**SUMMER INTERNSHIP PROJECT-REPORT**

TRAINER’S TOOLKIT

**by**

**Tushar Limaye**

**Ankur Garg**

**Feroz Khan**

**Sabiha Chogle**

**Gayatri Dwiwedi**

**Saurabh Goyal**

Supervised by:

**Mr. Mukund Mishra,**

Technical Training

Submitted to the Department

of

**Technical Training**



**Zensar Technologies**

**Plot No.4,Zensar Knowledge Park, Kharadi, MIDC, Off Nagar Road, Pune, Maharashtra, 411 014 020 6607 4000 (2012)**

#### CERTIFICATE

This is to certify that the Project entitled

TRAINER’S TOOLKIT

Submitted by:

|  |  |
| --- | --- |
| **Name** |  |
| Tushar Limaye |  |
| Ankur Garg  Saurabh Goyal  Feroz Khan  Gayatri Dwivedi  Sabiha Chogle |  |
|  |  |

for the partial fulfilment of the requirements of the course **Summer Internship Training.**

**Signature of Supervisor**

***(Designation)***

**Date:----------------------**

**ACKNOWLEDGEMENT**

Words are often too weak to express our feelings of indebtedness to ones benefactors. The same difficulty haunts to me in penning down the deep sense of gratitude.

We take this opportunity to thank the people at ZENSAR TECHNOLOGIES, who so generously contributed to our requirements and gathering work with their unfailing support. We appreciate the help from Mr. Mukund Mishra, who provided all required resources and made arrangements that every trainee needs from his/her guide . We are grateful to Mr.Prem Apte, Head of Technical Training, who provided many helpful suggestions and who had made this project possible by his support on technical points during the critical phases of this project. We humbly acknowledge our profound debt of gratitude to Mr.Robert Pillai and Mr.Pinkey Koley whose technical support in the field of IT helped us facilitating our work from time to time.

SUBMITTED BY-

Tushar Limaye

Ankur Garg

Saurabh Goyal

Feroz Khan

Gayatri Dwivedi

Sabiha Chogle

**TABLE OF CONTENTS**

**Topics PageNo.**

Objectives 4

Explanation of Project 5

Function Specifications 8

Data Flow Diagram 18

Source Code 21

Conclusion 215

## Objective:

The tool “Trainer’s Toolkit” is especially designed for trainers with which they can develop any training module. Its simplicity defines its usability.Trainers toolkit is provided with the help window and complete documentation which makes it very easy to use this tool.

Every year thousands of employees join I.T companies and these companies spend crores of rupees on their training. In addition to that they have to make arrangements for Classrooms, Systems and all necessity. This tool can help these companies to minimize this investment up to a great extent. It saves lot of hard labor also. Trainer can easily generate training module on per day basis and he can upload it on server or web. Now every trainee can download these module and can learn through it from anywhere and anytime he wants providing much more flexibility and learning process can be done in mass saving lot of time also.

There are lot of websites in the market which provides online training module(ex: w3schools.com) and compilation facility(ex:ideone.com) but there is no as such tool in the market with which we can develop these training modules that why Trainers Toolkit came into emergence. Module generated contains all the information about the subject and compilation facility where he can add, edit the code and see the output at the same instance.in addition to that trainer can also upload audio, video, images into that. It also includes email-id of trainer and some references where end user can browse for more information.

Modules generated maintain its uniqueness with a unique-id. Till now it is been restricted to develop training module for java but it can be extended for other platforms also like Python, .net, SQL by integrating their compiler.

## Explanation for the Project

## Introduction

* The Project – Trainer’s Toolkit is designed for the trainer’s with which they can create any training module to make the training programme easy to conduct for the trainer and to make the learning process easy for the trainee.
* It provides flexibility for the trainer to add media contents to its module and thus making it easy and efficient training tool.
* The output presented at the user’s end will be in HTML format.

## Features

* The trainer can add text, media files such as images, audio and videos in the module.
* The toolkit provides facility of selecting the compiler and integrating it with the training module.
* The training module includes quiz and a test engine for end user’s evaluation
* Random questions with their weightage and deadline time will appear on the screen.
* Once the test is completed it will give u a score card along with the remarks.
* SMTP mail server will be there with which score card will automatically be send to the mail address specified in the form filled before starting the test.
* The trainer can anytime upgrade any existing training module using this toolkit.
* The trainer can preview user’s view of the pages that will be displayed.
* The training module at last will be produced in HTML format.

## Module Details Page

* Module Id is the id generated by the toolkit randomly.
* Module Subject: It is provided by the trainer for which he/she is interested to create a training module.
* Author Name: Trainer has to enter his/her name.
* Select Complier: The trainer can select which compiler he/she wants to work upon.
* Next Button: Navigates you to the next Page.

## Table of Contents Page

* Module Subject: This is the subject carried forward from the previous page entered by the trainer.
* Topic Name: It’s the name of the topic.
* Add Topic Button: It adds the topic name in the list below which was provided by the trainer in Topic Name text field.
* Delete Button: This deletes the topic added in the list.
* Rename Button: This renames the topic name added in the list in a pop up window.
* List of topics: It is the list of all the topics provided by the trainer.
* ‘/\’, ‘\/’ button: This helps in rearranging the topics in the list according to the trainer.
* Topic Description: On selecting the topic, the trainer can enter a brief description in text area about the topics enlisted.
* Module Details: Takes you to the previous page where trainer may re-enter the module-id, etc.
* Add Content: On clicking this button you will be navigated to the next page where sub-topics and the description can be entered.
* The /\ and \/ buttons are for rearranging the order of topics.

## Add Content Page

* Module Name: This is the name carried forward from the previous page entered by the trainer.
* Topic Name: This is the name carried forward from the previous page entered by the trainer.
* Sub-Topic Name: The trainer has to enter the name of the sub topic.
* Add Subtopic Button- This button helps to add the topic in the list below which the trainer has entered in text field.
* List of sub-topics: It is the list of the all the sub topics added by the trainer.
* ‘/\’, ‘\/’ button: This helps in rearranging the subtopics in the list according to the trainer.
* Delete Button: This deletes the subtopic added in the list.
* Rename Button: This renames the subtopic name added in the list in a pop up window.
* Media type: Trainer can select any media like video, audio and images here.
* Add Media button: Trainer can add video, audio and images here by selecting the appropriate files in the browser menu.
* Add Text button: On clicking this button, trainer can add text area.
* Add Code button: On clicking this button, trainer can add text area.
* New Page Button: On clicking this button, trainer will gate a new empty space where he/she can add text, code and images.
* ‘<’, ‘>’ button: It helps in browsing the pages created by the trainer.
* Previous Button: Takes you to the previous page where trainer may re-enter the module-id, etc.
* Next Button: On clicking this button you will be navigated to the next page where sub-topics and the description can be entered.

## Completion Page

* Module Subject: This is the subject carried forward from the previous page entered by the trainer.
* List of Links to visit for further help.
* For query mail the trainer.
* Finish Button: On clicking this button the application created will be launched.
* Previous Button: Takes you to the previous page where trainer may re-enter the sub-topic, etc.

## Functional specifications:

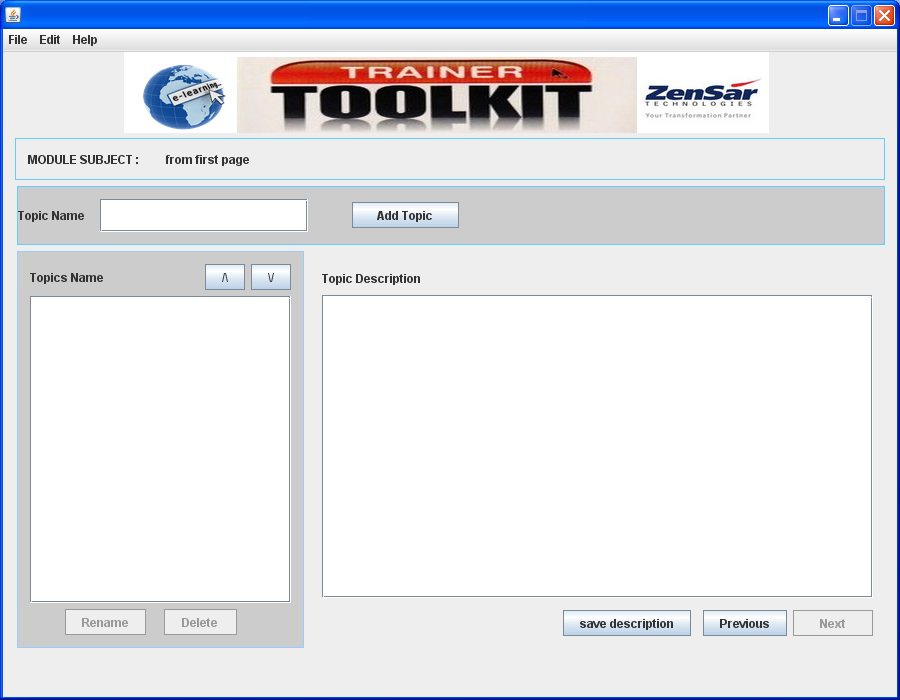
* The trainer can add text, media files such as images, audio and videos in the module.
* The toolkit provides facility of selecting the compiler and integrating it with the training module.
* The training module includes quiz and a test engine for end user’s evaluation
* Random questions with their weightage and deadline time will appear on the screen.
* Once the test is completed it will give u a score card along with the remarks.
* SMTP mail server will be there with which score card will automatically be send to the mail address specified in the form filled before starting the test.
* The trainer can anytime upgrade any existing training module using this toolkit.
* The trainer can preview user’s view of the pages that will be displayed.
* The training module at last will be produced in HTML format.
* Only required controls will be enabled at given point of time.
* Trainer can Undo/Redo the previous changes that he has made.
* Trainer can easily navigate between the pages of the same form.
* He can add quiz wherever he wants in the training module.
* End user have the facility to study the complete module in audio(means we have integrated a text to audio plug-in).

## Module Details:



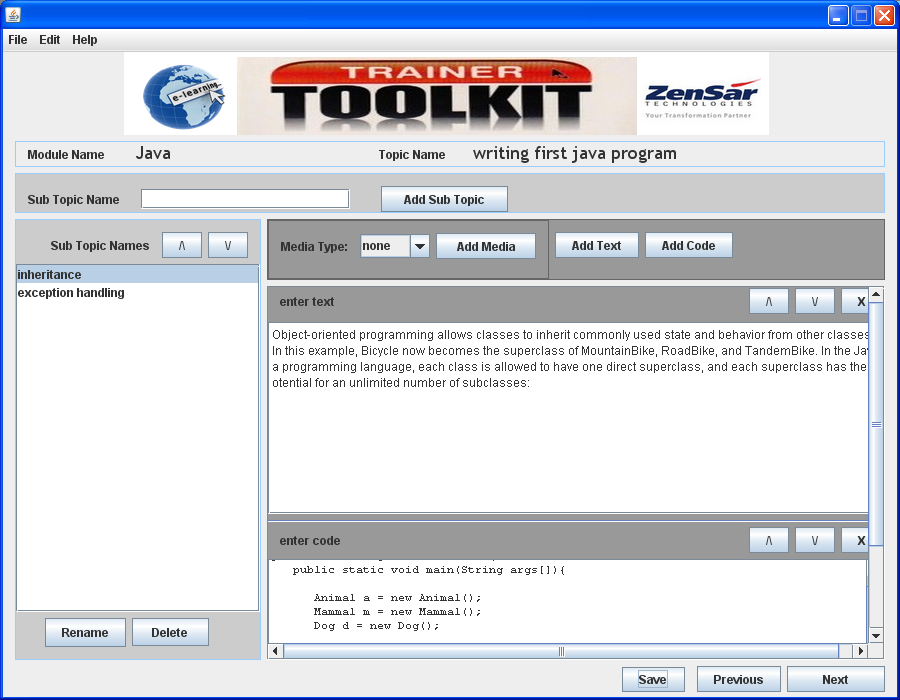
* Here module ID will be unique for each module (can include trainer’s signature also).
* The trainer has to enter the module subject, author name and select the compiler he wishes to build the training for.
* Only after filling all the details trainer can navigate to Table of Contents page.

## Table Of Contents:

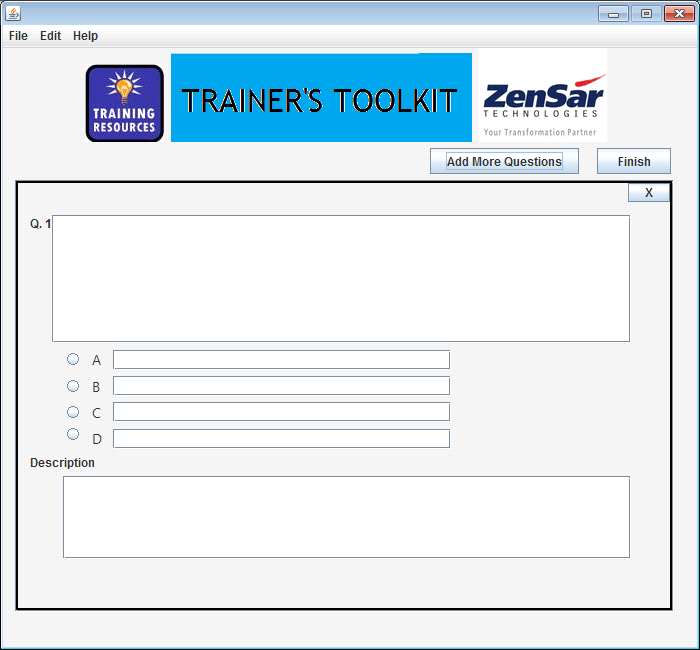


* Here the trainer will have to mention the topic name and give a brief description about it. This name will be added to the left hand corner panel and on selecting it the description will appear in the right hand corner panel. This information can be edited at any point of development process.
* The topics added can also be deleted and renamed.
* The trainer can navigate to next page or the previous page.
* The trainer can adjust the position of the topics using move-up and move-down.
* Renaming, deleting adjusting and navigation can only be done after selection of topic has been made.
* When topic is selected then only delete and rename button is enabled.
* When the trainer enters first character then only adds topic button is enabled.
* The description of any topic can be saved through add description.

## Add Content



* On this screen user can add subtopics.
* Trainer will add complete description of sub-topic (text, audio, video, images) on this page.
* Trainer can rename, delete any sub-topic he wants.
* The trainer has to select the media file he wishes to attach.
* The text area in the panel on the right hand side can be deleted.
* The trainer can add code also for the subtopic for which inbuilt compiler facility is also there for the user.
* Trainer can rearrange sub topics as per requirements.
* Trainer can save all the subtopics and its contents.
* Trainer can navigate to the finish screen.
* The trainer can adjust the text, code and media up and down.
* List of subtopics includes all the sub-topics added by the trainer.
* Trainer can navigate between pages (within a subtopic) in convenient way.
* Protection from accidental deletion—it will verify whether user want to delete or not.
* Trainer can also undo and redo the previous task on text area.
* Once the sub-topic is selected from the list then only the add text, add media, rename, delete control is enabled else they remain disabled.
* The trainer can add new page for the same sub-topic.
* The trainer can directly delete any text area he wants.
* He can anywhere add quiz (between the pages of training module).
* Media added into the toolkit goes to its respective folder in the toolkit.
* Quiz functionality
* On pressing add quiz button, a new page will appear in which we have a panel containing a question with it’s a four options. Trainer can add any number of questions he wants and when he completed the quiz he can end it with Finish Button.
* Trainer can also delete the question .Trainer will give the description of the answers along with the question that will not be visible to the user.
* User can only view the answers when he completes the quiz.

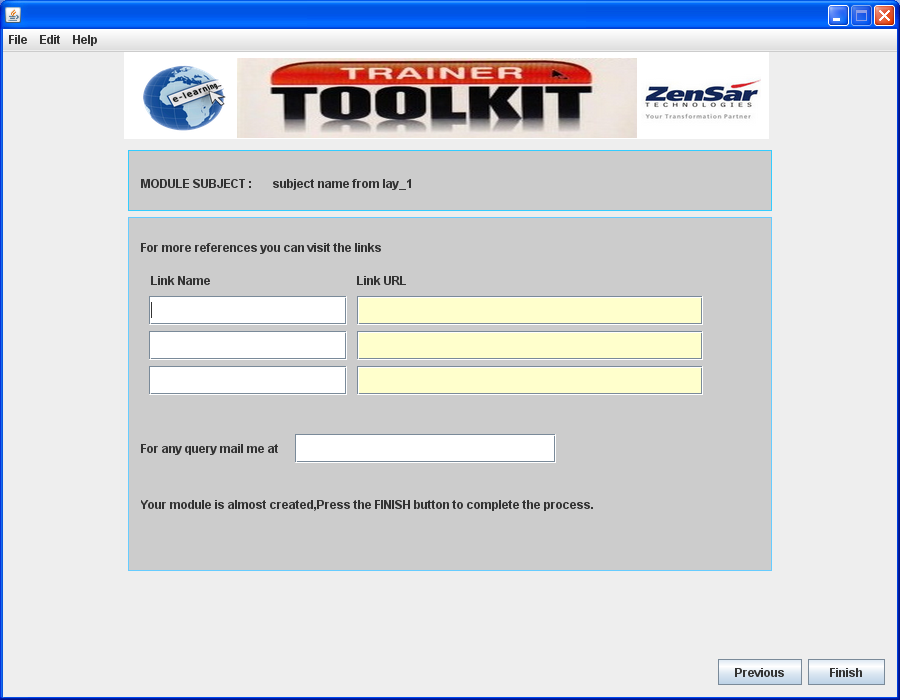


## Test Functionality

**We will discuss the functioning –**

* Initially there will be a form filling and Button to start a test.
* Now the role of trainer appears .He will enter the question along with the options. Trainer will set the question score and a timer.
* User will give this test and when he completes the test his score (graphs) will be displayed along with the comments. The score card will be sent to respective email id filled in the form.

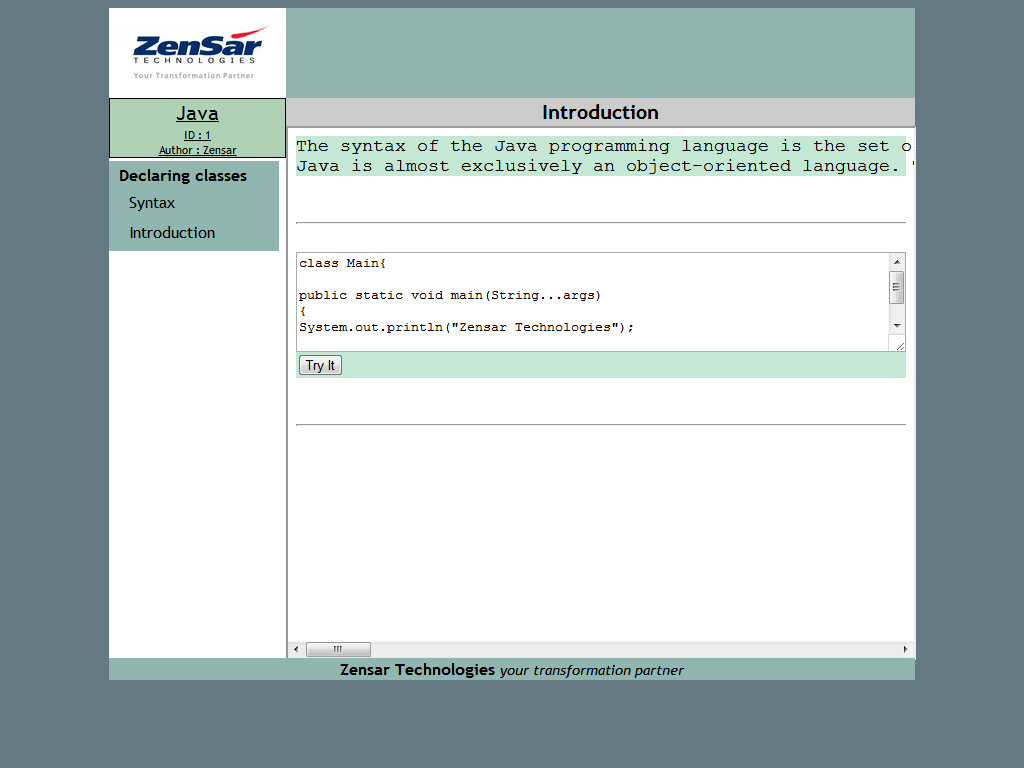
## Completion form



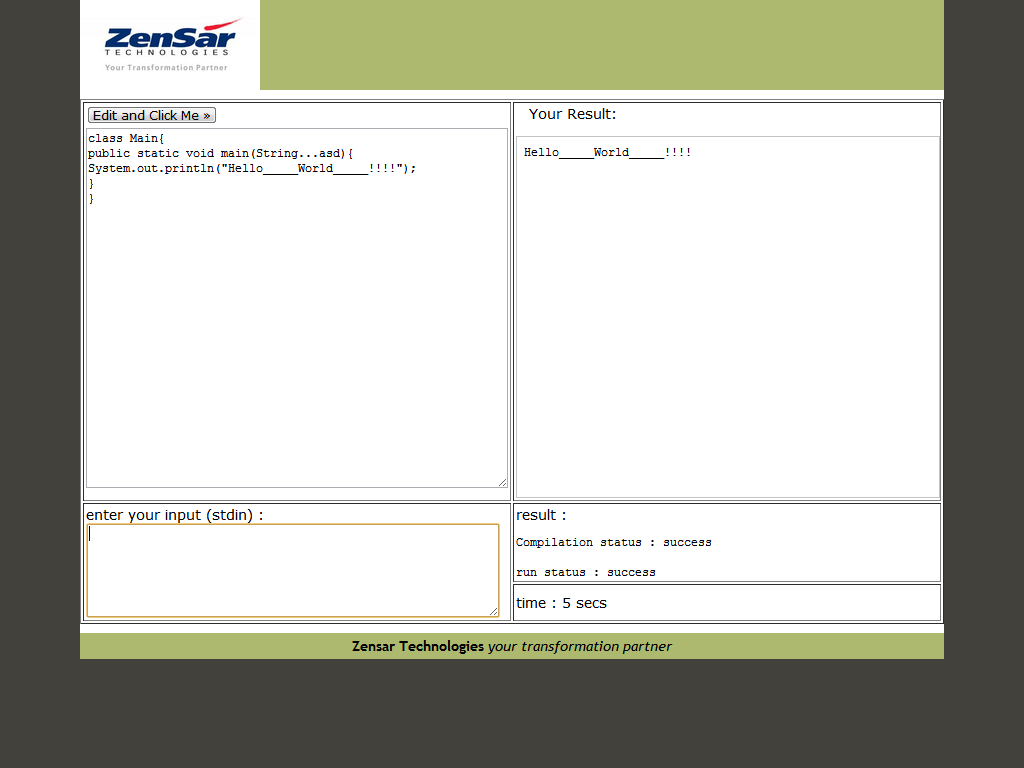
* Trainer can save the training module and then he may convert it into a zip file and upload it.
* Trainer can navigate between screens.
* At the user’s end the training module will be in HTML format only.
* Trainer can add references for additional information and can also create a section where user can ask him query on his mail.
* With the preview button trainer can view what all he has entered till now and he can see the complete training module (which the end user will see) before saving it.

## OUTPUT (Description how trainer and trainee will visual the module)

* As we see there is one text area including text and one code area including code and again we have one more text area (text area 2) so in the same sequence that is (text area 1, code area, text area 2) it will be added into our html format. This is a rough snapshot which gives the idea of how our training module will look like.



* Likewise when he selects any subtopic that text, code or any other thing related to it will be displayed. This how he can go through complete description of any subtopic. The trainer will add the description in the same sequence, same format and that will appear in the above displayed format to the trainee in sequence. Topic and sub-topic is easily distinguishable and color will automatically changes when the end user bring the cursor on it.
* Try It Editor Output- here the trainee can edit code , provide his own input or try his own new code and see the output



## 

## Technology Used

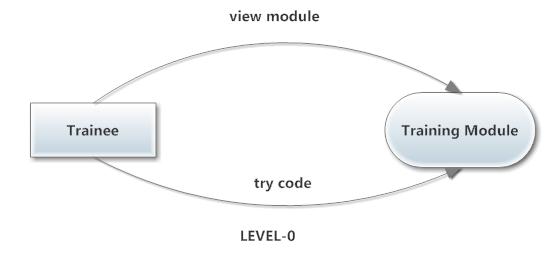
* Chrome (19.0.1084.52)
* Java Standard Edition( JDK 1.6)
* HTML 5
* JSP 2.1
* Servlet 2.4
* Web Server (Apache Tomcat 7.0)

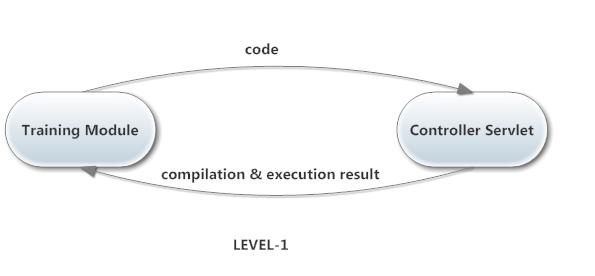
## Future Enhancements

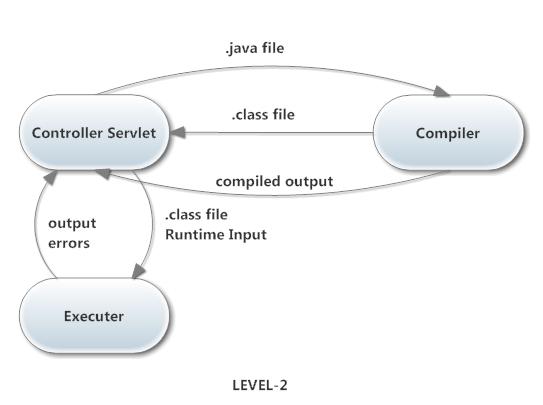
* Provide privileges to administrator.
* Admin will be able to see how many users are currently accessing the page (compiler).
* Admin can control the execution time according to network traffic (compiler).
* Provision for priorities of the user accounts and account management to allow different execution time.
* Embedding Test engine.
* Add more compilers like .net or python so that training module related to them can be prepared with this tool.
* Embedding online SQL compiler which executes SQL queries.
* Providing JDBC compiling capability. Integrating java and sql.

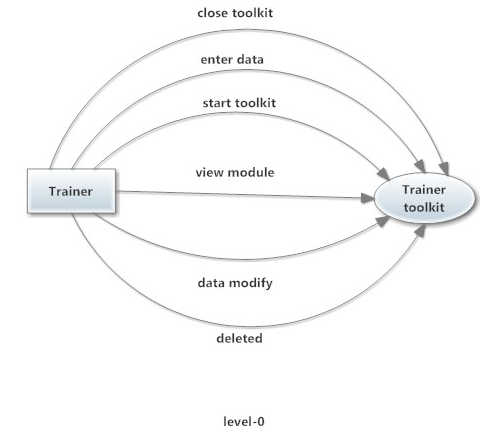
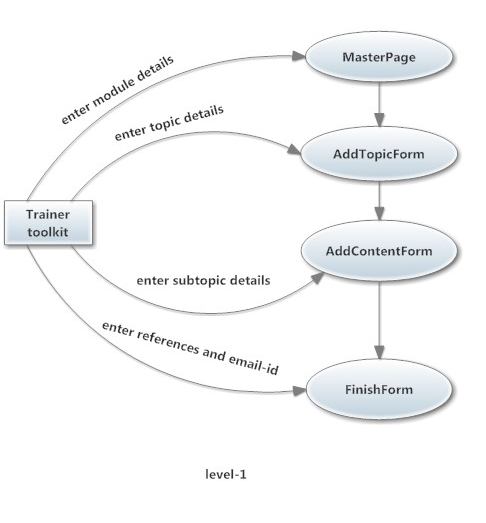
1. **Data Flow Diagram**

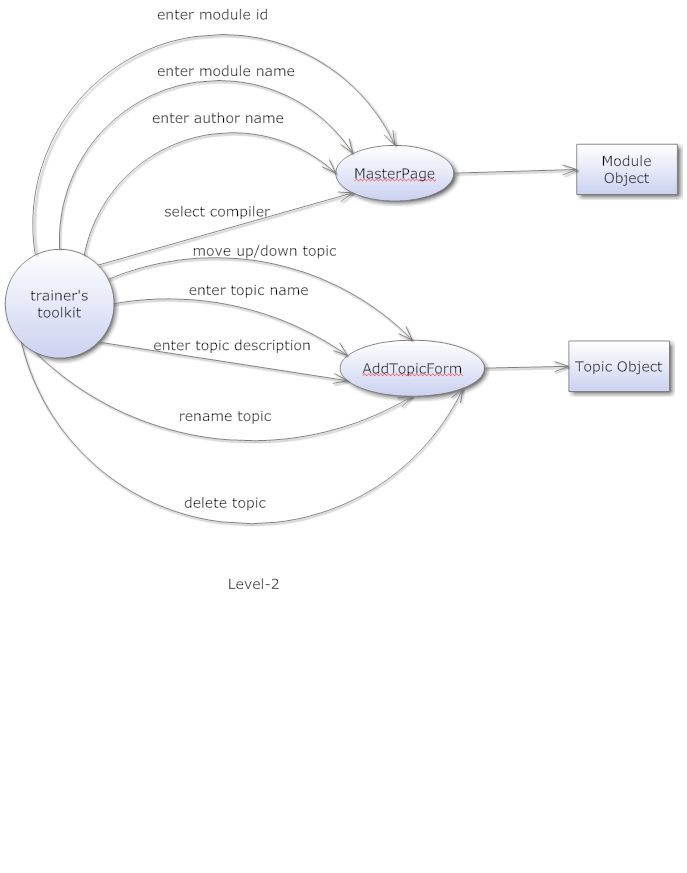
**Training Module**

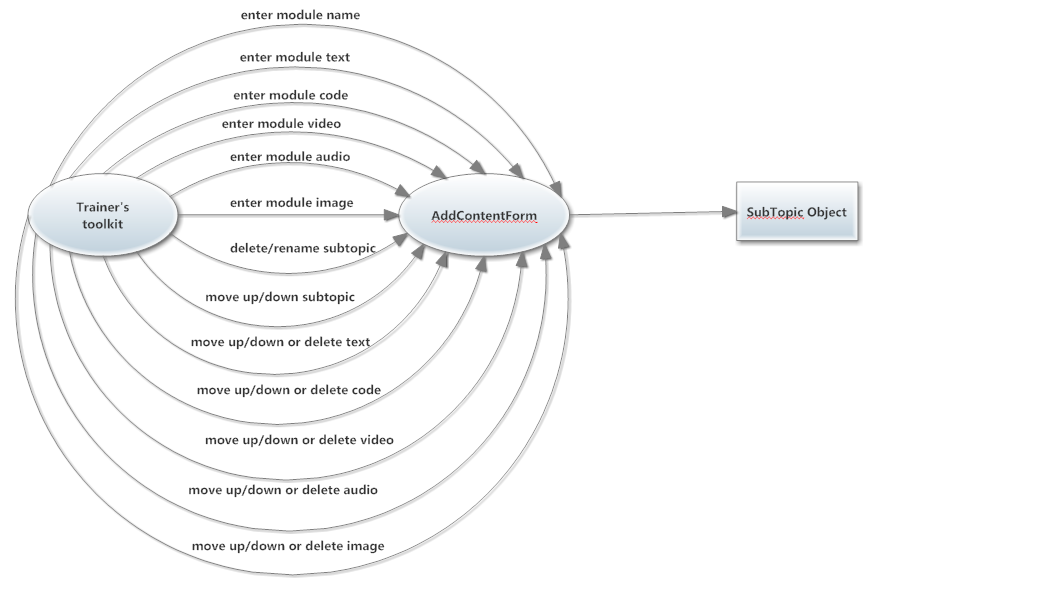
****

****

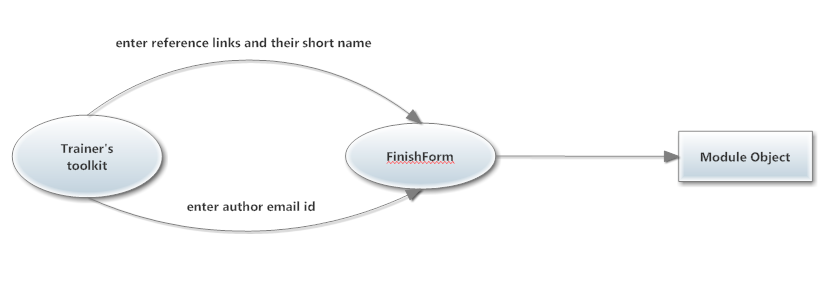
****

**Trainer’s Toolkit-DFD** ****

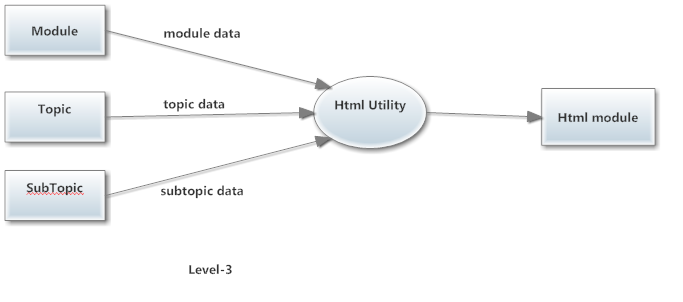
****



Level-2



Level-2

****

1. **Code**

**Package : com.html**

//FileSystem.java

**package** com.html;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.FileNotFoundException;

**import** java.io.FileOutputStream;

**import** java.io.IOException;

**import** java.util.ArrayList;

/\*

\* FileSystem class helps in generating html output files in proper directory structure

\*/

**public** **class** FileSystem {

**public** **static** File *rootDir*;

**protected** **static** File *dir*;

**static**{

*rootDir*=**new** File("modules");

System.*out*.println("rootDir.isDirectory() "+*rootDir*.isDirectory());

**try** {

System.*out*.println(*rootDir*.getCanonicalPath());

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

//System.out.println(target);

}

/\*

\* createDirectoryStructure() helps in creating the required environment in which the

\* html files gets finally placed. Besides it also provides some prerequesite files

\* such .css files, images and welcome\_page.html file.

\*/

**public** **static** String createDirectoryStructure(String moduleRoot){

*dir*=**new** File(*rootDir*,moduleRoot);

System.*out*.println("module root : "+moduleRoot+"\t dir.isDirectory() : "+*dir*.isDirectory());

**if**(!*dir*.isDirectory()){

**boolean** result=*dir*.mkdir();

System.*out*.println("dir.mkdir() : "+result);

**if**(result){

File videos=**new** File(*dir*,"videos");

File audios=**new** File(*dir*,"audios");

File images=**new** File(*dir*,"images");

File contents=**new** File(*dir*,"contents");

File styles=**new** File(*dir*,"styles");

**boolean** flag=**false**;

**if**(videos.mkdir()){

**if**(audios.mkdir()){

**if**(images.mkdir()){

File zensarLogo=**new** File("images/zensar\_logo.jpg");

*copyFile*(zensarLogo,images);

File images1=**new** File("images/images\_1.jpg");

File images2=**new** File("images/images\_2.jpg");

File images3=**new** File("images/images\_3.jpg");

File images4=**new** File("images/images\_4.jpg");

*copyFile*(images1,images);

*copyFile*(images2,images);

*copyFile*(images3,images);

*copyFile*(images4,images);

**if**(contents.mkdir()){

File welcomePageSrc=**new** File("welcome\_page.html");

*copyFile*(welcomePageSrc,*dir*);

**if**(styles.mkdir()){

File homeCSS=**new** File("styles"+File.*separatorChar*+"home.css");

*copyFile*(homeCSS,styles);

}

}

}

}

}

**if**(!flag){

**try** {

**throw** **new** IOException();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

System.*out*.println("completed");

}

**else**{

**try** {

**throw** **new** IOException();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

String root = **null**;

**try** {

root=*dir*.getCanonicalPath();

System.*out*.println("returning -> "+root);

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**return** root;

}

/\*

\* copyFile() copies the given source file to desired destination directory

\*/

**public** **static** **void** copyFile(File sourceFile,File destinationDir){

File targetFileName=**new** File(destinationDir,sourceFile.getName());

**int** sourceFileLength=(**int**) sourceFile.length();

**try** {

FileInputStream welcomeFis=**new** FileInputStream(sourceFile);

**byte** [] welcomeBuffer=**new** **byte**[sourceFileLength];

welcomeFis.read(welcomeBuffer);

FileOutputStream welcomeFos=**new** FileOutputStream(targetFileName);

welcomeFos.write(welcomeBuffer);

} **catch** (FileNotFoundException e1) {

// **TODO** Auto-generated catch block

e1.printStackTrace();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

/\*

\*main() is meant for unit testing of this class.

\*this is function should be used only for

\*testing purposes

\*/

**public** **static** **void** main(String...asd) **throws** IOException{

System.*out*.println(*createDirectoryStructure*("java"));

}

}

//HtmlUtility.java

**package** com.html;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.FileNotFoundException;

**import** java.io.FileOutputStream;

**import** java.io.IOException;

**import** java.util.ArrayList;

**import** java.util.Properties;

**import** com.toolkit.Integration;

**import** com.toolkit.Module;

**import** com.toolkit.SubTopic;

**import** com.toolkit.Topic;

/\*

\* HtmlUtility helps in generation of actual html contents from java object contents.

\*/

**public** **class** HtmlUtility {

Properties html;

/\*

\* HtmlUtility() performs some initialization tasks

\* which provide html tags name.

\*/

**public** HtmlUtility() **throws** FileNotFoundException, IOException{

html=**new** Properties();

html.load(**new** FileInputStream("properties/html.properties"));

}

/\*

\* getHeaderHtml() gives the header of the html file

\*/

**public** String getHeaderHtml(){

**return** html.getProperty("header");

}

/\*

\* getModuleHtml() gives the module details in html format

\*/

**public** String getModuleHtml(Module module){

String moduleHtml=html.getProperty("begin\_sidebar");

moduleHtml+=html.getProperty("begin\_module")+module.getModuleName();

moduleHtml+=html.getProperty("begin\_module\_id")+module.getModuleID()+html.getProperty("end\_module\_id");

moduleHtml+=html.getProperty("begin\_author\_name")+module.getAuthorName()+html.getProperty("end\_author\_name");

**if**(module.getAuthorMailId()!=**null** && module.getAuthorMailId().length()>0){

moduleHtml+=html.getProperty("begin\_author\_mail\_id")+module.getAuthorMailId()+html.getProperty("end\_author\_mail\_id");

}

moduleHtml+=html.getProperty("end\_module"); **return** moduleHtml;

}

/\*

\* getContentListHtml() gives the topic list and their respective sub topic list in

\* html format

\*/

**public** String getContentListHtml(ArrayList<Topic> topicList){

String contentList=html.getProperty("begin\_topic\_list");

**for**(Topic topic:topicList){

String topicFileName=getFileName(topic.getTopicName());

contentList+=html.getProperty("begin\_topic");

contentList+=html.getProperty("begin\_source")+topicFileName+html.getProperty("set\_target");

contentList+=html.getProperty("begin\_title")+topic.getTopicName()+html.getProperty("end\_title");

contentList+=topic.getTopicName()+html.getProperty("end\_source");

String description=topic.getTopicDescription()!=**null**?topic.getTopicDescription():"";

System.*out*.println("description -------getContentListHtml---- >"+description);

generateFile(description,topicFileName);

String subtopicList=html.getProperty("begin\_subtopic\_list");

**for**(SubTopic subtopic: topic.getSubTopicList()){

String subtopicFileName=getFileName(subtopic.getSubTopicName());

subtopicList+=html.getProperty("begin\_subtopic");

subtopicList+=html.getProperty("begin\_source")+subtopicFileName+html.getProperty("set\_target");

subtopicList+=html.getProperty("begin\_title")+subtopic.getSubTopicName()+html.getProperty("end\_title");

generateSubTopicContent(subtopicFileName,subtopic);

subtopicList+=subtopic.getSubTopicName()+html.getProperty("end\_source");

subtopicList+=html.getProperty("end\_subtopic");

}

subtopicList+=html.getProperty("end\_subtopic\_list");

contentList+=subtopicList+html.getProperty("end\_topic");

}

contentList+=html.getProperty("end\_topic\_list");

//adding references and email id of author.

**if**(Integration.*module*.references.size()>0){

String reference=html.getProperty("begin\_reference");

**for**(String linkName: Integration.*module*.references.keySet()){

reference+=html.getProperty("set\_ref\_source")+Integration.*module*.references.get(linkName)+html.getProperty("set\_ref\_name")+Integration.*module*.references.get(linkName)+html.getProperty("end\_ref");

}

contentList+=reference;

}

contentList+=html.getProperty("end\_sidebar");

**return** contentList;

}

/\*

\* generateSubTopicContent() helps in preparing the html version of sub topic contents

\* and saves the content in separate html file

\* in content directory

\*/

**private** **void** generateSubTopicContent(String subtopicFileName,

SubTopic subtopic) {

// **TODO** Auto-generated method stub

String html\_content="";

Properties content; //future enhancement

//System.out.println("description----------generateSubTopicContent------->"+);

System.*out*.println(subtopic.contentMap);

content=**new** Properties();

**try** {

content.load(**new** FileInputStream("properties/content.properties"));

File cssFile=**new** File("styles/content.css");

**int** size=(**int**) cssFile.length();

**byte**[] buffer=**new** **byte**[size];

FileInputStream fis=**new** FileInputStream(cssFile);

fis.read(buffer);

html\_content=**new** String(buffer);

html\_content=content.getProperty("begin\_style")+html\_content+content.getProperty("end\_style");

**for**(String key:subtopic.contentMap.keySet()){

String value=subtopic.contentMap.get(key);

**if**(key.contains("text")==**true**){

html\_content+=content.getProperty("begin\_text")+value+content.getProperty("end\_text");

}

**else** **if**(key.contains("code")==**true**){

html\_content+=content.getProperty("begin\_code")+value+content.getProperty("end\_code");

}

//added else if 26-06-12

**else**{

String fileName=value.split(",")[0];

//System.out.println(value.split(","));

//System.out.println(fileName);

**if**(key.contains("video")==**true**){

fileName=".."+File.*separatorChar*+"videos"+File.*separatorChar*+fileName;

html\_content+=content.getProperty("begin\_video")+fileName+content.getProperty("end\_video");

}

**else** **if**(key.contains("audio")==**true**){

fileName=".."+File.*separatorChar*+"audios"+File.*separatorChar*+fileName;

html\_content+=content.getProperty("begin\_audio")+fileName+content.getProperty("end\_audio");

}

**else** **if**(key.contains("image")==**true**){

fileName=".."+File.*separatorChar*+"images"+File.*separatorChar*+fileName;

html\_content+=content.getProperty("begin\_image")+fileName+content.getProperty("end\_image");

}

}

System.*out*.println("key-->"+key+"\thtml\_content--->"+html\_content);

}

System.*out*.println(html\_content);

subtopicFileName=FileSystem.*dir*.getCanonicalPath()+File.*separatorChar*+"contents"+File.*separatorChar*+subtopicFileName;

FileOutputStream fos=**new** FileOutputStream(subtopicFileName);

fos.write(html\_content.getBytes());

fos.close();

System.*out*.println("!!!!!!!!!!!!!!!!!!!!!\n"+html\_content);

} **catch** (FileNotFoundException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

/\*

\* generateFile() creates the file with given file name and contents

\* in the contents directory

\*/

**public** **void** generateFile(String content,String fileName){

FileOutputStream fos;

//String heading="<center><center>";

**try** {

fileName=FileSystem.*dir*.getCanonicalPath()+File.*separatorChar*+"contents"+File.*separatorChar*+fileName;

} **catch** (IOException e1) {

// **TODO** Auto-generated catch block

e1.printStackTrace();

}

**try** {

System.*out*.println(">>>>>>>>\_\_\_\_\_\_\_\_\_\_\_\_\_\_>>>>>>>>\n");

System.*out*.println(content+"\n"+fileName);

System.*out*.println(">>>>>>>>\_\_\_\_\_\_\_\_\_\_\_\_\_\_>>>>>>>>\n");

fos = **new** FileOutputStream(fileName);

**byte**[] buffer=content.getBytes();

fos.write(buffer);

fos.close();

} **catch** (FileNotFoundException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

/\*

\* getContentHtml() helps in providing the title bar for

\* highlighting the selected the topic or

\* subtopic from the html list

\*/

**public** String getContentHtml(){

String content=html.getProperty("begin\_content");

content+=html.getProperty("begin\_content\_title")+html.getProperty("end\_content\_title");

content+=html.getProperty("iframe");

content+=html.getProperty("end\_content");

**return** content;

}

/\*

\* getFooter() gives the footer for html file

\*/

**public** String getFooter(){

**return** html.getProperty("footer");

}

/\*

\* getFileName() helps in generating a decent name

\* for given topic or sub topic by removing

\* the special characters and white spaces

\*/

**private** String getFileName(String topicName){

String topicFileName=topicName;

topicFileName=topicFileName.toLowerCase();

topicFileName=topicFileName.trim();

topicFileName=topicFileName.replaceAll("[\\.`~!@#$%^&\*(){}\\\_+|<>?/:\"]","");

topicFileName=topicFileName.replaceAll("\\s+","\_");

//System.out.println(topicFileName);

topicFileName+=".html";

**return** topicFileName;

}

/\*

\* util() writes the module.html file to module root folder

\*/

**public** String util(Module module) **throws** FileNotFoundException, IOException{

HtmlUtility html=**new** HtmlUtility();

String temp=html.getHeaderHtml();//done

String output="";

output+=temp;

System.*out*.println("output after getHeaderHtml : "+output);line();

String html\_file\_code=html.getModuleHtml(module);//done

output+=html\_file\_code;

System.*out*.println("output after getModuleHtml : "+output);line();

ArrayList<Topic> topic = module.getTopicList();//done

String html\_topic\_page = html.getContentListHtml(topic);//done

output+=html\_topic\_page;

System.*out*.println("output after getContentListHtml : "+output);line();

output+=html.getContentHtml();

System.*out*.println("output after getContentHtml : "+output);line();

temp=html.getFooter();//done

output+=temp;

System.*out*.println("output after getFooter : "+output);

**char** sc=File.*separatorChar*;

String moduleFileName=FileSystem.*dir*.getCanonicalPath()+File.*separatorChar*+"module.html";

FileOutputStream fos=**new** FileOutputStream(moduleFileName);line();

**byte**[] buffer=output.getBytes();

fos.write(buffer);

fos.close();

**return** output;

}

**private** **void** line(){

System.*out*.println("------------------------------------------------");

}

/\*

\* main() is meant for unit testing for class

\*/

**public** **static** **void** main(String...asd) **throws** FileNotFoundException, IOException{

Module module=**new** Module();

/\*/module.setAuthorName("Zensar");

//module.setCompilerName("JDK");

//module.setModuleID("121");

//module.setModuleName("JAVA");

Topic topic=new Topic();

//topic.setTopicName("CLASSES");

//topic.setTopicDescription("this is a class...");

SubTopic st=new SubTopic();

//st.setSubTopicName("DEclaring");

//topic.subTopicList.add(st);

module.topicList.add(topic);

HtmlUtility hu=new HtmlUtility();

hu.util(module);\*/

}

}

**Package : com.toolkit**

//AddCodePanel.java

/\*

\* AddCodePanel.java

\*this panel is called in the add content page on click of AddCodeButton in "AddContentPage" form.

\*

\*/

**package** com.toolkit;

/\*\*

\* this class is used to add code for user

\* which can be executed online by user or trainer

\* **@author** \_\_USER\_\_

\*/

**public** **class** AddCodePanel **extends** javax.swing.JPanel {

/\*\*

\*

\*/

**private** **static** **final** **long** *serialVersionUID* = 1L;

/\*\* Creates new form AddCodePanel \*/

**private** **int** count = 0;

**public** String toString() {

**return** "code-" + (++count);

}

**public** AddCodePanel() {

initComponents();

}

/\*\* This method is called from within the constructor to

\* initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is

\* always regenerated by the Form Editor.

\*/

//GEN-BEGIN:initComponents

// <editor-fold defaultstate="collapsed" desc="Generated Code">

**private** **void** initComponents() {

enterCodeLabel = **new** javax.swing.JLabel();

jScrollPane1 = **new** javax.swing.JScrollPane();

codeEditorPane = **new** javax.swing.JEditorPane();

jSeparator1 = **new** javax.swing.JSeparator();

jButton1 = **new** javax.swing.JButton();

jButton2 = **new** javax.swing.JButton();

jButton3 = **new** javax.swing.JButton();

autoSelectOnDelClick = **new** javax.swing.JCheckBox();

setBackground(**new** java.awt.Color(153, 153, 153));

enterCodeLabel.setText("enter code");

codeEditorPane.setFont(**new** java.awt.Font("Courier New", 0, 11));

jScrollPane1.setViewportView(codeEditorPane);

jButton1.setText("/\\");

jButton2.setText("\\/");

jButton3.setText("X");

jButton3.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jButton3ActionPerformed(evt);

}

});

autoSelectOnDelClick.setVisible(**false**);

javax.swing.GroupLayout layout = **new** javax.swing.GroupLayout(**this**);

**this**.setLayout(layout);

layout.setHorizontalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

layout.createSequentialGroup()

.addContainerGap()

.addComponent(enterCodeLabel)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*,

233, Short.*MAX\_VALUE*)

.addComponent(autoSelectOnDelClick)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jButton1)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jButton2)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jButton3))

.addComponent(jSeparator1,

javax.swing.GroupLayout.*DEFAULT\_SIZE*, 451,

Short.*MAX\_VALUE*)

.addComponent(jScrollPane1,

javax.swing.GroupLayout.*DEFAULT\_SIZE*, 451,

Short.*MAX\_VALUE*));

layout.setVerticalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(jButton3)

.addComponent(jButton2)

.addComponent(jButton1)

.addComponent(enterCodeLabel)

.addComponent(

autoSelectOnDelClick))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jScrollPane1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

161,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jSeparator1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)));

}// </editor-fold>

//GEN-END:initComponents

/\*

\* jButton3 is a delete button.

\* this function helps in deleting the particular panel on clicking this button .

\* this button makes auto selection of "autoSelectOnDelClick" check-box ,which is checked by the

\* removeTextPanel() method present in AddContentPanel class, to remove the particular panel from the content panel

\*

\*

\*/

**private** **void** jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

//delete code

AddContentPage cp = (AddContentPage) (getParent().getParent()

.getParent().getParent().getParent().getParent().getParent());

//= new AddContentPage();

System.*out*.println("start delete method form addtextJPanel");

autoSelectOnDelClick.setSelected(**true**);

cp.removeTextPanel();

System.*out*.println("selection done auto "

+ autoSelectOnDelClick.isSelected());

System.*out*.println("end delete method form addtextJPanel");

**this**.validate();

}

//GEN-BEGIN:variables

// Variables declaration - do not modify

**private** javax.swing.JCheckBox autoSelectOnDelClick;

**public** javax.swing.JEditorPane codeEditorPane;

**private** javax.swing.JLabel enterCodeLabel;

**private** javax.swing.JButton jButton1;

**private** javax.swing.JButton jButton2;

**private** javax.swing.JButton jButton3;

**private** javax.swing.JScrollPane jScrollPane1;

**private** javax.swing.JSeparator jSeparator1;

// End of variables declaration//GEN-END:variables

/\*this method checks whether the autoSelectOnDelClick

\* check box is selected or not

\* for deletion purpose.

\*/

**public** **boolean** isSelected() {

System.*out*.println(autoSelectOnDelClick.isSelected() + " in AddText");

**return** autoSelectOnDelClick.isSelected();

}

/\* this method provides the text present in the addCodePanel

\* for storing purpose.

\*/

**public** String getText() {

**return** codeEditorPane.getText();

}

**public** **void** setText(String value) {

// **TODO** Auto-generated method stub

codeEditorPane.setText(value);

}

}

//AddContentPage.java

/\*

\* Layout\_three.java

\*

\* Created on \_\_DATE\_\_, \_\_TIME\_\_

\*/

**package** com.toolkit;

**import** java.awt.Component;

**import** java.awt.GridLayout;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.FocusEvent;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.FileNotFoundException;

**import** java.io.FileOutputStream;

**import** java.io.IOException;

**import** java.util.ArrayList;

**import** java.util.List;

**import** java.util.Stack;

**import** javax.swing.ImageIcon;

**import** javax.swing.JFileChooser;

**import** javax.swing.JOptionPane;

**import** javax.swing.JPanel;

**import** com.html.FileSystem;

/\*\* this class is used to subtopic content such as subtopic

\* explanation,code and media for better understanding

\*/

**public** **class** AddContentPage **extends** javax.swing.JFrame {

/\*\*

\*

\*/

**private** **static** **final** **long** *serialVersionUID* = 1L;

**private** **static** String *moduleRoot*;

**static**{

*moduleRoot*=Integration.*createModuleStructure*();

}

/\*\* Creates new form Layout\_three \*/

**public** AddContentPage() {

model = **new** javax.swing.DefaultListModel();

initComponents();

layout = **new** java.awt.GridLayout(0, 1, 5, 5);

contentPanel.setLayout(layout);

setSize(900, 700);

setResizable(**false**);

ImageIcon module\_strip = **new** ImageIcon("images/module\_strip.png");

//toolIcon.

imageLabel.setIcon(module\_strip);

/\*

ImageIcon titleIcon = new ImageIcon("images/tool\_title\_image.png");

titleLabel.setIcon(titleIcon);

ImageIcon companyIcon = new ImageIcon("images/zensar\_logo\_2.jpg");

companyLabel.setIcon(companyIcon);

\*/

listContentPanel = **new** java.util.ArrayList<JPanel>();

browseButton.setEnabled(**false**);

addTextButton.setEnabled(**false**);

addCodeButton.setEnabled(**false**);

moveUpSubTopicButton.setEnabled(**false**);

moveDownSubTopicButton.setEnabled(**false**);

newPagePanel.setVisible(**false**);

}

**public** AddContentPage(AddTopicForm addTopicForm) {

**this**();

**this**.addTopicForm = addTopicForm;

System.*out*.println("calling displayContentPage: " + **this**);

**this**.displayContentPage(**true**);

}

**public** AddContentPage(AddQuizPanel addQuizPanel) {

**this**();

**this**.addQuizPanel = addQuizPanel;

}

//GEN-BEGIN:initComponents

// <editor-fold defaultstate="collapsed" desc="Generated Code">

**private** **void** initComponents() {

java.awt.GridBagConstraints gridBagConstraints;

jPopupMenu1 = **new** javax.swing.JPopupMenu();

modulePanel = **new** javax.swing.JPanel();

moduleNameLabel = **new** javax.swing.JLabel();

topicNameLabel = **new** javax.swing.JLabel();

moduleNameValueLabel = **new** javax.swing.JLabel();

topicNameValueLabel = **new** javax.swing.JLabel();

subTopicPanel = **new** javax.swing.JPanel();

scrollbarTopicsLabel = **new** javax.swing.JLabel();

jScrollPane1 = **new** javax.swing.JScrollPane();

subTopicList = **new** javax.swing.JList();

renameSubTopicButton = **new** javax.swing.JButton();

deleteSubTopicButton = **new** javax.swing.JButton();

moveUpSubTopicButton = **new** javax.swing.JButton();

moveDownSubTopicButton = **new** javax.swing.JButton();

nextButtonLayout = **new** javax.swing.JButton();

topPanel = **new** javax.swing.JPanel();

imageLabel = **new** javax.swing.JLabel();

backButtonLayout = **new** javax.swing.JButton();

menuBar = **new** javax.swing.JMenuBar();

fileMenu = **new** javax.swing.JMenu();

openMenuItem = **new** javax.swing.JMenuItem();

saveMenuItem = **new** javax.swing.JMenuItem();

saveAsMenuItem = **new** javax.swing.JMenuItem();

exitMenuItem = **new** javax.swing.JMenuItem();

editMenu = **new** javax.swing.JMenu();

cutMenuItem = **new** javax.swing.JMenuItem();

copyMenuItem = **new** javax.swing.JMenuItem();

pasteMenuItem = **new** javax.swing.JMenuItem();

deleteMenuItem = **new** javax.swing.JMenuItem();

helpMenu = **new** javax.swing.JMenu();

contentsMenuItem = **new** javax.swing.JMenuItem();

aboutMenuItem = **new** javax.swing.JMenuItem();

contentScrollPane = **new** javax.swing.JScrollPane();

contentPanel = **new** javax.swing.JPanel();

subTopicInfoPanel = **new** javax.swing.JPanel();

subTopicLabel = **new** javax.swing.JLabel();

subTopicTextField = **new** javax.swing.JTextField();

addSubTopicButton = **new** javax.swing.JButton();

messageLabel = **new** javax.swing.JLabel();

jPanel1 = **new** javax.swing.JPanel();

addTextButton = **new** javax.swing.JButton();

addCodeButton = **new** javax.swing.JButton();

jPanel2 = **new** javax.swing.JPanel();

mediaLabel = **new** javax.swing.JLabel();

mediaComboBox = **new** javax.swing.JComboBox();

browseButton = **new** javax.swing.JButton();

newPagePanel = **new** javax.swing.JPanel();

newPageButton = **new** javax.swing.JButton();

jButton3 = **new** javax.swing.JButton();

jButton2 = **new** javax.swing.JButton();

addQuizButton = **new** javax.swing.JButton();

saveButton = **new** javax.swing.JButton();

saveButton.setText("Save");

jMenuBar1 = **new** javax.swing.JMenuBar();

jMenu1 = **new** javax.swing.JMenu();

jMenuItem1 = **new** javax.swing.JMenuItem();

jMenu2 = **new** javax.swing.JMenu();

undoMenuItem = **new** javax.swing.JMenuItem();

redoMenuItem = **new** javax.swing.JMenuItem();

jMenu3 = **new** javax.swing.JMenu();

ContentsMenuItem = **new** javax.swing.JMenuItem();

jMenuItem2 = **new** javax.swing.JMenuItem();

setDefaultCloseOperation(javax.swing.WindowConstants.*EXIT\_ON\_CLOSE*);

modulePanel.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(153, 204, 255)));

moduleNameLabel.setText("Module Name");

topicNameLabel.setText("Topic Name");

moduleNameValueLabel.setFont(**new** java.awt.Font("Trebuchet MS", 0, 18));

moduleNameValueLabel.setText("Java");

topicNameValueLabel.setFont(**new** java.awt.Font("Trebuchet MS", 0, 18));

topicNameValueLabel.setText("writing first java program");

javax.swing.GroupLayout modulePanelLayout = **new** javax.swing.GroupLayout(

modulePanel);

modulePanel.setLayout(modulePanelLayout);

modulePanelLayout.setHorizontalGroup(modulePanelLayout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

modulePanelLayout

.createSequentialGroup()

.addContainerGap()

.addComponent(moduleNameLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

89,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addGap(18, 18, 18)

.addComponent(moduleNameValueLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

226,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addGap(18, 18, 18)

.addComponent(topicNameLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

76,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addGap(18, 18, 18)

.addComponent(topicNameValueLabel,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

380, Short.*MAX\_VALUE*)));

modulePanelLayout

.setVerticalGroup(modulePanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

modulePanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

moduleNameLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

23,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addComponent(

topicNameLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

23,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addComponent(topicNameValueLabel)

.addComponent(moduleNameValueLabel)));

subTopicPanel.setBackground(**new** java.awt.Color(204, 204, 204));

subTopicPanel.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(153, 204, 255)));

scrollbarTopicsLabel.setText(" Sub Topic Names");

subTopicList.setModel(model);

subTopicList

.addListSelectionListener(**new** javax.swing.event.ListSelectionListener() {

**public** **void** valueChanged(

javax.swing.event.ListSelectionEvent evt) {

subTopicListValueChanged(evt);

}

});

subTopicList.addFocusListener(**new** java.awt.event.FocusAdapter() {

**public** **void** focusLost(java.awt.event.FocusEvent evt) {

subTopicListFocusLost(evt);

}

});

jScrollPane1.setViewportView(subTopicList);

renameSubTopicButton.setText("Rename");

renameSubTopicButton.setEnabled(**false**);

renameSubTopicButton

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

renameSubTopicButtonActionPerformed(evt);

}

});

deleteSubTopicButton.setText("Delete");

deleteSubTopicButton.setEnabled(**false**);

deleteSubTopicButton

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

deleteSubTopicButtonActionPerformed(evt);

}

});

moveUpSubTopicButton.setText("/\\");

moveUpSubTopicButton

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

moveUpSubTopicButtonActionPerformed(evt);

}

});

moveDownSubTopicButton.setText("\\/");

moveDownSubTopicButton

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

moveDownSubTopicButtonActionPerformed(evt);

}

});

javax.swing.GroupLayout subTopicPanelLayout = **new** javax.swing.GroupLayout(

subTopicPanel);

subTopicPanel.setLayout(subTopicPanelLayout);

subTopicPanelLayout

.setHorizontalGroup(subTopicPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

subTopicPanelLayout

.createSequentialGroup()

.addContainerGap(32, Short.*MAX\_VALUE*)

.addComponent(scrollbarTopicsLabel)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(moveUpSubTopicButton)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(moveDownSubTopicButton)

.addContainerGap())

.addGroup(

subTopicPanelLayout

.createSequentialGroup()

.addGap(29, 29, 29)

.addComponent(renameSubTopicButton)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

deleteSubTopicButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

77,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addContainerGap(35, Short.*MAX\_VALUE*))

.addComponent(jScrollPane1,

javax.swing.GroupLayout.*DEFAULT\_SIZE*, 218,

Short.*MAX\_VALUE*));

subTopicPanelLayout

.setVerticalGroup(subTopicPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

subTopicPanelLayout

.createSequentialGroup()

.addContainerGap()

.addGroup(

subTopicPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

scrollbarTopicsLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

26,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addComponent(

moveUpSubTopicButton)

.addComponent(

moveDownSubTopicButton))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

jScrollPane1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

348,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

subTopicPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

deleteSubTopicButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

28,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addComponent(

renameSubTopicButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

29,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addContainerGap(29, Short.*MAX\_VALUE*)));

nextButtonLayout.setText("Next");

nextButtonLayout.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

nextButtonLayoutActionPerformed(evt);

}

});

topPanel.setLayout(**new** java.awt.GridBagLayout());

imageLabel.setText(" ");

gridBagConstraints = **new** java.awt.GridBagConstraints();

gridBagConstraints.gridheight = 3;

gridBagConstraints.fill = java.awt.GridBagConstraints.*BOTH*;

topPanel.add(imageLabel, gridBagConstraints);

backButtonLayout.setText("Previous");

backButtonLayout.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

backButtonLayoutActionPerformed(evt);

}

});

fileMenu.setText("File");

openMenuItem.setText("Open");

fileMenu.add(openMenuItem);

saveMenuItem.setText("Save");

fileMenu.add(saveMenuItem);

saveAsMenuItem.setText("Save As ...");

fileMenu.add(saveAsMenuItem);

exitMenuItem.setText("Exit");

exitMenuItem.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

exitMenuItemActionPerformed(evt);

}

});

fileMenu.add(exitMenuItem);

menuBar.add(fileMenu);

editMenu.setText("Edit");

cutMenuItem.setText("Cut");

editMenu.add(cutMenuItem);

copyMenuItem.setText("Copy");

editMenu.add(copyMenuItem);

pasteMenuItem.setText("Paste");

editMenu.add(pasteMenuItem);

deleteMenuItem.setText("Delete");

editMenu.add(deleteMenuItem);

menuBar.add(editMenu);

helpMenu.setText("Help");

contentsMenuItem.setText("Contents");

helpMenu.add(contentsMenuItem);

aboutMenuItem.setText("About");

helpMenu.add(aboutMenuItem);

menuBar.add(helpMenu);

contentPanel.setBackground(**new** java.awt.Color(153, 153, 153));

contentPanel.setLayout(**new** java.awt.GridBagLayout());

contentScrollPane.setViewportView(contentPanel);

subTopicInfoPanel.setBackground(**new** java.awt.Color(204, 204, 204));

subTopicInfoPanel.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(153, 204, 255)));

subTopicLabel.setText("Sub Topic Name");

subTopicTextField

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

subTopicTextFieldActionPerformed(evt);

}

});

addSubTopicButton.setText("Add Sub Topic");

addSubTopicButton

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

addSubTopicButtonActionPerformed(evt);

}

});

messageLabel.setFont(**new** java.awt.Font("Trebuchet MS", 3, 14));

messageLabel.setForeground(**new** java.awt.Color(0, 153, 0));

javax.swing.GroupLayout subTopicInfoPanelLayout = **new** javax.swing.GroupLayout(

subTopicInfoPanel);

subTopicInfoPanel.setLayout(subTopicInfoPanelLayout);

subTopicInfoPanelLayout

.setHorizontalGroup(subTopicInfoPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

subTopicInfoPanelLayout

.createSequentialGroup()

.addContainerGap()

.addComponent(

subTopicLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

101,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

subTopicTextField,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

209,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addGap(31, 31, 31)

.addComponent(

addSubTopicButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

127,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*,

56, Short.*MAX\_VALUE*)

.addComponent(

messageLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

287,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addContainerGap()));

subTopicInfoPanelLayout

.setVerticalGroup(subTopicInfoPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

subTopicInfoPanelLayout

.createSequentialGroup()

.addContainerGap()

.addGroup(

subTopicInfoPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

subTopicInfoPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

subTopicLabel)

.addComponent(

addSubTopicButton)

.addComponent(

subTopicTextField,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addComponent(

messageLabel,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

23,

Short.*MAX\_VALUE*))

.addContainerGap()));

jPanel1.setBackground(**new** java.awt.Color(153, 153, 153));

jPanel1.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(102, 102, 102)));

addTextButton.setText("Add Text");

addTextButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

addTextButtonActionPerformed(evt);

}

});

addCodeButton.setText("Add Code");

addCodeButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

addCodeButtonActionPerformed(evt);

}

});

jPanel2.setBackground(**new** java.awt.Color(153, 153, 153));

jPanel2.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(102, 102, 102)));

mediaLabel.setText("Media Type:");

mediaComboBox.setModel(**new** javax.swing.DefaultComboBoxModel(

**new** String[] { "none", "videos", "audios", "images" }));

mediaComboBox.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

mediaComboBoxActionPerformed(evt);

}

});

browseButton.setText("Add Media");

browseButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

browseButtonActionPerformed(evt);

}

});

javax.swing.GroupLayout jPanel2Layout = **new** javax.swing.GroupLayout(

jPanel2);

jPanel2.setLayout(jPanel2Layout);

jPanel2Layout

.setHorizontalGroup(jPanel2Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel2Layout

.createSequentialGroup()

.addContainerGap()

.addComponent(

mediaLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

67,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

mediaComboBox,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

browseButton,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

100, Short.*MAX\_VALUE*)

.addContainerGap()));

jPanel2Layout

.setVerticalGroup(jPanel2Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel2Layout

.createSequentialGroup()

.addContainerGap()

.addGroup(

jPanel2Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

mediaLabel)

.addComponent(

mediaComboBox,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

24,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addComponent(

browseButton))

.addContainerGap(19, Short.*MAX\_VALUE*)));

newPagePanel.setBackground(**new** java.awt.Color(153, 153, 153));

newPagePanel.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(102, 102, 102)));

newPageButton.setText("New Page");

newPageButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

newPageButtonActionPerformed(evt);

}

});

jButton3.setText("<");

jButton2.setText(">");

jButton2.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

javax.swing.GroupLayout newPagePanelLayout = **new** javax.swing.GroupLayout(

newPagePanel);

newPagePanel.setLayout(newPagePanelLayout);

newPagePanelLayout

.setHorizontalGroup(newPagePanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

newPagePanelLayout

.createSequentialGroup()

.addContainerGap(40, Short.*MAX\_VALUE*)

.addGroup(

newPagePanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

newPagePanelLayout

.createSequentialGroup()

.addGap(6,

6,

6)

.addComponent(

jButton3)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*UNRELATED*)

.addComponent(

jButton2))

.addComponent(

newPageButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

104,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addContainerGap()));

newPagePanelLayout

.setVerticalGroup(newPagePanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

newPagePanelLayout

.createSequentialGroup()

.addComponent(newPageButton)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

newPagePanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(jButton2)

.addComponent(jButton3))));

javax.swing.GroupLayout jPanel1Layout = **new** javax.swing.GroupLayout(

jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout

.setHorizontalGroup(jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addComponent(

jPanel2,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(addTextButton)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(addCodeButton)

.addGap(18, 18, 18)

.addComponent(

newPagePanel,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)

.addContainerGap()));

jPanel1Layout

.setVerticalGroup(jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addContainerGap()

.addGroup(

jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

addTextButton)

.addComponent(

addCodeButton)))

.addGroup(

jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

**false**)

.addComponent(

jPanel2,

javax.swing.GroupLayout.Alignment.*LEADING*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)

.addComponent(

newPagePanel,

javax.swing.GroupLayout.Alignment.*LEADING*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

56, Short.*MAX\_VALUE*)));

addQuizButton.setText("Add Quiz");

addQuizButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

addQuizButtonActionPerformed(evt);

}

});

addQuizButton.setVisible(**false**);

saveButton.setText("Save");

saveButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

saveButtonActionPerformed(evt);

}

});

jMenu1.setText("File");

jMenuItem1.setText("Exit");

jMenu1.add(jMenuItem1);

jMenuBar1.add(jMenu1);

jMenu2.setText("Edit");

undoMenuItem.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_Z*,

java.awt.event.InputEvent.*CTRL\_MASK*));

undoMenuItem.setText("Undo");

jMenu2.add(undoMenuItem);

redoMenuItem.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_Y*,

java.awt.event.InputEvent.*CTRL\_MASK*));

redoMenuItem.setText("Redo");

jMenu2.add(redoMenuItem);

jMenuBar1.add(jMenu2);

jMenu3.setText("Help");

ContentsMenuItem.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_H*,

java.awt.event.InputEvent.*ALT\_MASK*));

ContentsMenuItem.setText("Contents");

ContentsMenuItem.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

ContentsMenuItemActionPerformed(evt);

}

});

jMenu3.add(ContentsMenuItem);

jMenuItem2.setText("About");

jMenu3.add(jMenuItem2);

jMenuBar1.add(jMenu3);

setJMenuBar(jMenuBar1);

javax.swing.GroupLayout layout = **new** javax.swing.GroupLayout(

getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addContainerGap()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

topPanel,

javax.swing.GroupLayout.Alignment.*TRAILING*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

837, Short.*MAX\_VALUE*)

.addGroup(

layout.createSequentialGroup()

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

subTopicPanel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

contentScrollPane,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

611,

Short.*MAX\_VALUE*)

.addComponent(

jPanel1,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

layout.createSequentialGroup()

.addComponent(

addQuizButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

94,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addGap(10,

10,

10)

.addComponent(

saveButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

63,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*UNRELATED*)

.addComponent(

backButtonLayout)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

nextButtonLayout,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

98,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))))

.addComponent(

subTopicInfoPanel,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)

.addComponent(

modulePanel,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*))

.addContainerGap()));

layout.setVerticalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addComponent(topPanel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

83,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(modulePanel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(subTopicInfoPanel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

40,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*)

.addGroup(

javax.swing.GroupLayout.Alignment.*LEADING*,

layout.createSequentialGroup()

.addComponent(

jPanel1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

contentScrollPane,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

393,

Short.*MAX\_VALUE*))

.addComponent(

subTopicPanel,

javax.swing.GroupLayout.Alignment.*LEADING*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(backButtonLayout)

.addComponent(nextButtonLayout)

.addComponent(addQuizButton)

.addComponent(

saveButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

25,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addContainerGap()));

pack();

}// </editor-fold>

//GEN-END:initComponents

/\* saveButton saves the content panel data in

\* correspondent to the subtopic selected from

\* the list

\*/

**protected** **void** saveButtonActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

String subTopicName = (String) subTopicList.getSelectedValue();

String topicName = topicNameValueLabel.getText();

saveButtonActionPerformed(topicName, subTopicName);

Thread message = **new** Thread(**new** Runnable() {

@Override

**public** **void** run() {

// **TODO** Auto-generated method stub

messageLabel.setText("subtopic saved.");

**try** {

Thread.*sleep*(2000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

messageLabel.setText("");

}

});

message.start();

}

**protected** **void** saveButtonActionPerformed(String topicName,

String subTopicName) {

System.*out*.println("performing SAVE BUTTON ACTION....");

**if** (contentPanel.getComponents().length != 0) {

Integration.*removeSubTopicContents*(topicName, subTopicName);

}

**for** (Component panel : contentPanel.getComponents()) {

//FOR ADDING PANEL

Integration.*setSubTopicContent*(topicName, subTopicName, panel);

System.*out*.println("adding contents to sub topic...." + panel);

/\* // FOR REMOVING PANEL

contentPanel.remove(panel);

\*/

**this**.repaint();

}

System.*out*.println("SAVE BUTTON ACTION completed....");

}

**private** **void** ContentsMenuItemActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

System.*out*.println("entered into help");

run = Runtime.*getRuntime*();

**try** {

File currentDir = **new** File(".");

String root = currentDir.getCanonicalPath();

System.*out*.println(root);

String helpLocation = root + File.*separatorChar* + "help"

+ File.*separatorChar* + "Help\_window.chm";

child = Runtime.*getRuntime*().exec(

"rundll32 url.dll,FileProtocolHandler " + helpLocation);

} **catch** (Exception ex) {

ex.printStackTrace();

System.*out*.println(ex.getMessage());

}

}

/\* addQuizButton creates the new object of AddQuizPanel

\* then hides the AddContentPage and makes the AddQuizPanel

\* visible to the trainer so that it can crate a quiz related

\* to particular subtopic .

\*/

**protected** **void** addQuizButtonActionPerformed(ActionEvent evt) {

// **TODO** Auto-generated method stub

**if** (addQuizPanel == **null**) {

addQuizPanel = **new** AddQuizPanel(**this**);

}

**this**.setVisible(**false**);

addQuizPanel.displayQuizPage(**true**);

}

**protected** **void** newPageButtonActionPerformed(ActionEvent evt) {

// **TODO** Auto-generated method stub

String pageName = "->Page ";

**int** subTopicIndex = subTopicList.getSelectedIndex();

SubTopic subtopic = (SubTopic) subTopicList.getSelectedValue();

subtopic.pageNo++;

pageName += subtopic.pageNo;

model.insertElementAt(pageName, subTopicIndex);

//subtopic.subTopicStack.push();

}

/\* removeTextPanel() performs the deletion of the

\* particular panel which is selected for deletion

\* by the trainer by clicking the ( "X" ) button available

\* in the panel .

\* this method delete the panel by checking its check-box

\* is selected or not (which is hidden) by calling isSelected()

\* method present in class of that panel.

\* panel can be of type AddTextJPanel,AddCodePanel or AddMediaJPanel

\*/

**public** **void** removeTextPanel() {

System.*out*.println("start removeTextpanel method");

**for** (Component panel : contentPanel.getComponents()) {

// } (Component panel : contentPanel.getComponents()) {

System.*out*.println("Entered");

**boolean** checkValue = **false**;

**if** (panel **instanceof** AddTextJPanel) {

AddTextJPanel tempPanel = (AddTextJPanel) panel;

checkValue = tempPanel.isSelected();

} **else** **if** (panel **instanceof** AddCodePanel) {

AddCodePanel tempPanel = (AddCodePanel) panel;

checkValue = tempPanel.isSelected();

} **else** **if** (panel **instanceof** AddMediaJPanel) {

AddMediaJPanel tempPanel = (AddMediaJPanel) panel;

checkValue = tempPanel.isSelected();

}

**if** (checkValue == **true**) {

contentPanel.remove(panel);

System.*out*

.println("-----------------------------------------------------------------------");

System.*out*.println(getClass() + ": deleting text panel ....");

shuffle();

}

}

}

/\*

\* This moveUpSubTopicButton performs the swap operation

\* which takes the subTopic name down-words (i.e increase the

\* index of selected subTopic) if its already present at the topic it

\* remains at the same place.

\*/

**private** **void** moveDownSubTopicButtonActionPerformed(

java.awt.event.ActionEvent evt) {

**int** moveMe = subTopicList.getSelectedIndex();

**if** (moveMe != model.getSize() - 1) {

// not already at bottom

swap(moveMe, moveMe + 1);

subTopicList.setSelectedIndex(moveMe + 1);

subTopicList.ensureIndexIsVisible(moveMe + 1);

}

}

/\*

\* This moveUpSubTopicButton performs the swap operation

\* which takes the subTopic name up-words (i.e decrease the

\* index of selected subTopic) if its already present at the topic it

\* remains at the same place.

\*/

**private** **void** moveUpSubTopicButtonActionPerformed(

java.awt.event.ActionEvent evt) {

**int** moveMe = subTopicList.getSelectedIndex();

**if** (moveMe != 0) {

// not already at top

swap(moveMe, moveMe - 1);

subTopicList.setSelectedIndex(moveMe - 1);

subTopicList.ensureIndexIsVisible(moveMe - 1);

}

}

/\*

\* This method performs the swap

\* operation between two subTopic names

\* up-words or down-words as per the arguments

\* provided to it.

\*/

**private** **void** swap(**int** a, **int** b) {

Object aObject = model.getElementAt(a);

Object bObject = model.getElementAt(b);

model.set(a, bObject);

model.set(b, aObject);

String topicName = topicNameValueLabel.getText();

Integration.*swapSubTopic*(topicName, a, b);

}

**private** **void** jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

}

**protected** **void** mediaComboBoxActionPerformed(ActionEvent evt) {

// **TODO** Auto-generated method stub

}

**protected** **void** subTopicTextFieldActionPerformed(ActionEvent evt) {

// **TODO** Auto-generated method stub

}

**protected** **void** subTopicListFocusLost(FocusEvent evt) {

}

/\* shuffle() method is used for rearranging the panels

\* in the contentPanel after deleting the any panel

\* from it.

\* it first removes all panel from the contentPanel

\* by pushing all panels into the stack .it also

\* uses another stack for sequencing the panels

\* in the contentPanel.

\*/

**public** **void** shuffle() {//very useful method

panels = **new** Stack<Component>();

**int** count = 0;

**for** (Component c : contentPanel.getComponents()) {

panels.push(c);

System.*out*.println(c + "\n----------->" + count++);

}

// this.getContentPane().removeAll();

// Collections.shuffle(panels);

Stack<Component> reversedPanels = **new** Stack<Component>();

**while** (!panels.isEmpty()) {

reversedPanels.push(panels.pop());

}

**while** (!reversedPanels.empty())

contentPanel.add(reversedPanels.pop());

((JPanel) getContentPane()).revalidate();

**this**.repaint();

}

/\*

\* browseButton first creates the new object of AddMediaJPanel

\* and then checks whether the item is selected

\* or not from the drop-down list or not .if not it gives

\* warning message .if item is selected from the drop-down

\* list ,then it opens file Chooser to file for uploading

\* and also checks the extension of the file supported by

\* the application , if the extension is not supported by

\* the application it provides the warning to the trainer.

\* it will add the AddMediaJPanel to contentPanel

\* if all conditions are satisfied.

\*/

**private** **void** browseButtonActionPerformed(java.awt.event.ActionEvent evt) {

mediaPanel = **new** AddMediaJPanel();

String selectedMedia = (String) mediaComboBox.getSelectedItem();

System.*out*.println("selected media : " + selectedMedia);

**if** (selectedMedia.equalsIgnoreCase("none")) {

JOptionPane.*showMessageDialog*(**this**,

"Please select any media type!!", "Error",

JOptionPane.*ERROR\_MESSAGE*);

} **else** {

JFileChooser jf = **new** JFileChooser();

jf.showOpenDialog(**this**);

File file = jf.getSelectedFile();

**final** String fileName = file.getName();

System.*out*.println("file name-->" + fileName + "--");

**if** (Utility.*validateFileFormat*(selectedMedia, fileName)) {

FileInputStream fis;

System.*out*.println("File.separatorChar-->" + File.*separatorChar*

+ "--");

**try** {

String location = *moduleRoot*+File.*separatorChar*+selectedMedia + File.*separatorChar*

+ fileName;

System.*out*.println("starting file operations...location-->"

+ location + "--");

fis = **new** FileInputStream(file);

**byte**[] buffer = **new** **byte**[(**int**) file.length()];

fis.read(buffer);

FileOutputStream fos = **new** FileOutputStream(location);

fos.write(buffer);

fis.close();

fos.close();

System.*out*

.println("file successfully copied to correct folder...");

} **catch** (FileNotFoundException e) {

JOptionPane.*showMessageDialog*(**this**,

"File does not exists!!", "Error",

JOptionPane.*ERROR\_MESSAGE*);

} **catch** (IOException e) {

JOptionPane

.*showMessageDialog*(

**this**,

"File upload failed due to some internal error!!\nPlease try after some time!!",

"Error", JOptionPane.*ERROR\_MESSAGE*);

}

mediaPanel.setText(file.getName(), selectedMedia);

contentPanel.add(mediaPanel);

**this**.validate();

} **else** {

JOptionPane.*showMessageDialog*(**this**,

"File extension not supported!!!", "Error",

JOptionPane.*ERROR\_MESSAGE*);

}

}

}

// \* deleteSubTopicButton delete the selected SubTopic form the list

**protected** **void** deleteSubTopicButtonActionPerformed(ActionEvent evt) {

**try** {

**if** (!subTopicList.isSelectionEmpty()) {

**int** subTopicIndex = subTopicList.getSelectedIndex();

model.remove(subTopicIndex);

String topicName = topicNameValueLabel.getText();

Integration.*deleteSubTopic*(topicName, subTopicIndex);

**if** (model.size() == 0) {

isModifiable(**false**);

}

} **else** {

JOptionPane.*showMessageDialog*(**this**, "Please Select Any Topic",

"Error", JOptionPane.*ERROR\_MESSAGE*);

}

} **catch** (java.lang.ArrayIndexOutOfBoundsException e) {

JOptionPane.*showMessageDialog*(**this**,

"Please select any sub topic!!", "Error",

JOptionPane.*ERROR\_MESSAGE*);

}

}

/\* renameSubTopicButton performs the renaming the selected subTopicName

\* it doesn't allow the blank field .or renaming the subTopic with the same name which

\* is already present in the SubTopicName list.

\*

\*/

**protected** **void** renameSubTopicButtonActionPerformed(ActionEvent evt) {

**try** {

**if** (!subTopicList.isSelectionEmpty()) {

String newSubTopicName = JOptionPane.*showInputDialog*(

"Enter new name for the Topic : ",

subTopicList.getSelectedValue());

**if** (model.contains(newSubTopicName)) {

JOptionPane.*showMessageDialog*(**this**, "Topic already exists",

"Error", JOptionPane.*ERROR\_MESSAGE*);

} **else** {

**int** subtopicIndex = subTopicList.getSelectedIndex();

model.set(subtopicIndex, newSubTopicName);

String topicName = topicNameValueLabel.getText();

Integration.*renameSubTopic*(topicName, subtopicIndex,

newSubTopicName);

}

} **else** {

JOptionPane.*showMessageDialog*(**this**,

"Please Select Any SubTopic", "Error",

JOptionPane.*ERROR\_MESSAGE*);

}

} **catch** (java.lang.ArrayIndexOutOfBoundsException e) {

JOptionPane.*showMessageDialog*(**this**,

"Please select any sub topic!!", "Error",

JOptionPane.*ERROR\_MESSAGE*);

}

}

**private** **void** subTopicListValueChanged(

javax.swing.event.ListSelectionEvent evt) {

System.*out*.println("performing SUB TOPIC LIST VALUE CHANGED event...");

isModifiable(**true**);

**if** (model.getSize() > 0) {

String subTopicName = (String) subTopicList.getSelectedValue();

String topicName = topicNameValueLabel.getText();

**for** (Component comp : contentPanel.getComponents()) {

contentPanel.remove(comp);

}

contentPanel.validate();

//loading the current subtopic contents

System.*out*.println("before integration subtopic name : "

+ subTopicName);

Integration.*getSubTopicContent*(topicName, subTopicName, **this**);

}

System.*out*.println("end of SUB TOPIC LIST VALUE CHANGED event...");

}

/\* this button add new subtopic(subTopicTextField) to the subTopicList,

\* it trims the text present in subTopicTextField then

\* it checks whether the particular subtopic already present or not

\* in subTopicList or not if its already present it gives error

\* and also checks whether subTopicName is provided properly not.

\* it can not accept the blank text field as subtopic name.

\*

\*/

**protected** **void** addSubTopicButtonActionPerformed(ActionEvent evt) {

String subTopicName = subTopicTextField.getText();

subTopicName = subTopicName.trim();

**if** (subTopicName.length() != 0) {

**if** (model.contains(subTopicName)) {

JOptionPane.*showMessageDialog*(**this**, "Topic already exists",

"Error", JOptionPane.*ERROR\_MESSAGE*);

} **else** {

model.addElement(subTopicName);

//adding Integration function....

Integration.*setSubtopicName*(topicNameValueLabel.getText(),

subTopicTextField.getText());

}

System.*out*.println(model.contains(subTopicName.trim()));

subTopicTextField.setText("");

} **else** {

JOptionPane.*showMessageDialog*(**this**, "Please enter any name!!",

"Error", JOptionPane.*ERROR\_MESSAGE*);

}

subTopicList.validate();

subTopicList.repaint();

}

/\* addTextButton creates the new object of AddTextJPanel

\* then adds it to the ContentPanel and places the courser

\* at the start of the editorPane.

\*/

**protected** **void** addTextButtonActionPerformed(ActionEvent evt) {

textJPanel = **new** AddTextJPanel();

contentPanel.add(textJPanel);

textJPanel.editorPane.requestFocusInWindow();

listContentPanel.add(textJPanel);

System.*out*.println("add text button");

//layout.addLayoutComponent("textJPanel",textJPanel);

**this**.validate();

}

/\* addCodeButton creates the new object of AddCodePanel

\* then adds it to the ContentPanel and places the courser

\* at the start of the codeEditorPane.

\*/

**protected** **void** addCodeButtonActionPerformed(java.awt.event.ActionEvent evt) {

codeJPanel = **new** AddCodePanel();

contentPanel.add(codeJPanel);

codeJPanel.codeEditorPane.requestFocusInWindow();

System.*out*.println("add code button");

**this**.validate();

}

**protected** **void** isModifiable(**boolean** setValue) {

deleteSubTopicButton.setEnabled(setValue);

renameSubTopicButton.setEnabled(setValue);

browseButton.setEnabled(setValue);

addTextButton.setEnabled(setValue);

addCodeButton.setEnabled(setValue);

addQuizButton.setEnabled(setValue);

moveDownSubTopicButton.setEnabled(setValue);

moveUpSubTopicButton.setEnabled(setValue);

}

//backButton takes it to the MasterPage

**private** **void** backButtonLayoutActionPerformed(java.awt.event.ActionEvent evt) {

/\*for(Component comp: subTopicList.getComponents()){

subTopicList.remove(comp);

}\*/

model.clear();

**this**.setVisible(**false**);

addTopicForm.setVisible(**true**);

}

//nextButton take it to the next page i.e to the Finish Page

//it also creates the object of finish page if not created already

**private** **void** nextButtonLayoutActionPerformed(java.awt.event.ActionEvent evt) {

**try** {

**int** checkList = model.size();

**if** (checkList == 0) {

**throw** **new** EmptyListException();

}

**if** (finishForm == **null**) {

finishForm = **new** FinishForm(**this**);

}

**this**.setVisible(**false**);

finishForm.setVisible(**true**);

} **catch** (EmptyListException e) {

JOptionPane.*showMessageDialog*(**this**, "please create any subtopic!!",

"Error", JOptionPane.*ERROR\_MESSAGE*);

e.printStackTrace();

}

}

**private** **void** exitMenuItemActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_exitMenuItemActionPerformed

System.*exit*(0);

}

/\*\*

\* **@param** args the command line arguments

\*/

**public** **void** displayContentPage(**boolean** value) {

System.*out*.println("inside displayContentPage :" + addTopicForm);

moduleNameValueLabel.setText(addTopicForm.masterPage

.getModuleSubjectTextField());

System.*out*.println("--" + addTopicForm.getTopicName() + "--");

topicNameValueLabel.setText(addTopicForm.getTopicName());

**this**.setVisible(value);

populateSubTopicList();

}

**public** **static** **void** main(String args[]) {

java.awt.EventQueue.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**new** AddContentPage().setVisible(**true**);

}

});

}

//GEN-BEGIN:variables

// Variables declaration - do not modify

**private** javax.swing.JMenuItem ContentsMenuItem;

**private** javax.swing.JMenuItem aboutMenuItem;

**private** javax.swing.JButton addCodeButton;

**private** javax.swing.JButton addQuizButton;

**private** javax.swing.JButton addSubTopicButton;

**private** javax.swing.JButton addTextButton;

**private** javax.swing.JButton backButtonLayout;

**private** javax.swing.JButton browseButton;

**public** javax.swing.JPanel contentPanel;

**private** javax.swing.JScrollPane contentScrollPane;

**private** javax.swing.JMenuItem contentsMenuItem;

**private** javax.swing.JMenuItem copyMenuItem;

**private** javax.swing.JMenuItem cutMenuItem;

**private** javax.swing.JMenuItem deleteMenuItem;

**private** javax.swing.JButton deleteSubTopicButton;

**private** javax.swing.JMenu editMenu;

**private** javax.swing.JMenuItem exitMenuItem;

**private** javax.swing.JMenu fileMenu;

**private** javax.swing.JMenu helpMenu;

**private** javax.swing.JLabel imageLabel;

**private** javax.swing.JButton jButton2;

**private** javax.swing.JButton jButton3;

**private** javax.swing.JMenu jMenu1;

**private** javax.swing.JMenu jMenu2;

**private** javax.swing.JMenu jMenu3;

**private** javax.swing.JMenuBar jMenuBar1;

**private** javax.swing.JMenuItem jMenuItem1;

**private** javax.swing.JMenuItem jMenuItem2;

**private** javax.swing.JPanel jPanel1;

**private** javax.swing.JPanel jPanel2;

**private** javax.swing.JPopupMenu jPopupMenu1;

**private** javax.swing.JScrollPane jScrollPane1;

**private** javax.swing.JComboBox mediaComboBox;

**private** javax.swing.JLabel mediaLabel;

**private** javax.swing.JMenuBar menuBar;

**private** javax.swing.JLabel messageLabel;

**private** javax.swing.JLabel moduleNameLabel;

**private** javax.swing.JLabel moduleNameValueLabel;

**private** javax.swing.JPanel modulePanel;

**private** javax.swing.JButton moveDownSubTopicButton;

**private** javax.swing.JButton moveUpSubTopicButton;

**private** javax.swing.JButton newPageButton;

**private** javax.swing.JPanel newPagePanel;

**private** javax.swing.JButton nextButtonLayout;

**private** javax.swing.JMenuItem openMenuItem;

**private** javax.swing.JMenuItem pasteMenuItem;

**private** javax.swing.JMenuItem redoMenuItem;

**private** javax.swing.JButton renameSubTopicButton;

**private** javax.swing.JMenuItem saveAsMenuItem;

**private** javax.swing.JButton saveButton;

**private** javax.swing.JMenuItem saveMenuItem;

**private** javax.swing.JLabel scrollbarTopicsLabel;

**private** javax.swing.JPanel subTopicInfoPanel;

**private** javax.swing.JLabel subTopicLabel;

**public** javax.swing.JList subTopicList;

**private** javax.swing.JPanel subTopicPanel;

**public** javax.swing.JTextField subTopicTextField;

**private** javax.swing.JPanel topPanel;

**private** javax.swing.JLabel topicNameLabel;

**private** javax.swing.JLabel topicNameValueLabel;

**private** javax.swing.JMenuItem undoMenuItem;

// End of variables declaration//GEN-END:variables

**public** javax.swing.DefaultListModel model;

**protected** AddTopicForm addTopicForm;

**private** FinishForm finishForm;

**private** List<JPanel> listContentPanel;

**public** AddTextJPanel textJPanel;

**private** GridLayout layout;

**public** AddCodePanel codeJPanel;

**public** AddMediaJPanel mediaPanel;

**public** AddQuizPanel addQuizPanel;

**public** Stack<Component> panels;

**private** Runtime run;

**private** Process child;

**public** String getTopicName() {

// **TODO** Auto-generated method stub

**return** topicNameValueLabel.getText();

}

**public** **void** populateSubTopicList() {

// **TODO** Auto-generated method stub

System.*out*.println("populating subtopic jlist...");

Integration.*setSubTopicJList*(topicNameValueLabel.getText(), **this**);

}

}

//AddMediaJPanel.java

/\*

\* AddMediaJPanel.java

\*

\* Created on \_\_DATE\_\_, \_\_TIME\_\_

\*/

**package** com.toolkit;

/\*\*

\* this class is used to add media file such as images,audio or video

\* for subtopic for its better understanding

\* **@author** \_\_USER\_\_

\*/

**public** **class** AddMediaJPanel **extends** javax.swing.JPanel {

/\*\*

\*

\*/

**private** **static** **final** **long** *serialVersionUID* = 1L;

/\*\* Creates new form AddMediaJPanel \*/

**private** **static** **int** *count* = 0;

**public** String toString() {

**return** mediaTypeLabel.getText() + "-" + (++*count*);

/\*

\* sample return value

\* videos-1

\* images-3

\* audios-6

\*/

}

**public** AddMediaJPanel() {

initComponents();

**this**.setSize(50, 75);

previewButton.setVisible(**false**);

}

/\*\* This method is called from within the constructor to

\* initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is

\* always regenerated by the Form Editor.

\*/

//GEN-BEGIN:initComponents

// <editor-fold defaultstate="collapsed" desc="Generated Code">

**private** **void** initComponents() {

mediaNameSelectedLabel = **new** javax.swing.JLabel();

jSeparator3 = **new** javax.swing.JSeparator();

mediaFileNameLabel = **new** javax.swing.JLabel();

jButton1 = **new** javax.swing.JButton();

jButton2 = **new** javax.swing.JButton();

jButton3 = **new** javax.swing.JButton();

previewButton = **new** javax.swing.JButton();

autoSelectOnDelClick = **new** javax.swing.JCheckBox();

fileNameLabel = **new** javax.swing.JLabel();

mediaTypeLabel = **new** javax.swing.JLabel();

setBackground(**new** java.awt.Color(153, 153, 153));

mediaNameSelectedLabel.setText("Media Selected : ");

jSeparator3.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(153, 204, 255)));

mediaFileNameLabel.setText("Media Name");

jButton1.setText("/\\");

jButton2.setText("\\/");

jButton3.setText("X");

jButton3.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jButton3ActionPerformed(evt);

}

});

previewButton.setText("preview");

previewButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

previewButtonActionPerformed(evt);

}

});

autoSelectOnDelClick.setVisible(**false**);

fileNameLabel.setText("File Name :");

mediaTypeLabel.setText("media type");

javax.swing.GroupLayout layout = **new** javax.swing.GroupLayout(**this**);

**this**.setLayout(layout);

layout.setHorizontalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addContainerGap()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(fileNameLabel)

.addComponent(

mediaNameSelectedLabel))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addComponent(

mediaTypeLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

117,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*,

29,

Short.*MAX\_VALUE*)

.addComponent(

autoSelectOnDelClick)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

previewButton)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

jButton1)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

jButton2,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

43,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

jButton3))

.addGroup(

layout.createSequentialGroup()

.addComponent(

mediaFileNameLabel,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

377,

Short.*MAX\_VALUE*)

.addContainerGap())))

.addComponent(jSeparator3,

javax.swing.GroupLayout.*DEFAULT\_SIZE*, 497,

Short.*MAX\_VALUE*));

layout.setVerticalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(jButton3)

.addComponent(jButton1)

.addComponent(previewButton)

.addComponent(jButton2)

.addComponent(

autoSelectOnDelClick)

.addComponent(mediaTypeLabel)

.addComponent(

mediaNameSelectedLabel,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

23, Short.*MAX\_VALUE*))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(fileNameLabel)

.addComponent(

mediaFileNameLabel))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*UNRELATED*)

.addComponent(jSeparator3,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)));

}// </editor-fold>

//GEN-END:initComponents

/\* future enhancement

\* this button will provide the view of the media

\*/

**private** **void** previewButtonActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

}

/\*

\* jButton3 is a delete button.

\* this function helps in deleting the particular panel on clicking this button .

\* this button makes auto selection of "autoSelectOnDelClick" check-box ,

\* which is checked by the removeMediaJPanel() method

\* present in AddContentPanel class,

\* to remove the particular panel from the content panel

\*/

**private** **void** jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

AddContentPage cp = (AddContentPage) (getParent().getParent()

.getParent().getParent().getParent().getParent().getParent());

//= new AddContentPage();

System.*out*.println("start delete method form addtextJPanel");

autoSelectOnDelClick.setSelected(**true**);

cp.removeTextPanel();

System.*out*.println("selection done auto "

+ autoSelectOnDelClick.isSelected());

System.*out*.println("end delete method form addtextJPanel");

**this**.validate();

}

//GEN-BEGIN:variables

// Variables declaration - do not modify

**private** javax.swing.JCheckBox autoSelectOnDelClick;

**private** javax.swing.JLabel fileNameLabel;

**private** javax.swing.JButton jButton1;

**private** javax.swing.JButton jButton2;

**private** javax.swing.JButton jButton3;

**private** javax.swing.JSeparator jSeparator3;

**private** javax.swing.JLabel mediaFileNameLabel;

**private** javax.swing.JLabel mediaNameSelectedLabel;

**private** javax.swing.JLabel mediaTypeLabel;

**private** javax.swing.JButton previewButton;

// End of variables declaration//GEN-END:variables

**public** **void** setText(String text, String type) {

mediaFileNameLabel.setText(text);

mediaTypeLabel.setText("[" + type + "]");

}

**public** **boolean** isSelected() {

System.*out*.println(autoSelectOnDelClick.isSelected() + " in AddText");

**return** autoSelectOnDelClick.isSelected();

}

**public** String getMediaInfo() {

**return** mediaFileNameLabel.getText() + "," + mediaTypeLabel.getText();

}

**public** **void** setMediaInfo(String mediaName) {

// **TODO** Auto-generated method stub

}

}

//AddQuizPanel.java

/\*

\* AddQuizPanel.java

\*

\* Created on \_\_DATE\_\_, \_\_TIME\_\_

\*/

**package** com.toolkit;

**import** java.awt.Component;

**import** java.awt.GridLayout;

**import** java.util.Stack;

**import** javax.swing.ImageIcon;

**import** javax.swing.JOptionPane;

**import** javax.swing.JPanel;

/\*\*

\* this class is used to add questions for particular

\* subtopic

\* **@author** \_\_USER\_\_

\*/

**public** **class** AddQuizPanel **extends** javax.swing.JFrame {

/\*\* Creates new form AddQuizPanel \*/

**public** AddQuizPanel() {

model = **new** javax.swing.DefaultListModel();

initComponents();

setSize(700, 650);

setResizable(**false**);

layout = **new** java.awt.GridLayout(0, 1, 5, 5);

contentPanel.setLayout(layout);

ImageIcon toolIcon = **new** ImageIcon("images/logo2.jpg");

//toolIcon.

toolLabel.setIcon(toolIcon);

ImageIcon titleIcon = **new** ImageIcon("images/tool\_title\_image.png");

titleLabel.setIcon(titleIcon);

ImageIcon companyIcon = **new** ImageIcon("images/zensar\_logo\_2.jpg");

companyLabel.setIcon(companyIcon);

quizPanel2 = **new** QuizJPanel();

contentPanel.add(quizPanel2);

**this**.validate();

}

**public** AddQuizPanel(AddContentPage addContentPage) {

**this**();

**this**.addContentPage = addContentPage;

}

//GEN-BEGIN:initComponents

// <editor-fold defaultstate="collapsed" desc="Generated Code">

**private** **void** initComponents() {

java.awt.GridBagConstraints gridBagConstraints;

jButton1 = **new** javax.swing.JButton();

jButton2 = **new** javax.swing.JButton();

jScrollPane1 = **new** javax.swing.JScrollPane();

contentPanel = **new** javax.swing.JPanel();

topPanel = **new** javax.swing.JPanel();

toolLabel = **new** javax.swing.JLabel();

titleLabel = **new** javax.swing.JLabel();

companyLabel = **new** javax.swing.JLabel();

menuBar = **new** javax.swing.JMenuBar();

fileMenu = **new** javax.swing.JMenu();

openMenuItem = **new** javax.swing.JMenuItem();

saveMenuItem = **new** javax.swing.JMenuItem();

saveAsMenuItem = **new** javax.swing.JMenuItem();

exitMenuItem = **new** javax.swing.JMenuItem();

editMenu = **new** javax.swing.JMenu();

cutMenuItem = **new** javax.swing.JMenuItem();

copyMenuItem = **new** javax.swing.JMenuItem();

pasteMenuItem = **new** javax.swing.JMenuItem();

deleteMenuItem = **new** javax.swing.JMenuItem();

helpMenu = **new** javax.swing.JMenu();

contentsMenuItem = **new** javax.swing.JMenuItem();

aboutMenuItem = **new** javax.swing.JMenuItem();

setDefaultCloseOperation(javax.swing.WindowConstants.*EXIT\_ON\_CLOSE*);

jButton1.setText("Add More Questions");

jButton1.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jButton2.setText("Finish");

jButton2.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

contentPanel.setBackground(**new** java.awt.Color(153, 153, 153));

contentPanel.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(0, 0, 0)));

contentPanel.setLayout(**new** java.awt.GridBagLayout());

jScrollPane1.setViewportView(contentPanel);

topPanel.setLayout(**new** java.awt.GridBagLayout());

toolLabel.setText(" ");

gridBagConstraints = **new** java.awt.GridBagConstraints();

gridBagConstraints.gridheight = 3;

gridBagConstraints.fill = java.awt.GridBagConstraints.*BOTH*;

topPanel.add(toolLabel, gridBagConstraints);

titleLabel.setText(" ");

gridBagConstraints = **new** java.awt.GridBagConstraints();

gridBagConstraints.gridwidth = 6;

gridBagConstraints.gridheight = 3;

gridBagConstraints.fill = java.awt.GridBagConstraints.*BOTH*;

topPanel.add(titleLabel, gridBagConstraints);

companyLabel.setText(" ");

gridBagConstraints = **new** java.awt.GridBagConstraints();

gridBagConstraints.gridheight = 3;

gridBagConstraints.fill = java.awt.GridBagConstraints.*BOTH*;

topPanel.add(companyLabel, gridBagConstraints);

fileMenu.setText("File");

openMenuItem.setText("Open");

fileMenu.add(openMenuItem);

saveMenuItem.setText("Save");

fileMenu.add(saveMenuItem);

saveAsMenuItem.setText("Save As ...");

fileMenu.add(saveAsMenuItem);

exitMenuItem.setText("Exit");

exitMenuItem.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

exitMenuItemActionPerformed(evt);

}

});

fileMenu.add(exitMenuItem);

menuBar.add(fileMenu);

editMenu.setText("Edit");

cutMenuItem.setText("Cut");

editMenu.add(cutMenuItem);

copyMenuItem.setText("Copy");

editMenu.add(copyMenuItem);

pasteMenuItem.setText("Paste");

editMenu.add(pasteMenuItem);

deleteMenuItem.setText("Delete");

editMenu.add(deleteMenuItem);

menuBar.add(editMenu);

helpMenu.setText("Help");

contentsMenuItem.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_H*,

java.awt.event.InputEvent.*ALT\_MASK*));

contentsMenuItem.setText("Contents");

contentsMenuItem.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

contentsMenuItemActionPerformed(evt);

}

});

helpMenu.add(contentsMenuItem);

aboutMenuItem.setText("About");

helpMenu.add(aboutMenuItem);

menuBar.add(helpMenu);

setJMenuBar(menuBar);

javax.swing.GroupLayout layout = **new** javax.swing.GroupLayout(

getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addContainerGap()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

topPanel,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

659,

Short.*MAX\_VALUE*)

.addComponent(

jScrollPane1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

659,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addContainerGap())

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

layout.createSequentialGroup()

.addComponent(

jButton1)

.addGap(18, 18,

18)

.addComponent(

jButton2,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

74,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addGap(26, 26,

26)))));

layout.setVerticalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addComponent(topPanel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

94,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(jButton2)

.addComponent(jButton1))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jScrollPane1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

432,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addContainerGap()));

pack();

}// </editor-fold>

//GEN-END:initComponents

**private** **void** contentsMenuItemActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

System.*out*.println("entered into help");

run = Runtime.*getRuntime*();

**try** {

child = Runtime.*getRuntime*().exec(

"rundll32 url.dll,FileProtocolHandler D:\\data\\help.chm");

} **catch** (Exception ex) {

ex.printStackTrace();

System.*out*.println(ex.getMessage());

}

}

**private** **void** jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

**if** (addContentPage == **null**) {

addContentPage = **new** AddContentPage(**this**);

}

**this**.setVisible(**false**);

addContentPage.displayContentPage(**true**);

}

/\*jButton1 is a Add More Questions button.

\* on clicking this button add one more question

\* to the contentPanel area,which is the instance of

\* QuizJPanel class

\*/

**private** **void** jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

quizPanel2 = **new** QuizJPanel();

System.*out*.println("obj created.");

contentPanel.add(quizPanel2);

**this**.validate();

}

**private** **void** exitMenuItemActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_exitMenuItemActionPerformed

System.*exit*(0);

}//GEN-LAST:event\_exitMenuItemActionPerformed

/\*\*

\* **@param** args the command line arguments

\*/

**public** **static** **void** main(String args[]) {

java.awt.EventQueue.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**new** AddQuizPanel().setVisible(**true**);

}

});

}

//GEN-BEGIN:variables

// Variables declaration - do not modify

**private** javax.swing.JMenuItem aboutMenuItem;

**private** javax.swing.JLabel companyLabel;

**private** javax.swing.JPanel contentPanel;

**private** javax.swing.JMenuItem contentsMenuItem;

**private** javax.swing.JMenuItem copyMenuItem;

**private** javax.swing.JMenuItem cutMenuItem;

**private** javax.swing.JMenuItem deleteMenuItem;

**private** javax.swing.JMenu editMenu;

**private** javax.swing.JMenuItem exitMenuItem;

**private** javax.swing.JMenu fileMenu;

**private** javax.swing.JMenu helpMenu;

**private** javax.swing.JButton jButton1;

**private** javax.swing.JButton jButton2;

**private** javax.swing.JScrollPane jScrollPane1;

**private** javax.swing.JMenuBar menuBar;

**private** javax.swing.JMenuItem openMenuItem;

**private** javax.swing.JMenuItem pasteMenuItem;

**private** javax.swing.JMenuItem saveAsMenuItem;

**private** javax.swing.JMenuItem saveMenuItem;

**private** javax.swing.JLabel titleLabel;

**private** javax.swing.JLabel toolLabel;

**private** javax.swing.JPanel topPanel;

// End of variables declaration//GEN-END:variables

**private** javax.swing.DefaultListModel model;

**public** AddCodePanel codeJPanel;

**public** QuizJPanel quizPanel2;

**private** GridLayout layout;

//public Txtbox tbox;

**private** AddContentPage addContentPage;

**private** Runtime run;

**private** Process child;

**public** **void** displayQuizPage(**boolean** value) {

setVisible(value);

}

/\* removeQuizPanel() performs the deletion of the

\* particular panel which is selected for deletion

\* by the trainer by clicking the ( "X" ) button available

\* in the panel .

\* this method delete the panel by checking its check-box

\* is selected or not (which is hidden) by calling isSelected()

\* method present in QuizJPanel class of that panel.

\*/

**public** **void** removeQuizPanel() {

// **TODO** Auto-generated method stub

System.*out*.println("start removeQuizpanel method");

**for** (Component panel : contentPanel.getComponents()) {

System.*out*.println("Entered component");

**boolean** checkValue = **false**;

**if** (panel **instanceof** QuizJPanel) {

QuizJPanel tempPanel = (QuizJPanel) panel;

System.*out*.println("checking check value..");

checkValue = tempPanel.isSelected();

System.*out*.println("CheckValue " + checkValue);

//code for saving calling integration

//Integration.setQuizPage(this);

}

**if** (checkValue == **true**) {

System.*out*.println("Entered to delete");

contentPanel.remove(panel);

System.*out*

.println("-----------------------------------------------------------------------");

System.*out*.println(getClass() + ": deleting quiz panel ....");

shuffle();

}

}

}

/\* shuffle() method is used for rearranging the panels

\* in the contentPanel after deleting the any panel

\* from it.

\* it first removes all panel from the contentPanel

\* by pushing all panels into the stack .it also

\* uses another stack for sequencing the panels

\* in the contentPanel.

\*/

**public** **void** shuffle() {//very useful method

Stack<Component> panels = **new** Stack<Component>();

**int** count = 0;

**for** (Component c : contentPanel.getComponents()) {

**if** (c **instanceof** QuizJPanel) {

count++;

QuizJPanel tempPanel = (QuizJPanel) c;

tempPanel.questionLabel.setText("Q. " + count);

}

panels.push(c);

System.*out*.println("shuffle count \n----------->" + count);

}

QuizJPanel.*i* = count;

// this.getContentPane().removeAll();

// Collections.shuffle(panels);

Stack<Component> reversedPanels = **new** Stack<Component>();

**while** (!panels.isEmpty()) {

reversedPanels.push(panels.pop());

}

**while** (!reversedPanels.empty())

contentPanel.add(reversedPanels.pop());

((JPanel) getContentPane()).revalidate();

**this**.repaint();

}

}

//AddTextJPanel.java

/\*

\* AddTextJPanel.java

\*

\* Created on \_\_DATE\_\_, \_\_TIME\_\_

\*/

**package** com.toolkit;

**import** java.awt.event.ActionEvent;

**import** javax.swing.JCheckBox;

**import** javax.swing.JComponent;

**import** javax.swing.event.AncestorListener;

/\*\*

\* this class is used to provide the explanation part

\* for subtopic

\*/

**public** **class** AddTextJPanel **extends** javax.swing.JPanel {

/\*\*

\*

\*/

**private** **static** **final** **long** *serialVersionUID* = 1L;

/\*\* Creates new form AddTextJPanel \*/

**private** **static** **int** *count* = 0;

**public** String toString() {

**return** "text-" + (++*count*);

}

**public** AddTextJPanel() {

initComponents();

model = **new** javax.swing.DefaultListModel();

autoSelectOnDelClick.setVisible(**false**);

}

//GEN-BEGIN:initComponents

// <editor-fold defaultstate="collapsed" desc="Generated Code">

**private** **void** initComponents() {

addTextPanelLabel = **new** javax.swing.JLabel();

jScrollPane1 = **new** javax.swing.JScrollPane();

editorPane = **new** javax.swing.JEditorPane();

jSeparator1 = **new** javax.swing.JSeparator();

jButton1 = **new** javax.swing.JButton();

jButton2 = **new** javax.swing.JButton();

jButton3 = **new** javax.swing.JButton();

autoSelectOnDelClick = **new** javax.swing.JCheckBox();

setBackground(**new** java.awt.Color(153, 153, 153));

addTextPanelLabel.setText("enter text ");

jScrollPane1.setViewportView(editorPane);

jButton1.setText("\\/");

jButton2.setText("/\\");

jButton2.addMouseListener(**new** java.awt.event.MouseAdapter() {

**public** **void** mouseClicked(java.awt.event.MouseEvent evt) {

jButton2MouseClicked(evt);

}

});

jButton2.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

jButton3.setText("X");

jButton3.addMouseListener(**new** java.awt.event.MouseAdapter() {

**public** **void** mouseClicked(java.awt.event.MouseEvent evt) {

jButton3MouseClicked(evt);

}

});

jButton3.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jButton3ActionPerformed(evt);

}

});

autoSelectOnDelClick

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

autoSelectOnDelClickActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = **new** javax.swing.GroupLayout(**this**);

**this**.setLayout(layout);

layout.setHorizontalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

layout.createSequentialGroup()

.addContainerGap()

.addComponent(addTextPanelLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

61,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*,

204, Short.*MAX\_VALUE*)

.addComponent(autoSelectOnDelClick)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*UNRELATED*)

.addComponent(jButton2)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jButton1)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jButton3))

.addComponent(jSeparator1,

javax.swing.GroupLayout.*DEFAULT\_SIZE*, 435,

Short.*MAX\_VALUE*).addComponent(jScrollPane1));

layout.setVerticalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

jButton3)

.addComponent(

jButton1)

.addComponent(

jButton2)

.addComponent(

addTextPanelLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

29,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addComponent(

autoSelectOnDelClick))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jScrollPane1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

192,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jSeparator1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)));

}// </editor-fold>

//GEN-END:initComponents

**private** **void** jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

}

**private** **void** jButton2MouseClicked(java.awt.event.MouseEvent evt) {

// **TODO** add your handling code here:Moving text Box up

AddContentPage cp = (AddContentPage) (getParent().getParent()

.getParent().getParent().getParent().getParent().getParent());

System.*out*.println("move up addtextJPanel");

//remaining

}

**protected** **void** jButton3ActionPerformed(ActionEvent evt) {

// **TODO** Auto-generated method stub

}

**private** **void** autoSelectOnDelClickActionPerformed(

java.awt.event.ActionEvent evt) {

System.*out*.println("CLicked.......");

}

/\*

\* jButton3 is a delete button.

\* this function helps in deleting the particular panel on clicking this button .

\* this button makes auto selection of "autoSelectOnDelClick" check-box ,which is checked by the

\* removeTextPanel() method present in AddContentPanel class,

\* to remove the particular panel from the content panel

\*/

**private** **void** jButton3MouseClicked(java.awt.event.MouseEvent evt) {

//delete code

AddContentPage cp = (AddContentPage) (getParent().getParent()

.getParent().getParent().getParent().getParent().getParent());

//= new AddContentPage();

System.*out*.println("start delete method form addtextJPanel");

autoSelectOnDelClick.setSelected(**true**);

cp.removeTextPanel();

System.*out*.println("selection done auto "

+ autoSelectOnDelClick.isSelected());

System.*out*.println("end delete method form addtextJPanel");

**this**.validate();

}

//GEN-BEGIN:variables

// Variables declaration - do not modify

**private** javax.swing.JLabel addTextPanelLabel;

**private** javax.swing.JCheckBox autoSelectOnDelClick;

**public** javax.swing.JEditorPane editorPane;

**private** javax.swing.JButton jButton1;

**private** javax.swing.JButton jButton2;

**private** javax.swing.JButton jButton3;

**private** javax.swing.JScrollPane jScrollPane1;

**private** javax.swing.JSeparator jSeparator1;

// End of variables declaration//GEN-END:variables

**public** javax.swing.DefaultListModel model;

**public** **void** setLabelText(String text) {

addTextPanelLabel.setText(text);

}

// this method is used for deleting the instance of AddTextJPanel

//from the contentPanel present in AddContentPage class

**public** **boolean** isSelected() {

// **TODO** Auto-generated method stub

System.*out*.println(autoSelectOnDelClick.isSelected() + " in AddText");

**return** autoSelectOnDelClick.isSelected();

}

**public** String getText() {

**return** editorPane.getText();

}

**public** **void** setText(String value) {

// **TODO** Auto-generated method stub

editorPane.setText(value);

}

}

//AddTopicForm.java

/\*

\* AddTopicForm.java

\*

\* Created on \_\_DATE\_\_, \_\_TIME\_\_

\*/

**package** com.toolkit;

**import** java.awt.Color;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.MouseEvent;

**import** java.io.File;

**import** javax.swing.ImageIcon;

**import** javax.swing.JOptionPane;

**import** javax.swing.undo.\*;

/\*\*

\* this class is used to add topics

\*/

**public** **class** AddTopicForm **extends** javax.swing.JFrame {

/\*\*

\*

\*/

**private** **static** **final** **long** *serialVersionUID* = 1L;

/\*\* Creates new form AddTopicForm \*/

**public** AddTopicForm() {

model = **new** javax.swing.DefaultListModel();

initComponents();

setSize(900, 700);

setResizable(**false**);

ImageIcon module\_strip = **new** ImageIcon("images/module\_strip.png");

//toolIcon.

imageLabel.setIcon(module\_strip);

/\*

ImageIcon titleIcon = new ImageIcon("images/tool\_title\_image.png");

titleLabel.setIcon(titleIcon);

ImageIcon companyIcon = new ImageIcon("images/zensar\_logo\_2.jpg");

companyLabel.setIcon(companyIcon);

\*/

manager = **new** UndoManager();

topicNameTextField.getDocument().addUndoableEditListener(manager);

descriptionTextArea.getDocument().addUndoableEditListener(manager);

}

**public** AddTopicForm(MasterPage masterPage) {

// **TODO** Auto-generated constructor stub

**this**();

**this**.masterPage = masterPage;

displayTopicForm(**true**);

System.*out*.println(masterPage.getModuleSubjectTextField());

}

//GEN-BEGIN:initComponents

// <editor-fold defaultstate="collapsed" desc="Generated Code">

**private** **void** initComponents() {

java.awt.GridBagConstraints gridBagConstraints;

topPanel = **new** javax.swing.JPanel();

imageLabel = **new** javax.swing.JLabel();

topicInfoPanel = **new** javax.swing.JPanel();

topicNameLabel = **new** javax.swing.JLabel();

topicNameTextField = **new** javax.swing.JTextField();

addTopicButton = **new** javax.swing.JButton();

menuBar = **new** javax.swing.JMenuBar();

fileMenu = **new** javax.swing.JMenu();

openMenuItem = **new** javax.swing.JMenuItem();

saveMenuItem = **new** javax.swing.JMenuItem();

saveAsMenuItem = **new** javax.swing.JMenuItem();

exitMenuItem = **new** javax.swing.JMenuItem();

editMenu = **new** javax.swing.JMenu();

cutMenuItem = **new** javax.swing.JMenuItem();

copyMenuItem = **new** javax.swing.JMenuItem();

pasteMenuItem = **new** javax.swing.JMenuItem();

deleteMenuItem = **new** javax.swing.JMenuItem();

helpMenu = **new** javax.swing.JMenu();

contentsMenuItem = **new** javax.swing.JMenuItem();

aboutMenuItem = **new** javax.swing.JMenuItem();

jPanel1 = **new** javax.swing.JPanel();

moduleSubjectNameLabel = **new** javax.swing.JLabel();

moduleSubjectLabel = **new** javax.swing.JLabel();

jPanel2 = **new** javax.swing.JPanel();

topicListScrollPane = **new** javax.swing.JScrollPane();

topicList = **new** javax.swing.JList();

topicListLabel = **new** javax.swing.JLabel();

jButton1 = **new** javax.swing.JButton();

jButton3 = **new** javax.swing.JButton();

deleteTopicButton = **new** javax.swing.JButton();

renameButton = **new** javax.swing.JButton();

jPanel3 = **new** javax.swing.JPanel();

jScrollPane3 = **new** javax.swing.JScrollPane();

descriptionTextArea = **new** javax.swing.JTextArea();

addDescriptionButton = **new** javax.swing.JButton();

backButton = **new** javax.swing.JButton();

addContentButton = **new** javax.swing.JButton();

messageLabel = **new** javax.swing.JLabel();

jLabel1 = **new** javax.swing.JLabel();

jMenuBar1 = **new** javax.swing.JMenuBar();

jMenu1 = **new** javax.swing.JMenu();

jMenuItem1 = **new** javax.swing.JMenuItem();

jMenu2 = **new** javax.swing.JMenu();

undoMenuItem = **new** javax.swing.JMenuItem();

redoMenuItem = **new** javax.swing.JMenuItem();

jMenu3 = **new** javax.swing.JMenu();

jMenuItem3 = **new** javax.swing.JMenuItem();

jMenuItem2 = **new** javax.swing.JMenuItem();

setDefaultCloseOperation(javax.swing.WindowConstants.*EXIT\_ON\_CLOSE*);

setBackground(**new** java.awt.Color(204, 204, 255));

setForeground(**new** java.awt.Color(204, 204, 255));

topPanel.setLayout(**new** java.awt.GridBagLayout());

imageLabel.setText(" ");

gridBagConstraints = **new** java.awt.GridBagConstraints();

gridBagConstraints.gridheight = 3;

gridBagConstraints.fill = java.awt.GridBagConstraints.*BOTH*;

topPanel.add(imageLabel, gridBagConstraints);

topicInfoPanel.setBackground(**new** java.awt.Color(204, 204, 204));

topicInfoPanel.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(102, 204, 255)));

topicNameLabel.setText("Topic Name");

topicNameTextField

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

topicNameTextFieldActionPerformed(evt);

}

});

addTopicButton.setText("Add Topic");

addTopicButton.addMouseListener(**new** java.awt.event.MouseAdapter() {

**public** **void** mouseClicked(java.awt.event.MouseEvent evt) {

addTopicButtonMouseClicked(evt);

}

});

addTopicButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

addTopicButtonActionPerformed(evt);

}

});

javax.swing.GroupLayout topicInfoPanelLayout = **new** javax.swing.GroupLayout(

topicInfoPanel);

topicInfoPanel.setLayout(topicInfoPanelLayout);

topicInfoPanelLayout

.setHorizontalGroup(topicInfoPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

topicInfoPanelLayout

.createSequentialGroup()

.addComponent(

topicNameLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

70,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

topicNameTextField,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

208,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addGap(44, 44, 44)

.addComponent(

addTopicButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

107,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addContainerGap(359, Short.*MAX\_VALUE*)));

topicInfoPanelLayout

.setVerticalGroup(topicInfoPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

topicInfoPanelLayout

.createSequentialGroup()

.addContainerGap()

.addGroup(

topicInfoPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

topicNameLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

22,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addComponent(

topicNameTextField,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

33,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addComponent(

addTopicButton))

.addContainerGap(

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)));

fileMenu.setText("File");

openMenuItem.setText("Open");

fileMenu.add(openMenuItem);

saveMenuItem.setText("Save");

fileMenu.add(saveMenuItem);

saveAsMenuItem.setText("Save As ...");

fileMenu.add(saveAsMenuItem);

exitMenuItem.setText("Exit");

exitMenuItem.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

exitMenuItemActionPerformed(evt);

}

});

fileMenu.add(exitMenuItem);

menuBar.add(fileMenu);

editMenu.setText("Edit");

cutMenuItem.setText("Cut");

editMenu.add(cutMenuItem);

copyMenuItem.setText("Copy");

editMenu.add(copyMenuItem);

pasteMenuItem.setText("Paste");

editMenu.add(pasteMenuItem);

deleteMenuItem.setText("Delete");

editMenu.add(deleteMenuItem);

menuBar.add(editMenu);

helpMenu.setText("Help");

contentsMenuItem.setText("Contents");

helpMenu.add(contentsMenuItem);

aboutMenuItem.setText("About");

helpMenu.add(aboutMenuItem);

menuBar.add(helpMenu);

jPanel1.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(102, 204, 255)));

moduleSubjectNameLabel.setText("from first page");

moduleSubjectLabel.setText("MODULE SUBJECT :");

javax.swing.GroupLayout jPanel1Layout = **new** javax.swing.GroupLayout(

jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(jPanel1Layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*).addGroup(

jPanel1Layout

.createSequentialGroup()

.addContainerGap()

.addComponent(moduleSubjectLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*, 120,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addGap(18, 18, 18)

.addComponent(moduleSubjectNameLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*, 429,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addContainerGap(217, Short.*MAX\_VALUE*)));

jPanel1Layout

.setVerticalGroup(jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addContainerGap()

.addGroup(

jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

moduleSubjectLabel)

.addComponent(

moduleSubjectNameLabel))

.addContainerGap(

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)));

jPanel2.setBackground(**new** java.awt.Color(204, 204, 204));

jPanel2.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(153, 204, 255)));

topicList.setModel(model);

topicList

.setSelectionMode(javax.swing.ListSelectionModel.*SINGLE\_INTERVAL\_SELECTION*);

topicList.addMouseListener(**new** java.awt.event.MouseAdapter() {

**public** **void** mouseClicked(java.awt.event.MouseEvent evt) {

topicListMouseClicked(evt);

}

});

topicList

.addListSelectionListener(**new** javax.swing.event.ListSelectionListener() {

**public** **void** valueChanged(

javax.swing.event.ListSelectionEvent evt) {

topicListValueChanged(evt);

}

});

topicListScrollPane.setViewportView(topicList);

topicListLabel.setText("Topics Name");

jButton1.setText("/\\");

jButton1.setToolTipText("move up topic");

jButton1.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jButton3.setText("\\/");

jButton3.setToolTipText("move down topic");

jButton3.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jButton3ActionPerformed(evt);

}

});

deleteTopicButton.setText("Delete");

deleteTopicButton.setEnabled(**false**);

deleteTopicButton

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

deleteTopicButtonActionPerformed(evt);

}

});

renameButton.setText("Rename");

renameButton.setEnabled(**false**);

renameButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

renameButtonActionPerformed(evt);

}

});

javax.swing.GroupLayout jPanel2Layout = **new** javax.swing.GroupLayout(

jPanel2);

jPanel2.setLayout(jPanel2Layout);

jPanel2Layout

.setHorizontalGroup(jPanel2Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel2Layout

.createSequentialGroup()

.addGroup(

jPanel2Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel2Layout

.createSequentialGroup()

.addGap(47,

47,

47)

.addComponent(

renameButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

81,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addGap(18,

18,

18)

.addComponent(

deleteTopicButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

73,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addGroup(

jPanel2Layout

.createSequentialGroup()

.addContainerGap()

.addGroup(

jPanel2Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

topicListScrollPane,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

249,

Short.*MAX\_VALUE*)

.addGroup(

jPanel2Layout

.createSequentialGroup()

.addComponent(

topicListLabel)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*,

101,

Short.*MAX\_VALUE*)

.addComponent(

jButton1)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

jButton3)))))

.addContainerGap()));

jPanel2Layout

.setVerticalGroup(jPanel2Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel2Layout

.createSequentialGroup()

.addContainerGap()

.addGroup(

jPanel2Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(jButton1)

.addComponent(jButton3)

.addComponent(

topicListLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

26,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

topicListScrollPane,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

307, Short.*MAX\_VALUE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

jPanel2Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

renameButton)

.addComponent(

deleteTopicButton))

.addContainerGap()));

descriptionTextArea.setColumns(20);

descriptionTextArea.setFont(**new** java.awt.Font("Trebuchet MS", 0, 14));

descriptionTextArea.setRows(5);

descriptionTextArea.setEnabled(**false**);

jScrollPane3.setViewportView(descriptionTextArea);

addDescriptionButton.setText("save description");

addDescriptionButton

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

addDescriptionButtonActionPerformed(evt);

}

});

backButton.setText("Previous");

backButton.addMouseListener(**new** java.awt.event.MouseAdapter() {

**public** **void** mouseClicked(java.awt.event.MouseEvent evt) {

backButtonMouseClicked(evt);

}

});

backButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

backButtonActionPerformed(evt);

}

});

addContentButton.setText("Next");

addContentButton.setEnabled(**false**);

addContentButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

addContentButtonActionPerformed(evt);

}

});

messageLabel.setFont(**new** java.awt.Font("Trebuchet MS", 3, 14));

messageLabel.setForeground(**new** java.awt.Color(0, 153, 51));

jLabel1.setText("Topic Description");

javax.swing.GroupLayout jPanel3Layout = **new** javax.swing.GroupLayout(

jPanel3);

jPanel3.setLayout(jPanel3Layout);

jPanel3Layout

.setHorizontalGroup(jPanel3Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel3Layout

.createSequentialGroup()

.addContainerGap()

.addGroup(

jPanel3Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

jScrollPane3,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

497,

Short.*MAX\_VALUE*)

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

jPanel3Layout

.createSequentialGroup()

.addComponent(

addDescriptionButton)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*UNRELATED*)

.addComponent(

backButton)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

addContentButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

80,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addGroup(

jPanel3Layout

.createSequentialGroup()

.addComponent(

jLabel1)

.addGap(37,

37,

37)

.addComponent(

messageLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

132,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)))

.addContainerGap()));

jPanel3Layout

.setVerticalGroup(jPanel3Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel3Layout

.createSequentialGroup()

.addGap(15, 15, 15)

.addGroup(

jPanel3Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(jLabel1)

.addComponent(

messageLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

19,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

jScrollPane3,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

303,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*UNRELATED*)

.addGroup(

jPanel3Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

addContentButton)

.addComponent(

backButton)

.addComponent(

addDescriptionButton))

.addContainerGap(

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)));

jMenu1.setText("File");

jMenuItem1.setText("Exit");

jMenu1.add(jMenuItem1);

jMenuBar1.add(jMenu1);

jMenu2.setText("Edit");

undoMenuItem.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_Z*,

java.awt.event.InputEvent.*CTRL\_MASK*));

undoMenuItem.setText("Undo");

undoMenuItem.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

undoMenuItemActionPerformed(evt);

}

});

jMenu2.add(undoMenuItem);

redoMenuItem.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_Y*,

java.awt.event.InputEvent.*CTRL\_MASK*));

redoMenuItem.setText("Redo");

redoMenuItem.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

redoMenuItemActionPerformed(evt);

}

});

jMenu2.add(redoMenuItem);

jMenuBar1.add(jMenu2);

jMenu3.setText("Help");

jMenuItem3.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_H*,

java.awt.event.InputEvent.*ALT\_MASK*));

jMenuItem3.setText("Contents");

jMenuItem3.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jMenuItem3ActionPerformed(evt);

}

});

jMenu3.add(jMenuItem3);

jMenuItem2.setText("About");

jMenu3.add(jMenuItem2);

jMenuBar1.add(jMenu3);

setJMenuBar(jMenuBar1);

javax.swing.GroupLayout layout = **new** javax.swing.GroupLayout(

getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addContainerGap()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

topPanel,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

796, Short.*MAX\_VALUE*)

.addComponent(

jPanel1,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)

.addGroup(

layout.createSequentialGroup()

.addGap(2, 2, 2)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

topicInfoPanel,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)

.addGroup(

layout.createSequentialGroup()

.addComponent(

jPanel2,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

jPanel3,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)))))

.addContainerGap()));

layout.setVerticalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addComponent(topPanel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

81,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addGap(5, 5, 5)

.addComponent(jPanel1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(topicInfoPanel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*)

.addComponent(

jPanel2,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addComponent(

jPanel3,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addContainerGap(81, Short.*MAX\_VALUE*)));

pack();

}// </editor-fold>

//GEN-END:initComponents

**private** **void** topicListMouseClicked(java.awt.event.MouseEvent evt) {

// **TODO** add your handling code here:

descriptionTextArea.setFocusable(isDisplayable());

}

/\*

\* addDescriptionButton button saves the text present

\* in descriptionTextArea for topic selected from topicList

\*/

**protected** **void** addDescriptionButtonActionPerformed(ActionEvent evt) {

// **TODO** Auto-generated method stub

Integration.*setTopicDescription*(**this**);

/\* if (model.getSize() > 1) {

descriptionTextArea.setText("");

}

\*/

Thread message = **new** Thread(**new** Runnable() {

@Override

**public** **void** run() {

// **TODO** Auto-generated method stub

messageLabel.setText("description saved.");

**try** {

Thread.*sleep*(2000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

messageLabel.setText("");

}

});

message.start();

}

**private** **void** jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt) {

System.*out*.println("entered into help");

run = Runtime.*getRuntime*();

**try** {

File currentDir = **new** File(".");

String root = currentDir.getCanonicalPath();

System.*out*.println(root);

String helpLocation = root + File.*separatorChar* + "help"

+ File.*separatorChar* + "Help\_window.chm";

child = Runtime.*getRuntime*().exec(

"rundll32 url.dll,FileProtocolHandler " + helpLocation);

} **catch** (Exception ex) {

ex.printStackTrace();

System.*out*.println(ex.getMessage());

}

}

**private** **void** redoMenuItemActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

**try** {

manager.redo();

} **catch** (Exception ex) {

}

}

**private** **void** undoMenuItemActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

**try** {

manager.undo();

} **catch** (Exception ex) {

}

}

**protected** **void** addTopicButtonMouseClicked(MouseEvent evt) {

// **TODO** Auto-generated method stub

}

//move up button action performed

**private** **void** jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

**int** moveMe = topicList.getSelectedIndex();

**if** (moveMe != 0) {

//except at top position

swap(moveMe, moveMe - 1);

topicList.setSelectedIndex(moveMe - 1);

topicList.ensureIndexIsVisible(moveMe - 1);

}

}

//move down button action performed

**private** **void** jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

**int** moveMe = topicList.getSelectedIndex();

**if** (moveMe != model.getSize() - 1) {

// except at bottom position

swap(moveMe, moveMe + 1);

topicList.setSelectedIndex(moveMe + 1);

topicList.ensureIndexIsVisible(moveMe + 1);

}

}

**protected** **void** backButtonMouseClicked(MouseEvent evt) {

// **TODO** Auto-generated method stub

}

**private** **void** topicNameTextFieldActionPerformed(

java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

}

/\* backButton button take the control to the previous

\* page i.e to the MasterPage

\*/

**protected** **void** backButtonActionPerformed(ActionEvent evt) {

**this**.setVisible(**false**);

masterPage.setAddTopicForm(**this**);

masterPage.setVisible(**true**);

}

**private** **void** addContentButtonActionPerformed(java.awt.event.ActionEvent evt) {

**if** (addContentPage == **null**) {

addContentPage = **new** AddContentPage(**this**);

}

**this**.setVisible(**false**);

System.*out*.println("calling displayContentPage: " + addContentPage);

addContentPage.displayContentPage(**true**);

}

**private** **void** swap(**int** a, **int** b) {

Object aObject = model.getElementAt(a);

Object bObject = model.getElementAt(b);

model.set(a, bObject);

model.set(b, aObject);

Integration.*swapTopic*(a, b);

}

/\* renameButton performs the renaming the selected TopicName

\* it doesn't allow the blank field .or renaming the Topic with the same name which

\* is already present in the TopicName list.

\*/

**private** **void** renameButtonActionPerformed(java.awt.event.ActionEvent evt) {

**try** {

**if** (!topicList.isSelectionEmpty()) {

String newTopicName = JOptionPane.*showInputDialog*(

"Enter new name for the Topic : ",

topicList.getSelectedValue());

newTopicName = newTopicName.trim();

**if** (newTopicName.length() == 0) {

JOptionPane.*showMessageDialog*(**this**,

"Please enter topic name", "Error",

JOptionPane.*ERROR\_MESSAGE*);

} **else** **if** (model.contains(newTopicName)) {

JOptionPane.*showMessageDialog*(**this**, "Topic already exists",

"Error", JOptionPane.*ERROR\_MESSAGE*);

} **else** {

**int** topicIndex = topicList.getSelectedIndex();

model.set(topicIndex, newTopicName);

Integration.*renameTopic*(newTopicName, topicIndex);

}

} **else** {

JOptionPane.*showMessageDialog*(**this**, "Please Select Any Topic",

"Error", JOptionPane.*ERROR\_MESSAGE*);

}

} **catch** (java.lang.ArrayIndexOutOfBoundsException e) {

JOptionPane.*showMessageDialog*(**this**,

"Please select any sub topic!!", "Error",

JOptionPane.*ERROR\_MESSAGE*);

}

}

**private** **void** topicListValueChanged(javax.swing.event.ListSelectionEvent evt) {

addContentButton.setEnabled(**true**);

deleteTopicButton.setEnabled(**true**);

renameButton.setEnabled(**true**);

descriptionTextArea.setCaretColor(Color.*blue*);

//descriptionTextArea.setCursor(null);

**int** topicIndex = topicList.getSelectedIndex();

//for debugging purpose

System.*out*.println("index,model-size value before if condition : "

+ topicIndex + "," + model.getSize());

**if** (topicIndex == -1 && model.getSize() >= 0) {

topicIndex = 0;

**int** topicListSize = Integration.*module*.topicList.size();

System.*out*.println("size of topic list : " + topicListSize);

**if** (topicListSize > 0) {

descriptionTextArea.setText(Integration.*module*.topicList.get(

topicIndex).getTopicDescription());

}

} **else** **if** (topicIndex > -1) {

//System.out.println("index value : " + topicIndex);//for debugging purpose

**int** topicListSize = Integration.*module*.topicList.size();

System.*out*.println("size of topic list : " + topicListSize);

**if** (topicListSize > 0) {

descriptionTextArea.setText(Integration.*module*.topicList.get(

topicIndex).getTopicDescription());

}

}

System.*out*.println("inside topic list value changed--->topic index : "

+ topicIndex);

descriptionTextArea.setEnabled(**true**);

descriptionTextArea.setEditable(**true**);

descriptionTextArea.requestFocusInWindow();

//descriptionTextArea.setText("");

isModifiable(**true**);

topicList.validate();

}

/\* addTopicButton button add new topic(TopicTextField) to the topicList,

\* it trims the text present in topicTextField then

\* it checks whether the particular topic already present or not

\* in topicList or not if its already present it gives warning

\* and also checks whether topicName is provided properly not.

\* i.e it can not accept the blank text field as topic name.

\*

\*/

**protected** **void** addTopicButtonActionPerformed(ActionEvent evt) {

String newTopic = topicNameTextField.getText();

newTopic = newTopic.trim();

**if** (newTopic.length() == 0) {

JOptionPane.*showMessageDialog*(**this**, "Please enter topic-name",

"Error", JOptionPane.*ERROR\_MESSAGE*);

}

**else** **if** (model.contains(newTopic)) {

JOptionPane.*showMessageDialog*(**this**, "Topic already exists",

"Error", JOptionPane.*ERROR\_MESSAGE*);

} **else** {

newTopic = newTopic.trim();

model.addElement(newTopic);

descriptionTextArea.setText("");

//adding Integration function....

Integration.*setTopicPage*(newTopic);

}

System.*out*.println(model.contains(newTopic.trim()));

topicNameTextField.setText("");

descriptionTextArea.setText("");

topicList.repaint();

}

// \* deleteTopicButton delete the selected Topic form the list

**protected** **void** deleteTopicButtonActionPerformed(ActionEvent evt) {

**if** (!topicList.isSelectionEmpty()) {

System.*out*.println("inside delete action.........");

**int** topicIndex = topicList.getSelectedIndex();

System.*out*.println("topic index : " + topicIndex

+ " \t model-size : " + model.getSize());

model.remove(topicIndex);

descriptionTextArea.setText("");

Integration.*deleteTopic*(topicIndex);

//model.removeElementAt(topicIndex);

**if** (model.size() == 0) {

isModifiable(**false**);

}

} **else** {

JOptionPane.*showMessageDialog*(**this**, "Please Select Any Topic",

"Error", JOptionPane.*ERROR\_MESSAGE*);

}

}

/\*

\* this method is used for setting property of button

\* i.e setEnabled true or false

\*/

**protected** **void** isModifiable(**boolean** setValue) {

deleteTopicButton.setEnabled(setValue);

renameButton.setEnabled(setValue);

addContentButton.setEnabled(setValue);

jButton1.setEnabled(setValue);

jButton3.setEnabled(setValue);

}

**private** **void** exitMenuItemActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_exitMenuItemActionPerformed

System.*exit*(0);

}//GEN-LAST:event\_exitMenuItemActionPerformed

/\*\*

\* **@param** args the command line arguments

\*/

**public** String getTopicName() {

// **TODO** Auto-generated method stub

**return** (String) topicList.getSelectedValue();

}

**public** **static** **void** main(String args[]) {

java.awt.EventQueue.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**new** AddTopicForm().setVisible(**true**);

}

});

}

//GEN-BEGIN:variables

// Variables declaration - do not modify

**private** javax.swing.JMenuItem aboutMenuItem;

**private** javax.swing.JButton addContentButton;

**private** javax.swing.JButton addDescriptionButton;

**private** javax.swing.JButton addTopicButton;

**private** javax.swing.JButton backButton;

**private** javax.swing.JMenuItem contentsMenuItem;

**private** javax.swing.JMenuItem copyMenuItem;

**private** javax.swing.JMenuItem cutMenuItem;

**private** javax.swing.JMenuItem deleteMenuItem;

**private** javax.swing.JButton deleteTopicButton;

**public** javax.swing.JTextArea descriptionTextArea;

**private** javax.swing.JMenu editMenu;

**private** javax.swing.JMenuItem exitMenuItem;

**private** javax.swing.JMenu fileMenu;

**private** javax.swing.JMenu helpMenu;

**private** javax.swing.JLabel imageLabel;

**private** javax.swing.JButton jButton1;

**private** javax.swing.JButton jButton3;

**private** javax.swing.JLabel jLabel1;

**private** javax.swing.JMenu jMenu1;

**private** javax.swing.JMenu jMenu2;

**private** javax.swing.JMenu jMenu3;

**private** javax.swing.JMenuBar jMenuBar1;

**private** javax.swing.JMenuItem jMenuItem1;

**private** javax.swing.JMenuItem jMenuItem2;

**private** javax.swing.JMenuItem jMenuItem3;

**private** javax.swing.JPanel jPanel1;

**private** javax.swing.JPanel jPanel2;

**private** javax.swing.JPanel jPanel3;

**private** javax.swing.JScrollPane jScrollPane3;

**private** javax.swing.JMenuBar menuBar;

**private** javax.swing.JLabel messageLabel;

**private** javax.swing.JLabel moduleSubjectLabel;

**private** javax.swing.JLabel moduleSubjectNameLabel;

**private** javax.swing.JMenuItem openMenuItem;

**private** javax.swing.JMenuItem pasteMenuItem;

**private** javax.swing.JMenuItem redoMenuItem;

**private** javax.swing.JButton renameButton;

**private** javax.swing.JMenuItem saveAsMenuItem;

**private** javax.swing.JMenuItem saveMenuItem;

**private** javax.swing.JPanel topPanel;

**private** javax.swing.JPanel topicInfoPanel;

**public** javax.swing.JList topicList;

**private** javax.swing.JLabel topicListLabel;

**private** javax.swing.JScrollPane topicListScrollPane;

**private** javax.swing.JLabel topicNameLabel;

**public** javax.swing.JTextField topicNameTextField;

**private** javax.swing.JMenuItem undoMenuItem;

// End of variables declaration//GEN-END:variables

**protected** MasterPage masterPage;

**private** javax.swing.DefaultListModel model;

**private** AddContentPage addContentPage;

**public** UndoManager manager;

**private** Runtime run;

**private** Process child;

**public** **void** displayTopicForm(**boolean** value) {

moduleSubjectNameLabel.setText(masterPage.getModuleSubjectTextField());

setVisible(value);

}

}

//EmptyListException.java

**package** com.toolkit;

**public** **class** EmptyListException **extends** Exception {

**public** String toString(){

**return** "The List is Empty!!\nPlease enter any entry!!";

}

}

//FileUpload.java

/\*

\* FileUpload.java

\*

\* Created on \_\_DATE\_\_, \_\_TIME\_\_

\*/

**package** com.toolkit;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.FileOutputStream;

**import** java.io.IOException;

**import** javax.swing.JFileChooser;

/\*\*

\*

\* **@author** \_\_USER\_\_

\*/

**public** **class** FileUpload **extends** javax.swing.JFrame {

/\*\*

\*

\*/

**private** **static** **final** **long** *serialVersionUID* = 1L;

/\*\* Creates new form FileUpload \*/

**public** FileUpload() {

initComponents();

}

/\*\* This method is called from within the constructor to

\* initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is

\* always regenerated by the Form Editor.

\*/

//GEN-BEGIN:initComponents

// <editor-fold defaultstate="collapsed" desc="Generated Code">

**private** **void** initComponents() {

jButton1 = **new** javax.swing.JButton();

l = **new** javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.*EXIT\_ON\_CLOSE*);

jButton1.setText("browse...");

jButton1.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

**try** {

jButton1ActionPerformed(evt);

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

});

l.setText("file name will be");

javax.swing.GroupLayout layout = **new** javax.swing.GroupLayout(

getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addContainerGap()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(jButton1)

.addComponent(l))

.addContainerGap(321, Short.*MAX\_VALUE*)));

layout.setVerticalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addContainerGap()

.addComponent(jButton1)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(l)

.addContainerGap(266, Short.*MAX\_VALUE*)));

pack();

}// </editor-fold>

//GEN-END:initComponents

**private** **void** jButton1ActionPerformed(java.awt.event.ActionEvent evt)

**throws** IOException {

// **TODO** add your handling code here:

JFileChooser jf = **new** JFileChooser();

jf.showOpenDialog(**this**);

File file = jf.getSelectedFile();

l.setText("");

l.setText(file.getName() + file.length());

**this**.add(l);

**this**.validate();

**this**.setVisible(**true**);

FileInputStream fis = **new** FileInputStream(file);

**byte**[] buffer = **new** **byte**[(**int**) file.length()];

fis.read(buffer);

FileOutputStream fos = **new** FileOutputStream("videos\\"+file.getName());

fos.write(buffer);

//fos

fis.close();

fos.close();

}

/\*\*

\* **@param** args the command line arguments

\*/

**public** **static** **void** main(String args[]) {

java.awt.EventQueue.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**new** FileUpload().setVisible(**true**);

}

});

}

//GEN-BEGIN:variables

// Variables declaration - do not modify

**private** javax.swing.JButton jButton1;

**private** javax.swing.JLabel l;

// End of variables declaration//GEN-END:variables

}

//FinishForm.java

/\*

\* layout\_4.java

\*

\* Created on \_\_DATE\_\_, \_\_TIME\_\_

\*/

**package** com.toolkit;

**import** java.awt.event.ActionEvent;

**import** javax.swing.undo.\*;

**import** java.io.File;

**import** java.io.FileNotFoundException;

**import** java.io.IOException;

**import** javax.swing.ImageIcon;

**import** javax.swing.JOptionPane;

**import** com.html.HtmlUtility;

/\*\*

\* this is used for proving references and saving all

\* the data related to this module

\*/

**public** **class** FinishForm **extends** javax.swing.JFrame {

/\*\*

\*

\*/

**private** **static** **final** **long** *serialVersionUID* = 1L;

/\*\* Creates new form layout\_4 \*/

**public** FinishForm() {

initComponents();

setSize(900, 700);

setResizable(**false**);

ImageIcon module\_strip = **new** ImageIcon("images/module\_strip.png");

//toolIcon.

imageLabel.setIcon(module\_strip);

/\*

ImageIcon titleIcon = new ImageIcon("images/tool\_title\_image.png");

titleLabel.setIcon(titleIcon);

ImageIcon companyIcon = new ImageIcon("images/zensar\_logo\_2.jpg");

companyLabel.setIcon(companyIcon);

\*/

manager = **new** UndoManager();

emailTextField.getDocument().addUndoableEditListener(manager);

//referenceTextArea.getDocument().addUndoableEditListener(manager);

}

**public** FinishForm(AddContentPage addContentPage) {

**this**();

**this**.addContentPage = addContentPage;

// **TODO** Auto-generated constructor stub

}

//GEN-BEGIN:initComponents

// <editor-fold defaultstate="collapsed" desc="Generated Code">

**private** **void** initComponents() {

java.awt.GridBagConstraints gridBagConstraints;

menuBar = **new** javax.swing.JMenuBar();

fileMenu = **new** javax.swing.JMenu();

openMenuItem = **new** javax.swing.JMenuItem();

saveMenuItem = **new** javax.swing.JMenuItem();

saveAsMenuItem = **new** javax.swing.JMenuItem();

exitMenuItem = **new** javax.swing.JMenuItem();

editMenu = **new** javax.swing.JMenu();

cutMenuItem = **new** javax.swing.JMenuItem();

copyMenuItem = **new** javax.swing.JMenuItem();

pasteMenuItem = **new** javax.swing.JMenuItem();

deleteMenuItem = **new** javax.swing.JMenuItem();

helpMenu = **new** javax.swing.JMenu();

contentsMenuItem = **new** javax.swing.JMenuItem();

aboutMenuItem = **new** javax.swing.JMenuItem();

jPanel1 = **new** javax.swing.JPanel();

jLabel1 = **new** javax.swing.JLabel();

jLabel2 = **new** javax.swing.JLabel();

emailTextField = **new** javax.swing.JTextField();

linkURLTextField1 = **new** javax.swing.JTextField();

linkURLTextField2 = **new** javax.swing.JTextField();

linkURLTextField3 = **new** javax.swing.JTextField();

linkNameTextField2 = **new** javax.swing.JTextField();

linkNameTextField1 = **new** javax.swing.JTextField();

linkNameTextField3 = **new** javax.swing.JTextField();

jLabel3 = **new** javax.swing.JLabel();

jLabel4 = **new** javax.swing.JLabel();

infoLabel1 = **new** javax.swing.JLabel();

jPanel3 = **new** javax.swing.JPanel();

moduleSubjectNameLabel = **new** javax.swing.JLabel();

moduleSubjectLabel = **new** javax.swing.JLabel();

finishButton = **new** javax.swing.JButton();

previousButton = **new** javax.swing.JButton();

topPanel = **new** javax.swing.JPanel();

imageLabel = **new** javax.swing.JLabel();

menuBar1 = **new** javax.swing.JMenuBar();

fileMenu1 = **new** javax.swing.JMenu();

openMenuItem1 = **new** javax.swing.JMenuItem();

saveMenuItem1 = **new** javax.swing.JMenuItem();

saveAsMenuItem1 = **new** javax.swing.JMenuItem();

exitMenuItem1 = **new** javax.swing.JMenuItem();

editMenu1 = **new** javax.swing.JMenu();

jMenuItem1 = **new** javax.swing.JMenuItem();

jMenuItem2 = **new** javax.swing.JMenuItem();

cutMenuItem1 = **new** javax.swing.JMenuItem();

copyMenuItem1 = **new** javax.swing.JMenuItem();

pasteMenuItem1 = **new** javax.swing.JMenuItem();

deleteMenuItem1 = **new** javax.swing.JMenuItem();

helpMenu1 = **new** javax.swing.JMenu();

contentsMenuItem1 = **new** javax.swing.JMenuItem();

aboutMenuItem1 = **new** javax.swing.JMenuItem();

setDefaultCloseOperation(javax.swing.WindowConstants.*EXIT\_ON\_CLOSE*);

fileMenu.setText("File");

openMenuItem.setText("Open");

fileMenu.add(openMenuItem);

saveMenuItem.setText("Save");

fileMenu.add(saveMenuItem);

saveAsMenuItem.setText("Save As ...");

fileMenu.add(saveAsMenuItem);

exitMenuItem.setText("Exit");

exitMenuItem.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

exitMenuItemActionPerformed(evt);

}

});

fileMenu.add(exitMenuItem);

menuBar.add(fileMenu);

editMenu.setText("Edit");

cutMenuItem.setText("Cut");

editMenu.add(cutMenuItem);

copyMenuItem.setText("Copy");

editMenu.add(copyMenuItem);

pasteMenuItem.setText("Paste");

editMenu.add(pasteMenuItem);

deleteMenuItem.setText("Delete");

editMenu.add(deleteMenuItem);

menuBar.add(editMenu);

helpMenu.setText("Help");

contentsMenuItem.setText("Contents");

helpMenu.add(contentsMenuItem);

aboutMenuItem.setText("About");

helpMenu.add(aboutMenuItem);

menuBar.add(helpMenu);

jPanel1.setBackground(**new** java.awt.Color(204, 204, 204));

jPanel1.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(102, 204, 255)));

jLabel1.setText("For more references you can visit the links");

jLabel2.setText("For any query mail me at");

linkURLTextField1.setBackground(**new** java.awt.Color(255, 255, 204));

linkURLTextField2.setBackground(**new** java.awt.Color(255, 255, 204));

linkURLTextField3.setBackground(**new** java.awt.Color(255, 255, 204));

jLabel3.setText("Link Name");

jLabel4.setText("Link URL");

infoLabel1

.setText("Your module is almost created,Press the FINISH button to complete the process.");

javax.swing.GroupLayout jPanel1Layout = **new** javax.swing.GroupLayout(

jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout

.setHorizontalGroup(jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addContainerGap()

.addGroup(

jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addGroup(

jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addGap(8,

8,

8)

.addGroup(

jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

linkNameTextField1,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

194,

Short.*MAX\_VALUE*)

.addComponent(

linkNameTextField2,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

194,

Short.*MAX\_VALUE*)

.addComponent(

linkNameTextField3,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

194,

Short.*MAX\_VALUE*))

.addGap(10,

10,

10))

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addGap(10,

10,

10)

.addComponent(

jLabel3,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

92,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)))

.addGroup(

jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*)

.addComponent(

linkURLTextField3,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

342,

Short.*MAX\_VALUE*)

.addComponent(

linkURLTextField2,

javax.swing.GroupLayout.Alignment.*LEADING*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

342,

Short.*MAX\_VALUE*)

.addGroup(

javax.swing.GroupLayout.Alignment.*LEADING*,

jPanel1Layout

.createSequentialGroup()

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

jLabel4,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

169,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addComponent(

linkURLTextField1,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

342,

Short.*MAX\_VALUE*))))

.addGap(68,

68,

68))

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addComponent(

jLabel1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

275,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addContainerGap(

347,

Short.*MAX\_VALUE*))

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addComponent(

jLabel2,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

142,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

emailTextField,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

261,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addContainerGap(

215,

Short.*MAX\_VALUE*))

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addComponent(

infoLabel1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

503,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addContainerGap(

119,

Short.*MAX\_VALUE*)))));

jPanel1Layout

.setVerticalGroup(jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addContainerGap()

.addComponent(

jLabel1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

34,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

jLabel4,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

20,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addComponent(

jLabel3,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

20,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addComponent(

linkNameTextField1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

29,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

linkNameTextField2,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

29,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

linkNameTextField3,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

29,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addGroup(

jPanel1Layout

.createSequentialGroup()

.addComponent(

linkURLTextField1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

29,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

linkURLTextField2,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

29,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

linkURLTextField3,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

29,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)))

.addGap(34, 34, 34)

.addGroup(

jPanel1Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

jLabel2,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

38,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addComponent(

emailTextField,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

29,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addGap(30, 30, 30)

.addComponent(

infoLabel1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

14,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addGap(19, 19, 19)));

jPanel3.setBackground(**new** java.awt.Color(204, 204, 204));

jPanel3.setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(51, 204, 255)));

moduleSubjectNameLabel.setText("subject name from lay\_1");

moduleSubjectLabel.setText("MODULE SUBJECT :");

javax.swing.GroupLayout jPanel3Layout = **new** javax.swing.GroupLayout(

jPanel3);

jPanel3.setLayout(jPanel3Layout);

jPanel3Layout

.setHorizontalGroup(jPanel3Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

jPanel3Layout

.createSequentialGroup()

.addContainerGap()

.addComponent(

moduleSubjectLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

120,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*UNRELATED*)

.addComponent(

moduleSubjectNameLabel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

152,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addContainerGap(340, Short.*MAX\_VALUE*)));

jPanel3Layout

.setVerticalGroup(jPanel3Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

jPanel3Layout

.createSequentialGroup()

.addContainerGap(18, Short.*MAX\_VALUE*)

.addGroup(

jPanel3Layout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*,

**false**)

.addComponent(

moduleSubjectLabel,

javax.swing.GroupLayout.Alignment.*TRAILING*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)

.addComponent(

moduleSubjectNameLabel,

javax.swing.GroupLayout.Alignment.*TRAILING*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

29,

Short.*MAX\_VALUE*))

.addContainerGap()));

finishButton.setText("Finish");

finishButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

finishButtonActionPerformed(evt);

}

});

previousButton.setText("Previous");

previousButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

previousButtonActionPerformed(evt);

}

});

topPanel.setLayout(**new** java.awt.GridBagLayout());

imageLabel.setText(" ");

gridBagConstraints = **new** java.awt.GridBagConstraints();

gridBagConstraints.gridheight = 3;

gridBagConstraints.fill = java.awt.GridBagConstraints.*BOTH*;

topPanel.add(imageLabel, gridBagConstraints);

fileMenu1.setText("File");

openMenuItem1.setText("Open");

fileMenu1.add(openMenuItem1);

saveMenuItem1.setText("Save");

fileMenu1.add(saveMenuItem1);

saveAsMenuItem1.setText("Save As ...");

fileMenu1.add(saveAsMenuItem1);

exitMenuItem1.setText("Exit");

exitMenuItem1.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

exitMenuItem1exitMenuItemActionPerformed(evt);

}

});

fileMenu1.add(exitMenuItem1);

menuBar1.add(fileMenu1);

editMenu1.setText("Edit");

jMenuItem1.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_Z*,

java.awt.event.InputEvent.*CTRL\_MASK*));

jMenuItem1.setText("Undo");

jMenuItem1.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jMenuItem1ActionPerformed(evt);

}

});

editMenu1.add(jMenuItem1);

jMenuItem2.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_Y*,

java.awt.event.InputEvent.*CTRL\_MASK*));

jMenuItem2.setText("Redo");

jMenuItem2.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jMenuItem2ActionPerformed(evt);

}

});

editMenu1.add(jMenuItem2);

cutMenuItem1.setText("Cut");

editMenu1.add(cutMenuItem1);

copyMenuItem1.setText("Copy");

editMenu1.add(copyMenuItem1);

pasteMenuItem1.setText("Paste");

editMenu1.add(pasteMenuItem1);

deleteMenuItem1.setText("Delete");

editMenu1.add(deleteMenuItem1);

menuBar1.add(editMenu1);

helpMenu1.setText("Help");

helpMenu1.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

helpMenu1ActionPerformed(evt);

}

});

contentsMenuItem1.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_H*,

java.awt.event.InputEvent.*ALT\_MASK*));

contentsMenuItem1.setText("Contents");

contentsMenuItem1.addMouseListener(**new** java.awt.event.MouseAdapter() {

**public** **void** mouseClicked(java.awt.event.MouseEvent evt) {

contentsMenuItem1MouseClicked(evt);

}

});

contentsMenuItem1

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

contentsMenuItem1ActionPerformed(evt);

}

});

helpMenu1.add(contentsMenuItem1);

aboutMenuItem1.setText("About");

helpMenu1.add(aboutMenuItem1);

menuBar1.add(helpMenu1);

setJMenuBar(menuBar1);

javax.swing.GroupLayout layout = **new** javax.swing.GroupLayout(

getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addContainerGap()

.addComponent(

topPanel,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

990,

Short.*MAX\_VALUE*))

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

layout.createSequentialGroup()

.addComponent(

previousButton)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

finishButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

77,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addGroup(

layout.createSequentialGroup()

.addGap(125,

125,

125)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

**false**)

.addComponent(

jPanel1,

javax.swing.GroupLayout.Alignment.*LEADING*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)

.addComponent(

jPanel3,

javax.swing.GroupLayout.Alignment.*LEADING*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*))))

.addContainerGap()));

layout.setVerticalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addComponent(topPanel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

87,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addGap(11, 11, 11)

.addComponent(jPanel3,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jPanel1,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

313, Short.*MAX\_VALUE*)

.addGap(88, 88, 88)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(finishButton)

.addComponent(previousButton))

.addContainerGap()));

pack();

}// </editor-fold>

//GEN-END:initComponents

**private** **void** helpMenu1ActionPerformed(java.awt.event.ActionEvent evt) {

}

**private** **void** contentsMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {

System.*out*.println("entered into help");

Runtime run = Runtime.*getRuntime*();

**try** {

File currentDir = **new** File(".");

String root = currentDir.getCanonicalPath();

System.*out*.println(root);

String helpLocation = root + File.*separatorChar* + "help"

+ File.*separatorChar* + "Help\_window.chm";

Process child = Runtime.*getRuntime*().exec(

"rundll32 url.dll,FileProtocolHandler " + helpLocation);

} **catch** (Exception ex) {

ex.printStackTrace();

System.*out*.println(ex.getMessage());

}

}

**private** **void** contentsMenuItem1MouseClicked(java.awt.event.MouseEvent evt) {

}

**private** **void** jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

**try** {

manager.redo();

} **catch** (Exception ex) {

}

}

**private** **void** jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

**try** {

manager.undo();

} **catch** (Exception ex) {

}

}

**private** **void** exitMenuItem1exitMenuItemActionPerformed(

java.awt.event.ActionEvent evt) {

//GEN-FIRST:event\_exitMenuItemActionPerformed

System.*exit*(0);

}

**protected** **void** previousButtonActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

**this**.setVisible(**false**);

addContentPage.setVisible(**true**);

}

/\* finishButton generates the final html

\* files and also includes the author email id and references if provided

\* by the trainer

\*/

**protected** **void** finishButtonActionPerformed(ActionEvent evt) {

// **TODO** Auto-generated method stub

**try** {

Integration.*module*.setAuthorMailId(emailTextField.getText());

Integration.*setReference*(linkNameTextField1,linkURLTextField1);

Integration.*module*.references.clear();

Integration.*setReference*(linkNameTextField1, linkURLTextField1);

Integration.*setReference*(linkNameTextField2, linkURLTextField2);

Integration.*setReference*(linkNameTextField3, linkURLTextField3);

HtmlUtility htmlUtility = **new** HtmlUtility();

String output = htmlUtility.util(Integration.*module*);

**int** optionType = JOptionPane.*OK\_CANCEL\_OPTION*;

**int** messageType = JOptionPane.*PLAIN\_MESSAGE*;

//ImageIcon icon = new ImageIcon("//images//logo.jpg", "logo");

**int** res = JOptionPane

.*showConfirmDialog*(

**this**,

"Module is successfully created and saved.!\nClick OK to exit.",

"Zensar Technologies", optionType, messageType,

**null**);

**this**.validate();

**if** (res == JOptionPane.*OK\_OPTION*) {

System.*out*.println("OK\_OPTION");

**this**.dispose();

}

} **catch** (FileNotFoundException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

// **TODO** Auto-generated method stub

}

**protected** **void** backToHomePageButtonActionPerformed(ActionEvent evt) {

// **TODO** Auto-generated method stub

**this**.setVisible(**false**);

addContentPage.addTopicForm.masterPage.setVisible(**true**);

}

**private** **void** exitMenuItemActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_exitMenuItemActionPerformed

System.*exit*(0);

}//GEN-LAST:event\_exitMenuItemActionPerformed

/\*\*

\* **@param** args the command line arguments

\*/

**public** **static** **void** main(String args[]) {

java.awt.EventQueue.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**new** FinishForm().setVisible(**true**);

}

});

}

//GEN-BEGIN:variables

// Variables declaration - do not modify

**private** javax.swing.JMenuItem aboutMenuItem;

**private** javax.swing.JMenuItem aboutMenuItem1;

**private** javax.swing.JMenuItem contentsMenuItem;

**private** javax.swing.JMenuItem contentsMenuItem1;

**private** javax.swing.JMenuItem copyMenuItem;

**private** javax.swing.JMenuItem copyMenuItem1;

**private** javax.swing.JMenuItem cutMenuItem;

**private** javax.swing.JMenuItem cutMenuItem1;

**private** javax.swing.JMenuItem deleteMenuItem;

**private** javax.swing.JMenuItem deleteMenuItem1;

**private** javax.swing.JMenu editMenu;

**private** javax.swing.JMenu editMenu1;

**private** javax.swing.JTextField emailTextField;

**private** javax.swing.JMenuItem exitMenuItem;

**private** javax.swing.JMenuItem exitMenuItem1;

**private** javax.swing.JMenu fileMenu;

**private** javax.swing.JMenu fileMenu1;

**private** javax.swing.JButton finishButton;

**private** javax.swing.JMenu helpMenu;

**private** javax.swing.JMenu helpMenu1;

**private** javax.swing.JLabel imageLabel;

**private** javax.swing.JLabel infoLabel1;

**private** javax.swing.JLabel jLabel1;

**private** javax.swing.JLabel jLabel2;

**private** javax.swing.JLabel jLabel3;

**private** javax.swing.JLabel jLabel4;

**private** javax.swing.JMenuItem jMenuItem1;

**private** javax.swing.JMenuItem jMenuItem2;

**private** javax.swing.JPanel jPanel1;

**private** javax.swing.JPanel jPanel3;

**private** javax.swing.JTextField linkNameTextField1;

**private** javax.swing.JTextField linkNameTextField2;

**private** javax.swing.JTextField linkNameTextField3;

**private** javax.swing.JTextField linkURLTextField1;

**private** javax.swing.JTextField linkURLTextField2;

**private** javax.swing.JTextField linkURLTextField3;

**private** javax.swing.JMenuBar menuBar;

**private** javax.swing.JMenuBar menuBar1;

**private** javax.swing.JLabel moduleSubjectLabel;

**private** javax.swing.JLabel moduleSubjectNameLabel;

**private** javax.swing.JMenuItem openMenuItem;

**private** javax.swing.JMenuItem openMenuItem1;

**private** javax.swing.JMenuItem pasteMenuItem;

**private** javax.swing.JMenuItem pasteMenuItem1;

**private** javax.swing.JButton previousButton;

**private** javax.swing.JMenuItem saveAsMenuItem;

**private** javax.swing.JMenuItem saveAsMenuItem1;

**private** javax.swing.JMenuItem saveMenuItem;

**private** javax.swing.JMenuItem saveMenuItem1;

**private** javax.swing.JPanel topPanel;

// End of variables declaration//GEN-END:variables

**private** AddContentPage addContentPage;

**public** UndoManager manager;

**public** String getMail() {

**return** emailTextField.getText();

}

**public** String getReference() {

**return** **null**;

}

}

//Integration.java

**package** com.toolkit;

**import** java.awt.Component;

**import** java.util.ArrayList;

**import** java.util.Iterator;

**import** javax.swing.JTextField;

/\*

\* Integration class helps in storing the values from the swing object

\* to the plain old java objects

\*/

**public** **class** Integration {

**static** {

*module*=**new** Module();

}

**public** **static** Module *module*;

/\*

\* setModulePage() saves the module details in the Module object

\*/

**public** **static** **void** setModulePage(MasterPage master){

*module*.setModuleID(master.moduleIdTextField.getText());

*module*.setModuleName(master.moduleSubjectTextField.getText());

*module*.setAuthorName(master.authorNameTextField.getText());

String mid= *module*.getModuleID();

String an=*module*.getAuthorName();

String mn=*module*.getModuleName();

System.*out*.println("module added--->"+*module*);

}

/\*

\* setTopicPage() saves the topic details inside Topic object which gets stored in topic-list of Module object

\*/

**public** **static** **void** setTopicPage(String topicName){

Topic topic = **new** Topic();

//topic.setTopicName(topicPage.topicNameTextField.getText().trim());

topic.setTopicName(topicName);

Integration.*module*.getTopicList().add(topic);

System.*out*.println("topic list--->"+*module*.getTopicList());

}

/\*

\* setSubtopicName() saves the subtopic details in the SubTopic object which gets stored in sub topic list of Topic object

\*/

**public** **static** **void** setSubtopicName(String topicName,String subTopicName) {

//String topicName=addContentPage.getTopicName();

System.*out*.println("SET SUB TOPIC NAME ---> "+*module*+"\t list---->"+*module*.topicList);

System.*out*.println(topicName);

Topic topic=*module*.getTopicByName(topicName);

System.*out*.println("topic----> "+topic);

SubTopic subtopic=**new** SubTopic();

//System.out.println(addContentPage.subTopicList.getSelectedValue());

System.*out*.println(subTopicName);

subtopic.setSubTopicName(subTopicName);

topic.getSubTopicList().add(subtopic);

System.*out*.println("subtopic list----->"+topic.getSubTopicList());

}

/\*

\* setSubTopicContent() saves the sub toipc contents in the SubTopic object which is already stored inside the Topic object

\*/

**public** **static** **void** setSubTopicContent(String topicName,String subTopicName,Component panel){

Topic topic=*module*.getTopicByName(topicName);

SubTopic subtopic=topic.getSubTopicByName(subTopicName);

System.*out*.println("inside set sub topic content \*\*\*\*\*\*\*\*\*\*\*");

String key="";

String value="";

System.*out*.println("content map ----> "+subtopic.contentMap);

**if** (panel **instanceof** AddTextJPanel)

{

AddTextJPanel tempPanel = (AddTextJPanel) panel;

key=tempPanel.toString();

value=tempPanel.getText();

System.*out*.println("AddTextJPanel found---->"+key+"------->"+value);

}

**else** **if** (panel **instanceof** AddCodePanel)

{

AddCodePanel tempPanel = (AddCodePanel) panel;

key=tempPanel.toString();

value=tempPanel.getText();

System.*out*.println("AddCodeJPanel found---->"+key+"------->"+value);

}

**else** **if** (panel **instanceof** AddMediaJPanel)

{

AddMediaJPanel tempPanel = (AddMediaJPanel) panel;

key=tempPanel.toString();

value=tempPanel.getMediaInfo();

System.*out*.println("AddMediaJPanel found---->"+key+"------->"+value);

}

subtopic.contentMap.put(key, value);

System.*out*.println("Values of content map: " + subtopic.contentMap.get(key));

System.*out*.println("content map @ end of the set sub topic content "+subtopic.contentMap);

}

/\*

\* setTopicDescription() saves the topic description inside the Topic object which is already stored in Module object

\*/

**public** **static** **void** setTopicDescription(AddTopicForm addTopicForm) {

**int** topicIndex=addTopicForm.topicList.getSelectedIndex();

*module*.topicList.get(topicIndex).setTopicDescription(addTopicForm.descriptionTextArea.getText());

}

/\*

\* setSubTopicJList() helps in loading the subtopic upon navigating to a different topic.

\*/

**public** **static** **void** setSubTopicJList(String topicName,AddContentPage addContentPage) {

// **TODO** Auto-generated method stub

Topic topic=*module*.getTopicByName(topicName);

System.*out*.println("topic name : "+topicName+"--------topic object :"+topic);

**if**(topic.subTopicList!=**null**){

**for**(SubTopic subtopic:topic.subTopicList){

addContentPage.model.addElement(subtopic.getSubTopicName());

}

}

}

/\*

\* removeSubTopicContents() removes the given SubTopic object from the given Topic object

\*/

**public** **static** **void** removeSubTopicContents(String topicName,

String subTopicName) {

Topic topic=*module*.getTopicByName(topicName);

SubTopic subtopic=topic.getSubTopicByName(subTopicName);

subtopic.contentMap.clear();

}

/\*

\* getSubTopicContent() loads the subtopic content in the contentPanel of AddContentPage

\* whenever the trainer navigates to a different SubTopic in the list

\*/

**public** **static** **void** getSubTopicContent(String topicName,String subTopicName,AddContentPage addContentPage) {

// **TODO** Auto-generated method stub

String message=**null**;

System.*out*.println("RETRIVING CONTENTS OF SUBTOPICS....");

Topic topic=*module*.getTopicByName(topicName);

SubTopic subtopic=topic.getSubTopicByName(subTopicName);

//System.out.println("subtopic before if condition : "+subtopic);

**if**(subtopic!=**null** && subtopic.contentMap!=**null**){

System.*out*.println("subtopic content map : "+subtopic.contentMap);

Component panel;

**for**(String key:subtopic.contentMap.keySet()){

message="";

**if**(key.contains("text")==**true**){

AddTextJPanel temp=**new** AddTextJPanel();

temp.setText(subtopic.contentMap.get(key));

panel=temp;

message+="text panel found--->"+temp.getText();

}

**else** **if**(key.contains("code")==**true**){

AddCodePanel temp=**new** AddCodePanel();

temp.setText(subtopic.contentMap.get(key));

panel=temp;

message+="code panel found--->"+temp.getText();

}

**else**{

AddMediaJPanel temp=**new** AddMediaJPanel();

String[] value=subtopic.contentMap.get(key).split(",");

temp.setText(value[0],value[1]);

panel=temp;

message+="media panel found--->"+temp.getMediaInfo();

}

addContentPage.contentPanel.add(panel);

System.*out*.println("adding...."+message);

}

addContentPage.contentPanel.repaint();

addContentPage.contentPanel.validate();

}

System.*out*.println("RETRIVIAL FINISHED.");

}

/\*

\* deleteTopic() Deletes the particular Topic from the Module object

\*/

**public** **static** **void** deleteTopic(**int** topicIndex) {

// **TODO** Auto-generated method stub

*module*.topicList.remove(topicIndex);

}

/\*

\* renameTopic() Rename the particular Topic with given new topic name

\*/

**public** **static** **void** renameTopic(String newTopicName, **int** topicIndex) {

// **TODO** Auto-generated method stub

*module*.topicList.get(topicIndex).setTopicName(newTopicName);

}

/\*

\* swapTopic() swaps the topic with in the topic list of Module object

\*/

**public** **static** **void** swapTopic(**int** a, **int** b) {

// **TODO** Auto-generated method stub

Topic t1=*module*.topicList.get(a);

*module*.topicList.set(a, *module*.topicList.get(b));

*module*.topicList.set(b, t1);

}

/\*

\* renameSubTopic() Renames the particular SubTopic object with given new sub topic name

\*/

**public** **static** **void** renameSubTopic(String topicName, **int** subtopicIndex,String newSubTopicName) {

// **TODO** Auto-generated method stub

Topic topic=*module*.getTopicByName(topicName);

topic.subTopicList.get(subtopicIndex).setSubTopicName(newSubTopicName);

}

/\*

\* deleteSubTopic() deletes the particular SubTopic object from sub topic list with in the Topic object

\*/

**public** **static** **void** deleteSubTopic(String topicName, **int** subTopicIndex) {

// **TODO** Auto-generated method stub

*module*.getTopicByName(topicName).subTopicList.remove(subTopicIndex);

}

/\*

\* swapSubTopic() swaps the subtopics mentioned with in the sub topic list of Topic object

\*/

**public** **static** **void** swapSubTopic(String topicName,**int** a, **int** b) {

// **TODO** Auto-generated method stub

Topic topic=*module*.getTopicByName(topicName);

SubTopic subtopic=topic.subTopicList.get(a);

topic.subTopicList.set(a, topic.subTopicList.get(b));

topic.subTopicList.set(b, subtopic);

}

/\*

\* setReference() saves the references name along with its name in the Module object

\*/

**public** **static** **void** setReference(JTextField linkNameTextField,

JTextField linkURLTextField) {

// **TODO** Auto-generated method stub

String linkName=linkNameTextField.getText();

String linkURL=linkURLTextField.getText();

*module*.putReference(linkName, linkURL);

}

/\*

\* createModuleStructure() helps in reducing the effect on toolkit package

\* in case of any changes done to actual method

\* responsible for module structure generation

\*/

**public** **static** String createModuleStructure() {

// **TODO** Auto-generated method stub

String moduleDirName=*module*.getModuleName()+*module*.getModuleID();

String moduleRoot=com.html.FileSystem.*createDirectoryStructure*(moduleDirName);

**return** moduleRoot;

}

}

//MasterPage.java

/\*

\* MasterPage.java

\*

\* Created on \_\_DATE\_\_, \_\_TIME\_\_

\*/

**package** com.toolkit;

/\* This form is called as Master Page and this includes 4 fields which are

\* module ID, Name , Author, Compiler Selection.

\*

\*/

**import** java.awt.event.ActionEvent;

**import** java.io.File;

**import** java.io.FileNotFoundException;

**import** java.io.FileOutputStream;

**import** javax.swing.undo.\*;

**import** javax.swing.ImageIcon;

**import** javax.swing.JOptionPane;

/\*\*

\* this class is used to provide info about the module

\*/

**public** **class** MasterPage **extends** javax.swing.JFrame {

/\*\*

\*

\*/

**private** **static** **final** **long** *serialVersionUID* = 1L;

/\*\* Creates new form MasterPage \*/

**public** MasterPage() {

initComponents();

setSize(900, 700);

setResizable(**false**);

ImageIcon module\_strip = **new** ImageIcon("images/module\_strip.png");

// ImageIcon toolIcon = new ImageIcon("images/elearning\_right\_most.png");

imageLabel.setIcon(module\_strip);

manager = **new** UndoManager();

moduleIdTextField.getDocument().addUndoableEditListener(manager);

moduleSubjectTextField.getDocument().addUndoableEditListener(manager);

authorNameTextField.getDocument().addUndoableEditListener(manager);

moduleIdTextField.requestFocusInWindow();

}

//GEN-BEGIN:initComponents

// <editor-fold defaultstate="collapsed" desc="Generated Code">

**private** **void** initComponents() {

java.awt.GridBagConstraints gridBagConstraints;

jPanel1 = **new** javax.swing.JPanel();

jPanel2 = **new** javax.swing.JPanel();

topPanel = **new** javax.swing.JPanel();

imageLabel = **new** javax.swing.JLabel();

contentPanel = **new** javax.swing.JPanel();

moduleIdLabel = **new** javax.swing.JLabel();

moduleSubjectLabel = **new** javax.swing.JLabel();

authorNameLabel = **new** javax.swing.JLabel();

selectCompilerLabel = **new** javax.swing.JLabel();

moduleIdTextField = **new** javax.swing.JTextField();

moduleSubjectTextField = **new** javax.swing.JTextField();

authorNameTextField = **new** javax.swing.JTextField();

compilerComboBox = **new** javax.swing.JComboBox();

moduleNextButton = **new** javax.swing.JButton();

menuBar = **new** javax.swing.JMenuBar();

fileMenu = **new** javax.swing.JMenu();

openMenuItem = **new** javax.swing.JMenuItem();

saveMenuItem = **new** javax.swing.JMenuItem();

saveAsMenuItem = **new** javax.swing.JMenuItem();

exitMenuItem = **new** javax.swing.JMenuItem();

editMenu = **new** javax.swing.JMenu();

cutMenuItem = **new** javax.swing.JMenuItem();

copyMenuItem = **new** javax.swing.JMenuItem();

pasteMenuItem = **new** javax.swing.JMenuItem();

deleteMenuItem = **new** javax.swing.JMenuItem();

helpMenu = **new** javax.swing.JMenu();

contentsMenuItem = **new** javax.swing.JMenuItem();

aboutMenuItem = **new** javax.swing.JMenuItem();

menuBar1 = **new** javax.swing.JMenuBar();

fileMenu1 = **new** javax.swing.JMenu();

openMenuItem1 = **new** javax.swing.JMenuItem();

saveMenuItem1 = **new** javax.swing.JMenuItem();

saveAsMenuItem1 = **new** javax.swing.JMenuItem();

exitMenuItem1 = **new** javax.swing.JMenuItem();

editMenu1 = **new** javax.swing.JMenu();

jMenuItem1 = **new** javax.swing.JMenuItem();

jMenuItem2 = **new** javax.swing.JMenuItem();

cutMenuItem1 = **new** javax.swing.JMenuItem();

copyMenuItem1 = **new** javax.swing.JMenuItem();

pasteMenuItem1 = **new** javax.swing.JMenuItem();

deleteMenuItem1 = **new** javax.swing.JMenuItem();

helpMenu1 = **new** javax.swing.JMenu();

contentsMenuItem1 = **new** javax.swing.JMenuItem();

aboutMenuItem1 = **new** javax.swing.JMenuItem();

setDefaultCloseOperation(javax.swing.WindowConstants.*EXIT\_ON\_CLOSE*);

jPanel1.setLayout(**new** java.awt.GridBagLayout());

jPanel2.setLayout(**new** java.awt.GridBagLayout());

topPanel.setLayout(**new** java.awt.GridBagLayout());

imageLabel.setText(" ");

gridBagConstraints = **new** java.awt.GridBagConstraints();

gridBagConstraints.gridwidth = 3;

gridBagConstraints.gridheight = 3;

gridBagConstraints.fill = java.awt.GridBagConstraints.*BOTH*;

topPanel.add(imageLabel, gridBagConstraints);

contentPanel.setBackground(**new** java.awt.Color(255, 255, 255));

contentPanel.setBorder(javax.swing.BorderFactory.*createTitledBorder*(

javax.swing.BorderFactory.*createTitledBorder*(""), "",

javax.swing.border.TitledBorder.*DEFAULT\_JUSTIFICATION*,

javax.swing.border.TitledBorder.*TOP*));

moduleIdLabel.setText("Module ID");

moduleSubjectLabel.setText("Module Subject");

authorNameLabel.setText("Author Name");

selectCompilerLabel.setText("Select Compiler");

moduleIdTextField.setMaximumSize(**new** java.awt.Dimension(6, 22));

moduleSubjectTextField.setMaximumSize(**new** java.awt.Dimension(6, 22));

moduleSubjectTextField

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

moduleSubjectTextFieldActionPerformed(evt);

}

});

authorNameTextField

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

authorNameTextFieldActionPerformed(evt);

}

});

compilerComboBox.setModel(**new** javax.swing.DefaultComboBoxModel(

**new** String[] { "Please Select", "Java Compiler",

".Net Compiler", "None" }));

compilerComboBox.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

compilerComboBoxActionPerformed(evt);

}

});

javax.swing.GroupLayout contentPanelLayout = **new** javax.swing.GroupLayout(

contentPanel);

contentPanel.setLayout(contentPanelLayout);

contentPanelLayout

.setHorizontalGroup(contentPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

contentPanelLayout

.createSequentialGroup()

.addContainerGap(

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)

.addGroup(

contentPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

moduleIdLabel)

.addComponent(

moduleSubjectLabel)

.addComponent(

authorNameLabel)

.addComponent(

selectCompilerLabel))

.addGap(93, 93, 93)

.addGroup(

contentPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*,

**false**)

.addComponent(

moduleIdTextField,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

107,

Short.*MAX\_VALUE*)

.addComponent(

moduleSubjectTextField,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)

.addComponent(

authorNameTextField)

.addComponent(

compilerComboBox,

0,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*))));

contentPanelLayout

.setVerticalGroup(contentPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

contentPanelLayout

.createSequentialGroup()

.addGap(50, 50, 50)

.addGroup(

contentPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

moduleIdLabel)

.addComponent(

moduleIdTextField,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addGap(38, 38, 38)

.addGroup(

contentPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

moduleSubjectLabel)

.addComponent(

moduleSubjectTextField,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*))

.addGap(38, 38, 38)

.addGroup(

contentPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

authorNameLabel)

.addComponent(

authorNameTextField,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addGap(44, 44, 44)

.addGroup(

contentPanelLayout

.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

selectCompilerLabel)

.addComponent(

compilerComboBox,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addContainerGap()));

moduleNextButton.setText("Next");

moduleNextButton.addMouseListener(**new** java.awt.event.MouseAdapter() {

**public** **void** mouseClicked(java.awt.event.MouseEvent evt) {

moduleNextButtonMouseClicked(evt);

}

});

moduleNextButton.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

moduleNextButtonActionPerformed(evt);

}

});

fileMenu.setText("File");

openMenuItem.setText("Open");

fileMenu.add(openMenuItem);

saveMenuItem.setText("Save");

fileMenu.add(saveMenuItem);

saveAsMenuItem.setText("Save As ...");

fileMenu.add(saveAsMenuItem);

exitMenuItem.setText("Exit");

exitMenuItem.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

exitMenuItemActionPerformed(evt);

}

});

fileMenu.add(exitMenuItem);

menuBar.add(fileMenu);

editMenu.setText("Edit");

cutMenuItem.setText("Cut");

editMenu.add(cutMenuItem);

copyMenuItem.setText("Copy");

editMenu.add(copyMenuItem);

pasteMenuItem.setText("Paste");

editMenu.add(pasteMenuItem);

deleteMenuItem.setText("Delete");

editMenu.add(deleteMenuItem);

menuBar.add(editMenu);

helpMenu.setText("Help");

contentsMenuItem.setText("Contents");

helpMenu.add(contentsMenuItem);

aboutMenuItem.setText("About");

helpMenu.add(aboutMenuItem);

menuBar.add(helpMenu);

fileMenu1.setText("File");

openMenuItem1.setText("Open");

fileMenu1.add(openMenuItem1);

saveMenuItem1.setText("Save");

fileMenu1.add(saveMenuItem1);

saveAsMenuItem1.setText("Save As ...");

fileMenu1.add(saveAsMenuItem1);

exitMenuItem1.setText("Exit");

exitMenuItem1.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

exitMenuItemActionPerformed(evt);

}

});

fileMenu1.add(exitMenuItem1);

menuBar1.add(fileMenu1);

editMenu1.setText("Edit");

jMenuItem1.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_Z*,

java.awt.event.InputEvent.*CTRL\_MASK*));

jMenuItem1.setText("Undo");

jMenuItem1.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jMenuItem1ActionPerformed(evt);

}

});

editMenu1.add(jMenuItem1);

jMenuItem2.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_Y*,

java.awt.event.InputEvent.*CTRL\_MASK*));

jMenuItem2.setText("Redo");

jMenuItem2.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

jMenuItem2ActionPerformed(evt);

}

});

editMenu1.add(jMenuItem2);

cutMenuItem1.setText("Cut");

editMenu1.add(cutMenuItem1);

copyMenuItem1.setText("Copy");

editMenu1.add(copyMenuItem1);

pasteMenuItem1.setText("Paste");

editMenu1.add(pasteMenuItem1);

deleteMenuItem1.setText("Delete");

editMenu1.add(deleteMenuItem1);

menuBar1.add(editMenu1);

helpMenu1.setText("Help");

helpMenu1.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

helpMenu1ActionPerformed(evt);

}

});

contentsMenuItem1.setAccelerator(javax.swing.KeyStroke.*getKeyStroke*(

java.awt.event.KeyEvent.*VK\_H*,

java.awt.event.InputEvent.*ALT\_MASK*));

contentsMenuItem1.setText("Contents");

contentsMenuItem1.addMouseListener(**new** java.awt.event.MouseAdapter() {

**public** **void** mouseClicked(java.awt.event.MouseEvent evt) {

contentsMenuItem1MouseClicked(evt);

}

});

contentsMenuItem1

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

contentsMenuItem1ActionPerformed(evt);

}

});

helpMenu1.add(contentsMenuItem1);

aboutMenuItem1.setText("About");

helpMenu1.add(aboutMenuItem1);

menuBar1.add(helpMenu1);

setJMenuBar(menuBar1);

javax.swing.GroupLayout layout = **new** javax.swing.GroupLayout(

getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(jPanel1, javax.swing.GroupLayout.*DEFAULT\_SIZE*,

582, Short.*MAX\_VALUE*)

.addGroup(

layout.createSequentialGroup()

.addGap(12, 12, 12)

.addComponent(jPanel2,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

560, Short.*MAX\_VALUE*).addContainerGap())

.addGroup(

layout.createSequentialGroup()

.addContainerGap()

.addComponent(topPanel,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

562, Short.*MAX\_VALUE*).addContainerGap())

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

layout.createSequentialGroup()

.addContainerGap(497, Short.*MAX\_VALUE*)

.addComponent(moduleNextButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

75,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addContainerGap())

.addGroup(

layout.createSequentialGroup()

.addGap(259, 259, 259)

.addComponent(contentPanel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addContainerGap(16, Short.*MAX\_VALUE*)));

layout.setVerticalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addComponent(jPanel1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jPanel2,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(topPanel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

94,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*UNRELATED*)

.addComponent(contentPanel,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*,

40, Short.*MAX\_VALUE*)

.addComponent(moduleNextButton)

.addContainerGap()));

pack();

}// </editor-fold>

//GEN-END:initComponents

**private** **void** compilerComboBoxActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

}

/\*

\* The below method performs redo operation for all the text fields in master PAge

\*/

**private** **void** jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

**try** {

manager.redo();

} **catch** (Exception ex) {

}

}

/\*

\* The below method performs undo operation for all the text fields in master PAge

\*/

**private** **void** jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

**try** {

manager.undo();

} **catch** (Exception ex) {

}

}

**private** **void** contentsMenuItem1MouseClicked(java.awt.event.MouseEvent evt) {

}

**private** **void** helpMenu1ActionPerformed(java.awt.event.ActionEvent evt) {

}

/\*

\* The below method gives help window by clicking help followed by Content or ALT+H

\*/

**private** **void** contentsMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {

System.*out*.println("entered into help");

run = Runtime.*getRuntime*();

**try** {

File currentDir = **new** File(".");

String root = currentDir.getCanonicalPath();

System.*out*.println(root);

String helpLocation = root + File.*separatorChar* + "help"

+ File.*separatorChar* + "Help\_window.chm";

child = Runtime.*getRuntime*().exec(

"rundll32 url.dll,FileProtocolHandler " + helpLocation);

} **catch** (Exception ex) {

ex.printStackTrace();

System.*out*.println(ex.getMessage());

}

}

/\*

\* The below method checks that all the text field

\* contains valid data and compiler selection is done or not and

\* on NEXT button it goes to the next

\* Page i.e is called ADD TOPIC PAGE and for consistency of the data of

\* master page we create object of add topic form and sends the object of master Page

\*

\*/

**protected** **void** moduleNextButtonActionPerformed(ActionEvent evt) {

// **TODO** Auto-generated method stub

**boolean** flag = **true**;

String msg = **null**;

System.*out*.println("moduleNextButtonMouseClicked");

**if** (moduleIdTextField.getText().trim().length() == 0) {

msg = "please provide the module Id";

flag = **false**;

} **else** **if** (moduleSubjectTextField.getText().trim().length() == 0) {

msg = "please provide the module subject";

flag = **false**;

} **else** **if** (authorNameTextField.getText().trim().length() == 0) {

msg = "please provide the author name";

flag = **false**;

} **else** **if** (compilerComboBox.getSelectedItem() == "Please Select") {

msg = "please select the compiler";

flag = **false**;

}

**if** (addTopicForm == **null**) {

addTopicForm = **new** AddTopicForm(**this**);

}

**if** (flag == **true**) {

**this**.setVisible(**false**);

addTopicForm.displayTopicForm(**true**);

System.*out*.println("start next button event....");

/// adding integration function.....

Integration.*setModulePage*(**this**);

System.*out*.println("end next button event....");

} **else** {

JOptionPane.*showMessageDialog*(**this**, msg, "Error",

JOptionPane.*ERROR\_MESSAGE*);

**this**.setVisible(**true**);

}

}

**protected** **void** moduleSubjectTextFieldActionPerformed(ActionEvent evt) {

// **TODO** Auto-generated method stub

}

**private** **void** moduleNextButtonMouseClicked(java.awt.event.MouseEvent evt) {

}

**private** **void** authorNameTextFieldActionPerformed(

java.awt.event.ActionEvent evt) {

// **TODO** add your handling code here:

}

**private** **void** exitMenuItemActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_exitMenuItemActionPerformed

System.*exit*(0);

}//GEN-LAST:event\_exitMenuItemActionPerformed

/\*\*

\* **@param** args the command line arguments

\*/

**public** **static** **void** main(String args[]) {

java.awt.EventQueue.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**new** MasterPage().setVisible(**true**);

}

});

}

**public** **void** setAddTopicForm(AddTopicForm addTopicForm) {

**this**.addTopicForm = addTopicForm;

}

//GEN-BEGIN:variables

// Variables declaration - do not modify

**private** javax.swing.JMenuItem aboutMenuItem;

**private** javax.swing.JMenuItem aboutMenuItem1;

**private** javax.swing.JLabel authorNameLabel;

**public** javax.swing.JTextField authorNameTextField;

**public** javax.swing.JComboBox compilerComboBox;

**private** javax.swing.JPanel contentPanel;

**private** javax.swing.JMenuItem contentsMenuItem;

**private** javax.swing.JMenuItem contentsMenuItem1;

**private** javax.swing.JMenuItem copyMenuItem;

**private** javax.swing.JMenuItem copyMenuItem1;

**private** javax.swing.JMenuItem cutMenuItem;

**private** javax.swing.JMenuItem cutMenuItem1;

**private** javax.swing.JMenuItem deleteMenuItem;

**private** javax.swing.JMenuItem deleteMenuItem1;

**private** javax.swing.JMenu editMenu;

**private** javax.swing.JMenu editMenu1;

**private** javax.swing.JMenuItem exitMenuItem;

**private** javax.swing.JMenuItem exitMenuItem1;

**private** javax.swing.JMenu fileMenu;

**private** javax.swing.JMenu fileMenu1;

**private** javax.swing.JMenu helpMenu;

**private** javax.swing.JMenu helpMenu1;

**private** javax.swing.JLabel imageLabel;

**private** javax.swing.JMenuItem jMenuItem1;

**private** javax.swing.JMenuItem jMenuItem2;

**private** javax.swing.JPanel jPanel1;

**private** javax.swing.JPanel jPanel2;

**private** javax.swing.JMenuBar menuBar;

**private** javax.swing.JMenuBar menuBar1;

**private** javax.swing.JLabel moduleIdLabel;

**public** javax.swing.JTextField moduleIdTextField;

**private** javax.swing.JButton moduleNextButton;

**private** javax.swing.JLabel moduleSubjectLabel;

**public** javax.swing.JTextField moduleSubjectTextField;

**private** javax.swing.JMenuItem openMenuItem;

**private** javax.swing.JMenuItem openMenuItem1;

**private** javax.swing.JMenuItem pasteMenuItem;

**private** javax.swing.JMenuItem pasteMenuItem1;

**private** javax.swing.JMenuItem saveAsMenuItem;

**private** javax.swing.JMenuItem saveAsMenuItem1;

**private** javax.swing.JMenuItem saveMenuItem;

**private** javax.swing.JMenuItem saveMenuItem1;

**private** javax.swing.JLabel selectCompilerLabel;

**private** javax.swing.JPanel topPanel;

// End of variables declaration//GEN-END:variables

**private** AddTopicForm addTopicForm;

**public** UndoManager manager;

**private** Runtime run;

**private** Process child;

/\*

\* the below method returns the the value of the module subject.

\*/

**public** String getModuleSubjectTextField() {

// **TODO** Auto-generated method stub

**return** moduleSubjectTextField.getText();

}

}

//Module.java

**package** com.toolkit;

**public** **class** Module **implements** java.io.Serializable{

/\*\*

\*

\*/

**private** **static** **final** **long** *serialVersionUID* = 1L;

**public** Module(){

references=**new** java.util.LinkedHashMap<String,String>();

setTopicList(**new** java.util.ArrayList<Topic>());

}

**public** **void** setModuleID(String moduleID){

**this**.moduleID=moduleID;

}

**public** String getModuleID(){

**return** **this**.moduleID;

}

**public** **void** setModuleName(String moduleName){

**this**.moduleName=moduleName;

}

**public** String getModuleName(){

**return** **this**.moduleName;

}

**public** **void** setAuthorName(String authorName){

**this**.authorName=authorName;

}

**public** String getAuthorName(){

**return** **this**.authorName;

}

**public** **void** setCompilerName(String compilerName){

**this**.compilerName=compilerName;

}

**public** String getCompilerName(){

**return** **this**.compilerName;

}

**public** java.util.ArrayList<Topic> getTopicList() {

**return** topicList;

}

**public** **void** setTopicList(java.util.ArrayList<Topic> topicList) {

**this**.topicList = topicList;

}

**private** String moduleID;

**private** String moduleName;

**private** String authorName;

**private** String compilerName;

**public** java.util.LinkedHashMap<String,String> references;

**public** java.util.ArrayList<Topic> topicList;

**private** String emailId;

**public** String toString(){

**return** getModuleName();

}

**public** Topic getTopicByName(String topicName){

**for**(Topic topic:topicList){

**if**(topicName.equalsIgnoreCase(topic.getTopicName())){

**return** topic;

}

}

**return** **null**;

}

**public** **void** setAuthorMailId(String emailId) {

// **TODO** Auto-generated method stub

**this**.emailId=emailId;

}

**public** String getAuthorMailId(){

**return** emailId;

}

**public** **void** putReference(String name,String url) {

// **TODO** Auto-generated method stub

**if**((url!=**null** && url.length()>0 ) && ( name!=**null** && name.length()>0) ){

references.put(name,url);

}

}

}

//Question.java

**package** com.toolkit;

**import** java.util.List;

**public** **class** Question {

**private** String description;

**private** List<String> options;

**private** String explanation;

**private** **int** answer;

**public** **void** setDescription(String description){

**this**.description=description;

}

**public** String getDescription(){

**return** description;

}

**public** **void** setExplanation(String explanation){

**this**.explanation=explanation;

}

**public** String getExplanation(){

**return** explanation;

}

**public** **void** setanswer(**int** answer){

**this**.answer=answer;

}

**public** **int** getanswer(){

**return** answer;

}

}

//Quiz.java

**package** com.toolkit;

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** Quiz {

**public** List<Question> questionList;

**public** Quiz(){

questionList=**new** ArrayList<Question>();

}

**public** **void** setQuestionList(List<Question> questionList){

**this**.questionList=questionList;

}

**public** List<Question> getQuestionList(){

**return** questionList;

}

}

//QuizJPanel.java

/\*

\* QuizJPanel2.java

\*

\* Created on \_\_DATE\_\_, \_\_TIME\_\_

\*/

**package** com.toolkit;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.MouseEvent;

**import** javax.swing.ButtonGroup;

/\*\*

\* this class is used to add questions for subtopic

\*/

**public** **class** QuizJPanel **extends** javax.swing.JPanel {

/\*\* Creates new form QuizJPanel2 \*/

**public** QuizJPanel() {

initComponents();

autoSelectOnDelClickCheckBox.setVisible(**false**);

deleteButton.setVisible(**true**);

*i* = *i* + 1;

System.*out*.println("i value " + *i*);

questionLabel.setText("Q. " + *i*);

**this**.validate();

}

//GEN-BEGIN:initComponents

// <editor-fold defaultstate="collapsed" desc="Generated Code">

**private** **void** initComponents() {

jScrollPane2 = **new** javax.swing.JScrollPane();

jTextArea2 = **new** javax.swing.JTextArea();

buttonGroup1 = **new** javax.swing.ButtonGroup();

questionLabel = **new** javax.swing.JLabel();

jScrollPane1 = **new** javax.swing.JScrollPane();

jTextArea1 = **new** javax.swing.JTextArea();

aRadioButton = **new** javax.swing.JRadioButton();

buttonGroup1.add(aRadioButton);

jLabel1 = **new** javax.swing.JLabel();

jTextField1 = **new** javax.swing.JTextField();

jLabel5 = **new** javax.swing.JLabel();

jScrollPane3 = **new** javax.swing.JScrollPane();

jTextArea3 = **new** javax.swing.JTextArea();

deleteButton = **new** javax.swing.JButton();

autoSelectOnDelClickCheckBox = **new** javax.swing.JCheckBox();

bRadioButton = **new** javax.swing.JRadioButton();

buttonGroup1.add(bRadioButton);

jLabel2 = **new** javax.swing.JLabel();

jTextField2 = **new** javax.swing.JTextField();

cRadioButton = **new** javax.swing.JRadioButton();

buttonGroup1.add(cRadioButton);

jLabel3 = **new** javax.swing.JLabel();

dRadioButton = **new** javax.swing.JRadioButton();

buttonGroup1.add(dRadioButton);

jLabel4 = **new** javax.swing.JLabel();

jTextField3 = **new** javax.swing.JTextField();

jTextField4 = **new** javax.swing.JTextField();

jTextArea2.setColumns(20);

jTextArea2.setRows(5);

jScrollPane2.setViewportView(jTextArea2);

setBorder(javax.swing.BorderFactory

.*createLineBorder*(**new** java.awt.Color(0, 0, 0)));

addMouseListener(**new** java.awt.event.MouseAdapter() {

**public** **void** mouseEntered(java.awt.event.MouseEvent evt) {

formMouseEntered(evt);

}

**public** **void** mouseExited(java.awt.event.MouseEvent evt) {

formMouseExited(evt);

}

});

questionLabel.setText("Q.");

jTextArea1.setColumns(20);

jTextArea1.setRows(5);

jScrollPane1.setViewportView(jTextArea1);

jLabel1.setFont(**new** java.awt.Font("Segoe UI", 0, 14));

jLabel1.setText("A");

jLabel5.setText("Description");

jTextArea3.setColumns(20);

jTextArea3.setRows(5);

jScrollPane3.setViewportView(jTextArea3);

deleteButton.setText("X");

deleteButton.addMouseListener(**new** java.awt.event.MouseAdapter() {

**public** **void** mouseClicked(java.awt.event.MouseEvent evt) {

deleteButtonMouseClicked(evt);

}

**public** **void** mouseEntered(java.awt.event.MouseEvent evt) {

deleteButtonMouseEntered(evt);

}

});

autoSelectOnDelClickCheckBox

.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

autoSelectOnDelClickCheckBoxActionPerformed(evt);

}

});

jLabel2.setFont(**new** java.awt.Font("Segoe UI", 0, 14));

jLabel2.setText("B");

jLabel3.setFont(**new** java.awt.Font("Segoe UI", 0, 14));

jLabel3.setText("C");

jLabel4.setFont(**new** java.awt.Font("Segoe UI", 0, 14));

jLabel4.setText("D");

javax.swing.GroupLayout layout = **new** javax.swing.GroupLayout(**this**);

**this**.setLayout(layout);

layout.setHorizontalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addContainerGap()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*)

.addGroup(

javax.swing.GroupLayout.Alignment.*LEADING*,

layout.createSequentialGroup()

.addComponent(

questionLabel)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*)

.addGroup(

layout.createSequentialGroup()

.addGap(11,

11,

11)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

aRadioButton)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

jLabel1))

.addGroup(

layout.createSequentialGroup()

.addComponent(

bRadioButton)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

jLabel2))

.addGroup(

layout.createSequentialGroup()

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

cRadioButton)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

jLabel3))

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

layout.createSequentialGroup()

.addComponent(

dRadioButton)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

jLabel4)

.addGap(10,

10,

10)))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

**false**)

.addComponent(

jTextField3)

.addComponent(

jTextField4,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

294,

Short.*MAX\_VALUE*)

.addComponent(

jTextField2,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

338,

Short.*MAX\_VALUE*)

.addComponent(

jTextField1)))

.addComponent(

jScrollPane3,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

380,

Short.*MAX\_VALUE*)))

.addGroup(

layout.createSequentialGroup()

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

jScrollPane1,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

391,

Short.*MAX\_VALUE*))))

.addGroup(

javax.swing.GroupLayout.Alignment.*LEADING*,

layout.createSequentialGroup()

.addComponent(

jLabel5)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*,

348,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)))

.addGap(39, 39,

39))

.addGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*,

layout.createSequentialGroup()

.addComponent(

autoSelectOnDelClickCheckBox)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

deleteButton)))));

layout.setVerticalGroup(layout

.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

autoSelectOnDelClickCheckBox)

.addComponent(

deleteButton,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

19,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addGap(13, 13, 13)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addComponent(

jScrollPane1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

128,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*TRAILING*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

jLabel1)

.addComponent(

jTextField1,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addComponent(

aRadioButton))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

bRadioButton)

.addComponent(

jLabel2)

.addComponent(

jTextField2,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addGroup(

layout.createSequentialGroup()

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*LEADING*)

.addComponent(

jLabel3)

.addComponent(

cRadioButton))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(

dRadioButton))

.addGroup(

layout.createSequentialGroup()

.addComponent(

jTextField4,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addGroup(

layout.createParallelGroup(

javax.swing.GroupLayout.Alignment.*BASELINE*)

.addComponent(

jTextField3,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addComponent(

jLabel4)))))

.addComponent(questionLabel))

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jLabel5)

.addPreferredGap(

javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)

.addComponent(jScrollPane3,

javax.swing.GroupLayout.*PREFERRED\_SIZE*,

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

javax.swing.GroupLayout.*PREFERRED\_SIZE*)

.addContainerGap(

javax.swing.GroupLayout.*DEFAULT\_SIZE*,

Short.*MAX\_VALUE*)));

}// </editor-fold>

//GEN-END:initComponents

**protected** **void** autoSelectOnDelClickCheckBoxActionPerformed(ActionEvent evt) {

// **TODO** Auto-generated method stub

}

**protected** **void** deleteButtonMouseEntered(MouseEvent evt) {

// **TODO** Auto-generated method stub

}

/\*

\* deleteButton helps in deleting the particular panel on clicking this button .

\* this button makes auto selection of "autoSelectOnDelClick" check-box ,

\* which is checked by the removeQuizPanel() method

\* present in AddMediaJPanel class,

\* to remove the particular panel from the contentPnel

\*/

**private** **void** deleteButtonMouseClicked(java.awt.event.MouseEvent evt) {

// **TODO** add your handling code here:

AddQuizPanel quizPanel = (AddQuizPanel) (getParent().getParent()

.getParent().getParent().getParent().getParent().getParent());

//= new AddQuizPanel();

System.*out*.println("start delete method form QuizJPanel2 ");

autoSelectOnDelClickCheckBox.setSelected(**true**);

quizPanel.removeQuizPanel();

System.*out*.println("selection done auto... "

+ autoSelectOnDelClickCheckBox.isSelected());

System.*out*.println("end delete method form addtextJPanel");

**this**.validate();

}

**private** **void** formMouseExited(java.awt.event.MouseEvent evt) {

// **TODO** add your handling code here:

}

**private** **void** formMouseEntered(java.awt.event.MouseEvent evt) {

// **TODO** add your handling code here:

}

//GEN-BEGIN:variables

// Variables declaration - do not modify

**private** javax.swing.JRadioButton aRadioButton;

**private** javax.swing.JCheckBox autoSelectOnDelClickCheckBox;

**private** javax.swing.JRadioButton bRadioButton;

**private** javax.swing.ButtonGroup buttonGroup1;

**private** javax.swing.JRadioButton cRadioButton;

**private** javax.swing.JRadioButton dRadioButton;

**private** javax.swing.JButton deleteButton;

**private** javax.swing.JLabel jLabel1;

**private** javax.swing.JLabel jLabel2;

**private** javax.swing.JLabel jLabel3;

**private** javax.swing.JLabel jLabel4;

**private** javax.swing.JLabel jLabel5;

**private** javax.swing.JScrollPane jScrollPane1;

**private** javax.swing.JScrollPane jScrollPane2;

**private** javax.swing.JScrollPane jScrollPane3;

**private** javax.swing.JTextArea jTextArea1;

**private** javax.swing.JTextArea jTextArea2;

**private** javax.swing.JTextArea jTextArea3;

**private** javax.swing.JTextField jTextField1;

**private** javax.swing.JTextField jTextField2;

**private** javax.swing.JTextField jTextField3;

**private** javax.swing.JTextField jTextField4;

**public** javax.swing.JLabel questionLabel;

// End of variables declaration//GEN-END:variables

**public** **static** **int** *i* = 0;

**public** **boolean** isSelected() {

// **TODO** Auto-generated method stub

System.*out*.println(autoSelectOnDelClickCheckBox.isSelected()

+ " in QuizPanel");

**return** autoSelectOnDelClickCheckBox.isSelected();

}

}

//SubTopic.java

**package** com.toolkit;

**import** java.util.LinkedHashMap;

**import** java.util.Set;

**public** **class** SubTopic **implements** java.io.Serializable{

/\*\*

\*

\*/

**private** **static** **final** **long** *serialVersionUID* = 1L;

**public** SubTopic(){

contentMap=**new** java.util.LinkedHashMap<String, String>();

}

**public** **void** setSubTopicName(String subTopicName){

**this**.subTopicName=subTopicName;

}

**public** String getSubTopicName(){

**return** **this**.subTopicName;

}

**public** String toString(){

**return** subTopicName;

}

**public** String getComponentByKey(String key1){

**for**(String key2:contentMap.keySet()){

**if**(key2.equals(key1)){

**return** contentMap.get(key2);

}

}

**return** **null**;

}

**private** String subTopicName;

**public** LinkedHashMap<String, String> contentMap;

**public** **int** pageNo;

}

//Topic.java

**package** com.toolkit;

**public** **class** Topic **implements** java.io.Serializable{

/\*\*

\*

\*/

**private** **static** **final** **long** *serialVersionUID* = 1L;

**public** Topic(){

subTopicList = **new** java.util.ArrayList<SubTopic>();

}

**public** **void** setTopicName(String topicName){

**this**.topicName=topicName;

}

**public** java.util.ArrayList<SubTopic> getSubTopicList() {

**return** subTopicList;

}

**public** **void** setSubTopicList(java.util.ArrayList<SubTopic> subTopicList) {

**this**.subTopicList = subTopicList;

}

**public** String getTopicName(){

**return** **this**.topicName;

}

**public** **void** setTopicDescription(String topicDescription){

**this**.topicDescription=topicDescription;

}

**public** String getTopicDescription(){

**return** **this**.topicDescription;

}

**private** String topicName;

**private** String topicDescription;

**public** java.util.ArrayList<SubTopic> subTopicList;

**public** String toString(){

**return** topicName;

}

**public** SubTopic getSubTopicByName(String subtopicName){

**for**(SubTopic subtopic : subTopicList){

**if**(subtopic.getSubTopicName()!=**null** && subtopic.getSubTopicName().equalsIgnoreCase(subtopicName)){

**return** subtopic;

}

}

**return** **null**;

}

}

//Utility.java

**package** com.toolkit;

**import** java.io.BufferedReader;

**import** java.io.File;

**import** java.io.FileNotFoundException;

**import** java.io.FileReader;

**import** java.io.IOException;

**public** **class** Utility {

**public** **static** **final** String *VIDEO\_FILE*="videos";

**public** **static** **final** String *AUDIO\_FILE*="audios";

**public** **static** **final** String *IMAGE\_FILE*="images";

**public** **static** **boolean** validateFileFormat(String type,String fileName){

//boolean extensionFound=false;

String[] fileTokens=fileName.split("\\.");

/\*

\* iterating throught the tokenized comppnents

\* for(int i=0;i<fileTokens.length;i++)

\* System.out.print("-----"+fileTokens[i]+"--");

\*/

System.*out*.println("inside validation function");

**if**(fileTokens.length==1){

**return** **false**;

}

**else**{

File file=**null**;

**if**(type.equalsIgnoreCase(*VIDEO\_FILE*)){

file=**new** File("file\_formats/video\_formats");

}**else** **if**(type.equalsIgnoreCase(*AUDIO\_FILE*)){

file=**new** File("file\_formats/audio\_formats");

}**else** **if**(type.equalsIgnoreCase(*IMAGE\_FILE*)){

file=**new** File("file\_formats/image\_formats");

}

**try** {

BufferedReader fr=**new** BufferedReader(**new** FileReader(file));

String formatName="";

**while**(formatName!=**null**){

formatName=fr.readLine();

System.*out*.println(formatName+"-->"+fileTokens[1]+">>>>"+fr);

**if**(formatName!=**null** && formatName.equalsIgnoreCase(fileTokens[1])){

**return** **true**;

}

}

} **catch** (FileNotFoundException e) {

e.printStackTrace();

**return** **false**;

} **catch** (IOException e) {

e.printStackTrace();

**return** **false**;

}

}

**return** **false**;

}

**public** **static** **void** main(String[] asd){

String fileName="file.rm";

System.*out*.println("validate file result : "+*validateFileFormat*(*AUDIO\_FILE*,fileName));

}

}

**Package : com.compiler**

//Compiler.java

**package** com.compiler;

**public** **interface** Compiler {

/\*

\* doCompile method takes the code as input

\* and performs the compilation on it

\* and returns the compilation result as Result

\* object.

\*/

**public** Result doCompile(String code);

}

//Controller.java

**package** com.compiler;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.IOException;

/\*

\* Controller class compiles and executes the given

\* code.

\*/

**public** **class** Controller{

/\*

\* compile() by default invokes the Java Compiler

\* and depending compilation result it executes the

\* generated .class file using Java Executer i.e. JVM

\*

\* future enhancement:

\* perform check on selected compiler by the trainer at the

\* time of making the module and invoke the required compiler

\* or interpreter.

\*/

**public** Result compile(String code) **throws** IOException{

Compiler com=**new** JavaCompiler();

System.*out*.println("COMPILING.....");

Result result=com.doCompile(code);

System.*out*.println("###########################################");

System.*out*.println(result.getResultStatus());

System.*out*.println("###########################################");

**if**(result.getExitValue()==0){

Executer jvm=**new** JavaExecuter();

System.*out*.println("Executing .....");

Result executeResult=jvm.doExecute("Main",timeLimit);

System.*out*.println("###########################################");

System.*out*.println(executeResult.getResultStatus());

System.*out*.println("###########################################");

result.setExitValue(executeResult.getExitValue());

result.setOutput(result.getOutput()+executeResult.getOutput());

result.setResultStatus(result.getResultStatus()+"\n"+executeResult.getResultStatus());

}

**return** result;

}

/\*

\* main is defined for debugging purposes (for unit testing purpose)

\* The working of Controller is tested using this main method

\*/

**public** **static** **void** main(String...args) **throws** IOException{

Controller ctrl=**new** Controller();

**char** sc=File.*separatorChar*;

File codeFile=**new** File("compiler"+sc+"sample.java");

**byte**[] codeBuffer=**new** **byte**[(**int**) codeFile.length()];

FileInputStream inputFile=**new** FileInputStream(codeFile);

inputFile.read(codeBuffer);

String code=**new** String(codeBuffer);

//System.out.println("@@@@"+code+"@@@@");

Result result=ctrl.compile(code);

System.*out*.println("!!!!!!!!!!!\n"+result);

}

/\*

\* setTimeLimit sets the time limit for execution of code

\*/

**public** **void** setTimeLimit(**long** timeLimit){

**this**.timeLimit=timeLimit;

System.*out*.println("=========================>>>>>>"+**this**.timeLimit);

}

/\*

\* getTimeLimit gets the time limit specified for the execution of code

\*/

**public** **long** getTimeLimit(){

**return** **this**.timeLimit;

}

**private** **long** timeLimit;

}

//Executer.java

**package** com.compiler;

**public** **interface** Executer {

/\*

\* doExecute method takes .class file name

\* and time limit for execution of .class file,

\* it returns the execution output, errors as Result

\* object

\*/

**public** Result doExecute(String fileName,**long** timeLimit);

}

//FileSystem.java

**package** com.compiler;

**import** java.io.File;

**import** java.io.FileNotFoundException;

**import** java.io.FileOutputStream;

**import** java.io.IOException;

/\*

\* FileSystem class performs file operations required during code

\* compilation and execution process.

\*/

**public** **class** FileSystem {

**public** **static** **final** String *compileResultFile*="compile\_result";

**public** **static** **final** **char** *sp*=File.*separatorChar*;

**public** **static** File *dir* = **new** File(".");

**public** **static** String *appRoot*;

**private** **static** String *currentDir*;

/\*

\* initializes the currentDir to current directory location

\*/

**static**{

**try**{

*currentDir*=*dir*.getCanonicalPath();

}**catch**(IOException e){

e.printStackTrace();

}

*appRoot*=*currentDir*+*sp*+"webapps"+*sp*+"test";

}

/\*

\*generateFile() takes content and filename as input

\*to generate the file with filename having "content"

\*as contents.

\*/

@SuppressWarnings("finally")

**public** String generateFile(String code,String fileName){

FileOutputStream fos = **null**;

**try** {

String compilerDirPath=FileSystem.*appRoot*+*sp*+"compiler";

File dir=**new** File(compilerDirPath);

**if**(dir.isDirectory()==**false**){

dir.mkdir();

}

File file=**new** File(dir,fileName);

fos = **new** FileOutputStream(file);

fos.write(code.getBytes());

} **catch** (FileNotFoundException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

}

**finally**{

**try** {

fos.close();

} **catch** (IOException e) {

e.printStackTrace();

}

**return** fileName;

}

}

/\*

\* removeFiles removes the temporary files generated during code

\* compilation and execution process.

\*/

**public** **void** removeFiles() {

// **TODO** Auto-generated method stub

**try** {

**char** sc=File.*separatorChar*;

String target="webapps"+sc+"test"+sc+"compiler"+sc;

**final** String WINDOWS\_DEL\_CMD="cmd /c del "+target+"\*.class "+target+"Main.java "+target+"input.txt";

System.*out*.println(WINDOWS\_DEL\_CMD);

Process pro=**null**;

pro=Runtime.*getRuntime*().exec(WINDOWS\_DEL\_CMD);

System.*out*.println("exit value : "+pro.waitFor());

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

/\*

\* main() is defined for debugging purposes.

\* It also serves a way to individually test the

\* purpose of this class.

\*/

**public** **static** **void** main(String...asd){

**new** FileSystem().removeFiles();

}

}

//JavaCompiler.java

**package** com.compiler;

**import** java.io.BufferedReader;

**import** java.io.File;

**import** java.io.IOException;

**import** java.io.InputStream;

**import** java.io.InputStreamReader;

/\*

\* JavaCompiler class performs code compilation using

\* java-compiler.

\*/

**public** **class** JavaCompiler **implements** Compiler{

**public** **static** **final** String *fileName*="Main.java";

/\*

\* doCompile() takes code as input ; convert it into

\* .java file which is compiled using javac command,

\* it also catches the compile-time error in Result object;

\* which is then returned to caller method

\*/

@Override

**public** Result doCompile(String code){

FileSystem fs=**new** FileSystem();

String error="";

//System.out.println("code received inside JavaCompiler : "+code);

String fileName=fs.generateFile(code,JavaCompiler.*fileName*);

String command="javac webapps"+FileSystem.*sp*+"test"+FileSystem.*sp*+"compiler"+FileSystem.*sp*+fileName;

Result result=**new** Result();

Process pro = **null**;

**try** {

pro = Runtime.*getRuntime*().exec(command);

lineBreak();

error = printLines("Error :", pro.getErrorStream());

//output = printLines("Output :", pro.getInputStream());

error=(error.length()==0)?"":"Error : "+error;

//output=(output.length()==0)?"":"Output : "+output+"\n";

result.setExitValue(pro.waitFor());

**if**(result.getExitValue()==0){

result.setResultStatus("Compilation status : success\n");

}

**else**{

result.setResultStatus("Compilation status : failed\n");

}

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**finally**{

result.setOutput(error);

}

System.*out*.println(command + "------exitValue() " + pro.exitValue()+"-------");

**return** result;

}

/\*

\* printLines() reads the data from the provided InputStream and

\* return it as a String object

\*/

**private** String printLines(String messageType, InputStream ins) **throws** IOException {

String line = **null**;

String buffer="";

BufferedReader in = **new** BufferedReader(**new** InputStreamReader(ins));

**while** ((line = in.readLine()) != **null**){

System.*out*.println(messageType+ " " + line);

buffer+=line+"\n";

}

String result=buffer.trim().length()==0?"No errors...\n":"errors found....\n";

System.*out*.print(result);

lineBreak();

**return** buffer;

}

/\*

\* lineBreak() is used to print a line in compilation process to standard output

\*/

**private** **void** lineBreak(){

System.*out*.println("cccccccccccccccccccccccccccccccccccccccccccc");

}

}

//JavaExecuter.java

**package** com.compiler;

**import** java.io.BufferedReader;

**import** java.io.BufferedWriter;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.IOException;

**import** java.io.InputStream;

**import** java.io.InputStreamReader;

**import** java.io.OutputStream;

**import** java.io.OutputStreamWriter;

**public** **class** JavaExecuter **implements** Executer {

/\*

\* doExecute() takes .class file name and time limit for execution

\* as input.It invokes local JVM to execute the given .class file.

\* It writes the input provided by user to standard input stream

\* of execution process and retrieves the output,error from stdout

\* and stderr respectively.

\*/

@SuppressWarnings("finally")

@Override

**public** Result doExecute(String fileName,**long** timeLimit) {

Result result=**new** Result();

String buffer="";

String error="";

String output="";

**char** psc=File.*pathSeparatorChar*;

timeLimit=timeLimit\*1000;

System.*out*.println(">>>>>>>------------------time limit ---->"+timeLimit);

**try** {

/\*

\*@tomcat root folder

\*java -cp webapps\test\WEB-INF\classes;. OsValidator

\*/

String command="java -cp webapps"+FileSystem.*sp*+"test"+FileSystem.*sp*+"compiler;. "+fileName;

**final** TimeLimitProcess pro=**new** TimeLimitProcess(Runtime.*getRuntime*().exec(command),timeLimit);

//error = printLines("Runtime Error :", pro.getErrorStream());

System.*out*.println("command-->"+command);

String line=**null**;

OutputStream outputstream = pro.getOutputStream();

OutputStreamWriter outputstreamwriter = **new** OutputStreamWriter(outputstream);

BufferedWriter bufferedwriter = **new** BufferedWriter(outputstreamwriter);

String inputLoc=FileSystem.*appRoot*+FileSystem.*sp*+"compiler"+FileSystem.*sp*+"input.txt";

System.*out*.println("input file location : "+inputLoc);

BufferedReader br=**new** BufferedReader(**new** InputStreamReader(**new** FileInputStream(inputLoc)));//new FileInputStream("compiler/input.txt")

lineBreak();

/\*

\* writing to stdin from input.txt

\*/

System.*out*.println("standard input from user : ");

**while** ( (line = br.readLine()) != **null**){

bufferedwriter.write(line+"\n");

bufferedwriter.flush();

System.*out*.println("["+line+"]");

buffer+=line;

}

lineBreak();

bufferedwriter.flush();

bufferedwriter.close();

**final** java.util.List<String> list=**new** java.util.ArrayList<String>();

Thread outputThread=**new** Thread(**new** Runnable(){

@Override

**public** **void** run() {

// **TODO** Auto-generated method stub

**try**{

list.add(printLines("Output : ", pro.getInputStream()));

}**catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

});

outputThread.start();

//output = printLines("Output : ", pro.getInputStream());

error=printLines("Error : ", pro.getErrorStream());

//output = printLines("Output : ", pro.getInputStream());

outputThread.join();

result.setExitValue(pro.waitFor());

System.*out*.println("exitValue -->"+result.getExitValue());

output=list.get(0);

FileSystem fs=**new** FileSystem();

fs.removeFiles();

lineBreak();

output=(output.length()==0)?"":""+output;

//System.out.println("after execution buffer : "+buffer);

**if**(result.getExitValue()==0){

result.setResultStatus("run status : success\n");

}

**else**{

**if**(error.length()==0)

result.setResultStatus("run status : execution period exceeded the time-limit\n");

**else**

result.setResultStatus("run status : runtime error\n");

}

} **catch** (IOException e) {

e.printStackTrace();

} **catch** (InterruptedException e) {

e.printStackTrace();

}

**finally**{

result.setOutput(output+error);

//System.out.println(result);

lineBreak();

**return** result;

}

}

/\*

\* printLines() reads the data from the provided InputStream and

\* return it as a String object

\*/

**private** String printLines(String messageType, InputStream ins) **throws** IOException {

String line = **null**;

String buffer="";

BufferedReader in = **new** BufferedReader(**new** InputStreamReader(ins));

**while** ((line = in.readLine()) != **null**) {

System.*out*.println(messageType+ " " + line);

buffer+=line+"\n";

}

lineBreak();

//System.out.println("buffer-->"+buffer+"\tinside printLine");

**return** buffer;

}

/\*

\* lineBreak() is used to print a line in execution process to standard output

\*/

**private** **void** lineBreak(){

System.*out*.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");

}

**public** **static** **void** main(String...asd){

**long** timeLimit=2;//in seconds

**new** JavaExecuter().doExecute("Main",timeLimit);

}

}

//Result

**package** com.compiler;

**import** java.io.Serializable;

/\*

\* Result is a Plain Old Java Bean Class.

\*/

**public** **class** Result **implements** Serializable {

**public** Result(){

exitValue=1;

output="";

resultStatus="";

}

**private** **int** exitValue;

**private** String output;

**private** String resultStatus;

**public** **int** getExitValue(){

**return** exitValue;

}

**public** **void** setExitValue(**int** exitValue){

**this**.exitValue=exitValue;

}

**public** String getOutput(){

**return** output;

}

**public** **void** setOutput(String Output){

**this**.output=Output;

}

**public** String getResultStatus(){

**return** resultStatus;

}

**public** **void** setResultStatus(String resultStatus){

**this**.resultStatus=resultStatus;

}

**public** String toString(){

**return** "--------------\n"+getResultStatus()+getOutput()+"\n--------------";

}

}

//TimeLimitProcess.java

**package** com.compiler;

**import** java.io.\*;

**import** java.util.\*;

**import** org.apache.commons.io.IOUtils;

/\*\*

\* TimeLimitProcess is a subclass of Process such that the process is

\* killed if it runs for more than the specified number of milliseconds.

\* Wall clock seconds, not CPU seconds, are measured.

\* The process should already be started when TimeLimitProcess is invoked.

\* Typical use:

\* <pre>

\* ProcessBuilder pb = ...;

\* TimeLimitProcess p = new TimeLimitProcess(pb.start(), TIMEOUT\_SEC \* 1000);</pre>

\*

\* <b>Note</b>: If a Java process is destroyed (e.g., because it times

\* out), then its output is unreadable: Java code trying to read its

\* output stream fails. Here are two ways to get around this problem:

\*

\* <ul>

\* <li>

\* The client of TimeLimitProcess can send the process output to a file (or

\* ByteArrayOutputStream, etc.), which can be read after the process

\* terminates. This is easy to do in Java 7, for example via

\* ProcessBuilder.redirectOutput(tempFile). There does not appear to be an

\* easy way to do it in Java 6.

\* <li>

\* This class provides a workaround, in which it busy-waits reading the

\* standard and error outputs and stores them away. Use

\* ...

\*\*/

**public** **class** TimeLimitProcess **extends** Process {

**private** Process p;

**private** **long** timeLimit;

**private** **boolean** timed\_out = **false**;

// can make public for testing

**private** /\*@LazyNonNull\*/ StringWriter cached\_stdout;

**private** /\*@LazyNonNull\*/ StringWriter cached\_stderr;

**private** Timer timer;

**private** **static** **boolean** *debug* = **false**;

/\*\*

\* Creates a TimeLimitProcess with the given time limit, in wall clock

\* milliseconds.

\* Requires: p != null

\* **@param** timeLimit in milliseconds

\*\*/

**public** TimeLimitProcess (Process p, **long** timeLimit) {

**this**(p, timeLimit, **false**);

}

/\*\*

\* Creates a TimeLimitProcess with the given time limit, in wall clock

\* milliseconds.

\* Requires: p != null

\* **@param** timeLimit in milliseconds

\* **@param** cacheStdout

\* If true, causes the TimeLimitProcess to consume the standard output of the

\* underlying process, and to cache it. After the process terminates (on

\* its own or by being timed out), the output is available via the

\* cached\_stdout method. This is necessary because when a Java process

\* is terminated, its standard output is no longer available.

\*/

**public** TimeLimitProcess (Process p, **long** timeLimit, **boolean** cacheStdout) {

**this**.p = p;

timer = **new** Timer(**true**);

**this**.timeLimit = timeLimit;

**if** (*debug*) {

System.*out*.printf("new timelimit process, timeLimit=%s, cacheStdout=%s%n",

timeLimit, cacheStdout);

}

timer.schedule(**new** TPTimerTask(**this**, timeLimit), timeLimit);

**if** (cacheStdout) {

cached\_stdout = **new** StringWriter();

cached\_stderr = **new** StringWriter();

**new** StdoutStreamReaderThread().start();

**new** StderrStreamReaderThread().start();

}

}

/\*\*

\* Returns true if the process has timed out (has run for more than the

\* timeLimit msecs specified in the constructor).

\*/

**public** **boolean** timed\_out() {

**return** (timed\_out);

}

/\*\*

\* Returns the timeout time in msecs.

\*/

**public** **long** timeout\_msecs() {

**return** (timeLimit);

}

// /\*\*

// \* Returns the standard output of the process, if the cacheStdout

// \* parameter was "true" when the constructor was invoked.

// \* Only for debugging.

// \*/

// public String cached\_stdout() {

// if (cached\_stdout == null) {

// throw new Error("called cached\_stdout() without previously calling cache\_stdout()");

// }

// return cached\_stdout.toString();

// }

/\*\*

\* Kills the subprocess.

\* **@see** Process#destroy()

\*\*/

**public** **void** destroy() {

p.destroy();

}

/\*\*

\* Returns the exit value for the subprocess.

\* **@see** Process#getErrorStream()

\*/

**public** **int** exitValue() {

// I'm not sure whether this is necessary; the Process.destroy()

// documentation doesn't specify the effect on the exit value.

**if** ((p.exitValue() == 0) && timed\_out) {

**return** 255;

} **else** {

**return** p.exitValue();

}

}

/\*\*

\* Gets the error stream of the subprocess.

\* **@see** Process#getErrorStream()

\*/

**public** InputStream getErrorStream() {

**if** (cached\_stderr == **null**) {

**return** p.getErrorStream();

} **else** {

// Convert a String to an InputStream

String text = cached\_stderr.toString();

**try** {

InputStream is = **new** ByteArrayInputStream(text.getBytes("UTF-8"));

**return** is;

} **catch** (UnsupportedEncodingException e) {

**throw** **new** Error(e);

}

}

}

/\*\*

\* Gets the input stream of the subprocess.

\* **@see** Process#getInputStream()

\*/

**public** InputStream getInputStream() {

**if** (cached\_stdout == **null**) {

**return** p.getInputStream();

} **else** {

**return** stringToInputStream(cached\_stdout.toString());

}

}

// Convert a String to an InputStream

**private** InputStream stringToInputStream(String text) {

**try** {

InputStream is = **new** ByteArrayInputStream(text.getBytes("UTF-8"));

**return** is;

} **catch** (UnsupportedEncodingException e) {

**throw** **new** Error(e);

}

}

/\*\*

\* Gets the output stream of the subprocess.

\* **@see** Process#getOutputStream()

\*/

**public** OutputStream getOutputStream() {

**return** p.getOutputStream();

}

/\*\*

\* Causes the current thread to wait, if necessary, until the process represented by this Process object has terminated.

\* **@see** Process#waitFor()

\*/

**public** **int** waitFor() **throws** InterruptedException {

**return** p.waitFor();

}

/\*\*

\* **@return** true if the process if finished, false otherwise

\*\*/

**public** **boolean** finished () {

**try** {

// Process.exitValue() throws an exception if the process is not

// finished.

p.exitValue();

**return** **true**;

} **catch** (IllegalThreadStateException ie) {

**return** **false**;

}

}

/\*\*

\* This TimerTask destroys the process that is passed to it.

\*\*/

**private** **static** **class** TPTimerTask **extends** TimerTask {

TimeLimitProcess tp;

**long** timeLimit;

**public** TPTimerTask(TimeLimitProcess tp, **long** timeLimit) {

**this**.tp = tp;

**this**.timeLimit = timeLimit;

}

**public** **void** run() {

// If exitValue is queried while the process is still running,

// the IllegalThreadStateException will be thrown. If that

// happens, we kill the process and note that so callers can

// tell that a timeout occurred.

**try** {

**int** exit = tp.p.exitValue();

**if** (*debug*) {

System.*out*.println();

System.*out*.println("Process exited with status " + exit);

System.*out*.println();

}

} **catch** (IllegalThreadStateException ie) {

tp.p.destroy();

tp.timed\_out = **true**;

**if** (*debug*) {

System.*out*.println("Terminated process after timelimit of "

+ timeLimit + " msecs expired");

System.*out*.println();

}

}

**this**.cancel();

}

}

// I'm not sure how to generalize the below two classes into one -- my

// attempt failed.

**private** **class** StdoutStreamReaderThread **extends** Thread {

@SuppressWarnings("nullness") // checker bug: NonNullOnEntry cannot access a variable in an enclosing class

/\*@NonNullOnEntry("cached\_stdout")\*/

**public** **void** run() {

// This thread will block as the process produces output. That's OK,

// because the blocking is happening in a separate thread.

**try** {

IOUtils.copy(p.getInputStream(), cached\_stdout);

} **catch** (IOException e) {

// assume the best

}

}

}

**private** **class** StderrStreamReaderThread **extends** Thread {

@SuppressWarnings("nullness") // checker bug: NonNullOnEntry cannot access a variable in an enclosing class

/\*@NonNullOnEntry("cached\_stderr")\*/

**public** **void** run() {

// This thread will block as the process produces output. That's OK,

// because the blocking is happening in a separate thread.

**try** {

IOUtils.copy(p.getErrorStream(), cached\_stderr);

} **catch** (IOException e) {

// assume the best

}

}

}

}

**Package : com.web**

//ControllerServlet.java

**package** com.web;

**import** java.io.\*;

**import** java.io.PrintWriter;

**import** javax.servlet.\*;

**import** javax.servlet.http.HttpServlet;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

**import** java.util.Properties;

**import** com.compiler.\*;

/\*

\* ControllerServlet is a servlet class that receives request

\* from the code html form inside the module's subtopic content

\* page as well as from within the TryIt Zensar Editor.

\* This request contains a piece of code and a runtime input

\* (incase request is coming from TryIt Editor), and then it

\* stores the input data in the filesystem and checks the total

\* active requests to the servlet and decides the maximum time

\* period for executing the code. Once the compilation and

\* execution (if any) is done it it stores the output (if any),

\* error reports(if any) during compilation or execution in the

\* request attribute and forwards the request processing task

\* to TryIt.jsp

\*/

**public** **class** ControllerServlet **extends** HttpServlet {

/\*\*

\* Constructor of the object.

\*/

**public** ControllerServlet() {

**super**();

}

/\*\*

\* Destruction of the servlet. <br>

\*/

**public** **void** destroy() {

**super**.destroy(); // Just puts "destroy" string in log

// Put your code here

}

/\*\*

\* The doGet method of the servlet. <br>

\* This method is called when a form has its tag value method equals to get.

\*/

**public** **void** doGet(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException {

PrintWriter out = response.getWriter();

out.flush();

out.close();

}

/\*\*

\* The doPost method of the servlet.

\* this method performs the task of handling the request,

\* processing the request and forwarding the request

\*/

**public** **void** doPost(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException {

System.*out*.println("inside doPost() of ControllerServlet ....");

/\* File dir1 = new File(".");

File dir2 = new File("..");

try {

System.out.println("Current dir : " + dir1.getCanonicalPath());

System.out.println("Parent dir : " + dir2.getCanonicalPath());

} catch (Exception e) {

e.printStackTrace();

}

\*/

PrintWriter out = response.getWriter();

response.setContentType("text/html");

String code=request.getParameter("code");

System.*out*.println("code received : "+code);

String input=request.getParameter("input");

System.*out*.println("input received : "+input);

ServletContext context=getServletContext();

FileSystem fs=**new** FileSystem();

fs.generateFile(input,"input.txt");

Properties time=**new** Properties();

String pptDir=FileSystem.*appRoot*+FileSystem.*sp*+"WEB-INF"+FileSystem.*sp*+"compiler.properties";

time.load(**new** FileInputStream(pptDir));

System.*out*.println("time\_slot\_1 : "+time.getProperty("time\_slot\_1")+"\t"+time.getProperty("time\_slot\_1").getClass());

System.*out*.println("time\_slot\_2 : "+time.getProperty("time\_slot\_2"));

System.*out*.println("user\_limit : "+time.getProperty("user\_limit"));

**long** timeLimit=Long.*parseLong*(time.getProperty("time\_slot\_1"));

**int** userCount=com.web.RequestCounter.*getActiveRequests*();

**int** userLimit=Integer.*parseInt*(time.getProperty("user\_limit"));

**if**(userCount > userLimit){

timeLimit = Long.*parseLong*(time.getProperty("time\_slot\_2"));

}

System.*out*.println(">>>>>>-----------doPost---------------@>"+timeLimit);

context.setAttribute("timeLimit",timeLimit);

System.*out*.println(">>>>>>------------doPost--------------@>"+timeLimit);

Controller ctrl=**new** Controller();

ctrl.setTimeLimit(timeLimit);

System.*out*.println(">>>>>----------------doPost with ctrl.getTimeLimit------------>"+ctrl.getTimeLimit());

Result result=ctrl.compile(code);

String resultOutput = "<pre>"+result.getOutput()+"</pre>";

//out.println(resultOutput);

String resultStatus = "<pre>"+result.getResultStatus()+"</pre>";

request.setAttribute("code",code);

request.setAttribute("input",input);

request.setAttribute("resultOutput",resultOutput);

fs.generateFile(resultOutput,"result.html");

request.setAttribute("resultStatus",resultStatus);

RequestDispatcher rd=request.getRequestDispatcher("/TryIt.jsp");

rd.forward(request,response);

// out.println("<pre>"+result.getResultStatus()+result.getOutput()+"</pre>");

// out.flush();

// out.close();

}

/\*\*

\* Initialization of the servlet. <br>

\*

\* **@throws** ServletException if an error occurs

\*/

**public** **void** init() **throws** ServletException {

// Put your code here

}

}

//RequestCounter.java

**package** com.web;

**import** javax.servlet.\*;

**import** javax.servlet.http.\*;

/\*

\* RequestCounter is a ServletRequestListener that

\* counts the total active request at any point of time.

\* This Listener needs to be configured inside web.xml

\*/

**public** **class** RequestCounter **implements** ServletRequestListener {

**private** **static** **int** *activeRequests* = 0;

/\*

\*this method is invoked by container whenever

\*any request is received for any Servlet.

\*/

**public** **void** requestInitialized(ServletRequestEvent se) {

*activeRequests*++;

System.*out*.println("\n[[[[[[[new request initialized]]]]]]]]---------->total active active requests : "+*activeRequests*);

}

/\*

\* This method is invoked by container whenever

\* response is delivered to the client for the

\* corresponding request to any Servlet.

\*/

**public** **void** requestDestroyed(ServletRequestEvent se) {

**if**(*activeRequests* > 0)

*activeRequests*--;

System.*out*.println("\n[[[[[[[request destroyed]]]]]]]]]]---------->total active requests : "+*activeRequests*);

}

/\*

\* This is a utility method to get the count of

\* total active requests made to the Servlet.

\*/

**public** **static** **int** getActiveRequests() {

**return** *activeRequests*;

}

}

//SessionCounter.java

**package** com.web;

**import** javax.servlet.\*;

**import** javax.servlet.http.\*;

/\*

\* This class counts the total active session at any point of time

\*/

**public** **class** SessionCounter **implements** HttpSessionListener {

**private** **static** **int** *activeSessions* = 0;

/\*

\* This method is invoked by container whenever a new session

\* is created.

\*/

**public** **void** sessionCreated(HttpSessionEvent se) {

*activeSessions*++;

System.*out*.println("\n[[[[[[[new session created]]]]]]]]---------->total active session : "+*activeSessions*+" @ "+se.getSession()+"\n");

}

/\*

\* This method is invoked by container whenever an

\* already created session is about to destroy.

\*/

**public** **void** sessionDestroyed(HttpSessionEvent se) {

**if**(*activeSessions* > 0)

*activeSessions*--;

System.*out*.println("\n[[[[[[[session destroyed]]]]]]]]]]---------->total active session : "+*activeSessions*+" @ "+se.getSession()+"\n");

}

/\*

\* This method gives the count of total active session.

\*/

**public** **static** **int** getActiveSessions() {

**return** *activeSessions*;

}

}

//TryIt.jsp

<%@ page language="java" import="java.util.\*" pageEncoding="ISO-8859-1"%>

<%@ page isELIgnored="false" %>

<%@ page session="false" %>

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<%

String path = request.getContextPath();

String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";

%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

<base href="<%=basePath%>">

<title>Editor</title>

<meta http-equiv="pragma" content="no-cache">

<meta http-equiv="cache-control" content="no-cache">

<meta http-equiv="expires" content="0">

<meta http-equiv="keywords" content="keyword1,keyword2,keyword3">

<meta http-equiv="description" content="This is my page">

<link rel="stylesheet" type="text/css" href="trycss.css">

</head>

<body>

<div class="container">

<div class="header"><a href="www:zensar.com"><img src="images/zensar\_logo.jpg" alt="Insert Logo Here" name="Insert\_logo" width="200" height="100" id="Insert\_logo" style="background-color: #C6D580; display:block;" /></a>

<!-- end .header --></div>

<div class="content">

<form style="margin:0px" action="./servlet/ControllerServlet" method="post" id="tryitform" name="tryitform" >

<table border="1 " class="maintable" cellpadding="3px" cellspacing="3px" width="100%">

<tr>

<td width="50%" valign="top">

<input style="margin-bottom:5px;font-family:verdana;" name="submit" type="submit" value="Edit and Click Me &raquo;" onClick="displayed()">

<br>

<!-- <a href="http://www.zensar.com/" target="view">click to ZenSar</a>-->

<textarea class="code\_input" id="pre\_code" wrap="logical" name="code">${requestScope.code}</textarea>

<input type="hidden" name="code" id="code" />

<input type="hidden" id="bt" name="bt" />

</td>

<td valign="top">

<p class="result\_header">Your Result:</p>

<iframe class="result\_output" frameborder="0" name="view" src="compiler/result.html"></iframe>

</td>

</tr>

<tr>

<td align="left" class="bottomtext" rowspan="2">

enter your input (stdin) :<br />

<textarea name="input" rows="6" cols="56" style="width:inherit" autofocus><c:out value="${requestScope.input}" default=""/></textarea>

</td>

<td>

result : <div id="result">${requestScope.resultStatus}</div>

</td>

</tr>

<tr>

<td>

time : <c:out value="${applicationScope.timeLimit}" default="5"/> secs

</td>

</tr>

</table>

</form>

<!-- end .content --></div>

<div class="footer">

<p style="text-align:center">some copyright information and privacy policies</p>

<!-- end .footer --></div>

<!-- end .container --></div>

</body>

</html>

//trycss.css

body {

font: 100%/1.4 Verdana, Arial, Helvetica, sans-serif;

background-color: #42413C;

margin: 0;

padding: 0;

color: #000;

}

/\* ~~ Element/tag selectors ~~ \*/

ul, ol, dl { /\* Due to variations between browsers, it's best practices to zero padding and margin on lists. For consistency, you can either specify the amounts you want here, or on the list items (LI, DT, DD) they contain. Remember that what you do here will cascade to the .nav list unless you write a more specific selector. \*/

padding: 0;

margin: 0;

}

h1, h2, h3, h4, h5, h6, p {

margin-top: 0; /\* removing the top margin gets around an issue where margins can escape from their containing div. The remaining bottom margin will hold it away from any elements that follow. \*/

padding-right: 15px;

padding-left: 15px; /\* adding the padding to the sides of the elements within the divs, instead of the divs themselves, gets rid of any box model math. A nested div with side padding can also be used as an alternate method. \*/

}

/\* ~~this fixed width container surrounds the other divs~~ \*/

.container {

width: 960px;

background-color: #FFF;

margin: 0 auto; /\* the auto value on the sides, coupled with the width, centers the layout \*/

}

/\* ~~ the header is not given a width. It will extend the full width of your layout. It contains an image placeholder that should be replaced with your own linked logo ~~ \*/

.header {

background-color: #ADB96E;

}

.content {

padding: 10px 0;

width: 100%;

height:inherit;

float: left;

}

/\* ~~ This grouped selector gives the lists in the .content area space ~~ \*/

.content ul, .content ol {

padding: 0 15px 15px 40px; /\* this padding mirrors the right padding in the headings and paragraph rule above. Padding was placed on the bottom for space between other elements on the lists and on the left to create the indention. These may be adjusted as you wish. \*/

}

/\* ~~ The navigation list styles (can be removed if you choose to use a premade flyout menu like Spry) ~~ \*/

ul.nav {

list-style: none; /\* this removes the list marker \*/

border-top: 1px solid #666; /\* this creates the top border for the links - all others are placed using a bottom border on the LI \*/

margin-bottom: 15px; /\* this creates the space between the navigation on the content below \*/

}

ul.nav li {

border-bottom: 1px solid #666; /\* this creates the button separation \*/

}

ul.nav a, ul.nav a:visited { /\* grouping these selectors makes sure that your links retain their button look even after being visited \*/

padding: 5px 5px 5px 15px;

display: block; /\* this gives the link block properties causing it to fill the whole LI containing it. This causes the entire area to react to a mouse click. \*/

width: 160px; /\*this width makes the entire button clickable for IE6. If you don't need to support IE6, it can be removed. Calculate the proper width by subtracting the padding on this link from the width of your sidebar container. \*/

text-decoration: none;

background-color: #C6D580;

}

ul.nav a:hover, ul.nav a:active, ul.nav a:focus { /\* this changes the background and text color for both mouse and keyboard navigators \*/

background-color: #ADB96E;

color: #FFF;

}

/\* ~~ The footer ~~ \*/

.footer {

padding: 10px 0;

background-color: #CCC49F;

position: relative;/\* this gives IE6 hasLayout to properly clear \*/

clear: both; /\* this clear property forces the .container to understand where the columns end and contain them \*/

}

/\* ~~ miscellaneous float/clear classes ~~ \*/

.fltrt { /\* this class can be used to float an element right in your page. The floated element must precede the element it should be next to on the page. \*/

float: right;

margin-left: 8px;

}

.fltlft { /\* this class can be used to float an element left in your page. The floated element must precede the element it should be next to on the page. \*/

float: left;

margin-right: 8px;

}

.clearfloat { /\* this class can be placed on a <br /> or empty div as the final element following the last floated div (within the #container) if the #footer is removed or taken out of the #container \*/

clear:both;

height:0;

font-size: 1px;

line-height: 0px;

}

-->

.code{

width:50%;

height:100%;

display:block;

border:#093;

background-color:#6FC;

}

.result\_output

{

border:1px solid #c3c3c3;

width:100%;

height:400px;

background-color:#ffffff;

color:#000000;

}

.code\_input{

height:400px;

width:100%;

}

**Some Other Necessary files related to coding**

//html.properties

header=<\!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http\://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"><html xmlns\="http\://www.w3.org/1999/xhtml" xmlns\:ice\="http\://ns.adobe.com/incontextediting"><head><meta http-equiv\="Content-Type" content\="text/html; charset\=utf-8" /><title>Zensar</title><link rel\="stylesheet" href\="styles/home.css" /><script type\="text/javascript">function displayTitle(title){var tag\=document.getElementById('contentTitle');tag.innerHTML\=title;}</script></head><body><div class\="container"><div class\="header"><a href\="http\://www.zensar.com/"><img src\="images/zensar\_logo.jpg" target\="\_blank" alt\="Insert Logo Here" name\="Insert\_logo" width\="22%" height\="90" id\="Insert\_logo" style\="background-color\: \#8090AB; display\:block;" /></a><\!-- end .header --></div>

begin\_sidebar=<div class\="sidebar1">

begin\_module=<div class\="module">

begin\_module\_id=<br /><a style\="font-size\:11px;text-decoration\:none;"> ID \:

end\_module\_id=</a><br />

begin\_author\_name=<a style\="font-size\:11px;text-decoration\:none;"> Author \:

end\_author\_name=</a>

begin\_author\_mail\_id=<br/><a style\="font-size\:11px;text-decoration\:none;"> Author mail ID\:

end\_author\_mail\_id=</a>

end\_module=</div>

begin\_topic\_list=<ul class\="nav">

begin\_topic=<li>

begin\_source=<a href\="contents/

set\_target=" target\="icontent"

begin\_title=\ onClick\="displayTitle('

end\_title=')" >

end\_source=</a>

begin\_subtopic\_list=<ul type\="none" class\="subtopic">

begin\_subtopic=<li>

end\_subtopic=</li>

end\_subtopic\_list=</ul>

end\_topic=</li>

end\_topic\_list=</ul>

begin\_reference=<div class\="reference">References</div>

set\_ref\_source=<a class\="wrap" href\="

set\_ref\_name=" target\="\_blank">

end\_ref=</a>

end\_reference=</div>

end\_sidebar=</div>

begin\_content=<div class\="content">

begin\_content\_title=<center><strong id\="contentTitle">

end\_content\_title=</strong></center>

iframe=<iframe id\="chgMe" src\="welcome\_page.html" width\="100%" height\="100%" name\="icontent"></iframe>

end\_content=</div>

footer=<div class\="footer"><p><b>Zensar Technologies</b> <i style\="font-size\:14px">your transformation partner</i></p></div></div></body></html>

//content.properties

begin\_style=<head><style type\="text/css">

end\_style=</style></head>

begin\_text=<div class\="text"><pre>

end\_text=</pre></div><br /><hr><br />

begin\_code=<form class\="code" action\="http\://10.42.12.234\:6002/test/servlet/ControllerServlet" method\="post" target\="\_blank"><textarea readonly\="readonly" style\="width\:100%;height\:100px;" name\="code">

end\_code=</textarea><br><input type\="submit" value\="Try It"/></form><br /><hr><br />

begin\_video=<video width\="795" height\="490" controls\="controls"><source src\="

end\_video="></video><br /><hr><br />

begin\_audio=<audio controls\="controls" width\="520" height\="340"><source src\="

end\_audio=" /></audio><br /><hr><br />

begin\_image=<img src\="

end\_image=" width\="795" height\="490" /><br /><hr><br />

separator=<br /><hr><br />

//home.css

@charset *"utf-8"*;

/\* CSS Document \*/

**body** {

font: *100%/1.4* *"Trebuchet MS",* *Arial,* *Helvetica,* *sans-serif,* *Helvetica,* *sans-serif*;

font-size:*16px*;

background-color: *#657982*;

}

**a***:link* {

color:*#000000*;

text-decoration:*none*; /\* unless you style your links to look extremely unique, it's best to provide underlines for quick visual identification \*/

}

**a***:visited* {

color: *#000000*;

text-decoration: *underline*;

}

**a***:hover***,** **a***:active***,** **a***:focus* { /\* this group of selectors will give a keyboard navigator the same hover experience as the person using a mouse. \*/

text-decoration:*underline*;

box-shadow:*#3C9*;

}

/\* ~~ this container surrounds all other divs giving them their percentage-based width ~~ \*/

*.container* {

width: *80%*;

max-width: *1260px*;/\* a max-width may be desirable to keep this layout from getting too wide on a large monitor. This keeps line length more readable. IE6 does not respect this declaration. \*/

min-width: *780px*;/\* a min-width may be desirable to keep this layout from getting too narrow. This keeps line length more readable in the side columns. IE6 does not respect this declaration. \*/

background-color: *#FFF*;

margin: *0* *auto*; /\* the auto value on the sides, coupled with the width, centers the layout. It is not needed if you set the .container's width to 100%. \*/

}

/\* ~~ the header is not given a width. It will extend the full width of your layout. It contains an image placeholder that should be replaced with your own linked logo ~~ \*/

*.header* {

background-color:*#999*;

}

/\* ~~ These are the columns for the layout. ~~

1) Padding is only placed on the top and/or bottom of the divs. The elements within these divs have padding on their sides. This saves you from any "box model math". Keep in mind, if you add any side padding or border to the div itself, it will be added to the width you define to create the \*total\* width. You may also choose to remove the padding on the element in the div and place a second div within it with no width and the padding necessary for your design.

2) No margin has been given to the columns since they are all floated. If you must add margin, avoid placing it on the side you're floating toward (for example: a right margin on a div set to float right). Many times, padding can be used instead. For divs where this rule must be broken, you should add a "display:inline" declaration to the div's rule to tame a bug where some versions of Internet Explorer double the margin.

3) Since classes can be used multiple times in a document (and an element can also have multiple classes applied), the columns have been assigned class names instead of IDs. For example, two sidebar divs could be stacked if necessary. These can very easily be changed to IDs if that's your preference, as long as you'll only be using them once per document.

4) If you prefer your nav on the right instead of the left, simply float these columns the opposite direction (all right instead of all left) and they'll render in reverse order. There's no need to move the divs around in the HTML source.

\*/

*.sidebar1* {

float: *left*;

width: *22%*;

height:*550px*;

background-color:*#FFF*;

padding-bottom: *10px*;

}

*.module*{

font-size:*20px*;

text-decoration:*underline*;

text-align:*center*;

height:*auto*;

background-color:*#B0D1B6*;

display:*block*;

outline-width:*1px*;

outline-color:*#000*;

outline-style:*solid*;

margin:*1px*;

}

*.content* {

padding: *10px* *0*;

width: *70%*;

float: *left*;

height:*500px*;

}

*.icontent*{

display:*block*;

border-width:*thin*;

border:*#000*;

width:*inherit*;

height:*inherit*;

background-color:*#CFF*;

}

/\* ~~ This grouped selector gives the lists in the .content area space ~~ \*/

*.content* **ul,** *.content* **ol** {

padding: *0* *15px* *15px* *40px*; /\* this padding mirrors the right padding in the headings and paragraph rule above. Padding was placed on the bottom for space between other elements on the lists and on the left to create the indention. These may be adjusted as you wish. \*/

}

/\* ~~ The navigation list styles (can be removed if you choose to use a premade flyout menu like Spry) ~~ \*/

*#contentTitle*{

display:*block*;

float:*inherit*;

background-color:*#CCC*;

color:*#000*;

font-size:*20px*;

text-align:*center*;

margin-top:*-10px*;

width:*111.56%*;

}

/\* ~~ The footer ~~ \*/

*.footer* {

background-color:*#91B5AF*;

text-align:*center*;

position: *relative*;/\* this gives IE6 hasLayout to properly clear \*/

clear: *both*; /\* this clear property forces the .container to understand where the columns end and contain them \*/

}

/\* ~~ miscellaneous float/clear classes ~~ \*/

*.fltrt* { /\* this class can be used to float an element right in your page. The floated element must precede the element it should be next to on the page. \*/

float: *right*;

margin-left: *8px*;

}

*.fltlft* { /\* this class can be used to float an element left in your page. The floated element must precede the element it should be next to on the page. \*/

float: *left*;

margin-right: *8px*;

}

*.clearfloat* { /\* this class can be placed on a <br /> or empty div as the final element following the last floated div (within the #container) if the #footer is removed or taken out of the #container \*/

clear:*both*;

height:*0*;

font-size: *1px*;

line-height: *0px*;

}

*.container* **a** *#Insert\_logo* {

font-family: *Trebuchet* *MS,* *Arial,* *Helvetica,* *sans-serif*;

background-color:*#91B5AF*;

}

*.header*{

background-color:*#91B5AF*;

}

-->

</**style**><!--[if lte IE 7]>

<**style**>

*.content* { margin-right: *-1px*; } /\* this 1px negative margin can be placed on any of the columns in this layout with the same corrective effect. \*/

**ul***.nav* **a** { zoom: *1*; } /\* the zoom property gives IE the hasLayout trigger it needs to correct extra whiltespace between the links \*/

*#chgMe*{

width:*111%*;

height:*106%*;

}

*.contents*{

width:*100%*;

height:*100%*;

}

**a***.wrap*{

display:*block*;

word-wrap:*break-word*;

background-color:*#91B5AF*;

outline-style:*solid*;

outline-width:*1px*;

outline-color:*#000*;

color:*#00F*;

}

**ul**{

list-style-type:*none*;

}

**ul***.nav* **li**{

display:*block*;

width:*115%*;

height:*auto*;

background-color:*#91B5AF*;

padding:*3px*;

padding-left:*10px*;

margin:*4px*;

border-width:*2px*;

border-color:*#000*;

margin-left:*-40px*;

margin-bottom:*4px*;

margin-top:*-12px*;

color:*#000000*;

font-weight:*bold*;

}

**ul***.subtopic* **li**{

margin-bottom:*2px*;

margin-top:*2px*;

font-weight:*normal*;

}

*.reference*{

display:*block*;

width:*auto*;

height:*auto*;

background-color:#*#91B5AF*;

font-size:*18px*;

padding-left:*10px*;

margin-top:*-13px*;

background-color:*#B0D1B6*;

text-decoration:*underline*;

}

//content.css

*.code*{

display:*block*;

background-color:*#C5E8D4*;

width:*100%*;

height:*auto*;

font-family:*"Courier New",* *Courier,* *monospace*;

font-size:*18px*;

}

*.text*{

display:*block*;

background-color:*#C5E8D4*;

width:*100%*;

height:*auto*;

font-family:*"Times New Roman",* *Times,* *serif*;

font-size:*18px*;

}

//audio\_formats

wav

ogg

mp3

//image\_formats

gif

png

jpeg

jpg

bmp

//video\_formats

webm

ogg

mp4

//web.xml

<web-app>

<servlet>

<servlet-name>ControllerServlet</servlet-name>

<display-name>Controller For Compilation Activities</display-name>

<description>Will Handle and delegate control to model and view</description>

<servlet-class>com.web.ControllerServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>ControllerServlet</servlet-name>

<url-pattern>/servlet/ControllerServlet</url-pattern>

</servlet-mapping>

<filter>

<filter-name>TryItFilter</filter-name>

<filter-class>com.web.TryItFilter</filter-class>

</filter>

<filter-mapping>

<filter-name>TryItFilter</filter-name>

<url-pattern>/TryIt.jsp</url-pattern>

</filter-mapping>

<listener>

<listener-class>com.web.RequestCounter</listener-class>

</listener>

</web-app>

1. **Conclusion**

The tool "Trainers Toolkit" fulfill the requirements and needs for which it was developed. Sample modules prepared from this tool proves this tool efficient and easy way for learning. It is ready to be handover to the companies which can use this tool for training purposes. Help window gives basic understanding of the tool.Any trainer can easily generate any java module.The module is in HTMl format which have text, audio, video, image.The java compiler is also successfully integrated with which user can do coding, testing,code practise part at the same instance. The modules prepared have its identity which makes them unique. The tool is very flexible and compatible.It is a stand -alone application and Its .exe can easily be deployed on any machine and can be used for developing training modules.

**Future enhancement**

* Provide privileges to administrator.
* Admin will be able to see how many users are currently accessing the page (compiler).
* Admin can control the execution time according to network traffic (compiler).
* Provision for priorities of the user accounts and account management to allow different execution time.
* Embedding Test engine.
* Add more compilers like .net or python so that training module related to them can be prepared with this tool.
* Embedding online SQL compiler which executes SQL queries.
* Providing JDBC compiling capability. Integrating java and sql.