DPROJECT

Project Management Tool

DEVELOPER'S MANUAL

Prepared by *D&D Software*:
Dogan Yazar
Firat Alpergin
Tuncay Namli
Mehmet Remzi Dogar

INTRODUCTION

DProject is a web based project management tool including many facilities for possible users. Not only can it be used for a single company to manage their multiple projects, it can also provide the opportunity to serve multi companies from a single DProject server. DProject enables the company managers to manage their different projects from a single application. They can monitor all their projects as a whole; also they can get into the detail of their projects. One of the most important facilities for DProject is the creation of tasks for projects, assigning them to other employees and monitoring these tasks. Another interface that DProject offers to project managers is the automated meeting arrangement. Both the tasks and meetings interfaces enable users to easily share their documents and information. The other main component for our project is management of resources that will be used in projects and tasks of these projects. The other facilities of the project mostly help the project managers and employees to monitor these main components related activities with the help of graphical and statistical data. These are Gantt chart construction, Critical Path and Dependency Chart, Calendar for employees to monitor their activities easily and Notification System to notify the users from related activities. There are some other opportunities for company administrators to track the system.

After briefly describing some facilities of the DProject, now we will present the system architecture and other software related issues to help the software developers who want to develop our product. Firstly we give the main system architecture; user login process, company login process and construction of the main page. In addition to system architecture, first part also will include the general software design and issue for all components for our system. Then we give the architecture of our components one by one. Finally we will give some development suggestions on some components.

GENERAL SYSTEM ARCHITECTURE

We use Model View Controller (MVC) in our architecture. As MVC suggests we divide our control, view and data parts and use java servlet, jsp and java scripts and java beans respectively for these parts. In this architecture all redirection between pages are controlled by the servlets. JSP pages only show the data which they get from the java bean classes. As a result for every page we want to show we have a jsp file to show the page and a java bean classes to store the data for that page. However, generally we combine some pages as module and have a single controller (servlet) for that module.

During the session starts with the user login we must store some general necessary data about user and company. We use a root class 'Session.java' to store all this global data. By the user login this class is initialized and stored in the HTMLSession as an attribute to reach it until the user logout. Below is the diagram of the class Session and its subclasses to store the global data for the http session.

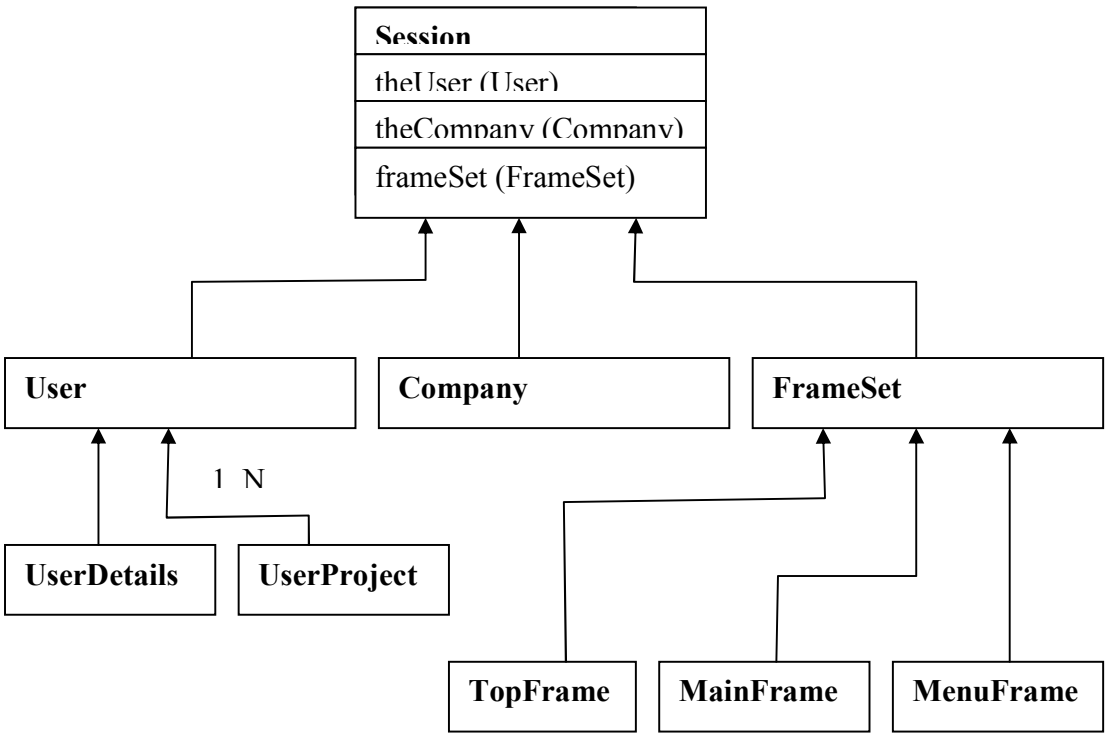

Figure 1

As seen from the figure we have some global data about the user, company. The User class stores the data for the user of the session to access easily when needed. It has a collection of 'UserProject' objects that holds the project ids and some other information

about the projects that the current user is assigned to. Company class holds the company information to be used in some pages. The ‘FrameSet’ class stores the information about the structure of the main page and its three frames as you will see when you login to the system. ‘TopFrame’ includes the name of tabs and other information. ‘MenuFrame’ holds menu options about selected tab and ‘MainFrame’ presents the content. These details and initializations are illustrated in the next section the login process for a user.

Company Creation

As described before, DProject enables multiple companies to register to the system. In our system we create a new database for each company. By this we prevent possible security issues and ease the management of database. We have a link from ‘login.jsp’ to ‘LoginServlet’ which controls the login procedures for both company creation and user login. The servlet will redirect the ‘companyLogin.jsp’ which shows the interface to create the company. After providing the data and submitting them the page is redirected to ‘LoginServlet’. The servlet gets the information, initializes a ‘Company’ object and calls the ‘createDB’ of the method of the object. This method first reads the ‘initialization.xml’ to get the database information for database connection. Connection related issues will be explained in the next section. After establishing connection, the method reads the sql file ‘db_create_1nisan.txt’ and executes the sql statements to create the database named with company id and tables of the database. Also a user is created with user id ‘admin’. After creation of database, servlet passes to the ‘login.jsp’ again.

User Login Process

We describe the initialization of the ‘Session’ object and construction of frames in addition to user login process. When the user enters the company id, user id and password submits the page, the method ‘processRequest’ of LoginServlet is called. This method differentiates the company creation and user login process and in user login case calls its ‘handleUserLogin’ method. This method takes the company id and establishes a connection with the static method of ‘MakeConnection’ with this id used as database name.

‘MakeConnection’ class includes java.sql.Connection object. It has two constructors that initialize the database related data (url, user name, password for database). The first constructor gets only url of the ‘initialization.xml’ to get the these data and it is used for database creation since the database name is unnecessary. Other constructor gets the database name (company id) with this xml and used in user login process. The ‘MakeConnection’ class also has the ‘getConnection’ method which establish a connection if the connection is closed. If already a connection is open, the method returns that connection.

After initializing the static ‘MakeConnection’ class, the servlet calls its ‘verify’ method to check the user id and password of the user. If the user is authorized to access the application, the method returns true and process continues. Otherwise, the servlet redirects to page ‘login.jsp’ with authentication failure message. After authentication, the

'init' method of 'Session' object is called and the initialization process starts. Recursively 'init' methods of all objects in 'Session' class are called. These 'init' methods get the required data from database and initialize the objects. FrameSet object includes the objects about the frames in the main page. In initialization, these objects are initialized with the initial configuration. For example, the 'MyOffice' tab is selected and its menu is set.

After the initialization, the servlet forwards the page 'BasicFrameSet.jsp' which is the main page structure. It is divided to three frames. At the top we have 'topFrame.jsp' which shows the tabs, username, projects combo box to select and DProject logo. The 'menuFrame.jsp' on the left shows the menu items for the selected tab. The 'MainFrame' forwards the page to a servlet which will show the main content. The name of the servlet is obtained from the 'MainFrame' object in 'FrameSet' object.

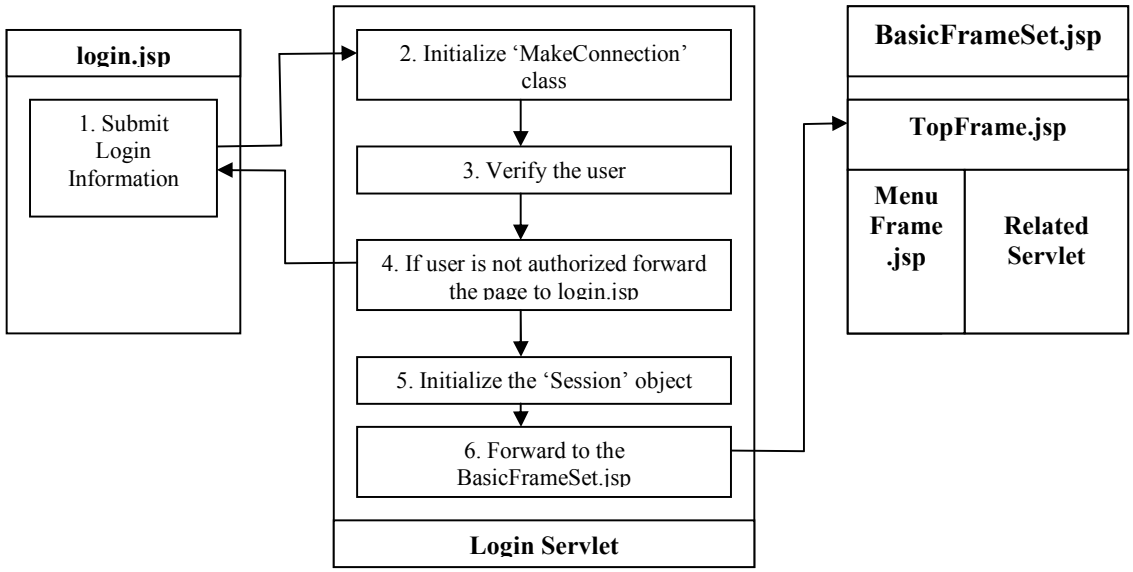

Figure 2 Login Process

Redirections from menu frame and top frame are done by calling the 'BasicFrameServlet'. The 'BasicFrameServlet' makes the necessary changes in the three frame class stored in session and redirects to 'BasicFrame.jsp' again. These changes may include the change of servlet name stored in 'MainFrame.java' and selected tabs. This process is illustrated at Figure 3.

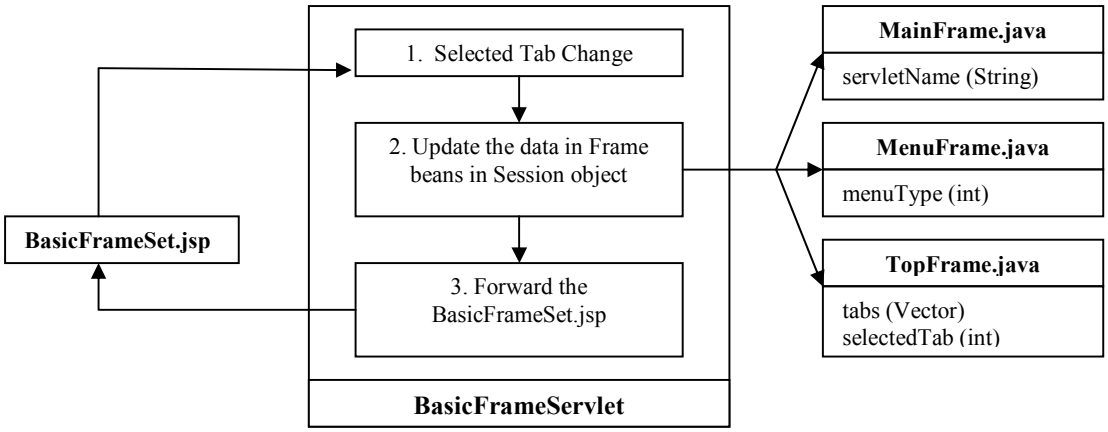

Figure 3 Change In the TopFrame

Other Controllers and Java Beans

In this section we mention about the modules and other controllers for listing the items. For every page which lists some information like listing the user tasks or listing user meetings we have a servlet that controls this listing. Generally when user selects an item from menu frame for such listings, the related servlet is called. This servlet obtain all information from database by its methods and constructs a vector of a bean related with data. Then servlet sends this vector and other information to the jsp page which shows this information. Below are these servlets and the process description.

a) mainFrame_Admin_UsersServlet :
This servlet is used to list the users in the company to administrators. The link to the servlet is from the menu item 'Users' when 'Administrator' tab is selected. When servlet is activated, it gets all users information in company and constructs a vector of 'User' objects. Then it sends this vector to the 'mainframe_Admin_Users.jsp' by setting it to the attribute of 'HttpRequest'.

b) mainFrame_Admin_UsersServlet :
This servlet is used to list all the projects in the company to administrators. The link to the servlet is from the menu item 'Projects' when 'Administrator' tab is selected. When servlet is activated, it gets projects information in company and constructs a vector of 'UserProject' objects which include restricted information about project. Then it sends this vector to the 'mainframe_Admin_Projects.jsp' by setting it to the attribute of 'HttpRequest'.

c) mainFrame_MyOffice_OtherUsersServlet :
This servlet is used to list other users in the company to employees. The link to the servlet is from the menu item 'Other Users' when 'MyOffice' tab is selected. When servlet is activated, it gets users information in company and constructs a vector of 'User' objects which include restricted information about user. Then it sends this vector to the 'mainframe_MyOffice_OtherUsers.jsp' by setting it to the attribute of 'HttpRequest'.

d) mainFrame_MyOffice_MyTasksServlet :
The servlet constructs a vector of objects from 'Task' class. It gets the tasks of current user and constructs the vector. Then it sends this vector to the 'mainframe_MyOffice_MyTasks.jsp' by setting it to the attribute of 'HttpRequest'.

e) mainFrame_MyOffice_MyProjectsServlet :
The servlet gets the projects information of current user and constructs a vector of 'UserProject' objects. Then it sends this vector to the 'mainframe_MyOffice_MyTasks.jsp' by setting it to the attribute of 'HttpRequest'.

f) mainFrame_MyOffice_MyNotificationsServlet :
This servlet is used to list the notifications come to the user and sending notifications to other users. The menu item 'MyNotifications' in the 'MyOffice' tab is linked with the listing purpose. In such case the servlet gets the notifications information from the database and constructs a vector of objects from 'Notification' class. Then it sends this vector as an attribute of 'HttpRequest' to 'mainFrame_MyOffice_MyNotifications.jsp'. All notification items (a row in table) in this page are linked to this servlet with some additional information. One of the information is the 'command' parameter of 'HttpRequest'. By this parameter servlet differentiate the action to be done that comes from the page. For example, all links on the items have 'go' as the value of 'command' attribute. Other action from the page is sending notifications to other employees in the company. Activating this action from page sends the parameter 'command' by value 'send_message' to the servlet. The servlet forward the page to the 'mainFrame_MyOffice_UserMessage.jsp'. When user sends the message from page with the parameters for message information and 'command' parameter with value 'add_message', the servlet constructs a 'Notification' object and call the 'insert' function to save this information to database.

g) mainFrame_MyOffice_MyMeetingsServlet :
The servlet gets the meeting information of current user and constructs a vector of 'UserMeetingDetails' objects. Then it sends this vector to the 'mainframe_MyOffice_MyMeetings.jsp' by setting it to the attribute of 'HttpRequest'.

h) mainFrame_aProject_ProjectUsersServlet :

This servlet is used to list the users who are assigned to projects. The link to the servlet is from the menu item ‘Users’ when the related project tab is selected. When servlet is activated, it gets users information and constructs a vector of ‘User’ objects. Then it sends this vector to the ‘mainframe_aProject _ProjectsUsers.jsp’ by setting it to the attribute of ‘HttpRequest’.

i) mainFrame_aProject_ProjectTasksServlet :
The servlet is used to list the tasks related with the selected project. The link to the servlet is from the menu item ‘Tasks’ when the related project tab is selected. When servlet is activated, it gets tasks information and constructs a vector of ‘Task’ objects. Then it sends this vector to the ‘mainframe_aProject _ProjectsTasks.jsp’ by setting it to the attribute of ‘HttpRequest’.

j) mainFrame_aProject_ProjectMeetingsServlet:
The servlet is used to list the meetings of the selected project. The link to the servlet is from the menu item ‘Meetings’ when the related project tab is selected. When servlet is activated, it gets meetings information and constructs a vector of ‘UserMeetingDetails’ objects. Then it sends this vector to the ‘mainframe_aProject _ProjectsMeetings.jsp’ by setting it to the attribute of ‘HttpRequest’.

After describing some servlets, now we describe the general structures of our java bean classes. Generally, all bean classes have ‘init’, ‘insert’, ’update’ and ‘delete’ commands which are for the database operations. The ‘init’ method gets the class related information from database and initializes the attributes of that object. The ‘insert’ method inserts the data in the object to the related tables in the database. The other methods ‘update’ and ‘delete’ updates and deletes the related entries in the database.

MODULES

Task Module

The controller of the module is the 'EditTaskServlet' servlet. The links to this servlet can be from the pages that list the tasks as 'Create New Task' link and from the items of these to update that task. This module handles all actions from the pages;

- mainFrame_Task_GeneralInformation.jsp
- mainFrame_Task_AssignedUsers.jsp
- mainFrame_Task_Assignation.jsp
- mainFrame_Task_AttachedFiles.jsp
- mainFrame_Task_Dependency.jsp
- mainFrame_Task_Resources.jsp
- mainFrame_Task_TaskNotes.jsp
- mainFrame_Task_aDependency.jsp
- mainFrame_Task_aResource.jsp

The first page that is shown to user is the 'mainFrame_Task_GeneralInformation.jsp'. Others are reached from the links that are on the right menu on all these pages. All links are linked to 'EditTaskServlet'. Actions from these different pages are differentiated with the parameter 'command' which comes with the http request. When the request comes to the servlet, the servlet first gets the task id, project id, user id and company id from the data in the object of 'Session' saved as an attribute of HttpSession. According to the command, the servlet gets other necessary information specific to command from the parameters of request which comes from jsp pages and it performs the necessary actions from this information. Then it puts the necessary information to the attributes of http request and forwards the page to related jsp page.

Project Module

The controller of the module is the 'EditProjectServlet' servlet. The links to this servlet can be from the pages that list the projects as 'Create New Project' link and from the items of these to update that project. This module handles all actions from the pages;

- mainFrame_Admin_Project_Assignation.jsp
- mainFrame_Admin_Project_GeneralInformation.jsp
- mainFrame_Admin_Project_ProjectPhases.jsp
- mainFrame_Admin_Project_TaskPriorities.jsp
- mainFrame_Admin_Project_TaskStatuses.jsp
- mainFrame_Admin_Project_TaskTypes.jsp
- mainFrame_Admin_Project_aPhase
- mainFrame_Admin_Project_aPriorityType
- mainFrame_Admin_Project_aStatusType
- mainFrame_Admin_Project_aTaskType
- mainFrame_Admin_Project_aUser

The sertvlet handles the actions as in the Task module. Six pages from above are linked with the right menu items shown in these pages. The others are linked from these pages and used to create related item or update the selected one from the list. For example 'mainFrame_Admin_Project_Assignation.jsp' includes the assigned users to project. This page includes a 'Assign New User' link when pressed the servlet is called with command value 'assign_new_user' and servlet will forward the page 'mainFrame_Admin_Project_aUser' which handles the assigning new user action after the necessary actions.

User Module

The controller of the module is the 'EditUserServlet' servlet. The links to this servlet can be from the pages that list the users as 'Create New User' link and from the items of these to update that user. This module handles all actions from the pages;

- mainFrame_Admin_User_Assignation
- mainFrame_Admin_User_GeneralInformation
- mainFrame_Admin_User_aProject

Handling the actions from pages is same as the other modules. This module is used to create new users and assigning them to existing project.

Meeting Module

The controller of the module is the 'EditMeetingServlet' servlet. The links to this servlet can be from the pages that list the meetings as 'Create New Meeting' link and from the items of these to update that meeting. This module handles all actions from the pages;

- mainFrame_Meeting_GeneralInformation
- mainFrame_Meeting_Assignation
- mainFrame_Meeting_AssignedUsers
- mainFrame_Meeting_AttachedFiles
- mainFrame_Meeting_MeetingNotes

This servlet control the actions of creating new servlets, updating them and other related issues like assigning users to meeting. 'Meeting', 'MeetingFile', 'MeetingNote' and 'UserMeeting' are bean clases for these pages. Meeting class includes the methods and algorithms for automation of meeting arrangement and all database related activities to provide these services.

Resource Module

The controller of the module is the ‘mainFrame_aProject_ProjectResourcesServlet’ servlet. This module handles all actions from the pages;

- mainFrame_aProject_ProjectResources.jsp
- mainFrame_aProject_aProjectResource.jsp

The servlet control the actions of creating resources for the project and listing the resources of project.

Gantt Charts Module

This module includes two servlet; ‘mainFrame_aProject_GanttChartServlet’ for the normal gantt chart and ‘mainFrame_aProject_DetailedGanttChartServlet’ for dependency chart. Handling of actions in these servlets is same with other modules. For charts we use an API named ChartDirector that constructs chart images from some data. The servlets construct the data to the pages ‘mainFrame_aProject_DetailedGanttChart.jsp’ and ‘mainFrame_aProject_GanttChart.jsp’ and by using of API the pages construct the images and show them with some other content to manage the actions.

Statistics Module

The controller of the module is the ‘statisticsServlet’ servlet. This module handles the page ‘mainFrame_aProject_Statistic.jsp’ which shows the general statistic about the project. ‘GeneralStats’,’UserStats’, ‘TaskPriorityStats’, ‘TaskStatusStats’ and ‘TaskTypeStats’ are the java beans for the page to store the data.

Time Report Module

The controller of the module is the ‘EditTimeReportServlet’ servlet. This module handles the pages ‘mainFrame_TimeReport.jsp’ and ‘mainFrame_ShowReport.jsp’ which shows the working times of the users for selected tasks.