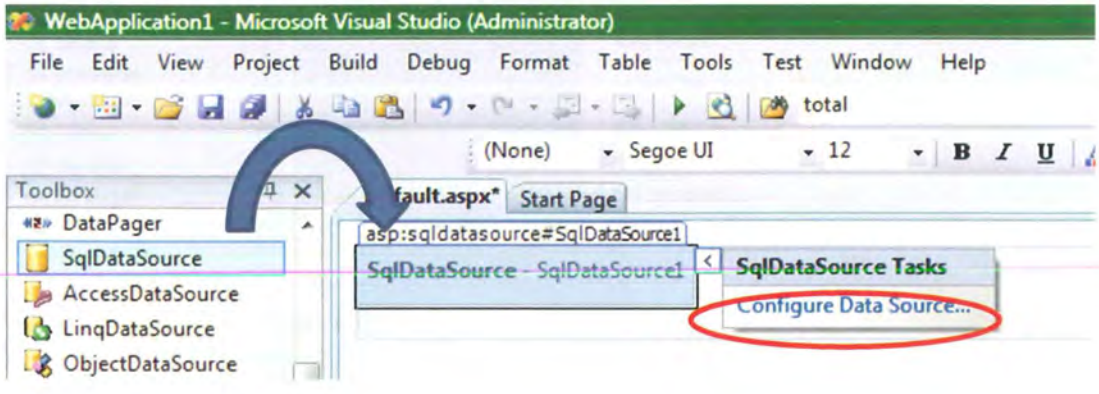
Displaying users and books from database



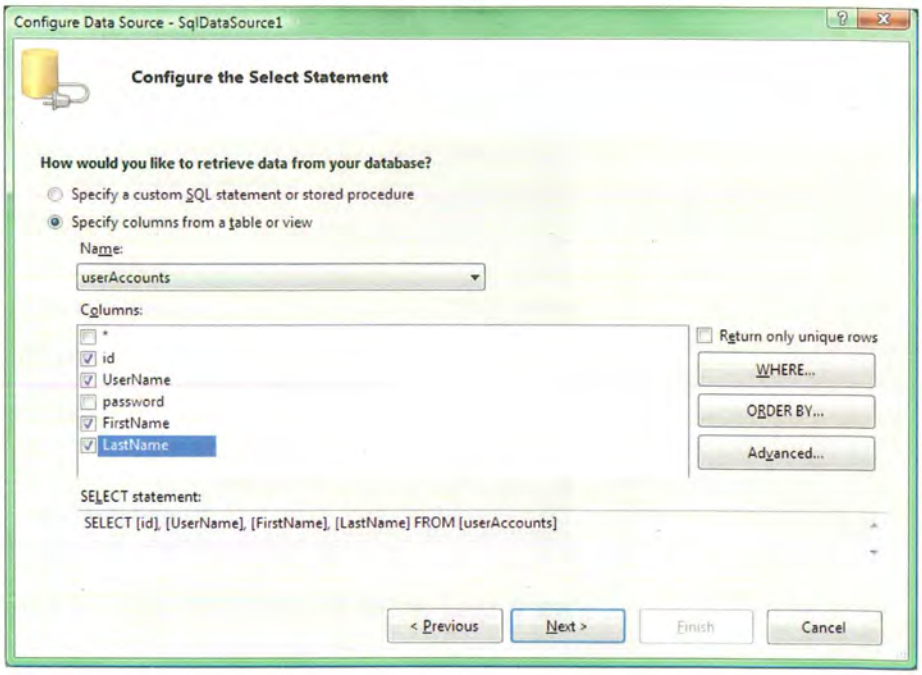
id	UserName	FirstName	LastName	Books		
				Book_Name	Author	Due_Date
1	testUser1	Test	User1	Book A	Author A	09/10/2009
				BookD	Author D	04/05/2009
				Book_Name	Author	Due_Date
2	testUser2	Test	User2	Book B	Author B	02/07/2009
				BookE	Author E	02/03/2009
				Book_Name	Author	Due_Date
3	testUser3	Test	User3	Book C	Author C	03/04/2009
4	testUser4	Test	User4			

Steps in RAD Tool (Visual Studio 2008)

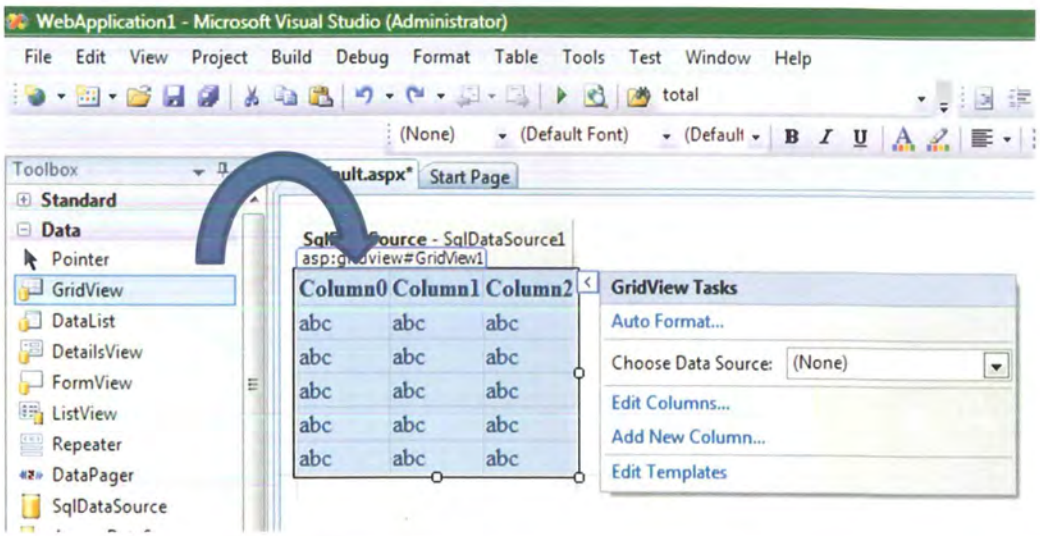
Step 1 – Drag and drop the SQL DataSource



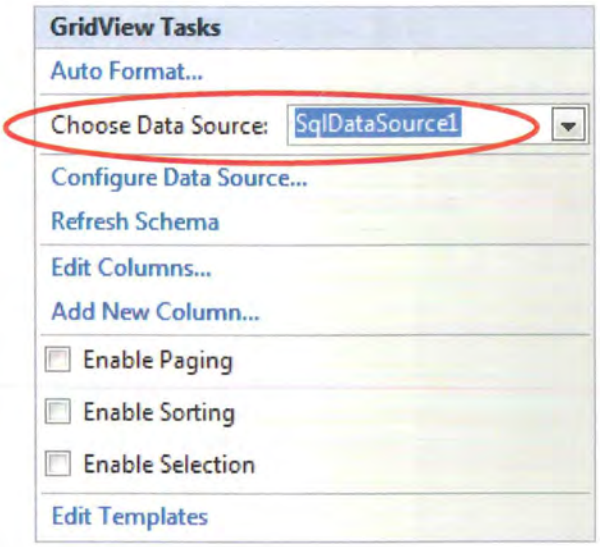
Step 2 – Configure the SQL DataSource



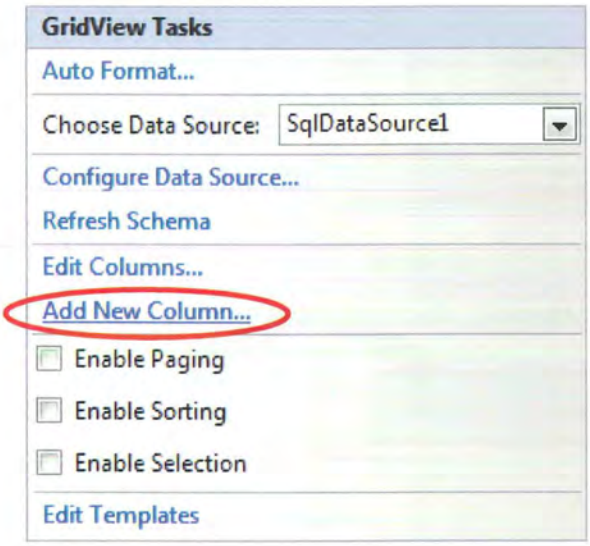
Step 3 – Drag and Drop the Grid View



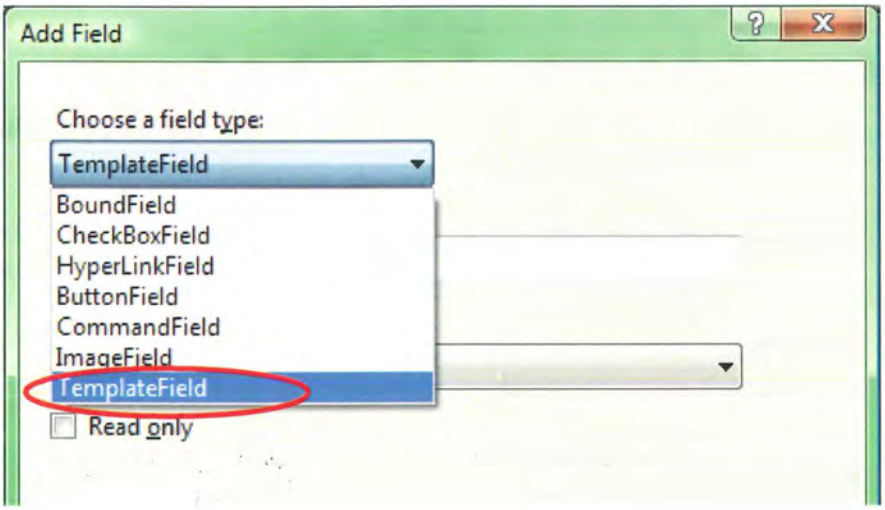
Step 4 – Link to the SQL DataSource



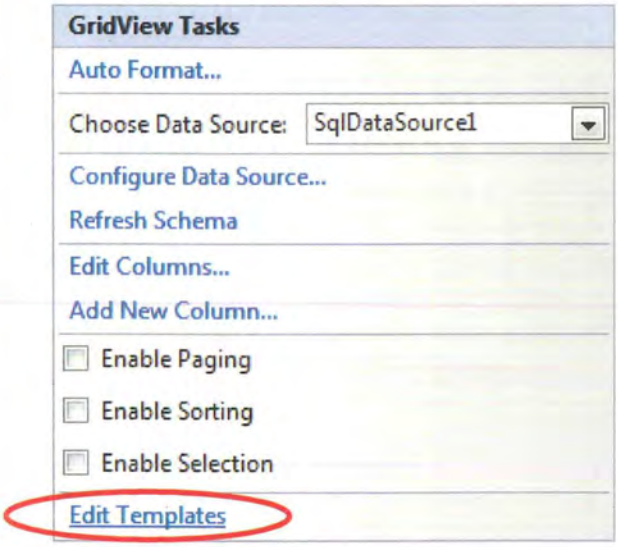
Step 5 – Add New Column in the Grid View



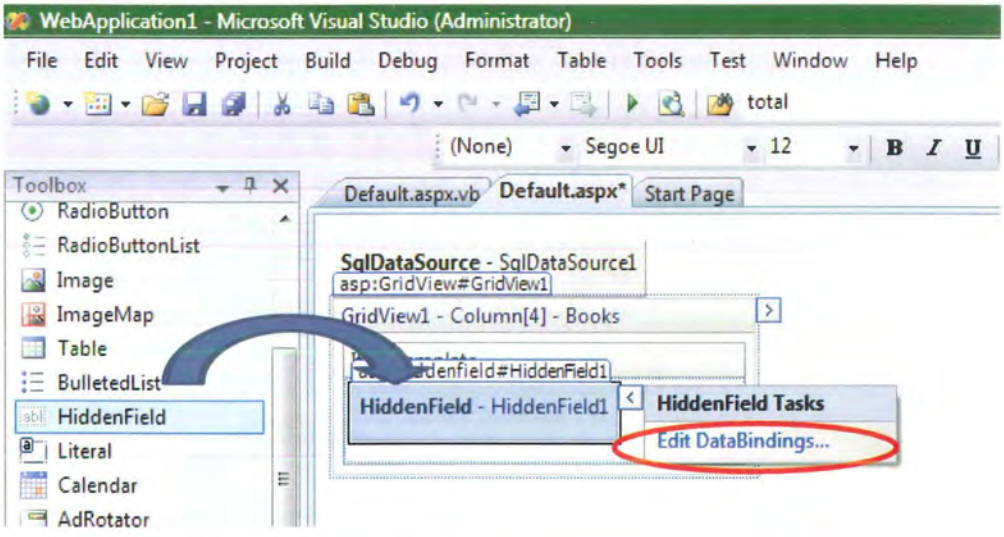
Step 6 – Configure the new field to be a template field



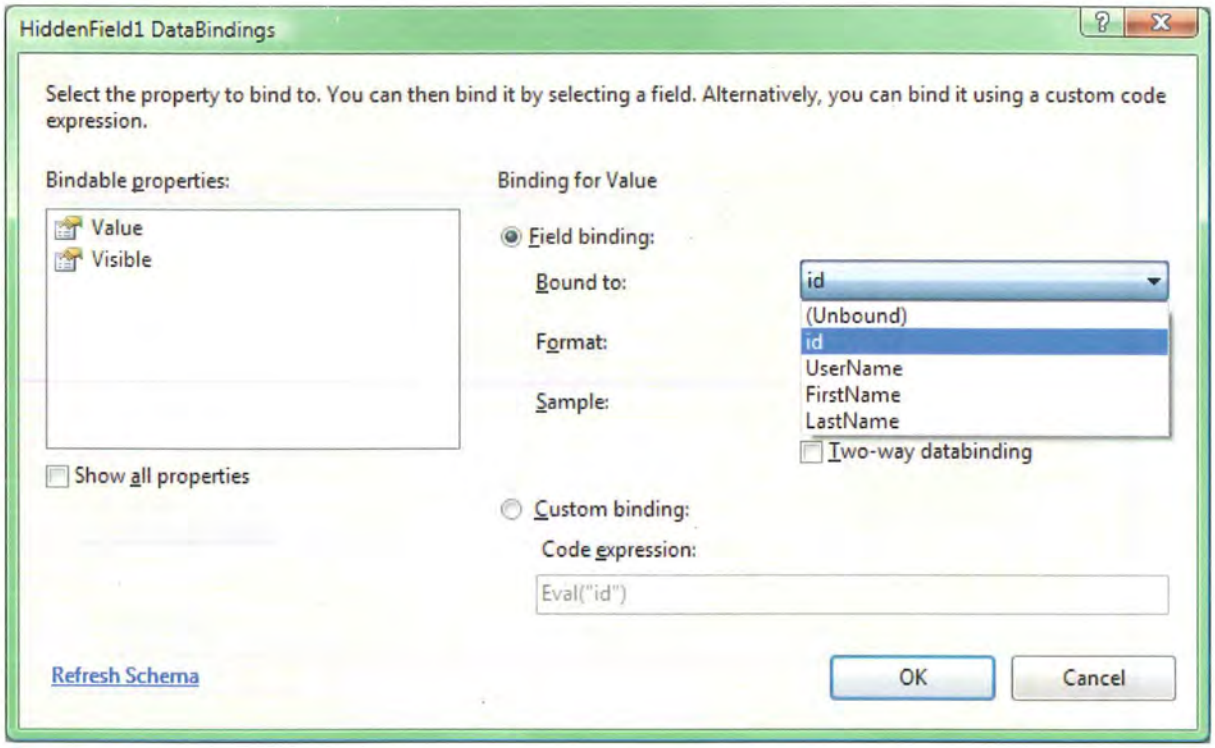
Step 7 – Edit the template field



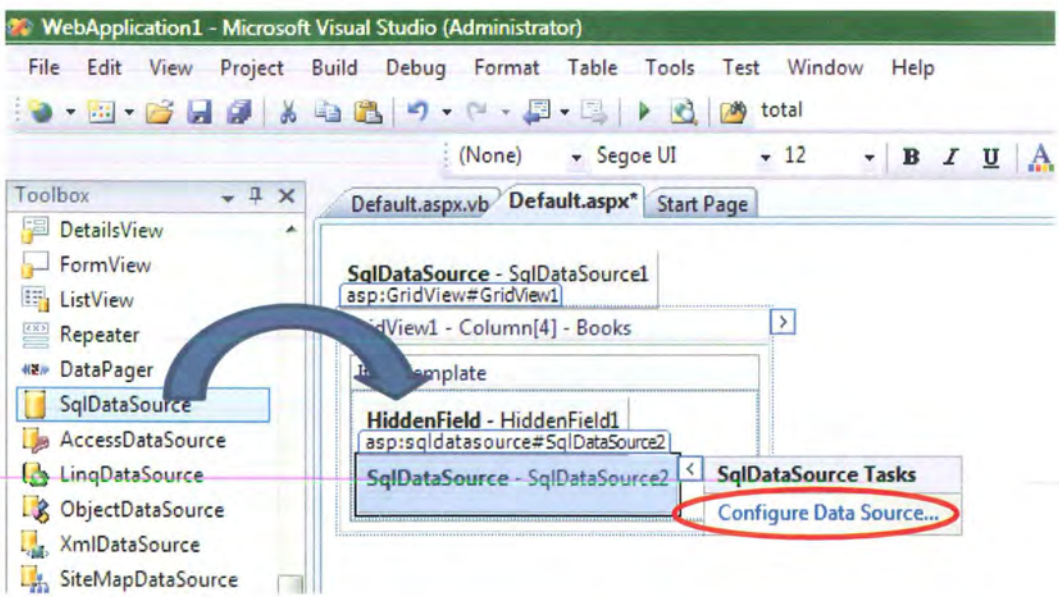
Step 8 – Drag and Drop a Hidden field into template field



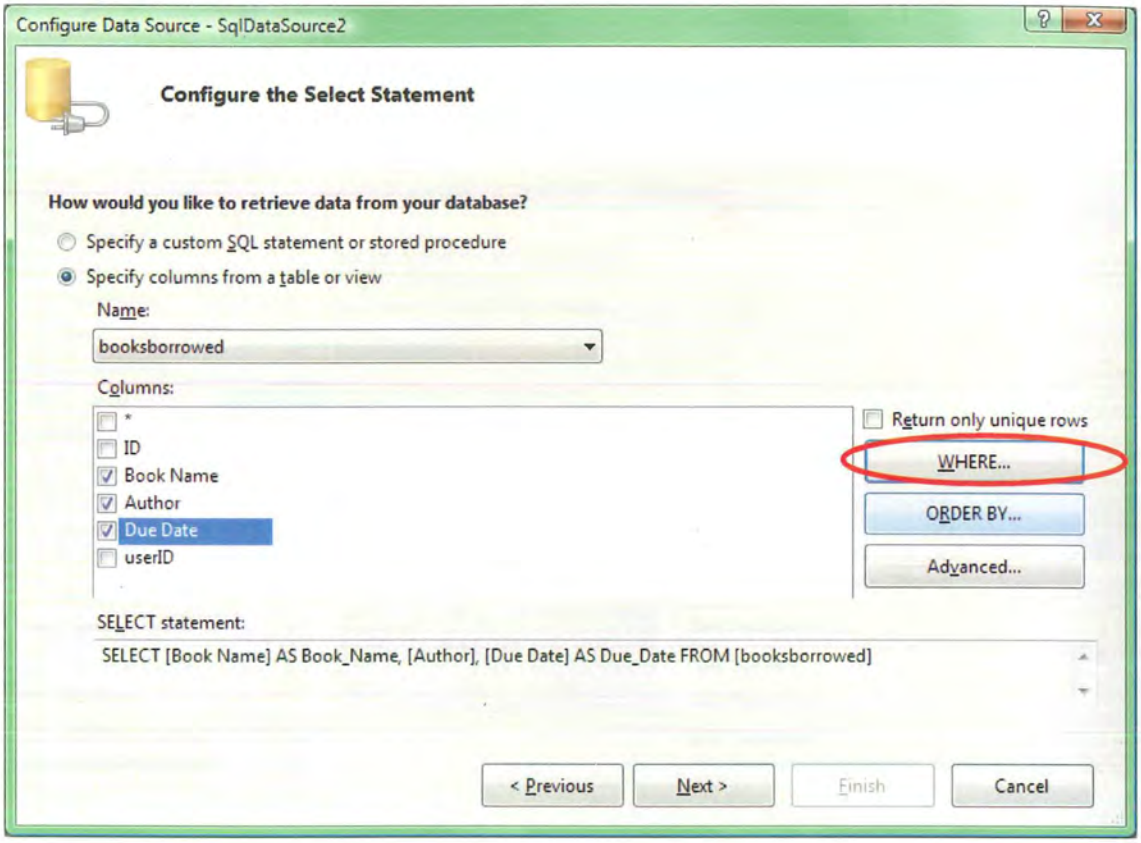
Step 9 – Bind the ID to hidden field



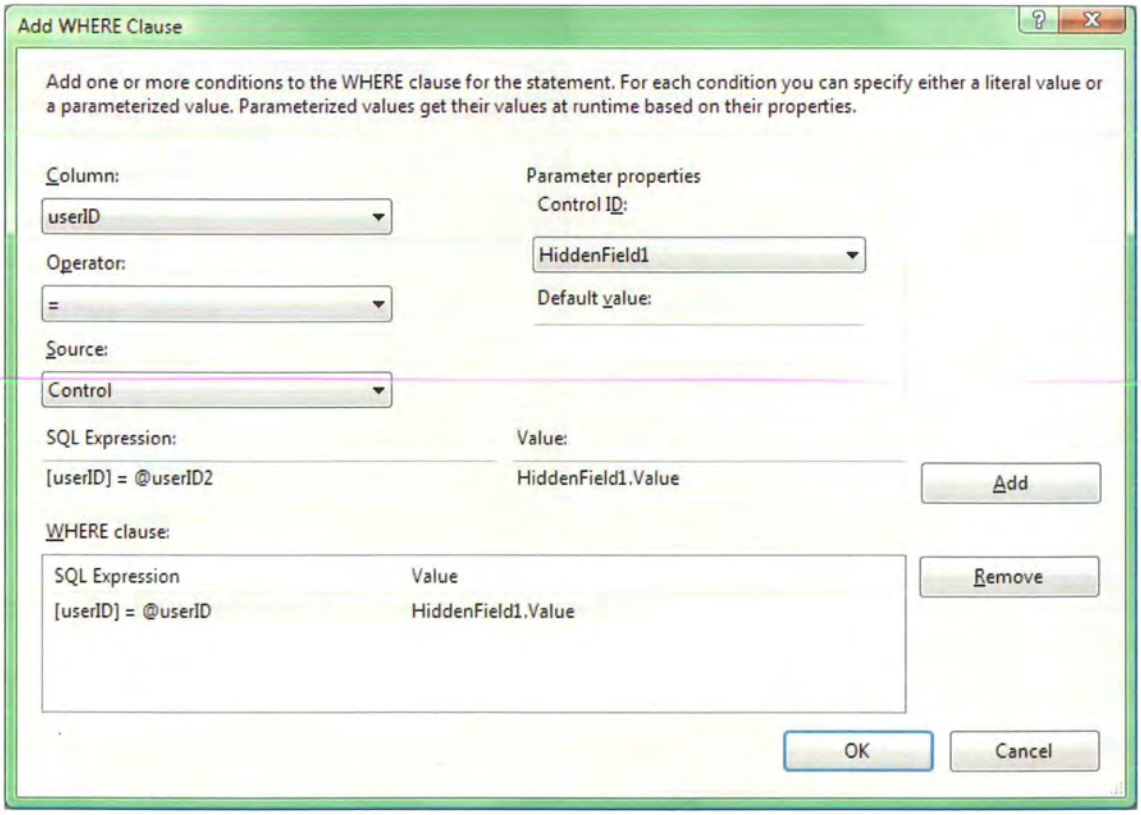
Step 10 – Drag and drop SQL DataSource into template field



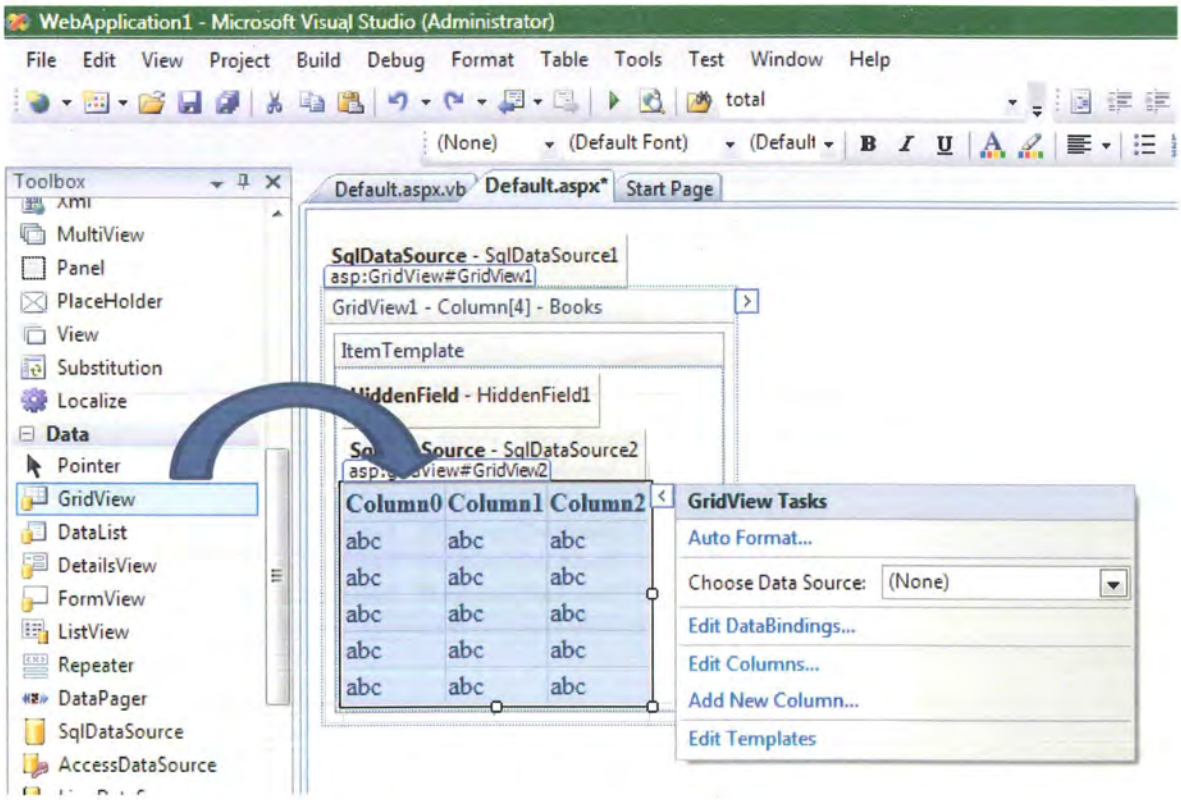
Step 11 – Configure the SQL DataSource



Step 12 – Set the userID as selection parameter and link it to HiddenField control

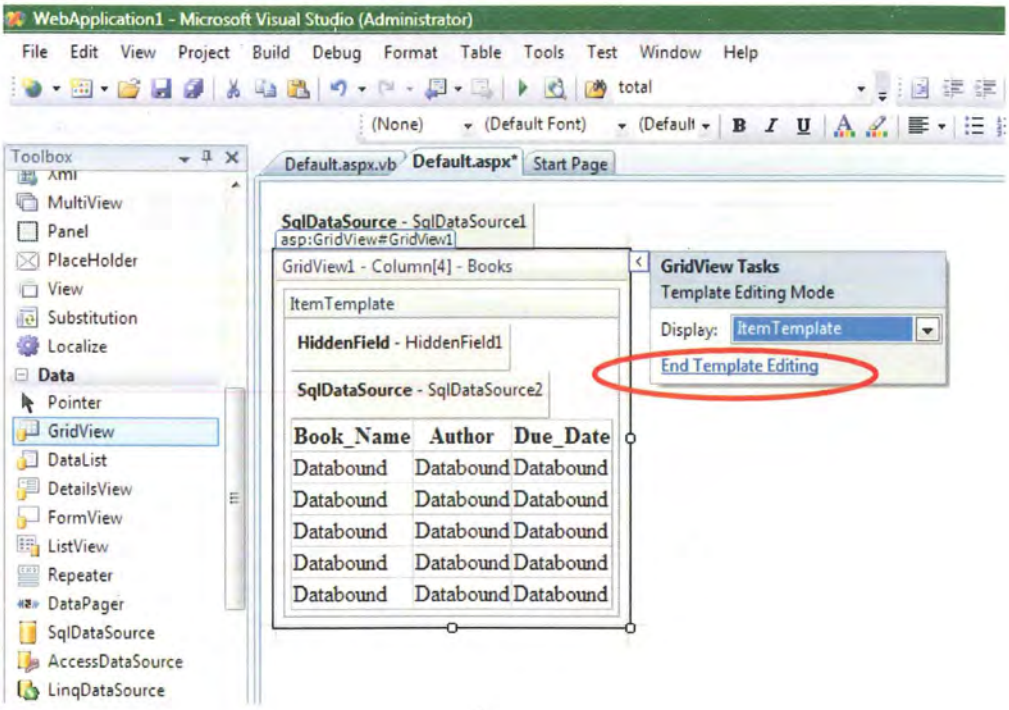


Step 13 – Drag and Drop the GridView to the template field



Step 14 – Link to the SQL DataSource

Step 15 – End Template Editing on GridView1


WebApplication1 - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Format Table Tools Test Window Help
total
(None) (Default Font) (Default
Toolbox
MultiView
Panel
PlaceHolder
View
Substitution
Localize
Data
Pointer
GridView
DataList
DetailsView
FormView
ListView
Repeater
DataPager
SqlDataSource
AccessDataSource
LinqDataSource
Default.aspx.vb
Default.aspx*
Start Page
SqlDataSource - SqlDataSource1
asp:GridView#GridView1
GridView1 - Column[4] - Books
ItemTemplate
HiddenField - HiddenField1
SqlDataSource - SqlDataSource2
Book_Name Author Due_Date
Databound Databound Databound
Databound Databound Databound
Databound Databound Databound
Databound Databound Databound
Databound Databound Databound
GridView Tasks
Template Editing Mode
Display: ItemTemplate
End Template Editing

Appendix B: Pre-exercise Questionnaire

An Investigation into Student Reaction to RAD vs Traditional Programming Environments for Novice Developers **Session ID - 15667d**

Informed Consent >> Pre-Exercise Questionnaire >> Setup and Configuration >> Traditional Exercises >> RAD Exercises >> Challenge Exercise >> Post-Exercise Survey >> End of Workshop

DEMOGRAPHICS

The following questions concern the demographic data.

1) What is your gender?
- ◎ Male
- ◎ Female

2) What of the following age groups do you fall into?
- ◎ < 18 years
- ◎ 18 – 21 years
- ◎ 22 – 25 years
- ◎ 26 – 29 years
- ◎ 30+ years

3) What is the current course that you are enrolled in?

4) Please specify your major field of study, if any.

5) How many units have you completed?
- ◎ Some or all of the first year units
- ◎ First and second year units
- ◎ First, second and some of the third year units

6) My main mode of study is;
- ◎ On-Campus
- ◎ Online
- ◎ Both

7) My preferred mode of study is;
- ◎ On-Campus
- ◎ Online
- ◎ Both

Next

PROGRAMMING EXPERIENCE

The following questions concern your previous experience with programming languages and environments.

8) Of the units you have completed so far, how many have had programming in them?

9) Have you used any programming languages before?

- ◎ Yes
- ◎ No

10) Have you done any web application development before?

- ◎ Yes
- ◎ No

11) Please specify the programming languages that you have used before (e.g. PHP, ASP, ASP.Net, Java, C, C++)

12) Please specify the tools that you have used to program (e.g. Text editor, Visual Studio, BlueJ, Eclipse)

13) How would you rate your experience as a programmer?

- ◎ Novice
- ◎ Intermediate
- ◎ Expert

Next

RAPID APPLICATION DEVELOPMENT VERSUS TRADITIONAL PROGRAMMING ENVIRONMENTS

Traditional programming environment refers to the environment used for programming in text-based format without any visual-aid. A typical example of traditional programming environment would be text-editor.

Rapid Application Development (RAD) environment refers to the drag-and-drop, visual, iconic programming environment which has 'pre-built components' or 'features' to help with the application development. Microsoft's Visual Studio is an example of RAD environment.

The following questions concern your thoughts and experiences with working with visual RAD environments versus traditional programming environments.

14) Have you ever programmed in a visual Rapid Application Development environment before (such as Microsoft's Visual Studio)?

- ○ Yes
- ○ No

Please describe:

15) I feel that visual RAD tools make programming easier.

Strongly Disagree	Disagree	Agree	Strongly Agree	Neutral
○	○	○	○	○

Please describe:

16) I feel that visual RAD features and functions can be hard to understand.

Strongly Disagree	Disagree	Agree	Strongly Agree	Neutral
○	○	○	○	○

Please describe:

17) I feel that traditional programming environments help me understand the programming processes better (e.g. variable declaration, condition, loops, recursion).

Strongly Disagree	Disagree	Agree	Strongly Agree	Neutral
○	○	○	○	○

Please describe:

18) I feel that learning syntax in traditional programming is difficult.

Strongly Disagree	Disagree	Agree	Strongly Agree	Neutral
○	○	○	○	○

Please describe:

19) If I were asked to program a web application, I think I would prefer to use

- ○ Traditional programming environment
- ○ Visual RAD environment

Please explain:

20) I have learnt

- ○ Traditional programming environment **first**
- ○ Visual RAD environment **first**
- ○ Both approximately at the same time
- ○ Neither of them

21) Which environment would you prefer to learn first as a novice programmer?

- ○ Traditional programming environment
- ○ Visual RAD environment
- ○ No Preference

Please explain:

Next

Learning Experience

The following questions concern your learning experience in introductory programming courses.

22) When doing programming exercises, I prefer to

- ◎ follow step-by-step written instructions
- ◎ follow a lecturer's on-screen example
- ◎ work on a solution on my own
- ◎ use a textbook and online resources

23) I find programming of any kind difficult to learn.

Strongly Disagree	Disagree	Agree	Strongly Agree	Neutral
◎	◎	◎	◎	◎

Please describe:

24) I expect to be able to program in a number of different environments over the duration of my studies.

Strongly Disagree	Disagree	Agree	Strongly Agree	Neutral
◎	◎	◎	◎	◎

Please describe:

25) Where possible, I would always like to use the same environment for all programming tasks.

Strongly Disagree	Disagree	Agree	Strongly Agree	Neutral
◎	◎	◎	◎	◎

Please describe:

26) From my experience, the first environment learned is still the most important.

Strongly Disagree	Disagree	Agree	Strongly Agree	Neutral
◎	◎	◎	◎	◎

Please describe:

27) In my future career, I expect to

- ◎ do programming
- ◎ program when I have to
- ◎ program as career
- ◎ I am not sure

Next

Appendix C: Post-exercise Questionnaire

An Investigation into Student Reaction to RAD vs Traditional Programming Environments for Novice Developers **Session ID - dc8fcb**

Informed Consent >> Pre-Exercise Questionnaire >> Setup and Configuration >> RAD Exercises >> Traditional Exercises >> Challenge Exercise >> Post-Exercise Survey >> End of Workshop

SECTION 1

The following questions concern your thoughts on Visual RAD versus Traditional programming environments based on completing the programming exercises.

1) Is this the first time you have used a visual RAD environment (certainly for building a working application)?

- ◯ Yes
- ◯ No

2) Based on the exercises, I feel that programming in

- ◯ Visual RAD environment is quicker than Traditional environment
- ◯ Traditional is quicker than Visual RAD environment
- ◯ I found each was equally quick to use

Please describe:

3) Based on the exercises, I feel that programming in

- ◯ Visual RAD environment is easier than traditional environment
- ◯ Traditional is easier than visual RAD environment
- ◯ I found that both were about as easy to use as the other

Please describe:

4) I feel that I would be able to write loops, variables and condition statements if I had started with visual RAD development.

Strongly Disagree	Disagree	Agree	Strongly Agree	Neutral
◯	◯	◯	◯	◯

Please describe:

5) I feel that I have or would have a deeper understanding of being able to write loops, variables and condition statements if I had started with Traditional development.

Strongly Disagree	Disagree	Agree	Strongly Agree	Neutral
◯	◯	◯	◯	◯

Please describe:

6) In web application development, I feel confident as a novice developer to use

- ◯ Visual RAD environment rather than traditional environment
- ◯ Traditional environment rather than visual RAD environment
- ◯ Both equally

Please describe:

7) I feel that the first environment has a significant impact on learning programming.

Strongly Disagree	Disagree	Agree	Strongly Agree	Neutral
◯	◯	◯	◯	◯

Please describe:

8) Which programming environment do you think should be introduced first to novice programmers in web application development?

- ◯ Visual RAD environment
- ◯ Traditional environment
- ◯ Does not matter

Please describe:

9) I feel that I have enough technical experience to use a visual RAD environment for actual development as presented in this workshop.

Strongly Disagree	Disagree	Agree	Strongly Agree	Neutral
◯	◯	◯	◯	◯

Please describe:

10) I feel that I learn more about actual programming syntax and concepts using

- ○ Visual RAD environment
- ○ Traditional environment
- ○ Both Equally

Please describe:

11) Which aspects of RAD do you think would help novice developers in learning programming?

12) Which aspects of RAD do you think is not suitable for novice developers?

13) Which aspects of traditional programming environment do you think would help novice developers in learning programming?

14) Which aspects of traditional programming environment do you think is not suitable for novice developers?

15) Which aspects do you think are important in choosing the first environment for novice developers?

Next

An Investigation into Student Reaction to RAD vs Traditional Programming Environments for Novice Developers

Session ID - dc8fcb

Informed Consent >> Pre-Exercise Questionnaire >> Setup and Configuration >> RAD Exercises >> Traditional Exercises >> Challenge Exercise >> Post-Exercise Survey >> End of Workshop

SECTION 2

The following questions concern your thoughts on RAD versus traditional programming environments based on completing the programming exercises.

16) I feel that I would need more programming experience to use visual RAD environments effectively.

Strongly Disagree ○ Disagree ○ Agree ○ Strongly Agree ○ Neutral ○

Please describe:

17) I feel that I would be able to program successfully in a visual RAD environment without traditional programming knowledge

Strongly Disagree ○ Disagree ○ Agree ○ Strongly Agree ○ Neutral ○

Please describe:

18) Given the nature of visual development in RAD, I feel that previous programming experience is not necessary.

Strongly Disagree ○ Disagree ○ Agree ○ Strongly Agree ○ Neutral ○

Please describe:

19) As a novice programmer, I feel that it is sufficient to program using a visual RAD environment as long as I know what components to use and when.

Strongly Disagree ○ Disagree ○ Agree ○ Strongly Agree ○ Neutral ○

Please describe:

20) I feel that it is not important to fully understand the underlying code that makes the visual RAD components work.

Strongly Disagree | Disagree | Agree | Strongly Agree | Neutral

Please describe:

21) I feel that being able to build a workable program is the most important aspect of learning programming, regardless of the environment.

Strongly Disagree | Disagree | Agree | Strongly Agree | Neutral

Please describe:

22) I feel that learning programming syntax first is the most important aspect of becoming a programmer.

Strongly Disagree | Disagree | Agree | Strongly Agree | Neutral

Please describe:

23) Regardless of Traditional or visual RAD methods of web programming, I feel that being able to learn any new environment quickly is more important than which type of environment it is.

Strongly Disagree | Disagree | Agree | Strongly Agree | Neutral

Please describe:

24) Which environment do you feel is appropriate for novice developers for self-learning in web application development context?

- Visual RAD environment
- Traditional environment
- Equally as appropriate

Please describe:

25) Which environment do you feel is appropriate for novice programmers for classroom-based learning in web application development context?

- Visual RAD environment
- Traditional environment
- Equally as appropriate

Please describe:

26) Which key aspects do you feel are important in learning programming?

27) Which aspects of RAD do you feel require traditional programming knowledge?

Next

SECTION 3

The following questions concern your thoughts on RAD versus traditional programming environments based on completing the programming exercises.

28) Which environment do you prefer for 'Search' exercise?

- ◯ Visual RAD environment
- ◯ Traditional environment
- ◯ Both about the same

Please describe:

29) Which environment do you prefer for 'Edit/ Delete' exercise?

- ◯ Visual RAD environment
- ◯ Traditional environment
- ◯ Both about the same

Please describe:

30) Which environment do you prefer for 'Insert' exercise?

- ◯ Visual RAD environment
- ◯ Traditional environment
- ◯ Both about the same

Please describe:

31) Did you manage to complete the challenge exercise using visual RAD environment?

- ◯ Yes
- ◯ No

Please discuss any problem encountered:

32) Did you manage to complete the challenge exercise using traditional environment?

- ◯ Yes
- ◯ No

Please discuss any problem encountered:

33) Overall, based on these exercises, I prefer

- ◯ Visual RAD Environment
- ◯ Traditional Environment
- ◯ Both about the same

Please describe:

34) If I had to further develop these exercises (with extra functions), I would use

- ◯ Visual RAD Environment
- ◯ Traditional Environment

Please describe:

35) Which set of exercises do you feel is easier to understand?

- ◯ Visual RAD Environment
- ◯ Traditional Environment
- ◯ Both about same

Please describe:

36) I feel that the teaching and learning materials are more important than the type of programming environments.

Strongly Disagree | Disagree | Agree | Strongly Agree | Neutral

Please describe:

37) I feel that availability of useful resources (textbooks or websites) influenced my reaction to visual RAD versus traditional programming environments.

Strongly Disagree | Disagree | Agree | Strongly Agree | Neutral

Please describe:

38) Which environment did you feel had the most useful online (web based) resources (such as tutorials / code examples)?

- Visual RAD Environment
- Traditional Environment
- Both about same

Please describe:

39) I feel that setup and configuration issues (of the environment) could affect my reaction to RAD versus Traditional programming environments.

Strongly Disagree | Disagree | Agree | Strongly Agree | Neutral

Please describe:

40) Please provide any additional factors that has influenced your reaction to RAD versus Traditional programming environments

41) Any additional comments on RAD versus Traditional programming environments

An Investigation into Student Reaction to RAD vs Traditional Programming Environments for Novice Developers **Session ID - dc8fcb**

Informed Consent >> Pre-Exercise Questionnaire >> Setup and Configuration >> RAD Exercises >> Traditional Exercises >> Challenge Exercise >> Post-Exercise Survey >> End of Workshop

THANK YOU FOR YOUR PARTICIPATION.....

Please leave your email address below, if you would like to participate in a face-to-face interview to discuss further on the Programming environments and this workshop.

Finish

Appendix D: Interview Questions

1) Given your indicated level of experience, how much actual development have you done in each of these environments?

-IF LITTLE OR NONE: How did you find these two methods as a first try?

-IF LOTS OF EXPERIENCE: Were you formally taught one or both of these environments or did you learn them on your own?

Can you explain that further?

2) Regardless of your level of expertise, which of the two techniques would you prefer to use if you were asked to develop a genuine web application Can you explain that further?

3) Given the two techniques shown, which would you like to see in your 1st year programming units and why? Can you explain further?

4) Can you see any disadvantages to one technique being taught before the other? Please explain further.

5) Given the abstracted nature of Visual RAD tools, do you think a traditional coding background is actually necessary before going into Visual RAD? Please explain further.

6) Do you think course structures need to take into account which units teach traditional coding and which teach visual RAD and so that a logical sequence exists? Please explain further.

7) Do you think these issues only apply to computer science students or to anyone studying in the area of IT? Please explain.

8) Finally, at this time which environment do you prefer and why? Please explain.

Appendix E: User Logs

SessionID	EventID	EventTime
Participant #1	Informed Consent Next	8/7/2009 10:24:54 AM
Participant #1	Session Message Next	8/7/2009 10:25:48 AM
Participant #1	Demographics Next	8/7/2009 10:27:11 AM
Participant #1	Programming Experience Next	8/7/2009 10:31:05 AM
Participant #1	RAD vs Trad Next	8/7/2009 10:34:28 AM
Participant #1	Learning Experience Next	8/7/2009 10:38:07 AM
Participant #1	Setup and Configuration Next	8/7/2009 10:38:37 AM
Participant #1	Traditional Exercise 1 Next	8/7/2009 10:39:28 AM
Participant #1	Traditional Exercise 2 Next	8/7/2009 10:39:36 AM
Participant #1	Traditional Exercise 3 Next	8/7/2009 10:39:41 AM
Participant #1	Setup and Configuration Next	8/7/2009 10:41:00 AM
Participant #1	Traditional Exercise 1 Next	8/7/2009 11:01:58 AM
Participant #1	Traditional Exercise 2 Next	8/7/2009 11:11:31 AM
Participant #1	Traditional Exercise 3 Next	8/7/2009 11:16:57 AM
Participant #1	Traditional Exercise 3 Next	8/7/2009 11:17:44 AM
Participant #1	RAD Exercise 1 Next	8/7/2009 11:48:48 AM
Participant #1	RAD Exercise 2 Next	8/7/2009 11:48:56 AM
Participant #1	RAD Exercise 3 Next	8/7/2009 11:49:01 AM
Participant #1	Challenge Next	8/7/2009 11:49:04 AM
Participant #1	Exercise Upload Next	8/7/2009 11:49:28 AM
Participant #1	Exercise Upload Next	8/7/2009 11:50:56 AM
Participant #1	Post Survey Section 1 Next	8/7/2009 11:54:49 AM
Participant #1	Post Survey Section 2 Next	8/7/2009 11:57:16 AM
Participant #1	Post Survey Section 3 Next	8/7/2009 12:00:33 PM
Participant #1	Finish	8/7/2009 12:02:39 PM
Participant #2	Informed Consent Next	8/7/2009 10:25:06 AM
Participant #2	Session Message Next	8/7/2009 10:26:23 AM
Participant #2	Demographics Next	8/7/2009 10:28:47 AM
Participant #2	Programming Experience Next	8/7/2009 10:29:14 AM
Participant #2	RAD vs Trad Next	8/7/2009 10:32:31 AM
Participant #2	Learning Experience Next	8/7/2009 10:35:11 AM
Participant #2	Setup and Configuration Next	8/7/2009 10:36:07 AM

Participant #2	Traditional Exercise 1 Next	8/7/2009 10:36:39 AM
Participant #2	Traditional Exercise 2 Next	8/7/2009 10:36:52 AM
Participant #2	Traditional Exercise 3 Back	8/7/2009 11:03:42 AM
Participant #2	Traditional Exercise 2 Next	8/7/2009 11:14:10 AM
Participant #2	Traditional Exercise 3 Next	8/7/2009 11:28:05 AM
Participant #2	RAD Exercise 1 Next	8/7/2009 11:32:05 AM
Participant #2	RAD Exercise 2 Next	8/7/2009 11:32:52 AM
Participant #2	RAD Exercise 3 Next	8/7/2009 11:34:52 AM
Participant #2	Challenge Trad Example	8/7/2009 11:35:01 AM
Participant #2	Challenge RAD Example	8/7/2009 11:36:11 AM
Participant #2	Challenge Next	8/7/2009 11:37:21 AM
Participant #2	Exercise Upload Next	8/7/2009 11:38:54 AM
Participant #2	Post Survey Section 1 Next	8/7/2009 11:43:48 AM
Participant #2	Post Survey Section 2 Next	8/7/2009 11:47:16 AM
Participant #2	Post Survey Section 3 Next	8/7/2009 11:47:43 AM
Participant #2	Finish	8/7/2009 11:48:58 AM
Participant #3	Informed Consent Next	8/7/2009 10:23:32 AM
Participant #3	Session Message Next	8/7/2009 10:24:11 AM
Participant #3	Demographics Next	8/7/2009 10:25:33 AM
Participant #3	Programming Experience Next	8/7/2009 10:29:28 AM
Participant #3	RAD vs Trad Next	8/7/2009 10:35:18 AM
Participant #3	Learning Experience Next	8/7/2009 10:36:09 AM
Participant #3	Setup and Configuration Next	8/7/2009 10:37:26 AM
Participant #3	Traditional Exercise 1 Next	8/7/2009 10:46:57 AM
Participant #3	Traditional Exercise 2 Next	8/7/2009 10:51:19 AM
Participant #3	Traditional Exercise 3 Back	8/7/2009 10:51:23 AM
Participant #3	Traditional Exercise 2 Next	8/7/2009 10:51:28 AM
Participant #3	Traditional Exercise 3 Next	8/7/2009 11:01:40 AM
Participant #3	RAD Exercise 1 Next	8/7/2009 11:19:12 AM
Participant #3	RAD Exercise 2 Next	8/7/2009 11:31:11 AM
Participant #3	RAD Exercise 3 Next	8/7/2009 11:55:04 AM
Participant #3	Challenge Next	8/7/2009 11:55:29 AM
Participant #3	Exercise Upload Back	8/7/2009 11:55:34 AM
Participant #3	Challenge Next	8/7/2009 11:55:41 AM

Participant #3	Exercise Upload Next	8/7/2009 11:59:37 AM
Participant #3	Post Survey Section 1 Next	8/7/2009 12:07:43 PM
Participant #3	Post Survey Section 2 Next	8/7/2009 12:10:38 PM
Participant #3	Post Survey Section 3 Next	8/7/2009 12:12:42 PM
Participant #3	Finish	8/7/2009 12:12:57 PM
Participant #4	Informed Consent Next	8/7/2009 10:27:15 AM
Participant #4	Session Message Next	8/7/2009 10:27:34 AM
Participant #4	Demographics Next	8/7/2009 10:28:43 AM
Participant #4	Programming Experience Next	8/7/2009 10:29:41 AM
Participant #4	RAD vs Trad Next	8/7/2009 10:33:47 AM
Participant #4	Learning Experience Next	8/7/2009 10:35:04 AM
Participant #4	Setup and Configuration Next	8/7/2009 10:35:10 AM
Participant #4	RAD Exercise 1 Next	8/7/2009 11:14:17 AM
Participant #4	RAD Exercise 2 Next	8/7/2009 11:31:53 AM
Participant #4	RAD Exercise 3 Next	8/7/2009 11:32:38 AM
Participant #4	Traditional Exercise 1 Next	8/7/2009 11:32:51 AM
Participant #4	Traditional Exercise 2 Next	8/7/2009 11:32:58 AM
Participant #4	Traditional Exercise 3 Back	8/7/2009 11:33:11 AM
Participant #4	Traditional Exercise 2 Back	8/7/2009 11:33:13 AM
Participant #4	Traditional Exercise 1 Back	8/7/2009 11:33:16 AM
Participant #4	RAD Exercise 3 Next	8/7/2009 11:52:26 AM
Participant #4	Traditional Exercise 1 Next	8/7/2009 11:52:29 AM
Participant #4	Traditional Exercise 2 Next	8/7/2009 11:52:31 AM
Participant #4	Traditional Exercise 3 Next	8/7/2009 11:52:35 AM
Participant #4	Challenge Next	8/7/2009 11:52:37 AM
Participant #4	Exercise Upload Next	8/7/2009 11:52:46 AM
Participant #4	Post Survey Section 1 Next	8/7/2009 11:56:07 AM
Participant #4	Post Survey Section 2 Next	8/7/2009 11:59:44 AM
Participant #4	Post Survey Section 3 Next	8/7/2009 12:02:08 PM
Participant #4	Finish	8/7/2009 12:02:10 PM
Participant #5	Informed Consent Next	8/7/2009 12:01:56 PM
Participant #5	Session Message Next	8/7/2009 12:03:22 PM
Participant #5	Demographics Next	8/7/2009 12:04:47 PM
Participant #5	Programming Experience Next	8/7/2009 12:06:21 PM

Participant #5	RAD vs Trad Next	8/7/2009 12:08:29 PM
Participant #5	Learning Experience Next	8/7/2009 12:12:44 PM
Participant #5	Setup and Configuration Next	8/7/2009 12:12:52 PM
Participant #5	Traditional Exercise 1 Next	8/7/2009 12:42:41 PM
Participant #5	Traditional Exercise 2 Next	8/7/2009 1:07:34 PM
Participant #5	Traditional Exercise 3 Back	8/7/2009 1:13:12 PM
Participant #5	Traditional Exercise 2 Back	8/7/2009 1:13:16 PM
Participant #5	Traditional Exercise 1 Next	8/7/2009 1:14:53 PM
Participant #5	Traditional Exercise 2 Next	8/7/2009 1:16:41 PM
Participant #5	Traditional Exercise 3 Next	8/7/2009 1:21:30 PM
Participant #5	RAD Exercise 1 Next	8/7/2009 1:22:17 PM
Participant #5	RAD Exercise 2 Next	8/7/2009 1:22:20 PM
Participant #5	RAD Exercise 3 Next	8/7/2009 1:22:22 PM
Participant #5	Challenge Next	8/7/2009 1:22:54 PM
Participant #5	Exercise Upload Next	8/7/2009 1:23:25 PM
Participant #5	Post Survey Section 1 Next	8/7/2009 1:29:59 PM
Participant #5	Post Survey Section 2 Next	8/7/2009 1:33:25 PM
Participant #5	Post Survey Section 3 Next	8/7/2009 1:34:52 PM
Participant #5	Finish	8/7/2009 1:35:14 PM
Participant #6	Informed Consent Next	8/7/2009 12:02:32 PM
Participant #6	Session Message Next	8/7/2009 12:03:20 PM
Participant #6	Demographics Next	8/7/2009 12:04:42 PM
Participant #6	Programming Experience Next	8/7/2009 12:06:16 PM
Participant #6	RAD vs Trad Next	8/7/2009 12:11:09 PM
Participant #6	Learning Experience Next	8/7/2009 12:14:10 PM
Participant #6	Setup and Configuration Next	8/7/2009 12:14:27 PM
Participant #6	Traditional Exercise 1 Back	8/7/2009 12:29:01 PM
Participant #6	Setup and Configuration Next	8/7/2009 12:29:03 PM
Participant #6	Traditional Exercise 1 Next	8/7/2009 12:45:52 PM
Participant #6	Traditional Exercise 2 Next	8/7/2009 12:45:55 PM
Participant #6	Traditional Exercise 3 Back	8/7/2009 12:46:15 PM
Participant #6	Traditional Exercise 2 Back	8/7/2009 12:46:20 PM
Participant #6	Traditional Exercise 1 Next	8/7/2009 12:47:56 PM
Participant #6	Traditional Exercise 2 Next	8/7/2009 12:48:04 PM

Participant #6	Traditional Exercise 3 Next	8/7/2009 12:48:10 PM
Participant #6	RAD Exercise 1 Next	8/7/2009 12:49:45 PM
Participant #6	RAD Exercise 2 Next	8/7/2009 1:12:10 PM
Participant #6	RAD Exercise 3 Back	8/7/2009 1:12:23 PM
Participant #6	RAD Exercise 2 Back	8/7/2009 1:12:49 PM
Participant #6	RAD Exercise 1 Next	8/7/2009 1:13:11 PM
Participant #6	RAD Exercise 2 Next	8/7/2009 1:15:10 PM
Participant #6	RAD Exercise 3 Next	8/7/2009 1:27:08 PM
Participant #6	Challenge Next	8/7/2009 1:27:12 PM
Participant #6	Exercise Upload Next	8/7/2009 1:28:41 PM
Participant #6	Post Survey Section 1 Next	8/7/2009 1:29:09 PM
Participant #6	Post Survey Section 2 Next	8/7/2009 1:29:22 PM
Participant #6	Post Survey Section 3 Next	8/7/2009 1:29:39 PM
Participant #6	Finish	8/7/2009 1:29:43 PM
Participant #7	Informed Consent Next	8/7/2009 12:02:00 PM
Participant #7	Session Message Next	8/7/2009 12:02:46 PM
Participant #7	Demographics Next	8/7/2009 12:03:43 PM
Participant #7	Programming Experience Next	8/7/2009 12:04:44 PM
Participant #7	RAD vs Trad Next	8/7/2009 12:06:28 PM
Participant #7	Learning Experience Next	8/7/2009 12:07:45 PM
Participant #7	Setup and Configuration Next	8/7/2009 12:10:46 PM
Participant #7	RAD Exercise 1 Next	8/7/2009 12:34:37 PM
Participant #7	RAD Exercise 2 Next	8/7/2009 12:34:48 PM
Participant #7	RAD Exercise 3 Next	8/7/2009 12:34:52 PM
Participant #7	Traditional Exercise 1 Next	8/7/2009 12:34:57 PM
Participant #7	Traditional Exercise 2 Next	8/7/2009 12:34:59 PM
Participant #7	Traditional Exercise 3 Next	8/7/2009 12:35:04 PM
Participant #7	Challenge Next	8/7/2009 12:35:12 PM
Participant #7	Exercise Upload Back	8/7/2009 12:35:23 PM
Participant #7	Challenge Back	8/7/2009 12:35:26 PM
Participant #7	Traditional Exercise 3 Back	8/7/2009 12:35:31 PM
Participant #7	Traditional Exercise 2 Back	8/7/2009 12:35:33 PM
Participant #7	Traditional Exercise 1 Back	8/7/2009 12:35:36 PM
Participant #7	RAD Exercise 3 Back	8/7/2009 12:35:38 PM

Participant #7	RAD Exercise 2 Next	8/7/2009 12:47:48 PM
Participant #7	RAD Exercise 3 Next	8/7/2009 12:47:51 PM
Participant #7	Traditional Exercise 1 Back	8/7/2009 12:47:59 PM
Participant #7	RAD Exercise 3 Back	8/7/2009 12:48:03 PM
Participant #7	RAD Exercise 2 Next	8/7/2009 12:48:46 PM
Participant #7	RAD Exercise 3 Next	8/7/2009 12:48:48 PM
Participant #7	Traditional Exercise 1 Next	8/7/2009 12:55:57 PM
Participant #7	Traditional Exercise 2 Next	8/7/2009 1:00:00 PM
Participant #7	Traditional Exercise 3 Next	8/7/2009 1:07:03 PM
Participant #7	Challenge Next	8/7/2009 1:07:11 PM
Participant #7	Exercise Upload Next	8/7/2009 1:08:08 PM
Participant #7	Post Survey Section 1 Next	8/7/2009 1:11:22 PM
Participant #7	Post Survey Section 2 Next	8/7/2009 1:13:15 PM
Participant #7	Post Survey Section 3 Next	8/7/2009 1:14:40 PM
Participant #7	Finish	8/7/2009 1:14:55 PM
Participant #8	Informed Consent Next	8/7/2009 1:19:08 PM
Participant #8	Session Message Next	8/7/2009 1:19:13 PM
Participant #8	Demographics Next	8/7/2009 1:19:31 PM
Participant #8	Programming Experience Next	8/7/2009 1:20:07 PM
Participant #8	RAD vs Trad Next	8/7/2009 1:20:41 PM
Participant #8	Learning Experience Next	8/7/2009 1:21:15 PM
Participant #8	Setup and Configuration Next	8/7/2009 1:21:19 PM
Participant #8	RAD Exercise 1 Next	8/7/2009 1:21:23 PM
Participant #8	RAD Exercise 2 Next	8/7/2009 1:21:29 PM
Participant #8	RAD Exercise 3 Next	8/7/2009 1:21:32 PM
Participant #8	Traditional Exercise 1 Next	8/7/2009 1:21:41 PM
Participant #8	Traditional Exercise 2 Next	8/7/2009 1:21:44 PM
Participant #8	Traditional Exercise 3 Next	8/7/2009 1:21:51 PM
Participant #8	Challenge Next	8/7/2009 1:22:03 PM
Participant #8	Exercise Upload Next	8/7/2009 1:22:22 PM
Participant #8	Post Survey Section 1 Next	8/7/2009 1:23:23 PM
Participant #8	Post Survey Section 2 Next	8/7/2009 1:23:39 PM
Participant #8	Post Survey Section 3 Next	8/7/2009 1:24:02 PM
Participant #8	Finish	8/7/2009 1:24:04 PM

Participant #9	Informed Consent Next	8/7/2009 3:45:40 PM
Participant #9	Session Message Next	8/7/2009 3:46:14 PM
Participant #9	Demographics Next	8/7/2009 3:47:15 PM
Participant #9	Programming Experience Next	8/7/2009 3:48:17 PM
Participant #9	RAD vs Trad Next	8/7/2009 3:54:54 PM
Participant #9	Learning Experience Next	8/7/2009 3:57:48 PM
Participant #9	Setup and Configuration Next	8/7/2009 4:00:57 PM
Participant #9	RAD Exercise 1 Next	8/7/2009 4:16:06 PM
Participant #9	RAD Exercise 2 Next	8/7/2009 4:31:41 PM
Participant #9	RAD Exercise 3 Back	8/7/2009 4:32:01 PM
Participant #9	RAD Exercise 2 Next	8/7/2009 4:35:55 PM
Participant #9	RAD Exercise 3 Next	8/7/2009 4:56:48 PM
Participant #9	Traditional Exercise 1 Next	8/7/2009 5:08:51 PM
Participant #9	Traditional Exercise 2 Next	8/7/2009 5:12:28 PM
Participant #9	Traditional Exercise 3 Next	8/7/2009 5:16:51 PM
Participant #9	Challenge Next	8/7/2009 5:22:22 PM
Participant #9	Exercise Upload Back	8/7/2009 5:22:30 PM
Participant #9	Challenge Next	8/7/2009 5:23:20 PM
Participant #9	Exercise Upload Next	8/7/2009 5:24:48 PM
Participant #9	Post Survey Section 1 Next	8/7/2009 5:33:57 PM
Participant #9	Post Survey Section 2 Next	8/7/2009 5:36:17 PM
Participant #9	Post Survey Section 3 Next	8/7/2009 5:37:51 PM
Participant #9	Finish	8/7/2009 5:38:00 PM
Participant #10	Informed Consent Next	8/7/2009 3:45:52 PM
Participant #10	Session Message Next	8/7/2009 3:46:19 PM
Participant #10	Demographics Next	8/7/2009 3:47:54 PM
Participant #10	Programming Experience Next	8/7/2009 3:50:31 PM
Participant #10	RAD vs Trad Next	8/7/2009 3:58:10 PM
Participant #10	Learning Experience Next	8/7/2009 4:05:34 PM
Participant #10	Setup and Configuration Next	8/7/2009 4:05:46 PM
Participant #10	Traditional Exercise 1 Next	8/7/2009 5:15:22 PM
Participant #10	Traditional Exercise 2 Next	8/7/2009 5:18:20 PM
Participant #10	Traditional Exercise 3 Next	8/7/2009 5:24:32 PM
Participant #10	RAD Exercise 1 Next	8/7/2009 5:24:38 PM

Participant #10	RAD Exercise 2 Next	8/7/2009 5:24:42 PM
Participant #10	RAD Exercise 3 Next	8/7/2009 5:24:46 PM
Participant #10	Challenge Next	8/7/2009 5:25:45 PM
Participant #10	Exercise Upload Next	8/7/2009 5:29:39 PM
Participant #10	RAD Exercise 1 Next	8/7/2009 5:53:51 PM
Participant #10	RAD Exercise 2 Next	8/7/2009 5:54:00 PM
Participant #10	RAD Exercise 3 Next	8/7/2009 5:54:02 PM
Participant #10	Challenge Next	8/7/2009 5:54:04 PM
Participant #10	Exercise Upload Next	8/7/2009 5:54:42 PM
Participant #10	Post Survey Section 1 Next	8/7/2009 5:55:24 PM
Participant #10	Post Survey Section 1 Next	8/7/2009 5:58:00 PM
Participant #10	Post Survey Section 2 Next	8/7/2009 6:00:06 PM
Participant #10	Post Survey Section 3 Next	8/7/2009 6:02:42 PM
Participant #10	Finish	8/7/2009 6:03:16 PM
Participant #11	Informed Consent Next	8/7/2009 4:16:10 PM
Participant #11	Session Message Next	8/7/2009 4:16:45 PM
Participant #11	Demographics Next	8/7/2009 4:17:10 PM
Participant #11	Programming Experience Next	8/7/2009 4:17:55 PM
Participant #11	RAD vs Trad Next	8/7/2009 4:24:33 PM
Participant #11	Learning Experience Next	8/7/2009 4:27:43 PM
Participant #11	Setup and Configuration Next	8/7/2009 4:28:25 PM
Participant #11	Traditional Exercise 1 Back	8/7/2009 4:41:04 PM
Participant #11	Setup and Configuration Next	8/7/2009 4:41:06 PM
Participant #11	Traditional Exercise 1 Next	8/7/2009 4:41:11 PM
Participant #11	Traditional Exercise 2 Next	8/7/2009 4:48:25 PM
Participant #11	Traditional Exercise 3 Next	8/7/2009 5:03:19 PM
Participant #11	RAD Exercise 1 Next	8/7/2009 5:12:05 PM
Participant #11	RAD Exercise 2 Next	8/7/2009 5:20:32 PM
Participant #11	RAD Exercise 1 Next	8/7/2009 5:25:01 PM
Participant #11	RAD Exercise 2 Next	8/7/2009 5:25:08 PM
Participant #11	RAD Exercise 3 Next	8/7/2009 5:42:01 PM
Participant #11	Challenge Next	8/7/2009 5:42:10 PM
Participant #11	Exercise Upload Next	8/7/2009 5:43:05 PM
Participant #11	Post Survey Section 1 Next	8/7/2009 5:51:14 PM

Participant #11	Post Survey Section 2 Next	8/7/2009 5:52:32 PM
Participant #11	Post Survey Section 3 Next	8/7/2009 5:53:40 PM
Participant #11	Finish	8/7/2009 5:53:48 PM
Participant #12	Informed Consent Next	8/7/2009 3:51:28 PM
Participant #12	Session Message Next	8/7/2009 3:52:23 PM
Participant #12	Demographics Next	8/7/2009 3:53:07 PM
Participant #12	Programming Experience Next	8/7/2009 3:54:27 PM
Participant #12	RAD vs Trad Next	8/7/2009 3:56:48 PM
Participant #12	Learning Experience Next	8/7/2009 3:57:44 PM
Participant #12	Setup and Configuration Next	8/7/2009 4:02:56 PM
Participant #12	RAD Exercise 1 Next	8/7/2009 4:22:14 PM
Participant #12	RAD Exercise 2 Next	8/7/2009 4:35:38 PM
Participant #12	RAD Exercise 3 Next	8/7/2009 4:57:29 PM
Participant #12	Traditional Exercise 1 Next	8/7/2009 5:09:17 PM
Participant #12	Traditional Exercise 2 Next	8/7/2009 5:13:44 PM
Participant #12	Traditional Exercise 3 Next	8/7/2009 5:25:10 PM
Participant #12	Challenge Next	8/7/2009 5:25:17 PM
Participant #12	Exercise Upload Next	8/7/2009 5:25:20 PM
Participant #12	Post Survey Section 1 Next	8/7/2009 5:31:36 PM
Participant #12	Post Survey Section 2 Next	8/7/2009 5:35:01 PM
Participant #12	Post Survey Section 3 Next	8/7/2009 5:42:31 PM
Participant #12	Finish	8/7/2009 5:42:39 PM
Participant #13	Informed Consent Next	8/14/2009 5:56:54 PM
Participant #13	Session Message Next	8/14/2009 5:57:09 PM
Participant #13	Demographics Next	8/14/2009 5:57:57 PM
Participant #13	Programming Experience Next	8/14/2009 5:59:00 PM
Participant #13	RAD vs Trad Next	8/14/2009 6:00:25 PM
Participant #13	Learning Experience Next	8/14/2009 6:01:30 PM
Participant #13	RAD Exercise 1 Next	8/14/2009 6:04:11 PM
Participant #13	RAD Exercise 2 Next	8/14/2009 6:04:17 PM
Participant #13	RAD Exercise 3 Next	8/14/2009 6:04:20 PM
Participant #13	Traditional Exercise 1 Next	8/14/2009 6:04:25 PM
Participant #13	Traditional Exercise 2 Next	8/14/2009 6:04:30 PM
Participant #13	Traditional Exercise 3 Next	8/14/2009 6:04:32 PM

Participant #13	RAD Exercise 1 Next	8/16/2009 11:43:34 AM
Participant #13	RAD Exercise 2 Next	8/16/2009 11:43:39 AM
Participant #13	RAD Exercise 3 Next	8/16/2009 11:43:43 AM
Participant #13	Traditional Exercise 1 Next	8/16/2009 11:43:59 AM
Participant #13	Traditional Exercise 2 Next	8/16/2009 11:44:02 AM
Participant #13	Traditional Exercise 3 Next	8/16/2009 11:44:04 AM
Participant #13	Challenge Next	8/16/2009 11:44:17 AM
Participant #13	Post Survey Section 1 Next	8/16/2009 11:46:50 AM
Participant #13	Post Survey Section 2 Next	8/16/2009 11:48:56 AM
Participant #13	Post Survey Section 3 Next	8/16/2009 11:51:31 AM
Participant #13	Finish	8/16/2009 11:51:45 AM
Participant #14	Informed Consent Next	8/14/2009 8:05:36 PM
Participant #14	Session Message Next	8/14/2009 8:05:46 PM
Participant #14	Demographics Next	8/14/2009 8:06:47 PM
Participant #14	Programming Experience Next	8/14/2009 8:07:34 PM
Participant #14	RAD vs Trad Next	8/14/2009 8:10:51 PM
Participant #14	Learning Experience Next	8/14/2009 8:11:39 PM
Participant #14	RAD Exercise 1 Next	8/14/2009 8:12:25 PM
Participant #14	RAD Exercise 2 Next	8/14/2009 8:12:28 PM
Participant #14	RAD Exercise 3 Next	8/14/2009 8:12:31 PM
Participant #14	Traditional Exercise 1 Next	8/14/2009 8:12:33 PM
Participant #14	Traditional Exercise 2 Next	8/14/2009 8:12:35 PM
Participant #14	Traditional Exercise 3 Next	8/14/2009 8:12:37 PM
Participant #14	Challenge Next	8/14/2009 8:12:40 PM
Participant #14	Post Survey Section 1 Next	8/14/2009 8:15:08 PM
Participant #14	Post Survey Section 2 Next	8/14/2009 8:17:49 PM
Participant #14	Post Survey Section 3 Next	8/14/2009 8:19:58 PM
Participant #14	Finish	8/14/2009 8:20:04 PM
Participant #15	Informed Consent Next	8/14/2009 9:44:43 PM
Participant #15	Session Message Next	8/14/2009 9:44:49 PM
Participant #15	Demographics Next	8/14/2009 9:45:30 PM
Participant #15	Programming Experience Next	8/14/2009 9:46:18 PM
Participant #15	RAD vs Trad Next	8/14/2009 9:47:32 PM
Participant #15	Learning Experience Next	8/14/2009 9:48:12 PM

Participant #15	RAD Exercise 1 Next	8/14/2009 9:49:57 PM
Participant #15	RAD Exercise 2 Next	8/14/2009 9:50:06 PM
Participant #15	RAD Exercise 3 Next	8/14/2009 9:50:10 PM
Participant #15	Traditional Exercise 1 Next	8/14/2009 9:50:14 PM
Participant #15	Traditional Exercise 2 Next	8/14/2009 9:50:16 PM
Participant #15	Traditional Exercise 3 Next	8/14/2009 9:50:17 PM
Participant #15	Challenge Next	8/14/2009 9:50:22 PM
Participant #15	Post Survey Section 1 Next	8/14/2009 9:51:33 PM
Participant #15	Post Survey Section 2 Next	8/14/2009 9:52:42 PM
Participant #15	Post Survey Section 3 Next	8/14/2009 9:53:36 PM
Participant #15	Finish	8/14/2009 9:53:42 PM
Participant #16	Informed Consent Next	8/15/2009 7:41:55 PM
Participant #16	Session Message Next	8/15/2009 7:42:04 PM
Participant #16	Demographics Next	8/15/2009 7:44:25 PM
Participant #16	Programming Experience Next	8/15/2009 7:46:20 PM
Participant #16	RAD vs Trad Next	8/15/2009 7:50:26 PM
Participant #16	Learning Experience Next	8/15/2009 7:52:09 PM
Participant #16	RAD Exercise 1 Next	8/15/2009 7:55:13 PM
Participant #16	RAD Exercise 2 Back	8/15/2009 7:56:16 PM
Participant #16	RAD Exercise 1 Next	8/15/2009 7:56:19 PM
Participant #16	RAD Exercise 2 Next	8/15/2009 7:56:38 PM
Participant #16	RAD Exercise 3 Next	8/15/2009 7:56:48 PM
Participant #16	Challenge Next	8/15/2009 7:58:05 PM
Participant #16	Post Survey Section 1 Next	8/15/2009 8:04:16 PM
Participant #16	Post Survey Section 2 Next	8/15/2009 8:14:38 PM
Participant #16	Post Survey Section 3 Next	8/15/2009 8:18:47 PM
Participant #16	Finish	8/15/2009 8:18:55 PM
Participant #17	Informed Consent Next	8/15/2009 9:14:43 PM
Participant #17	Session Message Next	8/15/2009 9:15:02 PM
Participant #17	Demographics Next	8/15/2009 9:19:46 PM
Participant #17	Programming Experience Next	8/15/2009 9:23:20 PM
Participant #17	RAD vs Trad Next	8/15/2009 9:37:59 PM
Participant #17	Learning Experience Next	8/15/2009 9:46:49 PM
Participant #17	RAD Exercise 1 Next	8/15/2009 9:58:07 PM

Participant #17	RAD Exercise 1 Next	8/15/2009 10:13:41 PM
Participant #17	RAD Exercise 1 Next	8/15/2009 10:24:57 PM
Participant #17	RAD Exercise 1 Next	8/16/2009 6:49:30 PM
Participant #17	RAD Exercise 2 Next	8/16/2009 6:52:37 PM
Participant #17	RAD Exercise 3 Next	8/16/2009 6:52:49 PM
Participant #17	Traditional Exercise 1 Next	8/16/2009 6:52:52 PM
Participant #17	Traditional Exercise 2 Next	8/16/2009 6:52:54 PM
Participant #17	Traditional Exercise 3 Next	8/16/2009 6:52:56 PM
Participant #17	Challenge Next	8/16/2009 6:52:58 PM
Participant #17	Challenge Back	8/16/2009 6:54:00 PM
Participant #17	Traditional Exercise 3 Back	8/16/2009 6:54:03 PM
Participant #17	Traditional Exercise 2 Back	8/16/2009 6:54:05 PM
Participant #17	Traditional Exercise 1 Back	8/16/2009 6:54:09 PM
Participant #17	RAD Exercise 3 Back	8/16/2009 6:54:14 PM
Participant #17	RAD Exercise 2 Next	8/16/2009 6:54:17 PM
Participant #17	RAD Exercise 2 Back	8/16/2009 8:37:48 AM
Participant #17	RAD Exercise 3 Back	8/17/2009 3:09:43 PM
Participant #17	RAD Exercise 2 Back	8/17/2009 3:09:45 PM
Participant #17	RAD Exercise 1 Next	8/17/2009 3:50:14 PM
Participant #17	RAD Exercise 2 Next	8/17/2009 3:50:16 PM
Participant #17	RAD Exercise 3 Back	8/17/2009 3:50:19 PM
Participant #17	RAD Exercise 2 Next	8/17/2009 3:50:20 PM
Participant #17	RAD Exercise 3 Next	8/17/2009 3:50:21 PM
Participant #17	Traditional Exercise 1 Back	8/17/2009 3:50:35 PM
Participant #17	RAD Exercise 3 Next	8/17/2009 3:50:38 PM
Participant #17	Traditional Exercise 1 Next	8/17/2009 3:50:39 PM
Participant #17	Traditional Exercise 2 Next	8/17/2009 3:50:49 PM
Participant #17	Traditional Exercise 3 Next	8/17/2009 3:52:17 PM
Participant #17	Challenge Next	8/17/2009 3:54:25 PM
Participant #17	Post Survey Section 1 Next	8/19/2009 12:46:22 PM
Participant #17	Post Survey Section 2 Next	8/19/2009 1:19:32 PM
Participant #17	Post Survey Section 3 Next	8/19/2009 1:44:11 PM
Participant #17	Finish	8/19/2009 1:45:35 PM
Participant #18	Informed Consent Next	8/16/2009 4:42:20 AM

Participant #18	Session Message Next	8/16/2009 4:43:30 AM
Participant #18	Demographics Next	8/16/2009 4:44:47 AM
Participant #18	Programming Experience Next	8/16/2009 4:47:54 AM
Participant #18	RAD vs Trad Next	8/16/2009 4:55:16 AM
Participant #18	Learning Experience Next	8/16/2009 5:02:10 AM
Participant #18	Traditional Exercise 1 Next	8/16/2009 6:19:31 PM
Participant #18	Traditional Exercise 2 Next	8/16/2009 6:29:35 PM
Participant #18	Traditional Exercise 3 Next	8/16/2009 6:49:26 PM
Participant #18	RAD Exercise 1 Next	8/16/2009 7:16:54 PM
Participant #18	RAD Exercise 2 Next	8/16/2009 7:16:56 PM
Participant #18	RAD Exercise 3 Back	8/16/2009 7:17:03 PM
Participant #18	RAD Exercise 2 Back	8/16/2009 7:17:05 PM
Participant #18	RAD Exercise 1 Back	8/16/2009 7:17:06 PM
Participant #18	Traditional Exercise 3 Next	8/16/2009 7:17:08 PM
Participant #18	RAD Exercise 1 Next	8/16/2009 7:17:10 PM
Participant #18	RAD Exercise 2 Next	8/16/2009 7:37:29 PM
Participant #18	RAD Exercise 3 Back	8/16/2009 7:37:33 PM
Participant #18	RAD Exercise 2 Next	8/16/2009 7:37:35 PM
Participant #18	RAD Exercise 3 Next	8/16/2009 7:39:20 PM
Participant #18	Challenge Next	8/16/2009 7:39:44 PM
Participant #18	Post Survey Section 1 Next	8/17/2009 4:52:47 AM
Participant #18	Post Survey Section 2 Next	8/17/2009 6:42:11 PM
Participant #19	Informed Consent Next	8/16/2009 11:30:51 AM
Participant #19	Session Message Next	8/16/2009 11:31:01 AM
Participant #19	Demographics Next	8/16/2009 11:32:00 AM
Participant #19	Programming Experience Next	8/16/2009 11:33:05 AM
Participant #19	RAD vs Trad Next	8/16/2009 11:37:48 AM
Participant #19	Learning Experience Next	8/16/2009 11:39:20 AM
Participant #19	RAD Exercise 1 Next	8/16/2009 11:48:32 AM
Participant #19	RAD Exercise 2 Next	8/16/2009 11:51:29 AM
Participant #19	RAD Exercise 3 Next	8/16/2009 11:51:40 AM
Participant #19	Traditional Exercise 1 Next	8/16/2009 11:52:05 AM
Participant #19	Traditional Exercise 2 Next	8/16/2009 11:52:10 AM
Participant #19	Traditional Exercise 3 Next	8/16/2009 11:52:36 AM

Participant #19	Challenge Next	8/16/2009 11:53:14 AM
Participant #19	Post Survey Section 1 Next	8/16/2009 11:55:35 AM
Participant #20	Informed Consent Next	8/16/2009 12:13:02 PM
Participant #20	Session Message Next	8/16/2009 12:13:11 PM
Participant #20	Demographics Next	8/16/2009 12:14:07 PM
Participant #20	Programming Experience Next	8/16/2009 12:14:31 PM
Participant #20	RAD vs Trad Next	8/16/2009 12:15:57 PM
Participant #20	Learning Experience Next	8/16/2009 12:16:48 PM
Participant #20	Traditional Exercise 1 Next	8/16/2009 12:17:43 PM
Participant #20	Traditional Exercise 2 Next	8/16/2009 12:17:47 PM
Participant #20	Traditional Exercise 3 Next	8/16/2009 12:17:56 PM
Participant #20	RAD Exercise 1 Next	8/16/2009 12:18:09 PM
Participant #20	RAD Exercise 2 Next	8/16/2009 12:18:26 PM
Participant #20	RAD Exercise 3 Next	8/16/2009 12:18:46 PM
Participant #20	Challenge Next	8/16/2009 12:19:00 PM
Participant #20	Post Survey Section 1 Next	8/16/2009 12:19:54 PM
Participant #20	Post Survey Section 2 Next	8/16/2009 12:20:30 PM
Participant #20	Post Survey Section 3 Next	8/16/2009 12:21:18 PM
Participant #20	Finish	8/16/2009 12:21:32 PM
Participant #21	Informed Consent Next	8/16/2009 4:10:47 PM
Participant #21	Session Message Next	8/16/2009 4:11:20 PM
Participant #21	Demographics Next	8/16/2009 4:12:04 PM
Participant #21	Programming Experience Next	8/16/2009 4:12:23 PM
Participant #21	RAD vs Trad Next	8/16/2009 4:14:13 PM
Participant #21	Learning Experience Next	8/16/2009 4:17:42 PM
Participant #21	Traditional Exercise 1 Next	8/16/2009 4:18:20 PM
Participant #21	Traditional Exercise 2 Next	8/16/2009 4:22:25 PM
Participant #21	Traditional Exercise 3 Next	8/16/2009 4:25:20 PM
Participant #21	RAD Exercise 1 Next	8/16/2009 4:25:35 PM
Participant #21	RAD Exercise 2 Next	8/16/2009 4:28:39 PM
Participant #21	RAD Exercise 3 Next	8/16/2009 4:31:41 PM
Participant #21	Challenge Next	8/16/2009 4:32:31 PM
Participant #21	Post Survey Section 1 Next	8/16/2009 4:35:40 PM
Participant #21	Post Survey Section 2 Next	8/16/2009 4:36:59 PM

Participant #21	Post Survey Section 3 Next	8/16/2009 4:38:16 PM
Participant #21	Finish	8/16/2009 4:38:51 PM
Participant #22	Informed Consent Next	8/16/2009 6:19:24 PM
Participant #22	Session Message Next	8/16/2009 6:19:30 PM
Participant #22	Demographics Next	8/16/2009 6:20:10 PM
Participant #22	Programming Experience Next	8/16/2009 6:21:38 PM
Participant #22	RAD vs Trad Next	8/16/2009 6:23:04 PM
Participant #22	Learning Experience Next	8/16/2009 6:23:45 PM
Participant #22	Traditional Exercise 1 Next	8/16/2009 6:23:59 PM
Participant #22	Traditional Exercise 2 Next	8/16/2009 6:24:02 PM
Participant #22	Traditional Exercise 3 Next	8/16/2009 6:24:05 PM
Participant #22	RAD Exercise 1 Next	8/16/2009 6:24:12 PM
Participant #22	RAD Exercise 2 Next	8/16/2009 6:24:13 PM
Participant #22	RAD Exercise 3 Next	8/16/2009 6:24:14 PM
Participant #22	Challenge Next	8/16/2009 6:24:17 PM
Participant #22	Post Survey Section 1 Next	8/16/2009 6:26:23 PM
Participant #22	Post Survey Section 2 Next	8/16/2009 6:28:48 PM
Participant #22	Post Survey Section 3 Next	8/16/2009 6:28:51 PM
Participant #22	Finish	8/16/2009 6:28:55 PM
Participant #23	Informed Consent Next	8/16/2009 11:43:15 PM
Participant #23	Session Message Next	8/16/2009 11:43:30 PM
Participant #23	Demographics Next	8/16/2009 11:44:05 PM
Participant #23	Programming Experience Next	8/16/2009 11:44:55 PM
Participant #23	RAD vs Trad Next	8/16/2009 11:47:48 PM
Participant #23	Learning Experience Next	8/16/2009 11:48:38 PM
Participant #23	RAD Exercise 1 Next	8/16/2009 11:48:40 PM
Participant #23	RAD Exercise 2 Back	8/16/2009 11:48:44 PM
Participant #23	RAD Exercise 1 Next	8/16/2009 11:49:15 PM
Participant #23	RAD Exercise 2 Next	8/16/2009 11:49:17 PM
Participant #23	RAD Exercise 3 Next	8/16/2009 11:49:21 PM
Participant #23	Traditional Exercise 1 Next	8/16/2009 11:49:23 PM
Participant #23	Traditional Exercise 2 Next	8/16/2009 11:49:26 PM
Participant #23	Traditional Exercise 3 Next	8/16/2009 11:49:28 PM
Participant #23	Challenge Next	8/16/2009 11:49:30 PM

Participant #23	Post Survey Section 1 Next	8/16/2009 11:50:37 PM
Participant #24	Informed Consent Next	8/17/2009 10:26:30 AM
Participant #24	Session Message Next	8/17/2009 10:26:52 AM
Participant #24	Demographics Next	8/17/2009 10:27:57 AM
Participant #24	Programming Experience Next	8/17/2009 10:29:24 AM
Participant #24	RAD vs Trad Next	8/17/2009 10:33:38 AM
Participant #24	Learning Experience Next	8/17/2009 10:35:26 AM
Participant #24	Traditional Exercise 1 Next	8/17/2009 10:36:07 AM
Participant #24	Traditional Exercise 2 Next	8/17/2009 10:36:18 AM
Participant #24	Traditional Exercise 3 Next	8/17/2009 10:36:21 AM
Participant #24	RAD Exercise 1 Next	8/17/2009 10:36:24 AM
Participant #24	RAD Exercise 2 Next	8/17/2009 10:36:25 AM
Participant #24	RAD Exercise 3 Next	8/17/2009 10:36:27 AM
Participant #24	Challenge Next	8/17/2009 10:36:30 AM
Participant #24	Post Survey Section 1 Next	8/17/2009 10:38:20 AM
Participant #24	Post Survey Section 2 Next	8/17/2009 10:39:25 AM
Participant #24	Post Survey Section 3 Next	8/17/2009 10:39:29 AM
Participant #24	Finish	8/17/2009 10:39:34 AM
Participant #25	Informed Consent Next	8/17/2009 12:52:06 PM
Participant #25	Session Message Next	8/17/2009 12:52:13 PM
Participant #25	Demographics Next	8/17/2009 12:53:21 PM
Participant #25	Programming Experience Next	8/17/2009 12:53:43 PM
Participant #25	RAD vs Trad Next	8/17/2009 12:57:50 PM
Participant #25	Learning Experience Next	8/17/2009 1:00:33 PM
Participant #25	RAD Exercise 1 Next	8/17/2009 1:00:47 PM
Participant #25	RAD Exercise 2 Next	8/17/2009 1:00:52 PM
Participant #25	RAD Exercise 3 Next	8/17/2009 1:00:57 PM
Participant #25	Traditional Exercise 1 Next	8/17/2009 1:00:59 PM
Participant #25	Traditional Exercise 2 Next	8/17/2009 1:01:01 PM
Participant #25	Traditional Exercise 3 Next	8/17/2009 1:01:03 PM
Participant #25	Challenge Next	8/17/2009 1:01:07 PM
Participant #25	Post Survey Section 1 Next	8/17/2009 1:11:59 PM
Participant #25	Post Survey Section 2 Next	8/17/2009 1:19:07 PM
Participant #25	Post Survey Section 3 Next	8/17/2009 1:23:41 PM

Participant #25	Finish	8/17/2009 1:23:55 PM
Participant #26	Informed Consent Next	8/17/2009 5:38:27 PM
Participant #26	Session Message Next	8/17/2009 5:38:34 PM
Participant #26	Demographics Next	8/17/2009 5:39:15 PM
Participant #26	Programming Experience Next	8/17/2009 5:41:53 PM
Participant #26	RAD vs Trad Next	8/17/2009 5:45:26 PM
Participant #26	Learning Experience Next	8/17/2009 5:46:45 PM
Participant #26	RAD Exercise 1 Next	8/17/2009 5:47:56 PM
Participant #26	RAD Exercise 2 Next	8/17/2009 5:48:03 PM
Participant #26	RAD Exercise 3 Next	8/17/2009 5:48:05 PM
Participant #26	Traditional Exercise 1 Next	8/17/2009 5:48:07 PM
Participant #26	Traditional Exercise 2 Next	8/17/2009 5:48:09 PM
Participant #26	Traditional Exercise 3 Next	8/17/2009 5:48:11 PM
Participant #26	Challenge Next	8/17/2009 5:48:14 PM
Participant #26	Post Survey Section 1 Next	8/17/2009 5:51:17 PM
Participant #26	Post Survey Section 2 Next	8/17/2009 5:54:45 PM
Participant #26	Post Survey Section 3 Next	8/17/2009 5:57:11 PM
Participant #26	Finish	8/17/2009 5:57:21 PM
Participant #27	Informed Consent Next	8/18/2009 12:01:14 PM
Participant #27	Session Message Next	8/18/2009 12:01:39 PM
Participant #27	Demographics Next	8/18/2009 12:02:18 PM
Participant #27	Programming Experience Next	8/18/2009 12:07:02 PM
Participant #27	Programming Experience Next	8/18/2009 12:23:11 PM
Participant #27	RAD vs Trad Next	8/18/2009 12:48:41 PM
Participant #27	Learning Experience Next	8/18/2009 12:56:38 PM
Participant #27	Traditional Exercise 1 Next	8/18/2009 1:02:42 PM
Participant #27	Traditional Exercise 2 Next	8/18/2009 1:06:08 PM
Participant #27	Traditional Exercise 3 Next	8/18/2009 1:14:00 PM
Participant #27	RAD Exercise 1 Next	8/18/2009 1:20:10 PM
Participant #27	RAD Exercise 2 Next	8/18/2009 1:20:11 PM
Participant #27	RAD Exercise 3 Next	8/18/2009 1:20:13 PM
Participant #27	Challenge Next	8/18/2009 1:20:32 PM
Participant #27	Post Survey Section 1 Next	8/18/2009 1:37:21 PM
Participant #27	Post Survey Section 2 Next	8/18/2009 1:41:32 PM

Participant #27	Post Survey Section 3 Next	8/18/2009 1:45:23 PM
Participant #27	Finish	8/18/2009 1:46:30 PM
Participant #28	Informed Consent Next	8/18/2009 2:14:49 PM
Participant #28	Session Message Next	8/18/2009 2:16:12 PM
Participant #28	Demographics Next	8/18/2009 2:17:09 PM
Participant #28	Programming Experience Next	8/18/2009 2:19:28 PM
Participant #28	RAD vs Trad Next	8/18/2009 2:22:25 PM
Participant #28	Learning Experience Next	8/18/2009 2:23:15 PM
Participant #28	Traditional Exercise 1 Next	8/18/2009 2:24:10 PM
Participant #28	Traditional Exercise 2 Next	8/18/2009 2:25:00 PM
Participant #28	Traditional Exercise 3 Next	8/18/2009 2:28:57 PM
Participant #28	RAD Exercise 1 Next	8/18/2009 2:39:12 PM
Participant #28	RAD Exercise 2 Next	8/18/2009 2:39:31 PM
Participant #28	RAD Exercise 3 Next	8/18/2009 2:39:56 PM
Participant #28	Challenge Next	8/18/2009 2:48:06 PM
Participant #28	Post Survey Section 1 Next	8/18/2009 3:06:04 PM
Participant #28	Post Survey Section 2 Next	8/18/2009 3:12:34 PM
Participant #28	Post Survey Section 3 Next	8/18/2009 3:15:31 PM
Participant #28	Finish	8/18/2009 3:16:02 PM
Participant #29	Informed Consent Next	8/26/2009 7:41:21 PM
Participant #29	Session Message Next	8/26/2009 7:41:47 PM
Participant #29	Demographics Next	8/26/2009 7:42:54 PM
Participant #29	Programming Experience Next	8/26/2009 7:44:51 PM
Participant #29	RAD vs Trad Next	8/26/2009 7:48:37 PM
Participant #29	Learning Experience Next	8/26/2009 7:52:26 PM
Participant #29	Traditional Exercise 1 Next	8/26/2009 7:56:01 PM
Participant #29	Traditional Exercise 2 Next	8/26/2009 7:56:55 PM
Participant #29	Traditional Exercise 3 Next	8/26/2009 7:57:39 PM
Participant #29	RAD Exercise 1 Next	8/26/2009 8:06:04 PM
Participant #29	RAD Exercise 2 Next	8/26/2009 8:13:01 PM
Participant #29	RAD Exercise 3 Next	8/26/2009 8:31:17 PM
Participant #29	Challenge Next	8/26/2009 8:49:32 PM
Participant #29	Post Survey Section 1 Next	8/26/2009 8:57:05 PM
Participant #29	Post Survey Section 2 Next	8/26/2009 9:06:16 PM

Participant #29	Post Survey Section 3 Next	8/26/2009 9:12:23 PM
Participant #29	Finish	8/26/2009 9:13:32 PM

Appendix F: Recruitment Notices

For Lab workshops

Home
Future Students
Courses
Facilities
Current Students
Blackboard
Student Support
CEED Program
FAQs
Student Resources
Facilities
International

« 2009 SECAU Security Congress – 1st Call for Papers
Looking to Launch Your ICT Career? »

Invitation to Participate in Programming Exercises for Honours Study

All students within the School of Computer and Security Science are invited to participate in the Honours Research of Pansy Colkers, a research student within this school. Pansy is looking at how novice and experienced programmers react to Traditional programming techniques (ie using text editors) versus using Visual Rapid Application Development methods that allow for drag and drop programming.

So, if you are new to programming, an experienced programmer or someone who has never programmed but would like to quickly try your hand at it, please assist Pansy with her research. The study will involve participants sitting in a lab and working through some predesigned programming exercises over the period of 60-90 minutes. The actual data collected during the exercises is totally anonymous, and has no impact on your studies whatsoever. Obviously, participation is appreciated but totally voluntary.

The exercises will run on Friday the 7th of August and there will be three lab times from which particpants can choose to attend. The times are available here. If you would like to participate, please email Pansy at pansyc@student.ecu.edu.au with the timeslot of your choosing. If you would like to know more about the study, please email Pansy or contact her thesis supervisor, Dr Justin Brown at j.brown@ecu.edu.au

For Online workshops

Home
Future Students
Courses
Facilities
Current Students
Blackboard
Student Support
CEED Program
FAQs
Student Resources
Facilities
International

« 2009 SECAU Security Congress – 1st Call for Papers
Looking to Launch Your ICT Career? »

Invitation to Participate in Programming Exercises for Honours Study

All students within the School of Computer and Security Science are invited to participate in the Honours Research of Pansy Colkers, a research student within this school. Pansy is looking at how novice and experienced programmers react to Traditional programming techniques (ie using text editors) versus using Visual Rapid Application Development methods that allow for drag and drop programming.

All you need to do is watch a couple of detailed videos that Pansy has put together looking at both traditional and visual RAD programming techniques and fill in an anonymous survey before and after watching the videos. You can either watch all the videos in one session, or record your session id and continue watching at a later date. Obviously, participation is greatly appreciated but totally voluntary.

If you would like to know more about the study, please email Pansy at pansyc@student.ecu.edu.au or contact her thesis supervisor, Dr Justin Brown at j.brown@ecu.edu.au

Participate

Appendix G: Informed Consent

An Investigation into Student Reaction to RAD vs Traditional Programming Environments for Novice Developers

Informed Consent >> Pre-Exercise Questionnaire >> Setup and Configuration >> Exercises >> Challenge Exercise >> Post-Exercise Survey >> End of Workshop

Informed Consent

This form regards the research project is being undertaken by Pansy Colkers for a Bachelor of Computer Science Honours at Edith Cowan University titled:

An Investigation into Student Reaction to RAD versus Traditional Programming Environments for Novice Developers

The research has been approved by the ECU Human Research Ethics Committee.

Contact Details

For further information, or any questions regarding the research, contact Pansy Colkers at pansyc@student.ecu.edu.au or on 0450 458 809.

You may also contact the supervisor of the research, Dr. Justin Brown, at j.brown@ecu.edu.au or on 0403 950 899.

Both the researcher and the supervisor are from the School of Computer and Information Science, in the Faculty of Computing, Health and Science.

If you have concerns about the research and would like to contact an independent person, Martin Masek can be contacted at m.masek@ecu.edu.au or on 9370 6410.

Intent to Participate

You have received an Information Letter describing the aims and procedures of the research. Participants are asked to participate in a programming workshop with three (3) web application development exercises in RAD (ASP.net exercises using Microsoft's Visual Studio) and traditional programming (PHP exercises using text editor). Short questionnaires will be administered before the programming exercises, and after completing these exercises.

All information collected will remain confidential and anonymous, and only be used to meet the aims of the research.

If you have any questions regarding the research which have not been answered, please ask the researcher now or contact one of the people listed in this form and the Information Letter.

Students are reminded that participation is entirely voluntary and will have no impact on their grade. Students may opt out of the research at any time.

☐ I have read and understood the information provided and wish to participate in this research.

Next

An Investigation into Student Reaction to RAD vs Traditional Programming Environments for Novice Developers **Session ID - 15667d**

Informed Consent >> Pre-Exercise Questionnaire >> Setup and Configuration >> Traditional Exercises >> RAD Exercises >> Challenge Exercise >> Post-Exercise Survey >> End of Workshop

Session ID - 15667d

Your current session ID is 15667d. You will need to enter this ID if you wish to stop the workshop at any time and resume it later.

Next

Appendix H: Traditional Programming Environment Exercises

TRADITIONAL PROGRAMMING ENVIRONMENT EXERCISE 1

Search and display data from database

External help on the exercise (Search Results from Google)

For the first exercise, we will be doing a simple search in a database and display the results on a web page. You will be provided with a search form to get you up and running. When a user enters a staff name in the search box, the system will look for the name in the database's StaffName field and display the results on the page.

Step 1:

Download the attached StaffDirectory.zip file and extract it to your desktop. Open the StaffDirectory folder on the desktop. Within the folder, you will probably see another folder called StaffDirectory, this being the folder which contains all the php and database files. Copy the StaffDirectory folder (the one with all the php and mdb files in it) to htdocs folder in xampp folder (e.g. It may look like this - C:\inetpub\wwwroot\xampp\htdocs\StaffDirectory).
StaffDirectory.zip

Step 2:

Open the C:\inetpub\wwwroot\xampp\htdocs\StaffDirectory\StaffSearch.php page in Editplus (Programs->Applications). StaffSearch.php contains the form to search staff details based on the name entered.

Step 3:

Insert the following piece of code after the line **//insert php codes here** in StaffSearch.php file. This code connects the application to the StaffDatabase.mdb file. You might need to change the line $db = 'C:\inetpub\wwwroot\xampp\htdocs\StaffDirectory\StaffDatabase.mdb' to point to the location of the StaffDabase file.

```
//connect to MS Access database

$conn = new COM('ADODB.Connection');

$db = 'C:\inetpub\wwwroot\xampp\htdocs\StaffDirectory\StaffDatabase.mdb';

$conn->Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=$db");
```

Step 4:

After connecting to database, we need to get the staff name from the text box and search the name in the database table. We use the $_POST["fieldname"] to get the data from the textbox (in the .php file) and store it in a variable. After that we use that variable to construct the select sql. SQL retrieves the records in the database and table which have the searched text in the staff name. The results are stored in a resultset called $rs. Place the following code after the database connection.

```
//Retrieve the staff name from the previous page

$varStaffName = $_POST["staffName"];

//Search the staff name in the staff table

$rs = $conn->Execute("SELECT * FROM staff WHERE staffName LIKE '%" . $varStaffName . "%'");
```

Step 5:

We'll use a while loop to iterate through the result set. But before that we need to do some error handling. if(!$rs) checks if resultset is a valid resultset. if(!$rs->EOF) before while loop checks if resultset contains any records. We display the appropriate message if there are no records to display. Place the following code after the records retrieval.

```
if(!$rs)

{

      echo "SQL query failed. Please check your query.<br />";

}else {

      if(!$rs->EOF){

            //display the results

            echo "<h2>Search Results</h2>";

            echo "<table border='1'><tr><th>ID</th>
```

```
                    <th>Faculty</th><th>Phone</th><th>Email</th><th> </th></tr>";
        while (!$rs->EOF)
        {
            $id = $rs->Fields("id");
            $name = $rs->Fields("staffname");
            $empNumber = $rs->Fields("empNumber");
                        $type = $rs->Fields("stafftype");
            $faculty = $rs->Fields("faculty");
            $phone = $rs->Fields("phone");
            $email = $rs->Fields("email");
            echo "<tr><td> ".$id ."</td>";
            echo "<td> " .$name . "</td>";
            echo "<td> " .$empNumber . "</td>";
            echo "<td> " .$type . "</td>";
            echo "<td> " .$faculty . "</td>";
            echo "<td> " .$phone . "</td>";
            echo "<td> " .$email . "</td>";
                        //create links for editing and deleting records (for exercise 3)
            echo "<td><a href='StaffEdit.php?id=" .$id . "'>Edit</a>
                    <a href='staffdelete.php?id=" .$id . "'>Delete</a></td></tr>";
            $rs->MoveNext();
           }
           echo "</table>";
       }else {
            echo "No record available to display";
       }
       $rs->Close();
   }
```

Step 6:

Finally, we need to close database connection.

```
$conn->Close();
```

You can go to http://localhost:8080/StaffDirectory/staffsearch.php to test the search page. The end result should look similar to this.

In summary, you have connected to a database, retrieved the submitted data from the form, performed a search using the select query and displayed the results (if any) on the form. You might have also noticed the syntax and commands in the PHP language such as **if (condition)---else, while(condition), echo and some variable declarations.** In the next exercise, you will be working on inserting records into the database.

Back Next

TRADITIONAL PROGRAMMING ENVIRONMENT EXERCISE 2

Inserting new records in database

External help on the exercise (Search Results from Google)

For this exercise, we will be inserting new records into the database and displaying the results on the page. You will be provided with a staff registration form with some input fields. This form was linked to the **Add New Staff** link on Staff Search page. After entering the information on the staff registration form, click on submit to insert the records. The system will store these information in the database and display the message if the insertion was successful.

Step 1:
Open the StaffReg.php file in EditPlus.

Step 2:
First we need to create a connection to the database to insert the records. It is the same as the creating database connection for previous exercise. Insert the following codes in php code section in StaffReg.php file.

```
//connect to MS Access database

$conn = new COM('ADODB.Connection');

$db = 'C:\inetpub\wwwroot\xampp\htdocs\StaffDirectory\StaffDatabase.mdb';

$conn->Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=$db");
```

Step 3:
After connecting to database, we need to get the data from the text boxes (in the form) and store them to the individual variables which we can use later to insert into the database. Place the following code after the database connection.

```
//Retrieve the staff info from the textboxes

$varStaffName = $_POST["staffName"];

$varEmpNumber = $_POST["empNumber"];

$varStaffType = $_POST["staffType"];

$varFaculty = $_POST["faculty"];

$varPhone = $_POST["phone"];

$varEmail = $_POST["email"];
```

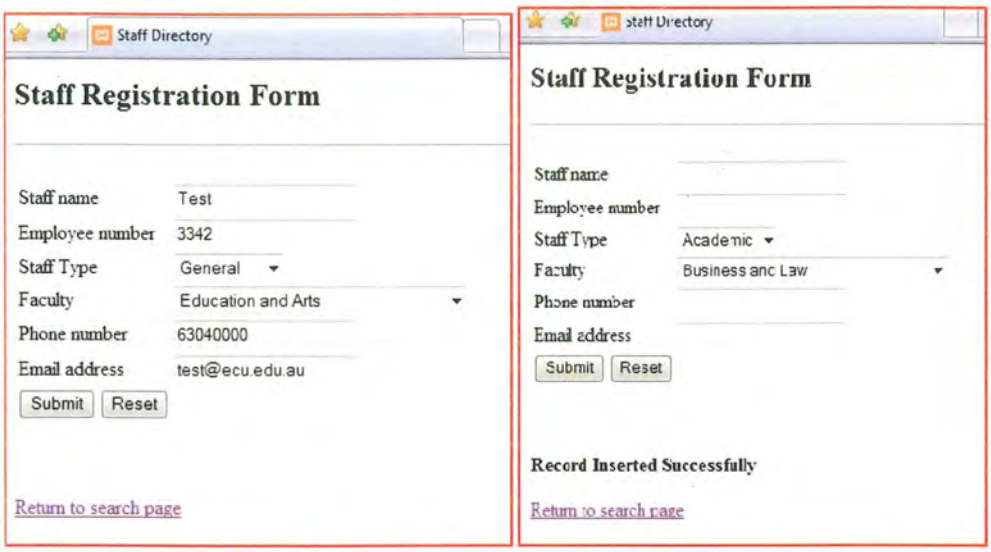
Step 4:
Then we generate the insert sql and execute the sql statement.

```
//Generate the insert sql

$insertSQL = "INSERT INTO Staff (StaffName, EmpNumber, StaffType, Faculty, Phone, Email) values

              ('" . $varStaffName . "', '" . $varEmpNumber . "', '" . $varStaffType . "',

               '" . $varFaculty . "', '" . $varPhone . "', '" . $varEmail ."')";

$conn->Execute($insertSQL);
```

Step 5:
Finally, we need to display the message and close the connection.

```
//display the results

echo "<h4>Record Inserted Successfully</h4>";

$conn->close;
```

You can go to http://localhost:8080/StaffDirectory/staffsearch.php and click on Add New Staff to test the page. The end result should look similar to this.



In summary, you have created a connection to a database, retrieved the submitted data from the form, inserted data into the database and displayed the result on the form. You might realize that there are no validations being performed before inserting to the database for this exercise. In real-life applications the validation on the user inputs is one of the most important requirements. We will do some validations in the challenged exercise. So far you have done searching and inserting the records in the database. In the next exercise, you will be working on deleting and editing records in the database.

Back Next

TRADITIONAL PROGRAMMING ENVIRONMENT EXERCISE 3

Editing and Deleting records in database

External help on the exercise (Search Results from G

For this exercise, we will be deleting and editing existing records in the database. You may have noticed the Edit and Delete links on Staff Search page. These links are linked to StaffEdit.php and StaffDelete.p files respectively. For deletion, we will look for the record id in the database and remove the record if the id is found. The user will be presented with a message after the deletion is completed. Editing the reco is a little trickier than other functions. It involves two major steps. First we need to present the user with the existing data on the form for the record that they have selected in order for them to make the change Then we will modify the data in the database after the user has made the changes and clicked Submit.

Step 1:
First we will start with deleting records. Open StaffDelete.php file in EditPlus.

Step 2:
Before deleting the record, we need to check if we managed to capture the staff id. This can be done by checking if there is a value in the id variable. please add the following code in StaffDelete.php

```
//check for record id

if ((!isset($_GET['id']) || trim ($_GET['id']) == ''))

{

     echo "Missing record ID.";

}else {

      //do the deleting here

}
```

Step 3:
Now the deleting part. First we connect to database as we have done in previous exercises

```
//connect to MS Access database

$conn = new COM('ADODB.Connection');

$db = 'C:\inetpub\wwwroot\xampp\htdocs\StaffDirectory\StaffDatabase.mdb';

$conn->Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=$db");
```

Step 4:
Then we retrieve the record id from the link and form the delete query. After that, we execute the delete query

```
//get id from the link

$id = $_GET['id'];

//generate delete sql

$deleteSQL = "DELETE FROM staff WHERE id=$id";

$conn->Execute($deleteSQL);
```

Step 5:
Finally, we need to display the message and close the connection

```
//display the results

echo "<h4>Record Deleted Successfully</h4>";

$conn->close;
```

Step 6:
For updating records, we need to populate the previous data in the form to allow the user to modify the record and later update the data in the database. Like Staffdelete.php, StaffEdit.php also receives t staff's id via the $id variable. Now we will modify the StaffEdit.php to check if the record id is valid. Open StaffEdit.php file in EditPlus. Place these codes under //codes to prefill the form.

```
//check for record id

if ((!isset($_GET['id']) || trim ($_GET['id']) == ''))

{

     echo "Missing record ID.";

}else {

          //do the searching here

}
```

Step 7:

Now we connect to the database and retrieve the id from the database. The process is similar to the staff search but in this case we are only retrieving one record from the database so we do not need to use the while loop.

```
//connect to MS Access database

$conn = new COM('ADODB.Connection');

$db = 'C:\inetpub\wwwroot\xampp\htdocs\StaffDirectory\StaffDatabase.mdb';

$conn->Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=$db");

//get id from the link

$id = $_GET['id'];

//Search the id in the staff table

$rs = $conn->Execute("SELECT * FROM staff WHERE id = $id");

if(!$rs) {

        echo "SQL query failed. Please check your query.";

}else {

        if(!$rs->EOF){

                $name = $rs->Fields("staffname");

                $empNumber = $rs->Fields("empNumber");

                $type = $rs->Fields("stafftype");

                $faculty = $rs->Fields("faculty");

                $phone = $rs->Fields("phone");

                $email = $rs->Fields("email");

                        //place the form here...

                        //close the record set

                        $rs->Close();

        }else{

                        //no result returned

                        echo "Unable to find record id " . $id . " in database";

        }

}
```

Step 8:

We have retrieved the values from the database so we can prefill the form now. We can use value=<?php echo $variable; ?> to print the php value of the variable in the text box. Replace the line //place the form here... with the code below

```
//prefill the form

?>

<body>

<h2>Staff Update Form </h2>

<hr />

<form name="StaffUpdateForm" id="StaffUpdateForm" method="post" action="StaffEdit.php?id=<?php echo $id; ?>">

<table> <tr>

        <td>Staff name </td>

        <td><input name="staffName" type="text" id="staffName" value="<?php echo $name; ?>" /></td>

</tr> <tr> <td>Employee number </td>

        <td><input name="empNumber" type="text" id="empNumber" value="<?php echo $empNumber; ?>" /></td>

</tr> <tr> <td>Staff Type</td>

        <td><select name="staffType" id="staffType">

                <option value="Academic" <?php if(strcmp($type,'Academic')==0) { ?>selected="selected" <?php } ?> >Academic</option>

                 <option value="General" <?php if(strcmp($type,'General')==0) { ?>selected="selected" <?php } ?>>General</option>

        </select> </td> </tr>
<tr> <td>Faculty </td>

        <td><select name="faculty" id="faculty">

                 <option value="F_BL" <?php if(strcmp($faculty,"F_BL")==0) { echo "selected='selected'"; } ?>>F_BL</option>

                 <option value="F_CHS" <?php if(strcmp($faculty,"F_CHS")==0) { echo "selected='selected'"; } ?>>F_CHS</option>

                 <option value="F_EA" <?php if(strcmp($faculty,"F_EA")==0) { echo "selected='selected'"; } ?>>F_EA</option>

                 <option value="F_RPS" <?php if(strcmp($faculty,"F_RPS")==0) { echo "selected='selected'"; } ; ?>>F_RPS</option>

        </select> </td> </tr>
```

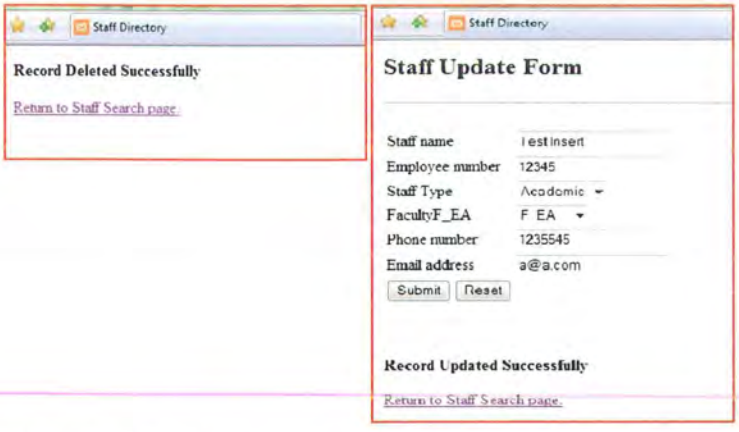
```
<tr> <td>Phone number</td>
        <td><input name="phone" type="text" id="phone" value="<?php echo $phone; ?>"/></td> </tr>

<tr> <td>Email address</td>
        <td><input name="email" type="text" id="email" value="<?php echo $email; ?>"/></td> </tr>

<tr> <td><input type="submit" name="Submit" value="Submit" />
        <input name="Reset" type="reset" id="Reset" value="Reset" /></td> <td> </td> </tr>

</table>
</form>

<?
```

Step 9:

Now it's time to update the values to the database after clicking on Submit button. This is very much similar to insert except that we do not have to reconnect to the database as it's already been connected in step 7. Place the codes below the line //codes to update the record

```
if($_POST['Submit']) {

//Retrieve the staff info from the textboxes

$varStaffName = $_POST["staffName"];

$varEmpNumber = $_POST["empNumber"];

$varStaffType = $_POST["staffType"];

$varFaculty = $_POST["faculty"];

$varPhone = $_POST["phone"];

$varEmail = $_POST["email"];

//Generate the update sql

$updateSQL = "Update Staff set StaffName= '$varStaffName' , EmpNumber= '$varEmpNumber' ,

            StaffType = '$varStaffType' , Faculty = '$varFaculty' , Phone = '$varPhone' ,

            Email = '$varEmail' WHERE id = $id" ;

$conn->Execute($updateSQL);

//display the results

echo "<h4>Record Updated Successfully</h4>";

$conn->close;

}
```

You can go to http://localhost:8080/StaffDirectory/staffsearch.php and click on Edit and Delete to test the page. The end result should look similar to this



The editing and deleting in traditional programming environment is very different from those in visual RAD environment. In visual RAD exercise that you have done (or will be doing), editing and deleting functions are part of the GridView control that is used for viewing the data.

Congratulations! you have done searching, inserting, editing and deleting records in database using Traditional Programming Environment.

Back Next

Appendix I: Visual RAD Environment Exercises

VISUAL RAD ENVIRONMENT EXERCISE 1

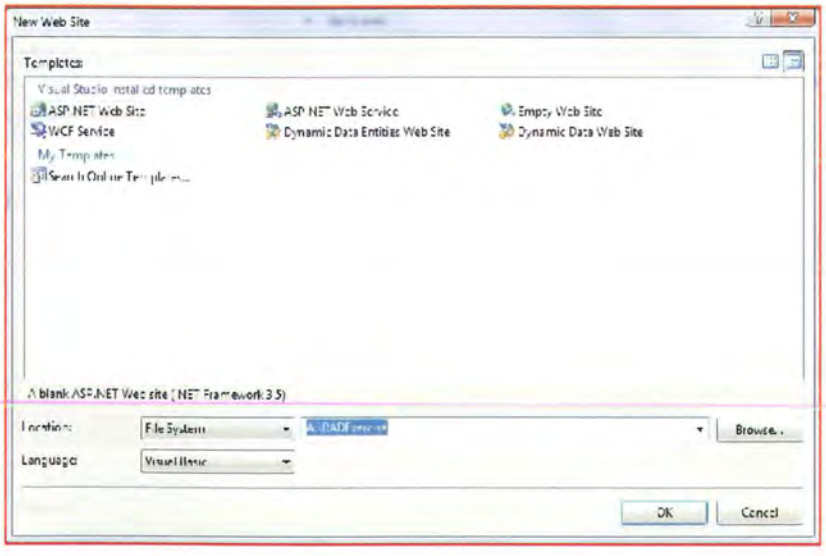
For the first exercise, we will be doing a simple search in the database and display the results on the page. We will be using some built-in controls in Microsoft's Visual Studio to perform the function. When a user enters a staff name in the search box, the system will look for the name in the database;s StaffName field and display the results on the page.

Search and display data from database

External help on the exercise (Search Results from Google)

Step 1:

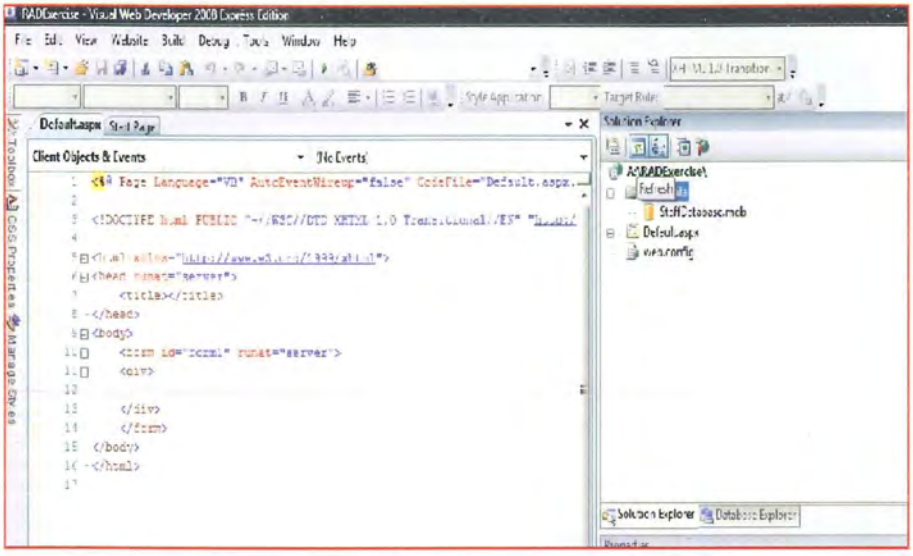
Create a new ASP.Net Web Site in Microsoft's Visual Studio by going to File => New Web Site. Select ASP.Net Template and store the website in folder RADExercises (e.g C:\Users\{username}\Documents\Visual Studio 2008\WebSites\RADExercises).



Step 2:

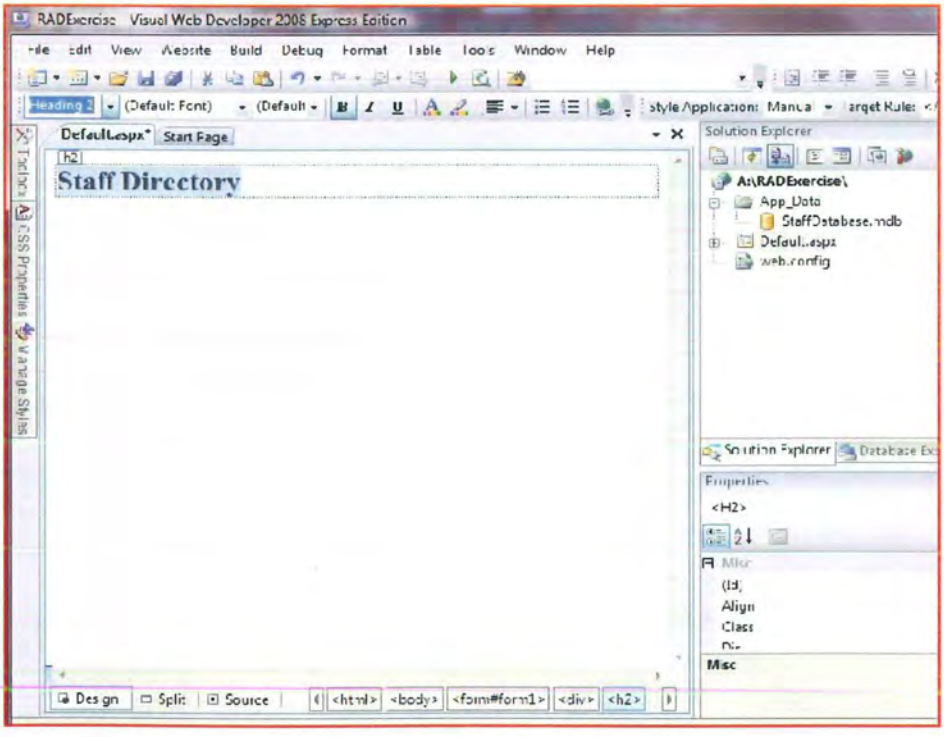
Download the attached StaffDatabase.zip file. Extract it and store on the Desktop, then copy StaffDatabase.mdb file to RADExercises\App_Data folder (You might find this folder under My Documents\Visual Studio 2008\Websites\ ). Refresh the Solution Explorer and you should be able to see the database file under the App_Data folder.

Staff Database (Right-click-> Save Target As)



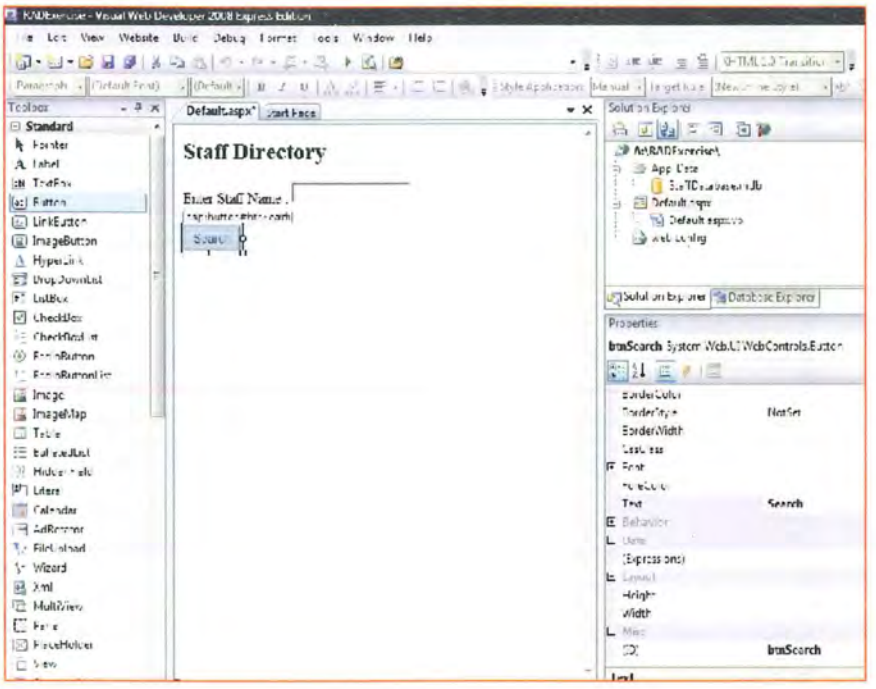
Step 3:

Switch your page into Design View, then type the title "Staff Directory" and set it to Heading 2.



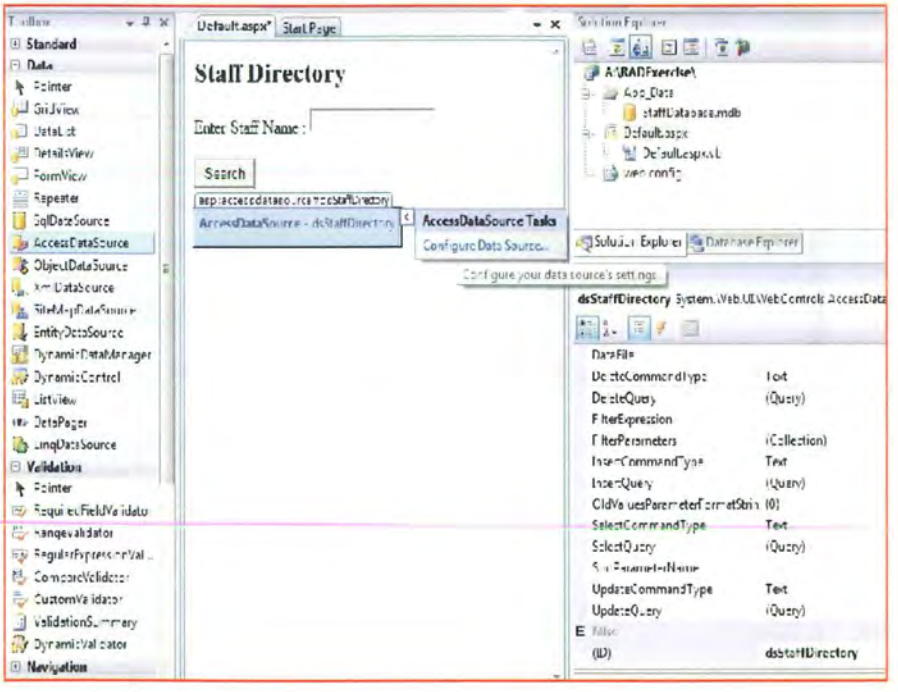
Step 4:

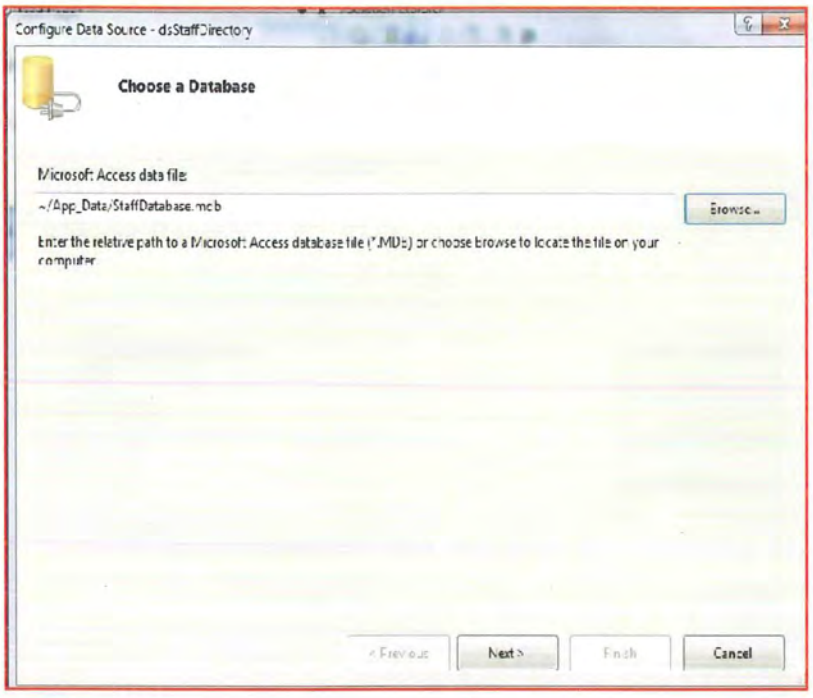
Now we will create a search form with some controls. Type in Enter Staff Name: and Drag and drop the TextBox and Button controls onto the form next to the text. Rename the TextBox id to txtStaffName and Button id to btnSearch in the Properties box.



Step 5:

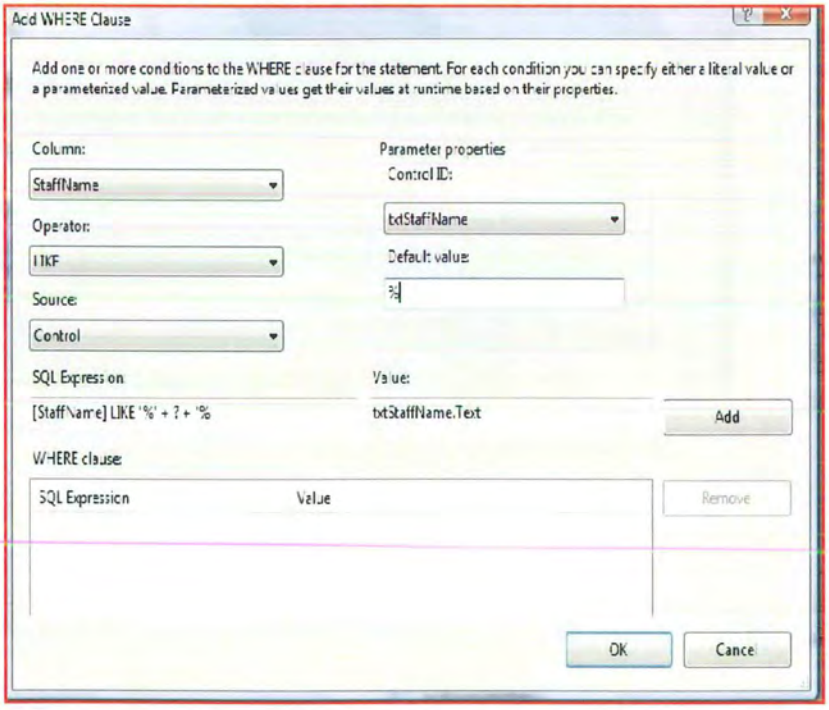
Drag and drop the AccessDataSource onto the form and name it dsStaffDirectory. Configure the datasource to point to StaffDatabase under App_Data folder

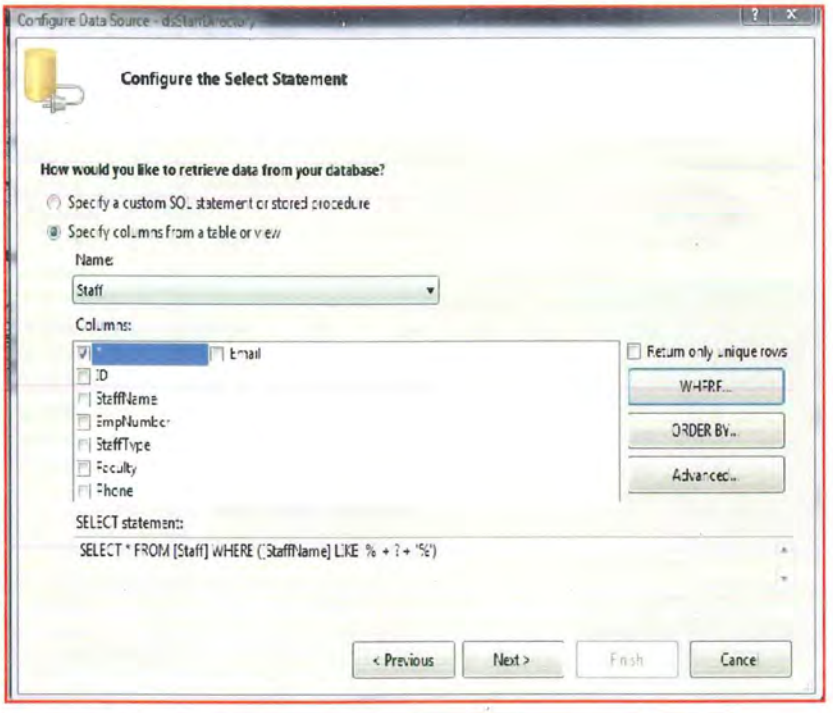


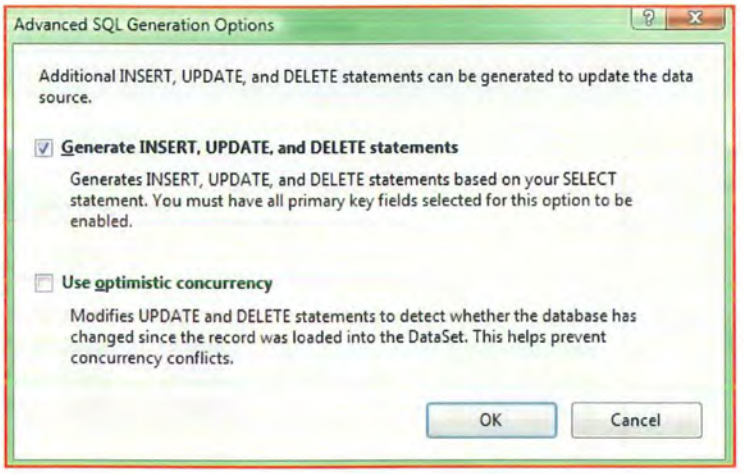


Step 6:

Select all the fields in the Staff table and click on WHERE to add a search condition. For this case, the search condition will be similar to the screenshot below. Also click on Advanced and Select Generate Insert, Update and Delete statements.

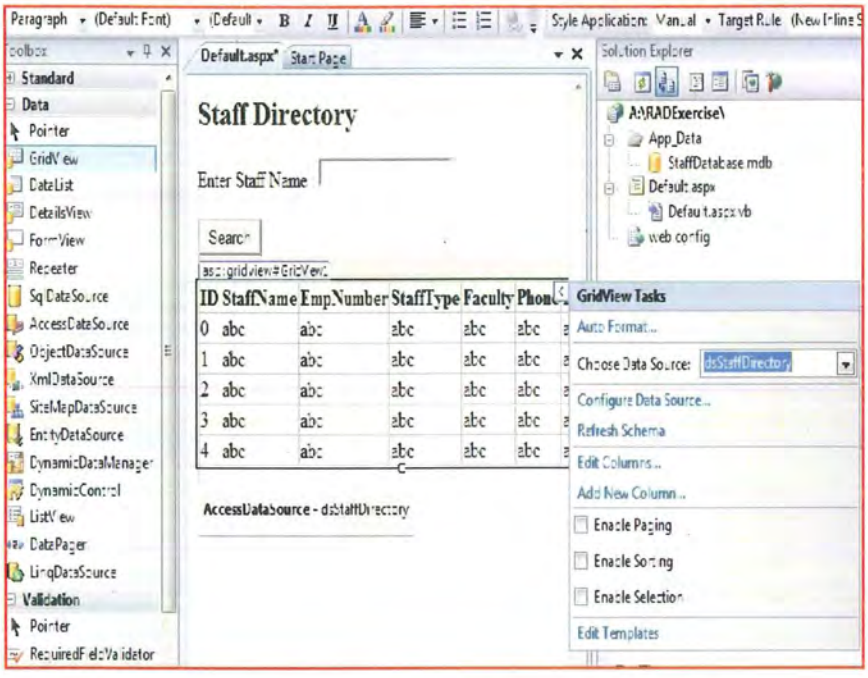




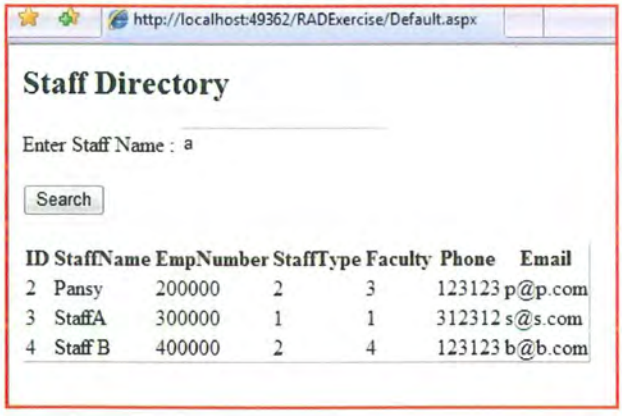


Step 7:

Drag and Drop the GridView control and link it to dsStaffDirectory DataSource.



Click on the Run icon to view the results. It should look similar to this.



In summary, you have used some standard controls such as textbox and button and data controls such as GridView and AccessDataSource to perform the simple search function. In visual RAD Environment, this sort of standard operations can done with simple drag-and-drop controls. Compare this process to that involved in the Traditional system when you have completed both sets of exercises. In the next exercise, you will be working on editing and deleting records in the database.

Back Next

VISUAL RAD ENVIRONMENT EXERCISE 2

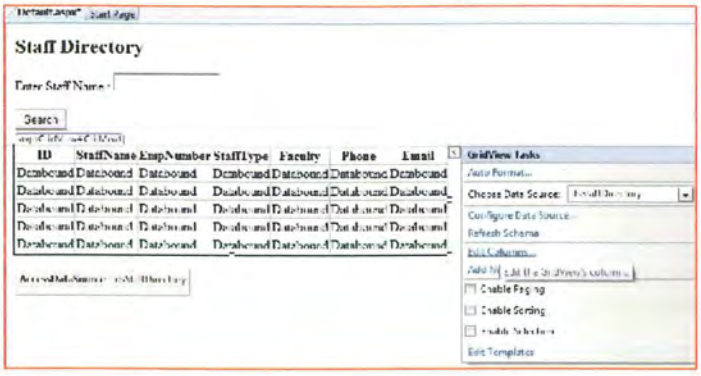
Editing and Deleting records in database

External help on the exercise (Search Results from Google)

For this exercise, we will be editing and deleting the records in the database. The GridView control in Microsoft's Visual Studio allows editing and deleting the records in the table without requiring any additional code. However, to modify the way data is displayed in the table, it requires more in-depth knowledge of the functions offered in the GridView control.

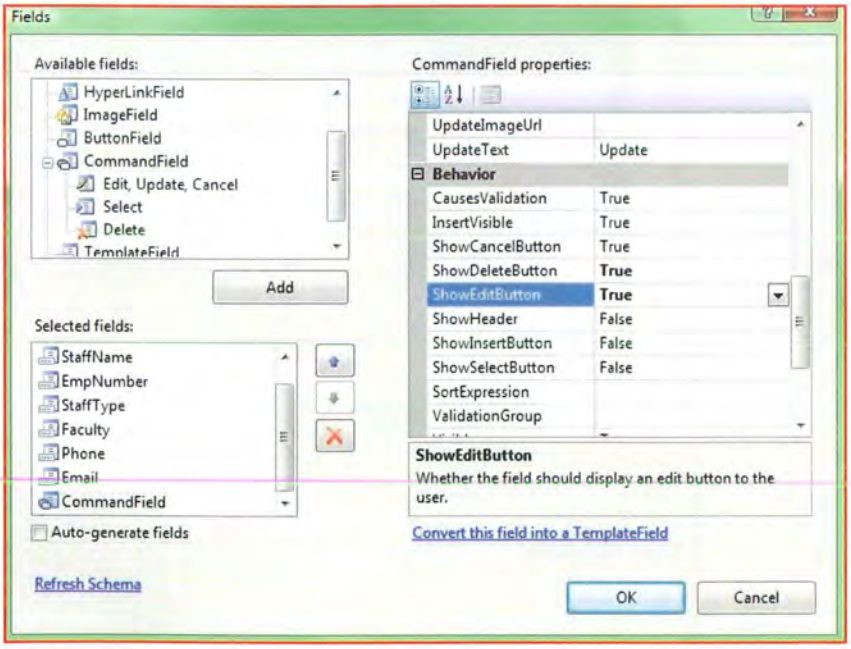
Step 1:

We will continue to use the RADExercise website that we created in the previous exercise. From GridView Tasks, Click Edit Columns.



Step 2:

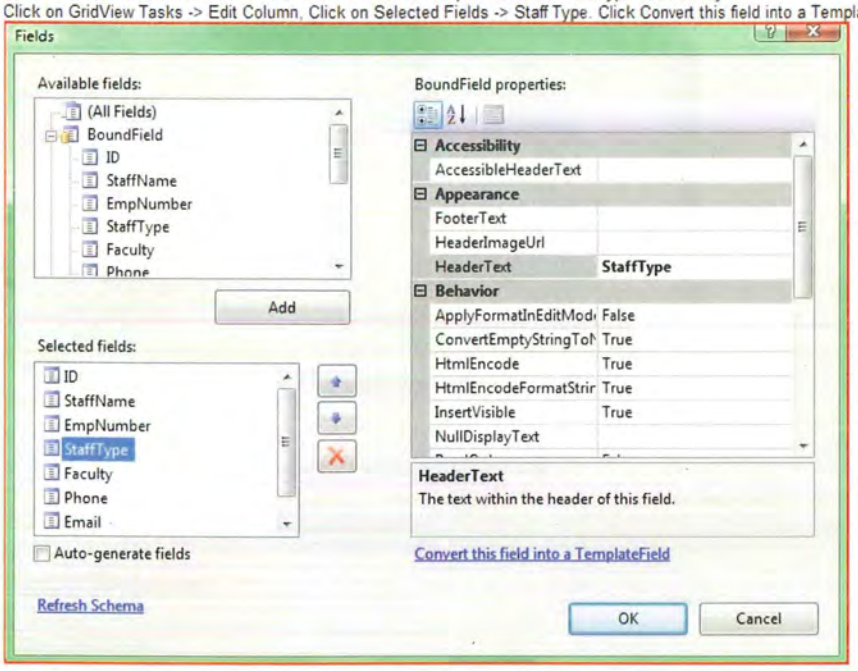
In Available fields, Select CommandField then click Add. In CommandField properties, Set true for ShowEditButton and ShowDeleteButton



Step 3:

This is all that is needed to edit or delete a simple record. When a user clicks on Edit, it will automatically display the value in text field for each column for the user to edit. However, for this case, we need to have a drop down list for Staff type and Faculty.

Click on GridView Tasks -> Edit Column, Click on Selected Fields -> Staff Type. Click Convert this field into a Template Field. Do the same for Faculty.

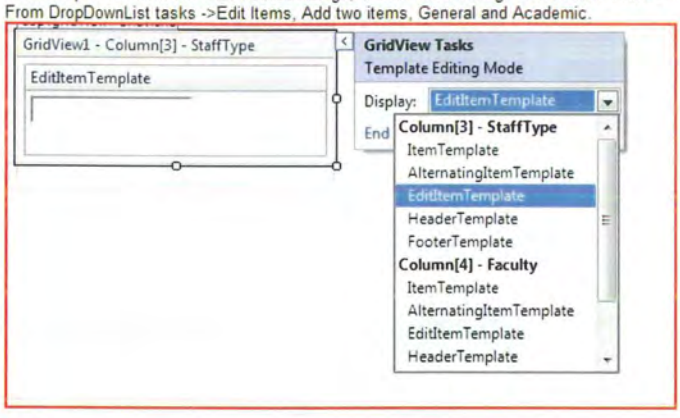


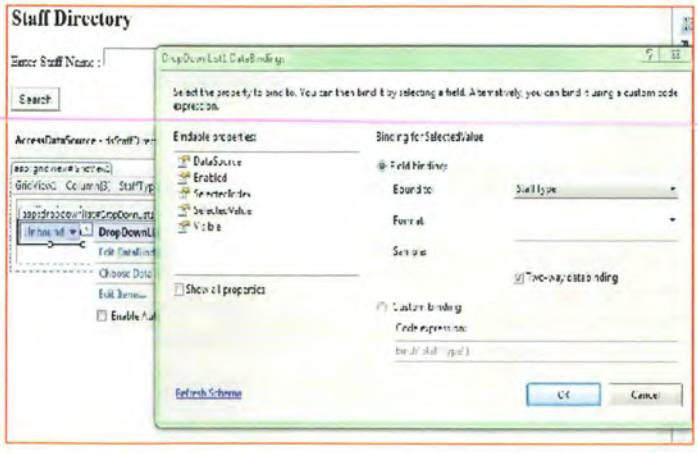
Step 4:

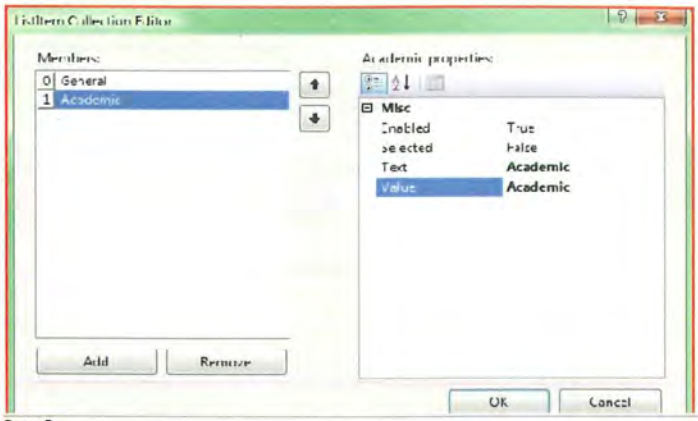
Click on GridView Tasks -> Edit Templates, Select StaffType -> EditItem Template. Delete the textbox and Drag and Drop a new dropdown list.

From DropDownList tasks ->Edit DataBindings, Choose the Binding For Selected Value -> Field Binding -> Bound To -> StaffType.

From DropDownList tasks ->Edit Items, Add two items, General and Academic.



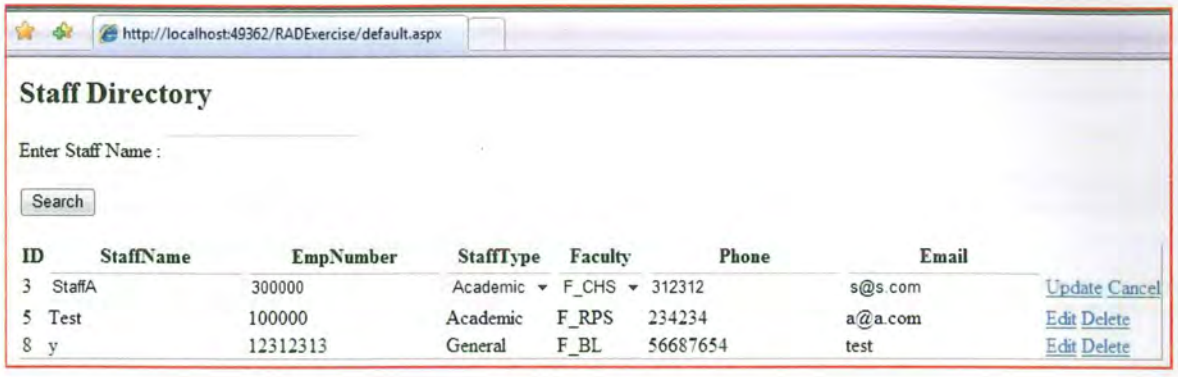




Step 5:

Repeat Step 4 for Faculty. Items in the faculty dropdown list will be F_BL, F_CHS, F_EA and F_RPS. Then click on End Template Editing.

Click on the Run icon  to view the results. It should look similar to this.



So far you have performed searching, editing and deleting records in database using visual RAD environment. In visual RAD environment, you are not required to know exactly how these editing and deleting are done but what controls and properties are needed to perform the function. In the next exercise, you will be working on inserting records into the database.

Back Next

VISUAL RAD ENVIRONMENT EXERCISE 3

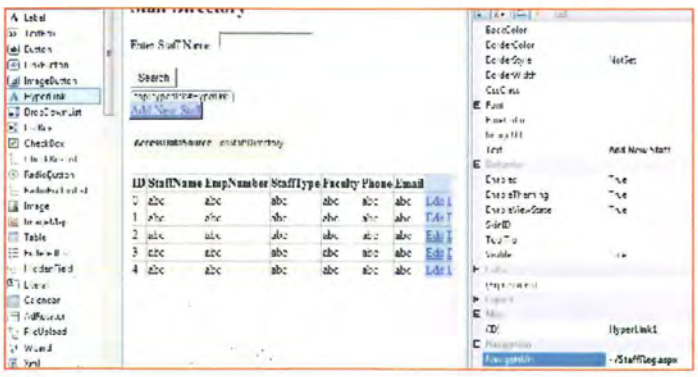
Inserting records in database

External help on the exercise (Search Results from Google)

For this exercise, we will be inserting some records into the database. We will be creating a link on our previous staff search page and link it to a new page called StaffReg.aspx. We will use some input controls to create a registration form and AccessDataSource to connect to the database and insert the record.

Step 1:

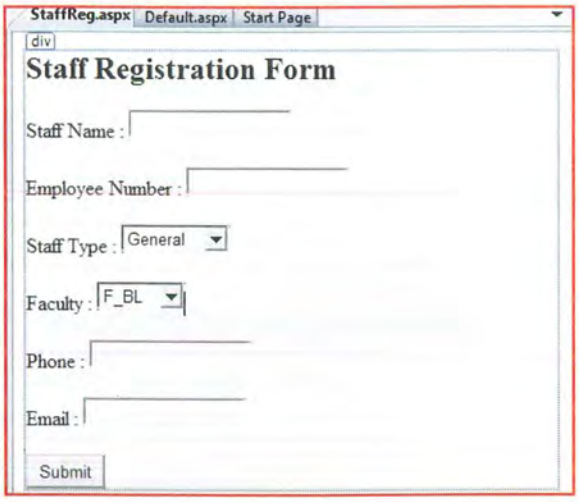
We will continue to use the RADExercise website that we created in the previous exercises. Right click on the project in Solution Explorer -> Add New item (Web Form) and name it StaffReg.aspx. Go back to Default.aspx and Add a hyperlink "Add New Staff" below Search Button and link it to StaffReg.aspx.



Step 2:

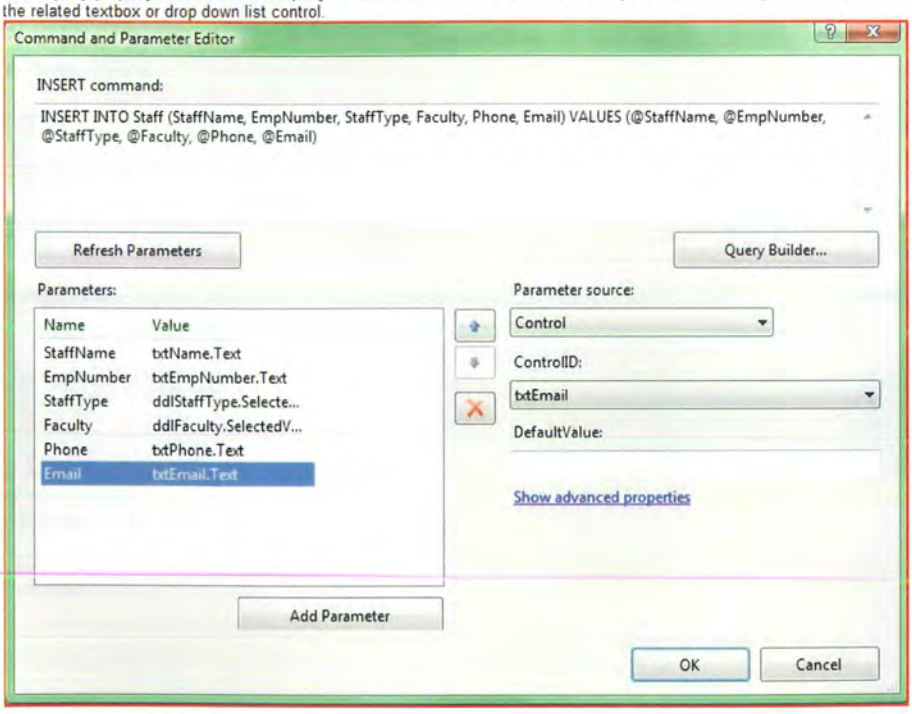
In StaffReg.aspx-> Design View, Create a form similar to the screen shot below. Here are the controls in the form.

Staff Name -> TextBox, ID = txtName
Employee Number -> TextBox, ID = txtEmpNumber
Staff Type -> DropDown List, ID = ddlStaffType, items = General, Academic
Faculty -> DropDown List, ID = ddlFaculty, items = F_BL, F_CHS, F_EA, F_RPS
Phone -> TextBox, ID = txtPhone
Email -> TextBox, ID = txtEmail
Submit -> Button, ID = btnSubmit, Text = Submit



Step 3:

Drag and Drop the AccessDataSource Control from the ToolBox. In AccessDataSource -> Properties, Select StaffDatabase.mdb for the DataFile Property. In Insert query property, Create an insert query similar to the screen shot below with 6 parameters. Each parameter is linked to the Control Parameter Source and to the related textbox or drop down list control.



Step 4:

Double click on Submit button to go to the Code View. In btnSubmit_Click function, enter these lines

```
AccessDataSource1.Insert()
Response.Redirect("~\Default.aspx")
```

StaffReg.aspx.vb | StaffReg.aspx | Default.aspx | Start Page

StaffReg (Declarations)

```
Partial Class StaffReg
    Inherits System.Web.UI.Page

    Protected Sub btnSubmit_Click(ByVal sender As Object, ByVal e As Syste
        AccessDataSource1.Insert()
        Response.Redirect("~\Default.aspx")
    End Sub
End Class
```

Click on the Run icon  to view the results. It should look similar to this.

Congratulations! you have done searching, inserting, editing and deleting records in database using Visual RAD Environment.

Back Next

An Investigation into Student Reaction to RAD vs Traditional Programming Environments for Novice Developers **Session ID - dc8fcb**

Informed Consent >> Pre-Exercise Questionnaire >> Setup and Configuration >> RAD Exercises >> Traditional Exercises >> Challenge Exercise >> Post-Exercise Survey >> End of Workshop

THANK YOU FOR YOUR PARTICIPATION.....

Please leave your email address below, if you would like to participate in a face-to-face interview to discuss further on the Programming environments and this workshop.

Finish