

Using Incremental JavaServer Faces Projects for Promoting Active Learning in Teaching Web Applications & Security

Cheer-Sun Yang^{*,1}

¹ Computer Science Department, West Chester University, West Chester, PA 19383, USA

Teaching emerging technologies with an active learning approach in Computer Science Department is generally perceived as a challenging task. This is mainly due to the fact that students usually lack the knowledge for getting involved in dialogues about the course subjects. This paper introduces the pedagogical approach of using incremental lab projects for promoting active learning in teaching *Web applications and Security*. These incremental lab projects are easy to learn, easy to implement, and relevant with the targeted subjects. As a result of using the incremental projects, students become motivated and actively participate in the discussion about various subjects. In this paper, the concepts of web applications and security, JavaServer Faces, Java Studio Creator are introduced. Then the approach of using incremental JSF projects is described. Finally, the experiences learned in the process of taking this approach and the future tasks are summarized at the end.

Keywords Active Learning; Web Applications and Security; JavaServer Faces (JSF)

1. Introduction

Teaching emerging technologies in Computer Science Department is generally perceived as a challenging task. Not only is teaching preparation a burden to the instructor, but also the learning model may become teacher-centered lectures. This is mainly due to the fact that students usually lack the knowledge for getting involved in discussions about the course subjects. For example, the use of an Integrated Development Environment (IDE) with JavaServer Faces (JSF) to develop a Web application can be an interesting and appealing approach for teaching Web Applications and Security. However, since the Java Studio Creator IDE and JSF are emerging technologies, they keep evolving [1]. The course preparation must consider what version of the software to use, which textbook to follow, what subjects to cover, and which lab exercises to implement as well as one major challenge--how the course can be delivered with a student-centered teaching/learning approach. When most students are new to JSF and the Studio Creator together, even active students may be reluctant to participate due to the lacking of knowledge such as Dynamic Navigation, JavaServer™ Beans, Validations and Conversions, Data Source Providers, and Database Accessing, Enterprise Java Beans (EJB), AJAX (Asynchronous JavaScript and XML), and Portlets.

This paper introduces the pedagogical approach of using incremental lab projects for promoting active learning in teaching Web applications and Security. These incremental lab projects are easy to learn, easy to implement, and relevant with the targeted subjects in nature. This set of lab projects starts with a base project and progressively becomes more complicated as more requirements are introduced when more course subjects are covered in the classroom. In the remaining part of this paper, the fact about teaching *Web Applications and Security* is introduced first. The danger of falling into the teacher-centered model is emphasized as the pedagogical issues are discussed. Then this paper proposes to supplement the course material with a set of incremental projects for promoting the active learning and connecting these seemingly un-related course subjects together. Finally, this paper summarizes the experiences learned in the process of taking this approach and lists the future tasks at the end.

2. Web Applications & Security under Java 2 Enterprise Edition (J2EE)

* Corresponding author: e-mail: cyang@wcupa.edu, Phone: +1 610 518 7678

Since the advent of the Internet, more and more applications are designed and developed with the Internet as the execution platform. Web applications are computing systems designed with the concept of client-server architecture. The Model-View-Controller (MVC) framework can be applied to most enterprise applications using the concept of Web Applications. As a result, teaching web applications and security has become essential in Computer Science Department.

In the area of web application design and development, several commercial platforms are available. Among them, Java 2 Enterprise Edition (J2EE) [2] and Microsoft .NET [3] are two popular choices; many companies in the industry has adopted one or both for various purposes when developing web applications. In our course, J2EE has been chosen for teaching this course. Another course focuses on Web Applications with Microsoft's Active Server Page (ASP) and C# under .NET environment. In this paper, the experiences of developing web applications with JavaServer Faces for the server-side user interface design and the Java Studio Creator IDE are introduced.

3. JavaServer Faces and Java Studio Creator

3.1 JavaServer Faces (JSF)

Traditionally, the major server-side programming languages are Servlet or JavaServer Pages (JSP) under the J2EE platform. However, there is an emerging technology known as JavaServer Faces (JSF). The main objective of JSF technology is to provide an alternative for applying rapid prototyping to the server-side programming. With the assistance of JSF expression languages and the event-driven programming model, the user interface of a web application can be designed and tested by using a drag-and-drop visual editor and the corresponding JSP program can be generated.

Some major JSF features for supporting the rapid-prototyping of server-side user interface design include the following [4]: static and dynamic page navigation, data conversion, data validation, event handling, database accessing with or without an additional data provider layer between the user code and the database, JavaBeans support, and Enterprise JavaBeans (EJB). With the help of a visual-oriented IDE known as the Java Studio Creator or simply Creator, the prototyping of web applications using J2EE can be completed much effectively. The basic features of the Creator are briefly introduced in the next section.

3.2 Java Studio Creator (Creator)

The Java Studio Creator is an IDE for using J2EE to develop web applications [5]. It provides several (sub)-windows for various purposes: Palette window, Servers Window, Outline Window, Navigator Window, Properties Window, Project Window, Files Window, and a Main Window that combines the Design View Window, JSP Source Editor Window and the Java Source Editor Window. In this section, some important Creator features are discussed for the purpose of understanding the basic project for teaching Web Applications and Security under J2EE. With the support of Creator, a rapid deployment of a server-side user interface can be completed. For example, a basic web login project can be developed for supporting a better web security. This project allows a user to enter a user id and a password prior to accessing any other web resources. After the project name is entered, the design view window will be displayed in the middle. A developer can click on the Label icon in the Palette Window, drag and drop it onto the Design View Window. The text string can be modified in the Design View or using the Properties Window. After the textboxes for user id and the password and the two buttons, i.e., a Login button and a Reset button, are added to the Design View, the Creator can automatically generate the corresponding JSP program for displaying this user interface page. The JSP program can be displayed by simply clicking the "JSP" tab on top of the Main Window. The properties about each web component, i.e., textboxes, labels, etc, can be changed by using the Properties Window,

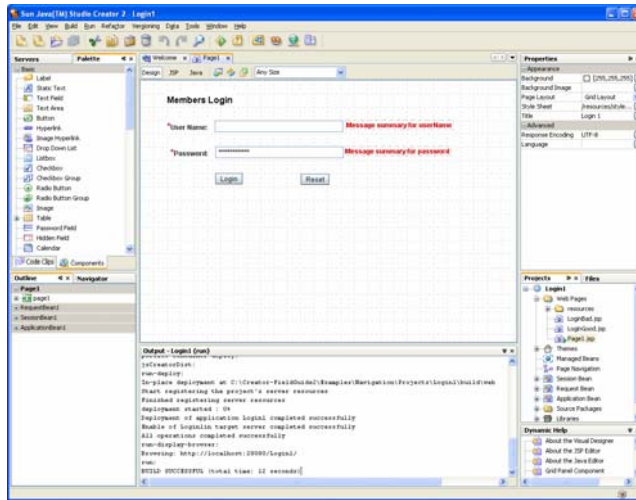


Fig. 1 *Java Studio Creator*. A sample user authentication project.

As a starting point, students will be introduced to become familiar with the Creator tool with the basic Login page. For simplicity, a specific login id and password pair is hard-coded in the Java source code. The Java source code can also be accessed or edited with the name of the page. The default page is called Page1. In Fig. 1, the first incremental project demonstrates how an incremental project can be devised and how incremental projects can be used to inspire students. If the user enters a correct combination of user id and the password, another welcome page will be displayed. Despite the simplicity, this application demonstrates several features of the Creator and JSF in addition to some basic functions the Creator provides. For example, the concepts of event-driven programming, static page navigation, and the drag-and-drop server-side user interface design model.

4. Incremental JSF Projects

In this section, the design of a set of incremental JSF projects is described. These projects will be used for illustrating the topics of Static Navigation, Dynamic Navigation, JavaBeans Components, Virtual Forms, Data Providers, and Web Services, respectively. All of them are easy to implement, easy to understand, and instructive. The main purpose is to lead students toward active thinking on problem solving.

4.1 Static Navigation

The first incremental project requires students to one more textboxes for entering the user's age, changing the "Login" button to "Compute" button, adding another "Reset" button, and adding another Static Text field. If the user id and the password pair is correct, the user's age of 35 will be incremented and the Static Text will show the text "Your age will become 36 next year!!" This simple change on the requirements will force the students to understand the original program completely first. For example, the navigation of JavaServer Page (JSP) requires an event-handling method to return a character string that has to be used in the Navigation Editor as a label for the arrow connecting the first page and the second page.

Originally, after the authentication is verified, another web page is displayed to show “Welcome to the member’s page!!” With the new requirement, the new page is not going to be displayed. Instead, the current age is incremented by one and displayed in the static text field. Students will be inspired to consider how this is achieved. Students may immediately consider changing the welcome page to the page for displaying the user’s age for next year. But, this approach will make the static text field useless. Then, the instructor can demonstrate how to display the age in the same web page instead of navigating to another page by simply changing the returned value to a blank string in the event-handling method.

This solution is far beyond any student’s expectation. Without using the incremental project, an instructor will be “forced” to elaborate all the details about page navigation and demonstrate how it works. Student may or may not truly understand the details since the control is not intuitive. This is the traditional “passive-learning” approach. With the incremental project, the instructor can demonstrate the basic project after illustrating the requirements first. Then, the instructor can change the requirements that are simple and easy to understand. The critical step follows. The instructor can invite students to use intuitive or creativity to consider how the project should be completed. With each solution students suggest, the instructor can add comments and lead students to the correct solution of changing the returned value from a string to a blank string. With this approach, the students will be impressed easily on how the navigation is actually completed and will be inspired to know more about JSF and the Creator. The minor change actually allows the application server to display the current page with a new content.

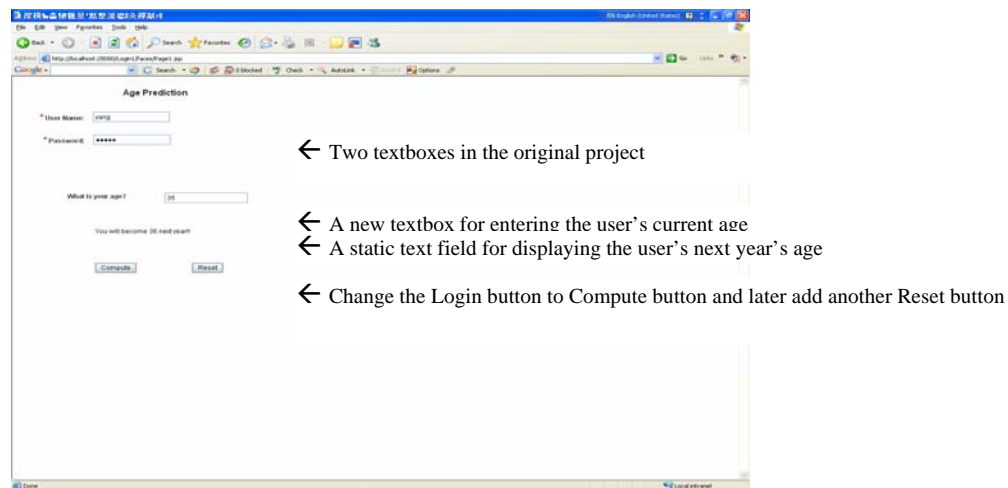


Fig. 2 *Age Prediction Project*. An incremental project #1.

4.2 Dynamic Navigation

Of course, the incremental project won’t stop here. The previous example only demonstrates the feature known as “Static Navigation” when the target web page is known at the development time. There are times when the target web page cannot be determined at the run-time. For example, if the checking of the user id/password fails, the navigation will display an error page instead of displaying the current page with the user’s age for next year. After the “basic” project is completed, the instructor can add the new requirement—if the user enters either an incorrect user ID or password, another web page is displayed to indicate that “The user ID or the password you entered is incorrect.” (Here the concept of web security can also be introduced to remind students that the error message should not indicate which one—the user id or the password—is incorrect.) Also, if a user enters an incorrect number as the current age, e.g., a

negative number or a number greater than 120, the error page should be displayed. The first incremental project is named as the “Age Prediction” project.

At this point, students have learned the basics of using the Creator as well as the concept of JSF. They are more likely to join the discussion on dynamic navigation. Many students also show strong interests on implementing more features such as displaying a Welcome Page if the user id and the password are correct—the learning style tends to change from passive to more active.

4.3 JavaBeans Components

As of now, the Age Prediction project only indicates that the incorrect (user id, password) pair is entered. It does not display what user id was entered. As a motivation statement, the instructor can point out that “data sharing” is the additive requirement. This is actually another incremental project with JavaBeans Components. A JavaBeans component is a Java class with a default constructor that does not have any parameter and some other properties. For example, for each instance variable in the class, there is a Getter and a Setter. Although it is easy to explain the concept of a class, the usage of a JavaBeans components is not intuitive. With the new requirement, a JavaBeans component has to be used to keep some information about the user, i.e., user id and the user’s current age. The error indication can become more user-friendly in this case. Students will be motivated to design another page for displaying the user’s age for next year instead of using a static textbox. This is another indication of active learning style.

4.4 More Incremental Projects

The pedagogical approach of applying incremental projects can be used throughout the whole semester. They look simple yet instructive. For example, the concept of developing virtual forms can be the next incremental project. The Age Prediction page can be used for authentication as well as for implementing a business logic. As a result, some data fields must be grouped together and used for one purpose, and some other fields have to be ignored for another purpose. In other words, the same web page may be used for various purposes. The Creator’s solution is known as “Virtual Forms”, i.e., a virtual form is created for each purpose, if needed. Again, the concept seems to be easy, yet hard to accept. With the help of another incremental project—adding a Reset Button to reset the user id and password, the concept can be illustrated nicely. The same page can be used for user authentication as well as for resetting user input. Of course, later, a more complicated project can also be used to further illustrate the concept of virtual forms. This approach can be used to introduce Data Providers and Web Services.

5 Experiences Learned and Future Works

Without using incremental projects, students can only listen to what the instructor presents and absorb as much as possible. Passive learning becomes a barrier for achieving more effective learning and teaching. With the help of incremental projects, students begin to become motivated on finding alternative solutions, asking questions. Most noticeably, students are eager to learn more about the course subjects. The experiences are reflected in the course evaluation taken later. In the future, more incremental projects will be designed for the purpose of assessing the effectiveness of the active learning.

References

- [1] JavaServer Faces Technology, <http://java.sun.com/javaee/javaserverfaces/index.jsp> (accessed in October 2006).
- [2] Java 2 Enterprise Edition, <http://java.sun.com/javaee/> (accessed in October 2006).
- [3] Microsoft .NET, <http://www.microsoft.com/net/default.mspx> (accessed in October 2006).
- [4] David Geary and Cay Horstmann, Core JavaServer Faces, Sun Microsystems Press (2004).
- [5] Gail Anderson and Paul Anderson, Java Studio Creator Field Guide, Sun Microsystems Press (2006).