A Model-Controller Interface for Struts-Based Web Applications

A Writing Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Computer Science

By

Deepti Bhardwaj

Spring 2010

ABSTRACT

A Model-Controller Interface for Struts-Based Web Applications

by Deepti Bhardwaj

There are a number of Integrated Development Environments (IDE) available which provide environments to develop Struts applications for example JDeveloper and Eclipse. However, these IDEs are desktop-based applications. Moreover, programmers have to ensure that their favorite IDEs are installed and properly configured on their computers before they are able to start working, which takes a substantial amount of time. Even running a properly configured IDE takes a long time to load.

In the proposed project, I will develop a web-based application which will act as an IDE for interfacing and integrating the model and controller of MVC architecture based applications. The proposed IDE will maximize the programmer productivity and will be available across the network. A web-based IDE will eliminate the inconvenience of installing or configuring the desktop-based IDEs. The most exciting advantage of a web-based IDE is that, with the new generation of smart phones and PDAs that support AJAX and Java, programming jobs can be done from any remote location and at any time. Developers will be able to map user requests to an appropriate action which implements the model in the controller by few clicks and drag-drop functionality in web-based IDE.

Table of Contents

1. Introduction

The goal of this project is to develop a web-based Integrated Development Environment (IDE) for MVC architecture based applications. The client only needs a web browser and an internet connection. By connecting to the server, an IDE is downloaded by the browser. The client is able to access previous code stored in the database or create new projects. The proposed web-based IDE is a Struts application. It provides an easy and convenient way to create model component for a web application. This IDE also helps the user to interface and integrate the model and controller components of a web application.

Using this application, user does not need to have a good knowledge of SQL queries for the creation of relational databases. The user should be able to create database table with drag and drop mechanism. The IDE runs the queries in background to create the exact structure on the database server. The proposed IDE also provides a functionality to drag a model object and drop onto one of the controller methods for interfacing. This will save the time of the user to write the code for connecting the controller and model components.

This report provides detailed information about the work done in CS297, Preparation for Writing Project or Thesis course. As this project is build using Struts2.0 and jQuery (JavaScript framework), this report gives a short introduction of Struts2.0 and jQuery frameworks. It also explains all the deliverables that were prepared for the different experiments and studies. These deliverables helped to understand the features or functionalities provided by the frameworks which can be used for various component developments in the project.

1.1. Struts 2.0 Framework

Struts provides a framework for servlet and JSP applications that uses Model-View-Controller (MVC) architecture where Model is business logic or business processes such as Java Beans, View is dialogs such as Java Server Pages and Controller is central control unit such as Java Servlets [5].

1.2. jQuery Framework

jQuery provides capabilities to create plugins on top of the JavaScript library. jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. This contributes to the creation of powerful and dynamic web pages [6].

2. Overview of Deliverables

2.1. Deliverable 1: File Creation Application

2.1.1. Description

In this deliverable, I developed an application for creating files on the server and storing an entry into the database table. The application was developed using Struts2.0 framework which follows MVC architecture. MySQL database is used on the backend to store the file details. The goal of this deliverable was to implement a simple Struts-based application and to understand Struts framework.

It provides an interface that takes file name and file type as inputs for creating files. On submit, the application validates the inputs and in case of failure, displays the error on the same page. In case of valid inputs, it creates a blank file with the name and type provided by the user on the server. The application also inserts the information of the file created into the database table for future reference.

2.1.2. Implementation

In the View component of the application, Struts 2.0 tags are used to create the form elements [7]. On submit an Ajax call is made for an action class which interacts with the model component to create the file. The action class returns ERROR in case of validation error otherwise SUCCESS. The following are the main modules of framework:

View (*CreateFile.jsp*): Displays the form to the user.
Action Class (*createFile.java*): Initialize the Model object and returns SUCCESS or ERROR.
Model (*createFileModel.java*): Contains method to create and save the file on server.

Struts configuration (*Struts.xml*): Contains all the action class mappings.

The application is deployed on Apache Tomcat application server. A database table *FILES* with fields id and filename is created on MySQL database server. The *struts2-core-2.0.6.jar* file is included inside the Lib directory of the application for Struts2.0 libraries. A model java class will connect to the MySQL database running on local system using *mysql-connector-java-5.1.10-bin.jar* connector file.

On request, the server lookups the *createForm* action mapping in *struts.xml* and invokes *CreateFile.jsp* servlet. The input form page is displayed on the client browser. On submit, an Ajax call is sent and based on the mapping in *struts.xml*, *execute()* method of the action class *createFile.java* is called. Based on what the execute method returns, framework determines which page needs to be called from the mapping in *struts.xml*. In case of empty string for file name, "*Invalid file name*" error message is displayed on the screen. For not selecting file type from the drop down "*Invalid file type*" error message is shown to the user. When both the inputs file name and file type are valid inputs, a file is created on server and a success message for example "*File Created Test.jsp*" is displayed.

To develop the application one of the challenge was to understand Struts2.0 framework and how to make Ajax call using Struts library. Struts tag for submit button with *Ajax* theme is used to make an asynchronous call to the server.

2.1.3. Screenshots

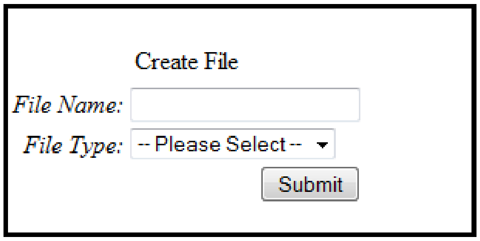
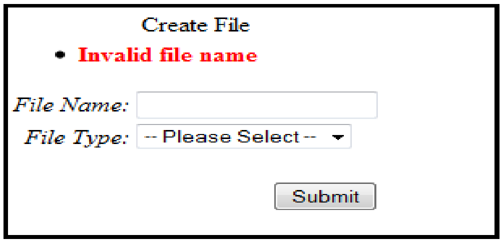
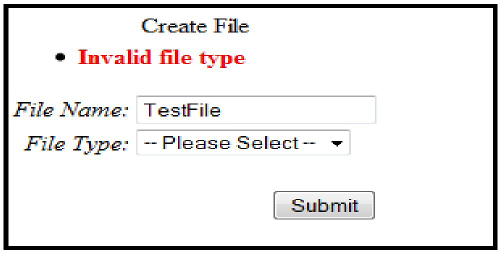

Create File

Fig 1. Input form for user's input.


Create File

- Invalid file name

Fig 2. On submit without file name input, an error message is displayed on the same page.



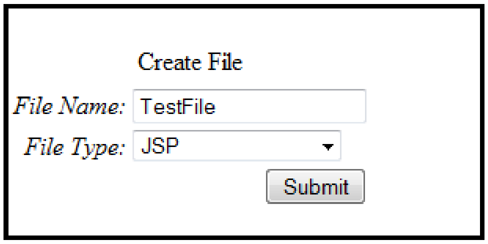
Create File

- **Invalid file type**

File Name: TestFile

File Type: -- Please Select --

Fig 3. On submit without selecting file's type, an error message is displayed on the same page.



Create File

File Name: TestFile

File Type: JSP

Fig 4. For valid inputs, success message is displayed and file is created on the server.

2.2. Deliverable 2: Study design architecture of the web-based IDE "Aurorasdk"

2.2.1. Description

In this deliverable, I studied architecture of the web-based IDE "Aurarasdk" [3]. Aurarasdk is an IDE for developing Java based applications. The web application is mainly implemented using Google Web Toolkit (GWT), Servlet and MySQL. This project was done under the guidance of Professor Gail C. Murphy at University of British Columbia. Aurarasdk supports full syntax highlighting and compilation of Java source codes with different plug-in compilers. Two compilers are implemented in Aurora SDK: Eclipse and JDK. It also supports auto completion and provides an environment to run the compiled code.

The Aurorasdk IDE refers to the architecture of Eclipse IDE. The architecture used for the IDE has four layers which makes the design flexible. In Application layer, GWT widgets are built to develop UI. In Service layer, GWT Remote Service is used to communicate between different layers. Two compilers, Eclipses and JDK, are implemented in Compiler layer. Aurorasdk uses MySQL database as data store which is in Persistence layer.

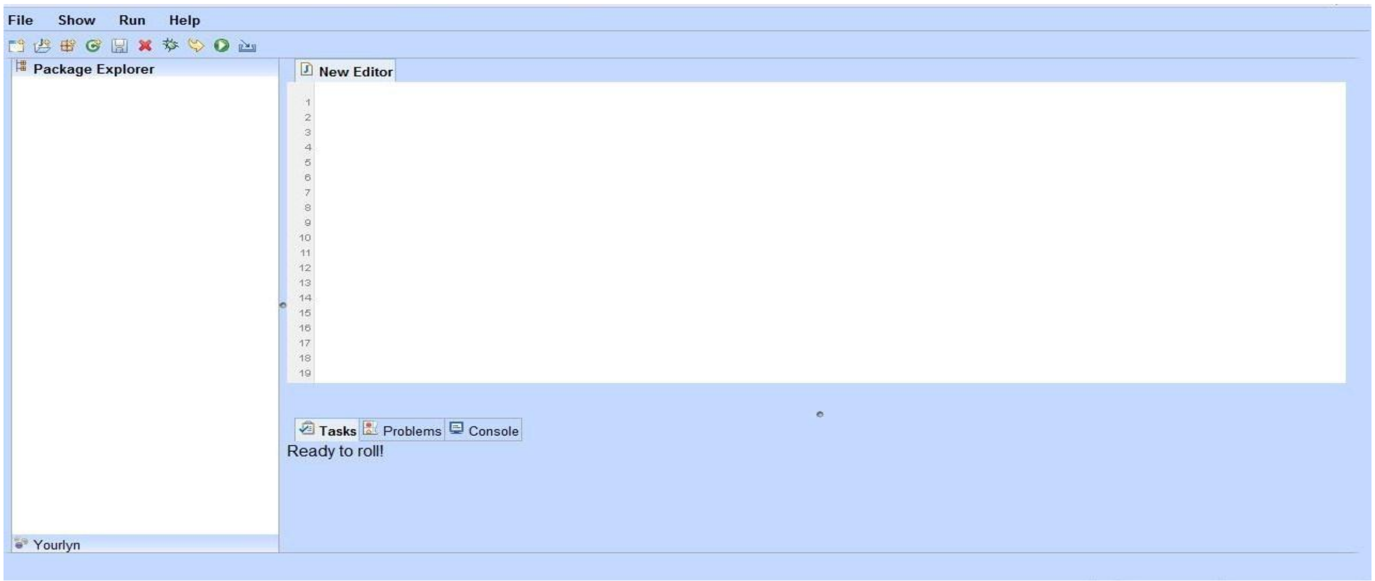
In future releases, the team has planned to enhance the IDE with new features like allow multiple users to work on a single project; to provide a functionality to debug the code; and plug-in more compilers other then Eclipse and JDK. Currently, the compilation feature of Aurorasdk breaks and generates incorrect compiled code when two users compile the files at the same time. The Aurorasdk team is working on fixing this issue.

2.2.2. Study observations and Issues faced

In order to experience working of Aurarasdk IDE, I downloaded the war file from the Aurorasdk official web page [3]. Than deployed the .war file of the application on Apache Tomcat. For database server I used MySQL5.0, as suggested by Aurarasdk team, and created application tables.

To run the application, start the application server and invoke the Aurorasdk index page on web browser. The home page is displayed with the menu bar with options to create new file, save file, etc. It has expandable and collapsible frames for package explorer, editor and tasks. I observed that there is no documented instruction provided for connecting the database and application files. As the application stores the details of the projects into database table, new project creation functionality of the Aurorasdk IDE did not work. The reason was the failure to establish connection between application and database server. Also the application editor does not work properly in Internet Explorer web browser.

2.2.3. Screenshots



Ready to roll!

Fig 5: Aurorasdk home page.

2.3. Deliverable 3: Application to create Model

2.3.1. Description

The goal of this deliverable is to learn jQuery and develop an application. In this deliverable, I developed an application for creating model components for Struts based applications. It helped to visualize the implementation of the IDE component to create models. This application is developed using XHTML elements and jQuery framework. It also provides functionality to perform different operations on the model object. I used animation libraries to build animations such as drag and drop, drawing lines between tables, and a right click pop-up menu.

The application has a menu bar to create a new table. It allows the user to add fields to the table and create relationships between the tables to build a model component. The user can also delete the table and can rename it. A right clicks on table object, pop-up a menu list of various operations that can be performed on the object. It provides an option to drag the table objects within the main frame.

In addition to above, the application provides the functionalities to create relationships between tables. In this application, the two types of relationships are supported: "has a", Composition, relationship and "Is a", Sub-class, relationship. In database design and object oriented program architecture, "has-a" is a relationship where one object is a part another object. "Is a" is a relationship where one class is a subclass of another class. These relationships are used to connect different tables in order to create a Model object.

2.3.2. Implementation

The application uses jQuery framework for drag and drop animations. It includes *"wz_jsgraphics.js"* JavaScript library available online for drawing lines and different type of shapes. To run the application "jquery-ui.min.js" and "wz_jsgraphics.js" JavaScript files are included in the application folder.

On requesting for index page, the browser renders the HTML code and displays the index page. The web page with menu on the top allows the user to create new table. When user clicks "New Table" link a new table is created with default name.

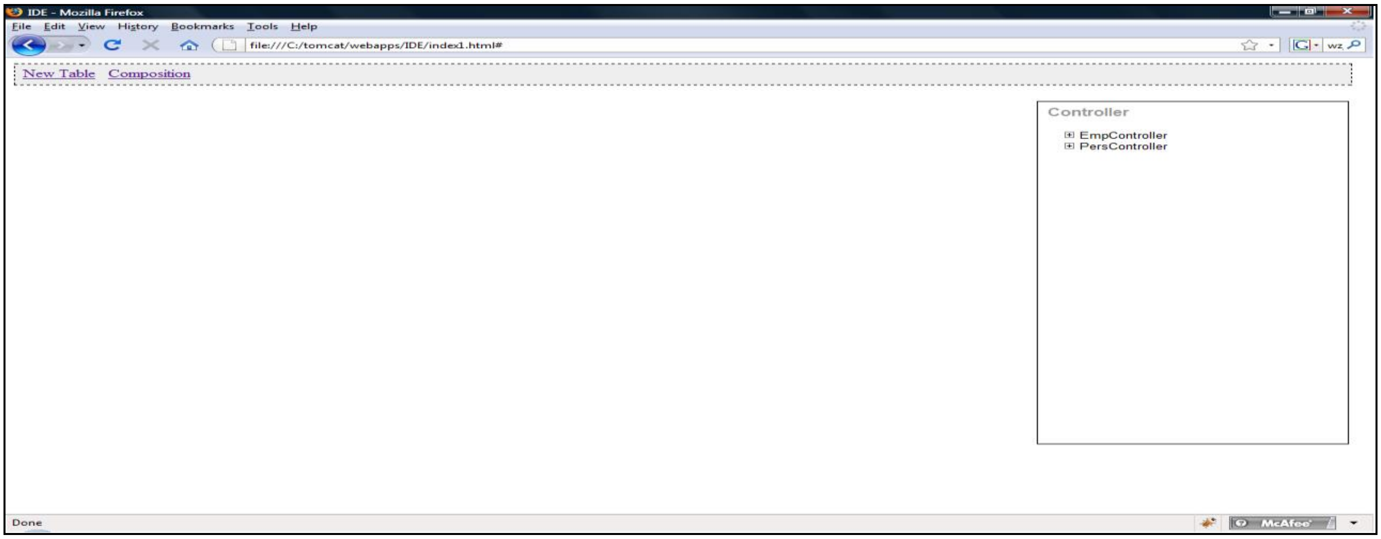


Fig 6: Home page with menu bar

A menu list is displayed on right click on the table created. The user can add new fields into the table by clicking "Add Field" option from the pop-up menu. The "Delete" option is used to delete a particular table along with its fields. To rename the default table name, click on "Edit" and a textbox is displayed. After entering a new name, press enter and table renames to the value entered.

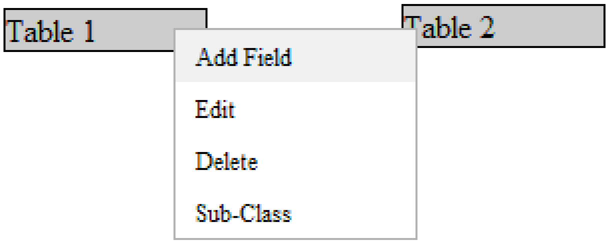


Fig 7: On right click; a menu is displayed with the list of actions.

This application takes advantage of jQuery features to provide the dragging of the table object. The table object created has a draggable ( ) method which is being called when user drag the table. It also has a droppable ( ) method which is being called when a field is being dropped on it. The fields added to the table are draggable fields. They can be dragged and dropped onto another table and fields of another table. When a field is dragged a clone of that field is generated and dragged instead of the original object.

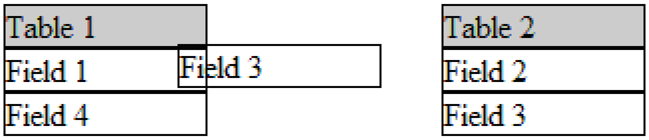


Fig 8: Drag and drop a clone of field from one table to another to connect two tables.

The code snippet below shows the draggable and droppable methods of a field object. It creates a clone of the object dragged using *helper* function and when dropped calls *drawLineBetweenBox ()* function to draw the selected relationship.

```
$('.attdiv').draggable ({helper:'clone'}).droppable ({

        drop: function (ui, event)     {
        var source = $(event.draggable).clone ().attr ("id");
        var dest = $(this).attr ("id");
        drawLineBetweenBox (source, dest);
      }
});
```

Creating relationships between tables was a challenging task. First, I tried with jpeg images for rectangle and triangle on the tip of the lines to display the relationships. But the angle for displaying shapes changes for every drag action on the connected tables. To avoid this problem, *wz_jsgraphics.js* online library is used to display shapes. I used function *drawLine ()* to draw lines and *drawPolygon ()* to draw diamond and triangular shapes to represent different relationships. Click on Composition link provided in menu bar and drag a field from one table to another will create a composition relationship.

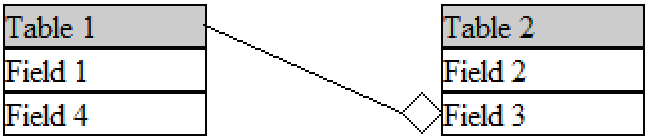


Fig 9: Composition or "has a" relationship denoted by line with a diamond shape on tip.

On right click popup menu, the option "Sub-Class" creates a new table which is a subclass of the existing one. It represents the sub-class relationship with a line and triangle pointing to the base or parent class.

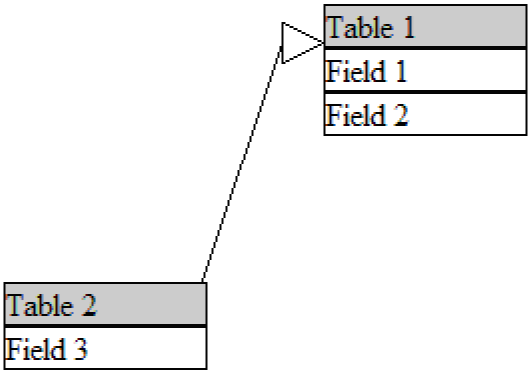


Fig 10: "Is a" relationship, Table 2 is a sub-class of Table 1.

2.4. Deliverable 4: Application to interface Model and Controller components.

2.4.1. Description

In Deliverable 3, the user can develop model objects by creating tables and relationships between them. Therefore in Deliverable 4, I have enhanced the application developed in Deliverable 3 to provide additional functionality. The targeted functionality for this deliverable is to connect Model and Controller components. This interfacing is highly important for the MVC based applications.

The Controller list is available to the user along with the all method names. The user has to create a Model object and decide to which Controller method it should be connected. Just dragging and dropping the Model object connects Model object with the Controller. The user can drag the Model object and drop onto the Controller method. This user action opens up a menu list at the Controller panel with all the method names. After the model object is dropped onto a Controller, the object name is added to the list of methods of that Controller.

In future work, I will implement logic to allow the user to drop Model object onto a particular method from the menu list. This user action will connect the Model object with the desired Controller method.

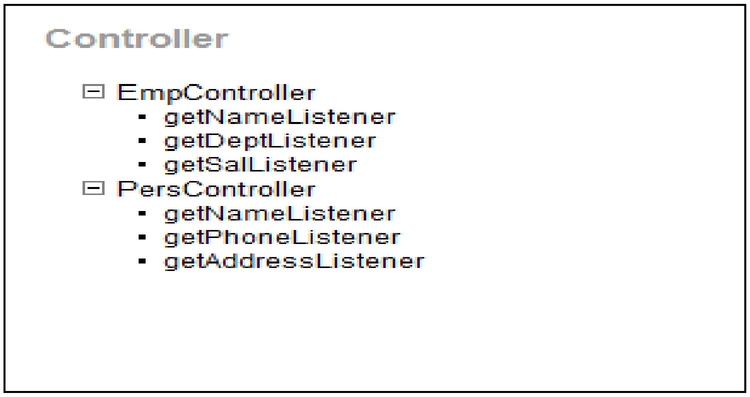
2.4.2. Implementation

The tree menu of the Controllers and their methods is implemented using CSS styles. The images are used for expand and collapse icons. jQuery library methods are used for drag and drop functionality. I included an online available JavaScript library to implement expand and collapse functionality.

On page load, a list of existing Controllers is displayed to the user on the right hand panel. Initially, the list is in collapsed state which means it just displays the names of the Controllers. When the user clicks on the expand icon, expandTree () method is called which takes element id as input. A call to this method expands the list and displays all the method names of the Controller file. On clicking collapse icon, it calls collapseTree () method to collapse the list.

The Model objects have draggable property which allows the user to drag them and the Controller list items have droppable property attached to them. When a Model object is dragged and dropped on a particular Controller name, droppable method calls another method to open up a menu list with all the method names inside that Controller. All the Model objects and controller elements have unique id associated which is used to perform various operations on them. The list of Controllers and methods are included on the code and in future work, this list will be fetched from a XML file saved on the server.

2.4.3. Screenshots



Controller

⊟ EmpController
- getNameListener
- getDeptListener
- getSalListener

⊟ PersController
- getNameListener
- getPhoneListener
- getAddressListener

Fig 11: Collapsible and Expandable list of Controllers and Methods.

3. Conclusion

This semester, I have developed an application for creating Model objects and for interfacing Model and Controller components. This application provided me with a good knowledge of jQuery which I will use in CS 298 for GUI animations. It also helped to visualize the project and to understand the technical challenges involved in it. The deliverables that I worked in CS297 are the foundation for the development of my final thesis project.

I will be implementing the Web-based IDE for Struts based application during Fall 2010 semester. In CS 298, JavaScript created for Deliverable 3 and Deliverable 4 during CS 297 will be used for drag and drop functionality. As the architecture of the IDE will be based on MVC pattern, Deliverable 1 provided a base knowledge to create web application using Struts framework. Struts validator framework provides robust functionality to validate user inputs. In Deliverable 1, I used validator.xml of Struts validator framework for validating file name and file type inputs. This experiment will be helpful in implementing the validator framework in CS 298.

The study of web-based IDE Aurorasdk provided a basic knowledge to design the architecture of an IDE. Using my observations from the study, I designed the database tables for my CS 298 thesis project implementation of web-based IDE. With the deliverables produced in CS 297, I will be able to implement the web-based IDE for Model and Controller interfacing.

4. References

1. [Dionysios G. Synodinos] Web-based IDEs to become mainstream. Retrieved from - http://www.infoq.com/news/2009/02/web-based-ide

2. [Heroku] Heroku Documentation. Retrieved from - http://docs.heroku.com/

3. [Jingwen Ou] Aurorasdk. Retrieved from - http://code.google.com/p/aurorasdk/

4. [Jacob T. Biehl] FASTDash: A Visual Dashboard for Fostering Awareness in Software Teams. Retrieved from - http://research.microsoft.com/pubs/64290/chi2007-fastdash.pdf

5. Struts. Retrieved from - http://www.ibm.com/developerworks/library/j-struts/

6. JQuery. Retrieved from - http://jquery.com/

7. Struts Tags. Retrieved from - http://struts.apache.org/2.0.14/docs/tag-reference.html

8. Walterzorn. Retrieved from - http://www.walterzorn/jsgraphics/jsgraphics_e.htm

9. Bespin. Retrieved from - https://mozillalabs.com/bespin