ગુજરાત રાજ્યના શિક્ષણવિભાગના પત્ર-ક્રમાંક
મશબ/1214/15/છ, તા. 29-1-2014થી–મંજૂર

COMPUTER STUDIES

Standard 12

PLEDGE

India is my country.
All Indians are my brothers and sisters.
I love my country and I am proud of its rich and varied heritage.
I shall always strive to be worthy of it.
I shall respect my parents, teachers and all my elders and treat everyone with courtesy.
I pledge my devotion to my country and its people.
My happiness lies in their well-being and prosperity.

Price : ₹ 105.00

Gujarat State Board of School Textbooks
'Vidyayan', Sector 10-A, Gandhinagar-382010

© Gujarat State Board of School Textbooks, Gandhinagar
Copyright of this book is reserved by Gujarat State Board of School Textbooks.
No reproduction of this book in whole or in part, in any form is permitted without written permission of the Director, Gujarat State Board of School Textbooks.

Subject Adviser

Prof. R. P. Soni

Authors

Dr Harshal Arolkar (Convener)
Shri Jigneshbhai Smart
Shri Jyotikaben Doshi
Shri Triptiben Dodiya
Shri Hardikbhai Joshi

Reviewers

Shri Girish S. Brahmbhatt
Shri Saket A. Dave
Shri Bimal K. Raval
Shri Rajanikant A. Pandya
Shri Ramaniklal L. Gilatar
Shri Ankitaben S. Dwivedi
Shri Pankaj R. Shukla
Shri Mayuriben I. Shah
Shri Rajshreeben N. Padia
Shri Nishitaben N. Gandhi
Shri Sejalben D. Trivedi

Co-ordinator

Shri Ashish H. Borisagar
(Subject Co-ordinator : Mathematics)

Preparation and Planning

Shri C. D. Pandya
(Dy. Director : Academic)

Lay-out and Planning

Shri Haresh S. Limbachiya
(Dy. Director : Production)

PREFACE

The Gujarat State Secondary and Higher Secondary Education Board has prepared new syllabi based on the open source operating system and compitible open source software tools for various topics of Computer Studies. These syllabi are sanctioned by the Government of Gujarat.

It is a matter of pleasure for the Gujarat State Board of School Textbooks to place this textbook of **Computer Studies** prepared according to the new syllabus before the students of **Standard 12**.

Before publishing the textbook, its manuscript has been fully reviewed by experts and teachers teaching at this level. Carrying out suggestions given by teachers and experts, we have made necessary changes in the manuscript and then have published the textbook.

The board has taken special care to ensure that this textbook is interesting, useful and free from errors. However, we welcome suggestions to enhance the quality of the textbook.

Dr Bharat Pandit	**Dr Nitin Pethani**
Director	Executive President
Date : 1-2-2014	Gandhinagar

First Edition : 2014
Published by : Bharat Pandit, Director, on behalf of Gujarat State Board of School Textbooks, 'Vidyayan', Sector 10-A, Gandhinagar
Printed by :

FUNDAMENTAL DUTIES

It shall be the duty of every citizen of India :

(a) to abide by the Constitution and respect its ideals and institutions, the National Flag and the National Anthem;

(b) to cherish and follow the noble ideals which inspired our national struggle for freedom;

(c) to uphold and protect the sovereignty, unity and integrity of India;

(d) to defend the country and render national service when called upon to do so;

(e) to promote harmony and the spirit of common brotherhood amongst all the people of India transcending religious, linguistic and regional or sectional diversities; to renounce practices derogatory to the dignity of women;

(f) to value and preserve the rich heritage or our composite culture;

(g) to protect and improve the natural environment including forests, lakes, rivers and wild life, and to have compassion for living creatures;

(h) to develop the scientific temper, humanism and the spirit of inquiry and reform;

(i) to safeguard public property and to abjure violence;

(j) to strive towards excellence in all spheres of individual and collective activity so that the nation constantly rises to higher levels of endeavour and achievement.

(k) who is a parent or guardian to provide opportunities for education to his child or, as the case may be, ward between the age of six and fourteen years.

CONTENTS

1. Creating HTML forms using KompoZer 1
2. Cascading Style Sheets and Java script 25
3. Designing simple website using KompoZer 52
4. Introduction to E-Commerce 72
5. Introduction to M-Commerce 90
6. Object-Oriented concepts 116
7. Java Basics 127
8. Classes and objects in Java 158
9. Working with Array and String 186
10. Exception handling in Java 203
11. File handling 220
12. Publishing documents using LaTeX 241
13. Other useful free tools and services 260

About This Textbook...

Dear Teachers,

With a mission to spread computer literacy on a fast track, the Gujarat Government has provided latest computer equipments to more than 6000 aided schools under the ICT@School program. As a new policy initiative, all the schools are given the Ubuntu (a variant of Linux) Operating System and other Open Source software packages so that schools can freely use and exchange the software without bothering about the licensing issues. Since earlier text books were largely based on proprietary software, there was a need to rewrite the textbooks based on new syllabus. This was also necessary in view of the fact that the 8th standard has been transferred to primary section. Therefore, new content has been provided for 9 to 12 standard in a phased manner based on the open source Operating System and compatible open source software tools for various topics of computer studies.

Students are now quite familiar with computer software fundamentals such as Operating System, Open Office components, HTML, C Programming, Internet and Web surfing etc. This textbook for 12th standard is the fourth in series for the subject of 'Computer Studies'. It aims at introducing creation of HTML forms, Writing Javascripts, Using Kompozer for designing simple websites, and E-Commerce as well as M-Commerce. Moreover, after introducing new paradigm of Object Oriented Concepts, it describes basic JAVA language which is gaining popularity since it is the language extensively used in web applications. At the end, some useful and popular open source packages like LaTeX, and other miscellaneous tools are introduced for the knowledge of students.

We hope, this coverage will be useful to the students of 12th standard and you will enjoy teaching and conducting practicals on these topics using open source Ubuntu operating system environment and related tools.

Dear Students,

So far in standards 9 to 11, you have learnt Operating System Ubuntu, Open Office components such as Writer, Calc, Impress; and database management tool Base as well as web page presentation in HTML. You also learnt algorithm development and basic C Programming which is considered as procedure oriented programming language. In addition you were introduced to multimedia tools for animation.

This text of 12th standard presents Web Page designing using Kompozer, E-Commerce, fundamentals of Object Oriented Concepts, JAVA programming and other useful tools.

Chapters 1 to 3 lead you to try your hand at creating simple yet working websites. For this, HTML forms, Stylesheets, JavaScripts and Kompozer tool for creation of websites are discussed in these chapters. E-Commerce, M-Commerce and L-Commerce are introduced in chapters 4 to 5. Nowadays most of the software development is done using Object Oriented Methodology. Therefore, you will learn Object Oriented Concepts in chapter 6. The prominent and current web application development language Java which is also object oriented language is introduced in chapter 7. The features of Java such as Classes and Objects, Arrays and Strings, Exception handling and File handling are covered in chapters 8 to 11. There is another open source word processing package very popular amongst book authors, writers, publishers and scientists known as LaTeX; and it is introduced in chapter 12. In the last chapter, some useful open source tools i.e., Archive Manager, VLC Media Player, Google Maps, statistical tool R, Skype and Rational Plan are briefly discussed just to apprise you about their usefulness in normal working and doing communication with computers.

It is expected that if you carefully study the text and practice the laboratory exercises, you will develop reasonable confidence in creating simple websites and writing simple Java programs.

Creating HTML forms using KompoZer 1

With increase in the use of Internet many activities have become online. We may use a web page to fill information about ourselves or a product. HTML forms are used to help the visitors of the website to input data. It allows for more interactivity and control in data entry. For instance, if you want to open a mail account or register on a website, you need to enter your personal details in a form. This information is used to setup the account for the user. In order to obtain such information on Internet, HTML forms are used. This data is further stored by the application and used to retrieve the details about the users registered on the website.

A form in HTML is a container used to collect different kinds of inputs from the user. HTML forms contains elements like label, checkbox, text input field, radio button, submit button, reset button and many more. These elements are used to enter the data as well as validate the data within the forms. We will create a simple form to enter the data using HTML tags, but before that let us discuss about the elements used to create HTML forms. The elements used are described in the section below :

- Form
- Input
- Textarea
- Select and Option

Form elements

The form element is used to create an HTML form. It acts as a container for all the elements used in the form. The tag <form>… </form> is used to implement this element. The example shows the use of the form element :

```
<form action="register.html" method="post">
.
.
input elements
.
.
</form>
```

Observe that the form element uses two attributes namely action and method. The action attribute is used to specify where to send the form data when the form is submitted. It takes a filename as value. This file is opened when the user clicks on the submit button after filling the data in the form.

The method attribute specifies the HTTP method to be used when sending the data. It can take

two values; GET and POST. The GET method retrieves the data from the form and sends it to the server by attaching it at the end of the URL. This method allows only a limited amount of information to be sent at a time. In the POST method, the data is sent as a block through the HTTP transaction. The data is included in the body of the request. This method does not have any restrictions on data length. The default value for method attribute is GET.

Input element

The input elements are used to insert various fields like radio button, text box and checkbox in the form. The tag <input> ….</input> or <input> is used to implement this element. The input tag has different attributes like type, name and value.

The type attribute of the input element specifies the field that is to be created in the form. The name attribute specifies the name to be used for the field in the form. The value attribute specifies the default value of the field in the form. Table 1.1 shows different values of attribute and its usage.

Type	Description	Example
Radio	Creates radio buttons in the form. Any one radio button can be selected at a time from a group of radio buttons. Generally used to select a single item from a given group of items.	<INPUT TYPE = "radio" NAME = "var" VALUE = "txt">
Checkbox	Creates checkboxes in the form. Multiple check boxes can be selected at a time. Generally used to select a multiple items from a given group of items.	<INPUT TYPE="checkbox" NAME = "var" VALUE ="txt" >
Text	Creates a text field to enter text in the form. A user can enter any data of his choice in the text field.	<INPUT TYPE = "text" NAME = "var" VALUE = "txt" >
Password	Creates a password field in the form. Similar to the text field but the characters are not displayed to the user. Instead the character typed is converted into non readable format.	<INPUT TYPE = "password" NAME = "var" >
Submit	Creates a submit button in the form. On clicking the submit button, the values of data entered in the form is submitted to the file specified in the action attribute of the form element.	<INPUT TYPE = "submit" VALUE = "label" >
Reset	Creates a reset button in the form. On clicking the reset button, the values of data entered in the form are cleared and set back to default values.	<INPUT TYPE = "reset" VALUE = "label" >

Table 1.1: Values of Type attribute used with input tag

Textarea element

The Textarea element allows multi-line text input. The tag <textarea>…</textarea> is used to implement this element. It allows entering unlimited number of characters. It can be used to enter comment, report or a long description of product. The size of a textarea element can be specified using rows and cols attributes. The rows attribute is used to set the number of rows of text that will be visible without scrolling up or down. The cols attribute is used to set the number of columns of text that will be visible without scrolling right or left. The following example shows how to insert a textarea in the form.

```
<form method="post" action="comment.html">
Input your comments: <br /> <textarea name="comments" rows="4" cols="20">
...Your comments here...
</textarea>
</form>
```

Select and Option element

The select element is used to create a drop down list or menu in a form. The option element is used to specify the values that are to be displayed in the menu. The tag <select>….</select> is used to create a drop down menu. The tag <option>…</option> is used to create the elements within the menu. Following example shows the use of select and option element.

```
<select>
<option value="Ahmedabad" >Ahmedabad</option>
<option value="Rajkot" >Rajkot</option>
<option value="Surat" >Surat </option>
</select>
```

Let us now create a sample registration form using the elements learned so far. Code listing 1.1 shows the HTML source code used to generate the form. The output of the code is shown in figure 1.1.

```
<HEAD>
        <TITLE>Registration Form</TITLE>
</HEAD>
<BODY bgcolor="lightblue">
<h1> <center>Registration Form</center></h1>
<FORM name="frmRegistration" action="form.html">
        <center>
        <TABLE BORDER="0">
<TR>
        <TD width="12%">First Name</TD>
        <TD width="1%"> </TD>
```

```
        <TD> <INPUT type="textbox" name="txtFirstName"></TD>
</TR>
<TR>
        <TD>Middle Name</TD>
        <TD> </TD>
        <TD><INPUT type="text box" name="txtMiddleName"></TD>
</TR>
<TR>
        <TD>Last Name</TD>
        <TD> </TD>
        <TD> <INPUT type="text box" name="txtLastName"></TD>
</TR>
<TR>
        <TD>Gender</TD>
        <TD> </TD>
        <TD>
        <INPUT type="radio" name="Gender" value="male" CHECKED>Male
        <INPUT type="radio" name="Gender" value="female" >Female
        </TD>
</TR>
<TR>
        <TD>Hobby</TD>
        <TD> </TD>
        <TD>
        <INPUT type="checkbox" name="chkSinging" value="Sing" CHECKED>Singing
        <INPUT type="checkbox" name="chkDancing" value="Dance">Dancing
        <INPUT type="checkbox" name="chkReading" value="Read">Reading
        </TD>
</TR>
<TR>
        <TD>Address</TD>
        <TD> </TD>
        <TD>
        <Textarea name="txtAddress" rows="5" cols="70">Insert Address Here</Textarea>
        </TD>
</TR>
<TR>
        <TD>City</TD>
        <TD> </TD>
        <TD>
```

```
            <Select Name="cmbCity">
            <Option >Ahmedabad</Option>
            <Option >Baroda</Option>
            <Option selected>Rajkot</Option>
            <Option >Surat</Option>
            </Select>
            </TD>
      </TR>
      <TR>
            <TD> </TD>
            <TD> </TD>
            <TD> <INPUT type="submit" name="cmdSubmit" value="Submit">
            <INPUT type="reset" name="cmdReset" value="Reset">
            </TD>
      </TR>
      </TABLE>
      </center>
</FORM>
</BODY>
</HTML>
```

Code listing 1.1: HTML code to generate sample registration form

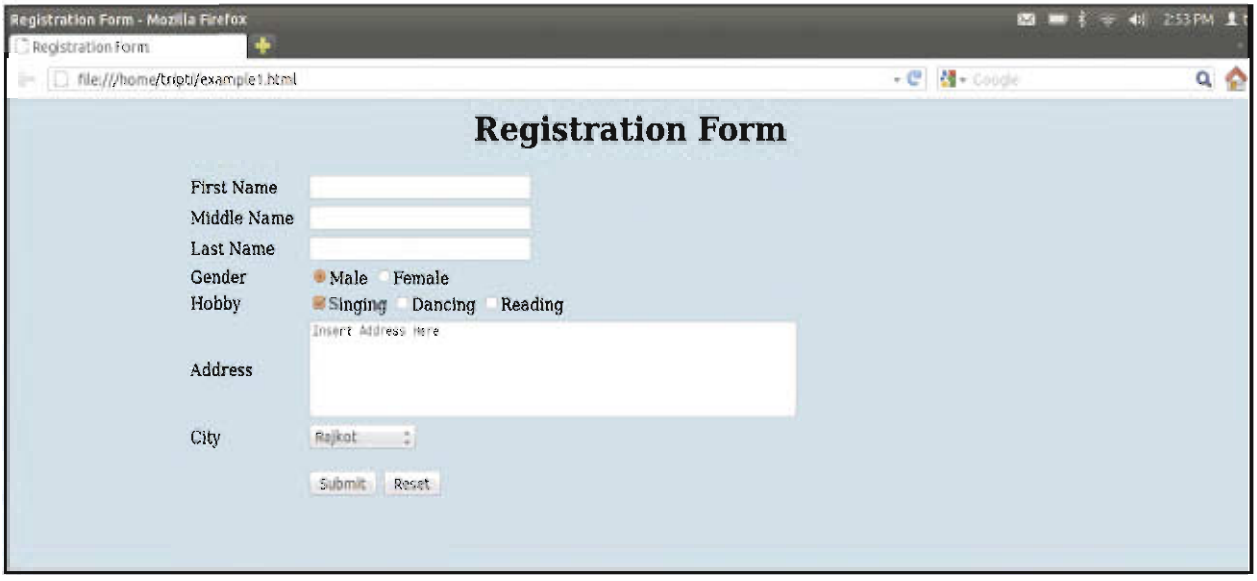


Figure 1.1: Registration Form as displayed in the web browser

Thus, you can see that creating a form using HTML tags is a tedious process. A simpler method is to use an IDE (Integrated Development Environment). An IDE is a software application that provides complete facilities to programmer to develop software. It provides a GUI (Graphical User Interface), text or code editor, a compiler and/or interpreter and a debugger. KompoZer, Eclipse, JBuilder and Netbeans are all examples of some open source IDEs. Let us discuss how to use the KompoZer to create web pages.

Introduction to KompoZer

KompoZer is a free open source web development IDE. It can be downloaded from http://www.KompoZer.net. It provides a web page editor which has a simple graphical interface known as WYSIWYG "what you see is what you get". It is a complete web authoring system which integrates web page development and web file management. Creating new web pages using KompoZer is quick and easy. The users can also edit the web pages by using the source code and making changes. KompoZer incorporates a Site Manager which gives rapid access to the files on both local machines and remote servers. Web pages and associated files can be uploaded to a remote server from within KompoZer. It also supports the use of "Styles" through Cascading Style Sheet (CSS). We will learn about CSS in the next chapter.

Before we learn how to create forms using KompoZer, let us first learn the KompoZer interface. Open KompoZer by locating its icon. To view the different toolbars and status bar (if not visible) click on **View → Show/Hide.** Select all the options listed: Composition Toolbar, Format Toolbar1, Format Toolbar2, Edit Mode Toolbar and Status bar. Site Manager and Rulers option should also be checked. Figure 1.2 shows the window that will be displayed after selecting the toolbars.

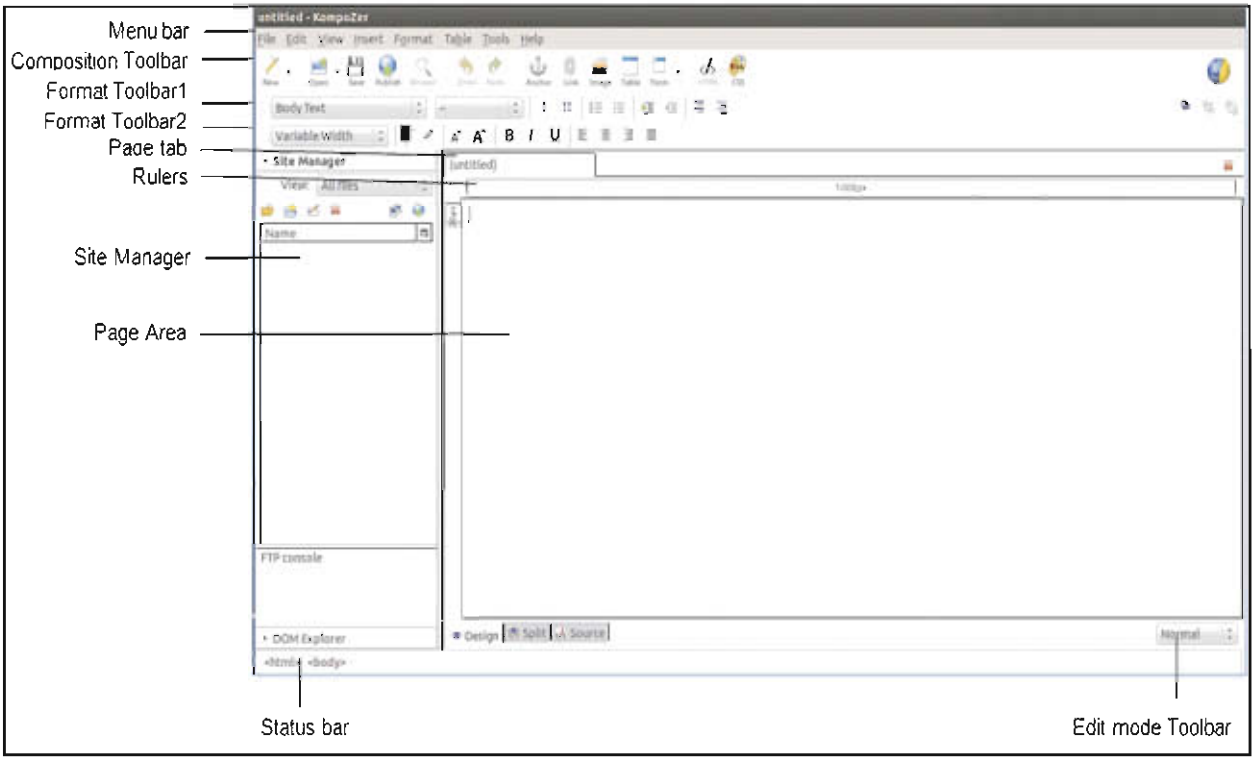


Figure 1.2: KompoZer Interface

Observe that in figure 1.2 you can see the menu bar on the top with the options like File, Edit, View, Insert, Format, Table, Tools and Help. Below the menu bar there are three Toolbars: Composition, Format Toolbar1 and Format Toolbar2.

The composition toolbar is used to create new file, open a file, save a file or publish a web page. The Format toolbar1 and Format toolbar2 are used to format the text, add bullets, numbering and perform similar formatting operations.

In the centre of the window, you can see two panes: Site Manager and blank web page. Site manager is a powerful tool used to navigate within the site or between the sites. You can close the site manager pane by clicking on close button or press F9. The page pane shows a blank untitled web page. The bottom right side of the window shows Edit mode toolbar with three viewing modes: Normal, HTML Tags and Preview. All the three viewing modes provide editing facilities.

The Preview mode offers the page view as seen in a browser. The difference is that in the preview mode the scripts do not run and therefore their effects will not be seen. The links also does not operate in preview mode.

The Normal view is very similar to preview mode. In this mode the table outlines are visible.

The HTML Tags view helps those who are familiar with HTML. A yellow marker is used to indicate the start tag for all elements. Clicking on a marker selects and highlights whole of the element.

The left side of the page pane shows Design, Split and Source tabs. The Design tab is used to design the web page. The Split tab displays the HTML source of the current element. Source tab shows all details of the HTML code. It helps in editing the source code.

On the bottom of the window you can see the status bar. When we click on any item in page, its structure appears in the status bar. If you want to customize any toolbar, right click on the respective toolbar and click on customize toolbar option. You can then customize the toolbar as per your choice.

Create a New File

To create a new file, open KompoZer. In the menu bar click **File → New**. A dialog box as shown in figure 1.3 appears with title "Create a new document or template". Select "Blank document" option from the options that are visible in the dialog box. At the bottom of the dialog box you can see a label "Create in". Select New Tab option from the drop down menu next to it. This allows us to create the page in a new tab. Click on the Create button.

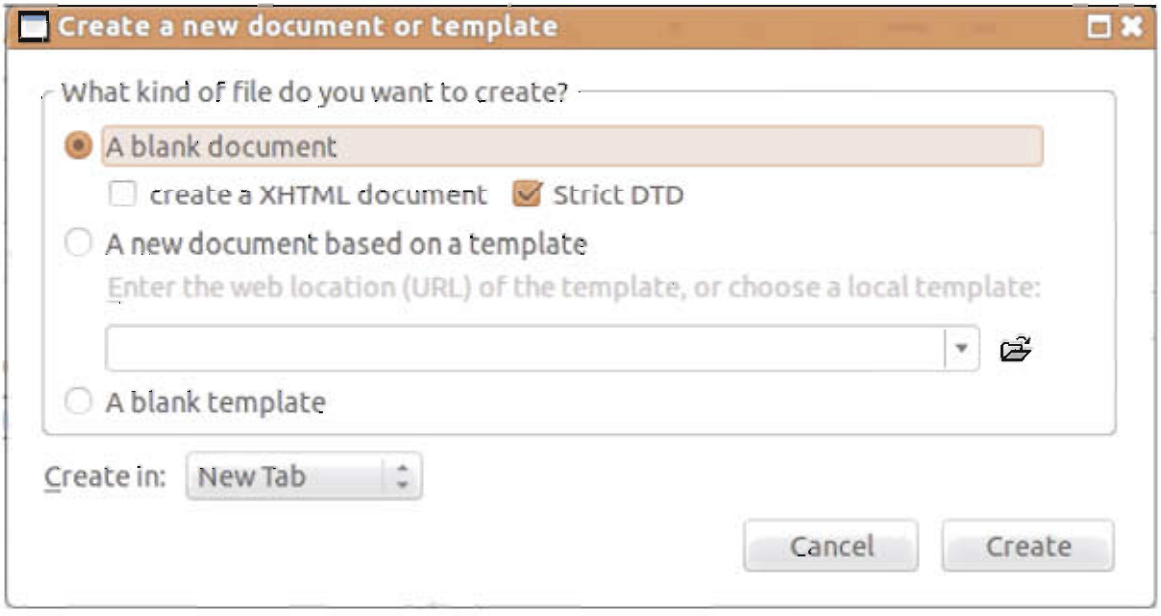


Figure 1.3 : Create a new file

Open an existing File

To open an existing file, click on

icon present on the composition toolbar. Alternatively, you can also click on **File → Open.** If the file has been opened recently, then you can also open the file from **File → Recent Pages.**

Now let us learn to create a forms using KompoZer. We will create a simple form with two input fields: Name and E-mail address and a submit button. Follow the steps given to create the form.

- Open KompoZer. Create a new file.
- From the menu bar, select **Insert → Form → Define Form.** Alternatively, you can also click on

in the composition toolbar. This will open a Form Properties dialog box as shown in figure 1.4. Clicking on More Properties shows added options for the form.

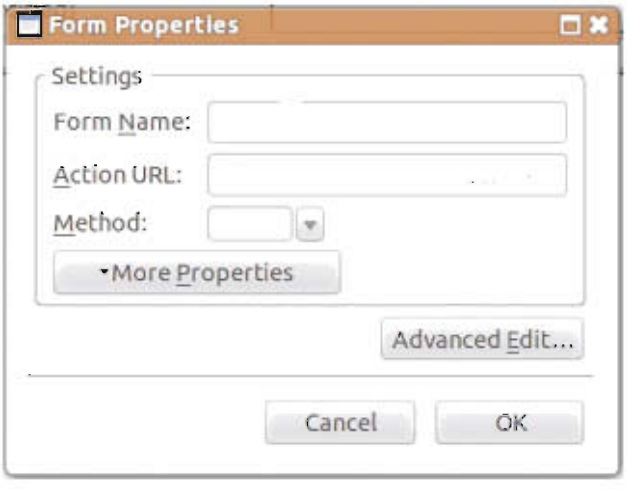


Figure 1.4 : Form properties dialog box

- Enter appropriate name for the form. In the Action URL, enter the file name where you want the form data to be submitted. Select the method POST from the method drop down menu, and click on the OK button. Figure 1.5 shows the details added in the form property dialog box.

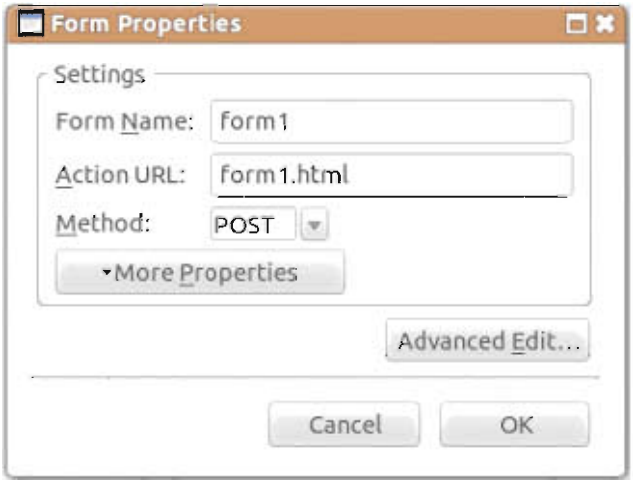


Figure 1.5 : Details entered in Form properties dialog box

- You can see the form inserted with light blue colored outline in the untitled page as shown in figure 1.6. In normal view, the forms are shown surrounded by a dotted blue box. All the form elements like text box, radio button, check box and drop down box will be placed within this box. Press the Enter key a few times to give some space to work on the form.

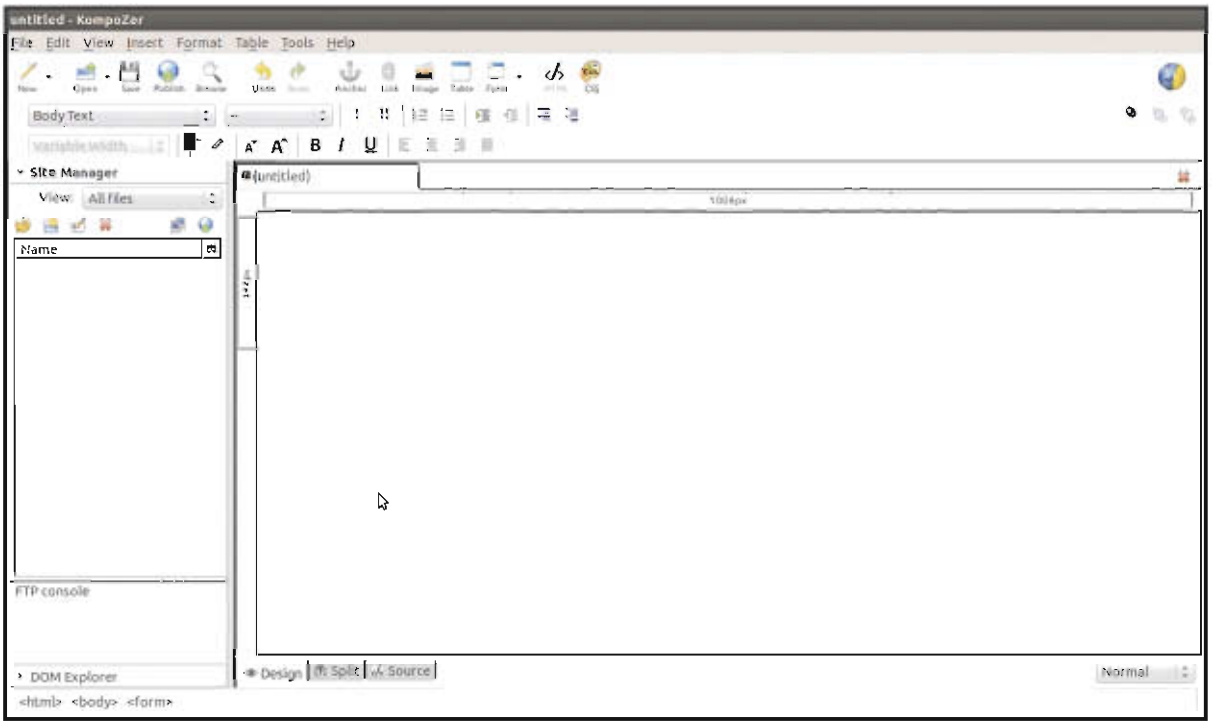


Figure 1.6 : Light blue colored form outline.

- First, we will insert a label for the name field. Click on **Insert → Form → Define label.** Place the cursor in the form where you want your label to appear. Type the text "Name" in the label as shown in figure 1.7. To come out of the label field, click outside the field.

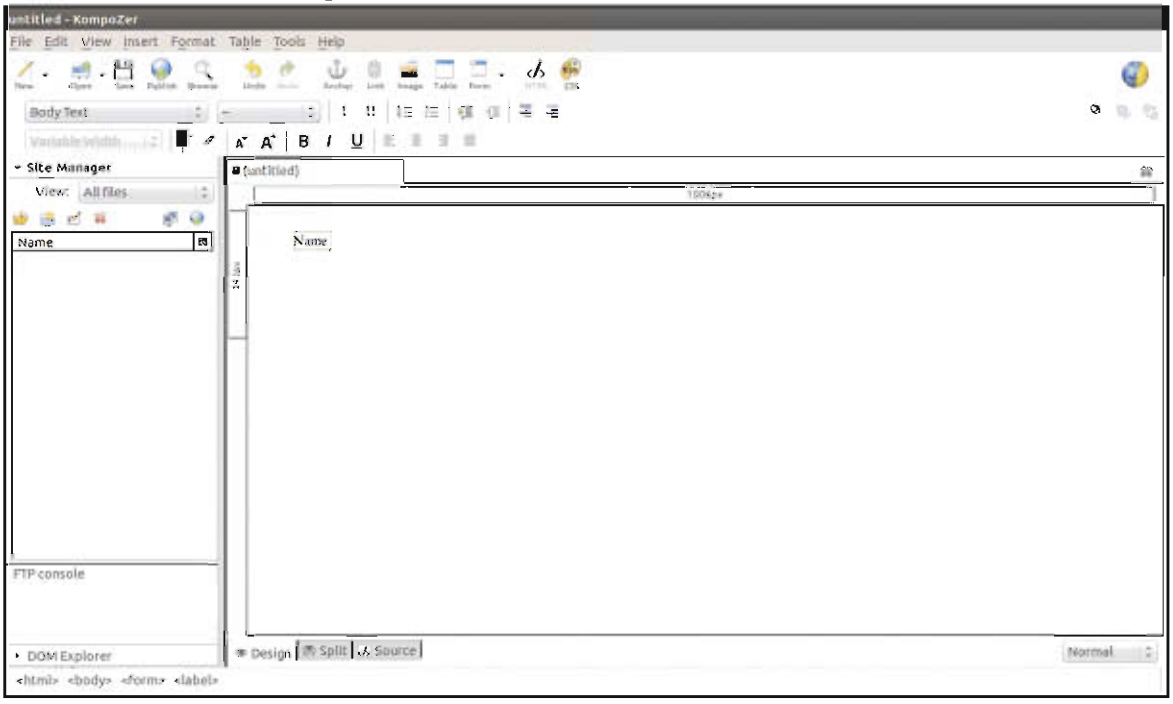


Figure 1.7 : Label field inserted in the form

- To insert an input text field in the form, click **Insert → Form → Form Field.** Figure 1.8 shows the form field properties dialog box. The drop down menu shows various input field type options which we had discussed earlier. Click on more properties, it shows advanced properties related to the field like field size, maximum length.

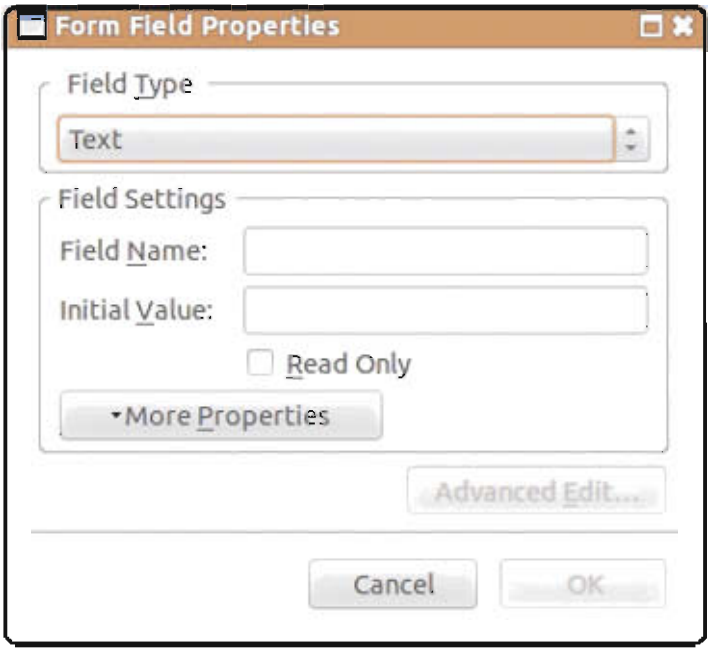


Figure 1.8 : Add form field

- From the drop down menu select Text. Under the Field Settings heading, enter a name in the Field Name text box. In our case we have used name as the field name. Enter some text in the Initial Value field, if you want to show some text before the user actually enters data. Here we have left this field empty. Click on the OK button. Figure 1.9 shows the form after text input field is added.

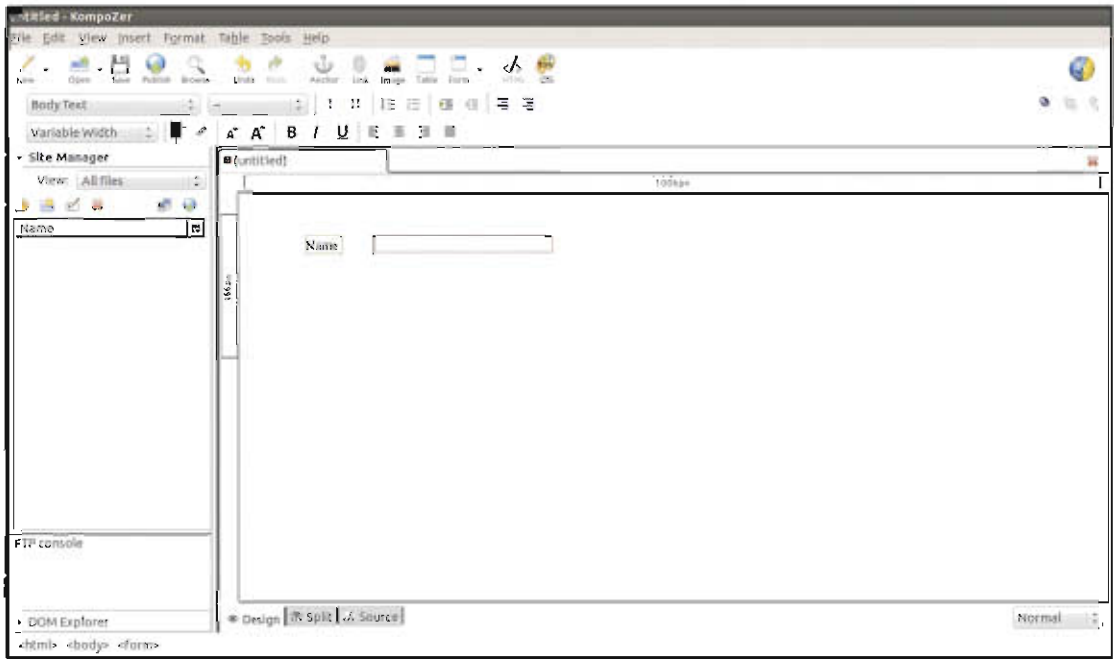


Figure 1.9 : Label and Text field added

- Similarly, we will add another label "E-mail" below the name label field. Add an input text field for E-mail, just like we did for the name field. Here enter text abc@xyz.com in the Initial Value textbox. This will help the user understand the format of E-mail address. Figure 1.10 shows the form with both the text field added.

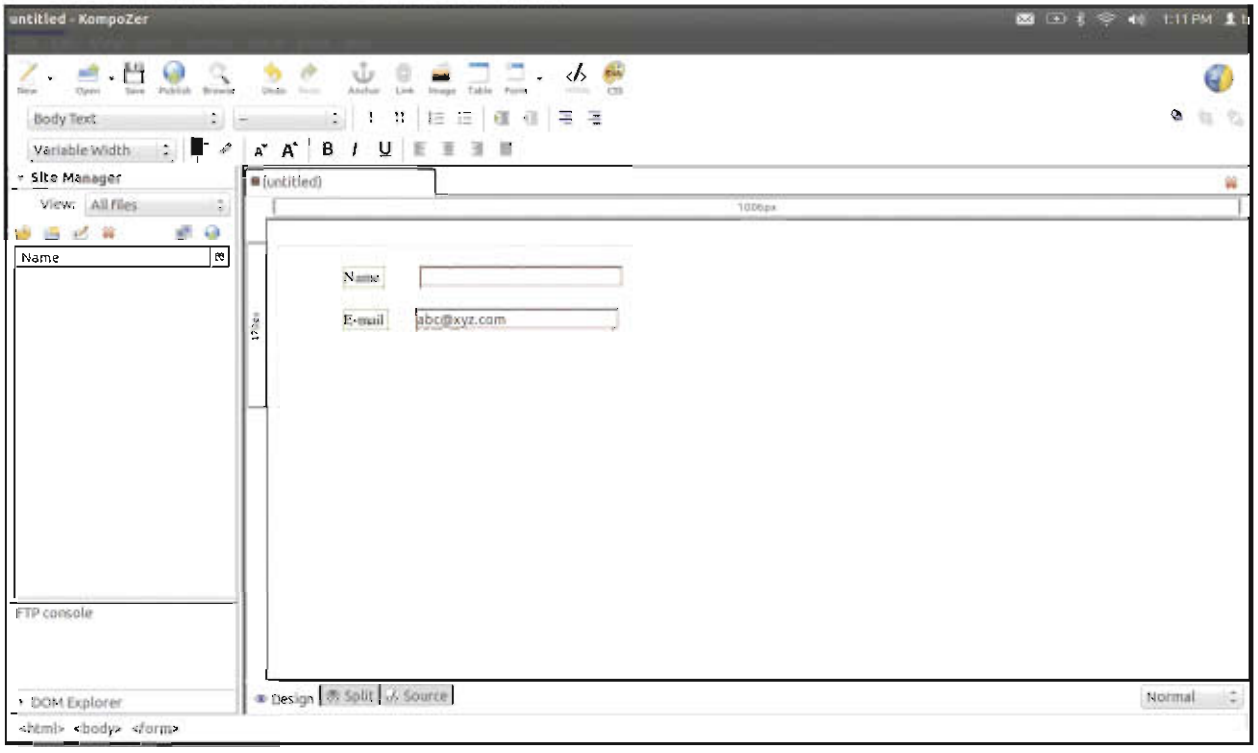


Figure 1.10 : Form with both text fields added

- Finally we will add a submit button in the form. Click **Insert → Form → Form Field.** From the drop-down menu select Submit Button. Type Submit in the both Field Name and Field Value text boxes, and click on the OK button. Figure 1.11 shows the look of form properties dialog box for submit button.

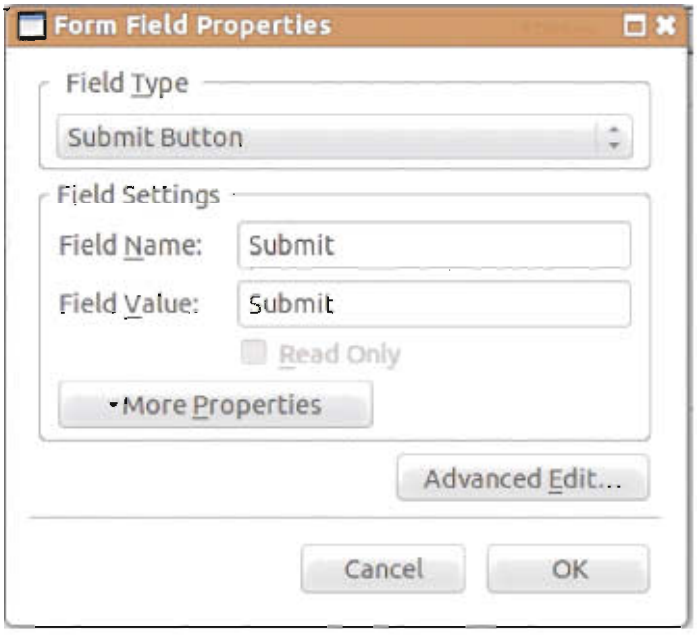


Figure 1.11 : Input field submit button

- The form at present is in the normal view. To have a preview of the form click on Edit mode toolbar and select Preview. Figure 1.12 shows the form in the preview mode.

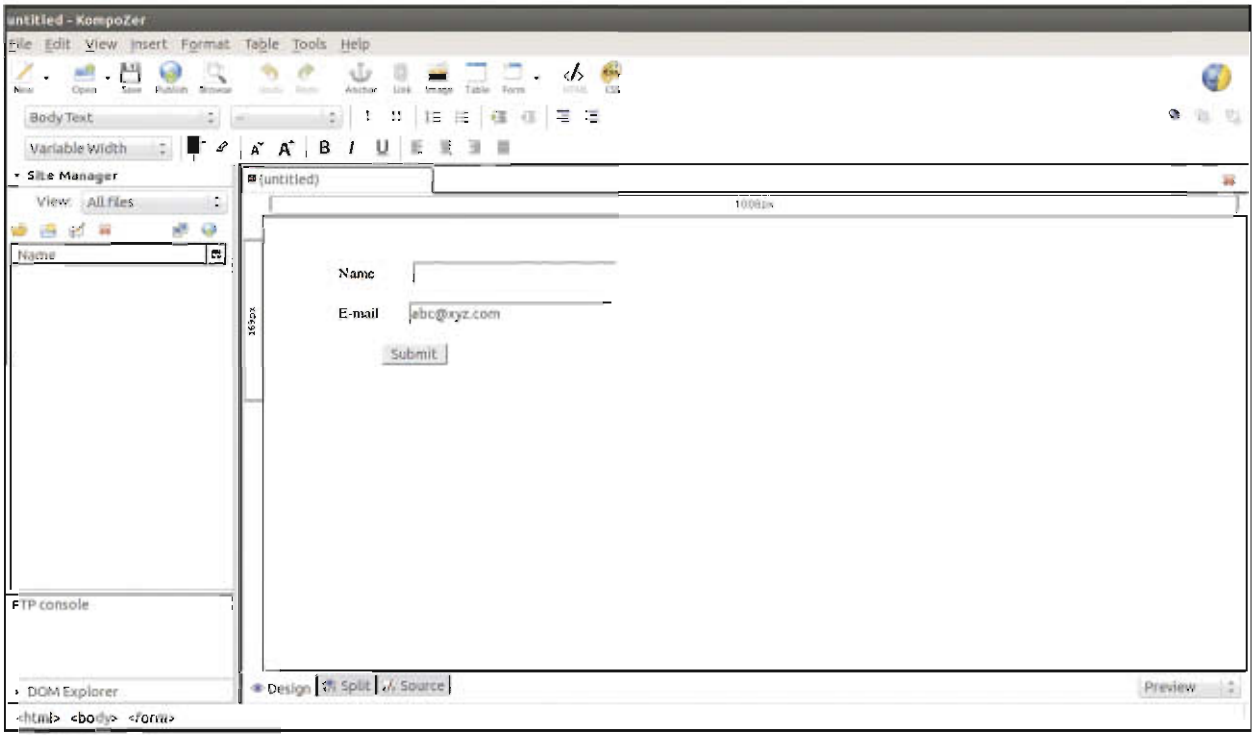


Figure 1.12 : Form as seen in preview mode

Thus, we have created our first form using KompoZer. You can see how KompoZer helps you creating forms within a short span and also relieves you from the tedious job of writing the source code which takes a long time. You can also see the source code of the form just created by clicking on the Source tab. See figure 1.13.

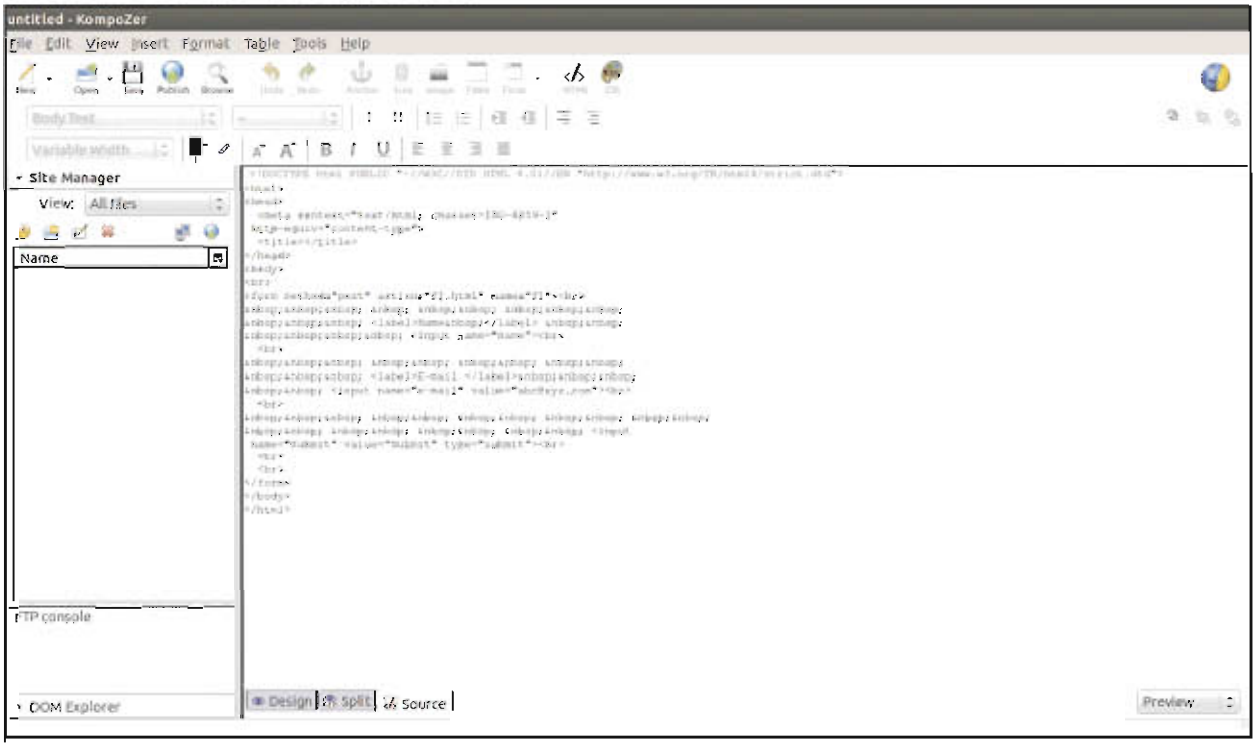


Figure 1.13 : Source code view of form

- Let us save the file which we have created. Click **File → Save.** Alternatively you can also click on the save button in the composition toolbar. This opens a Page Title dialog box as shown in figure 1.14. We can give a suitable title to the web page here. We have given the name "example1" to the title page. Now click on OK button.

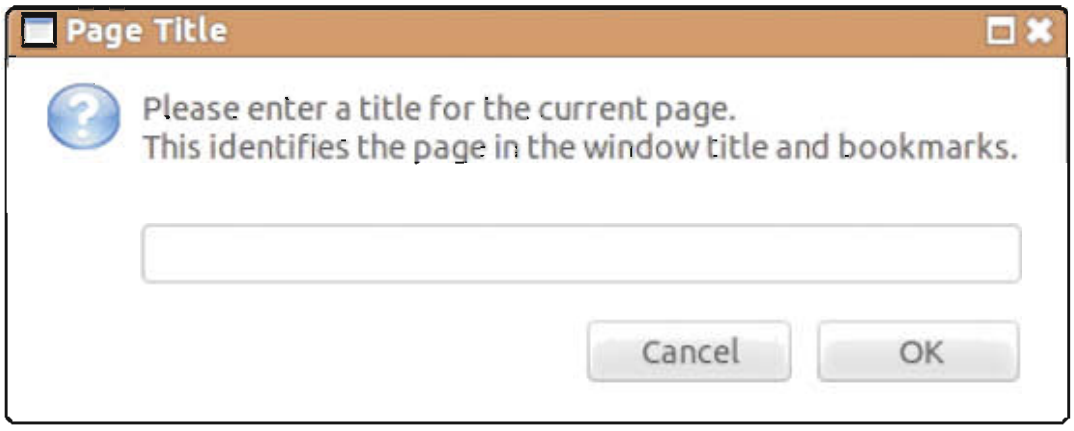


Figure 1.14 : Page title dialog box

The page title will be displayed in the browser windows title bar when viewed in the browser. In case we have created multiple web pages we should give the name of the website as the title page. In this example, since we created a single web page with a form, the title page is named as example1.

- After clicking the OK button, a new dialog box "Save Page As" is opened which prompts to enter a filename and specify the location where you want to save the file as shown in figure 1.15.

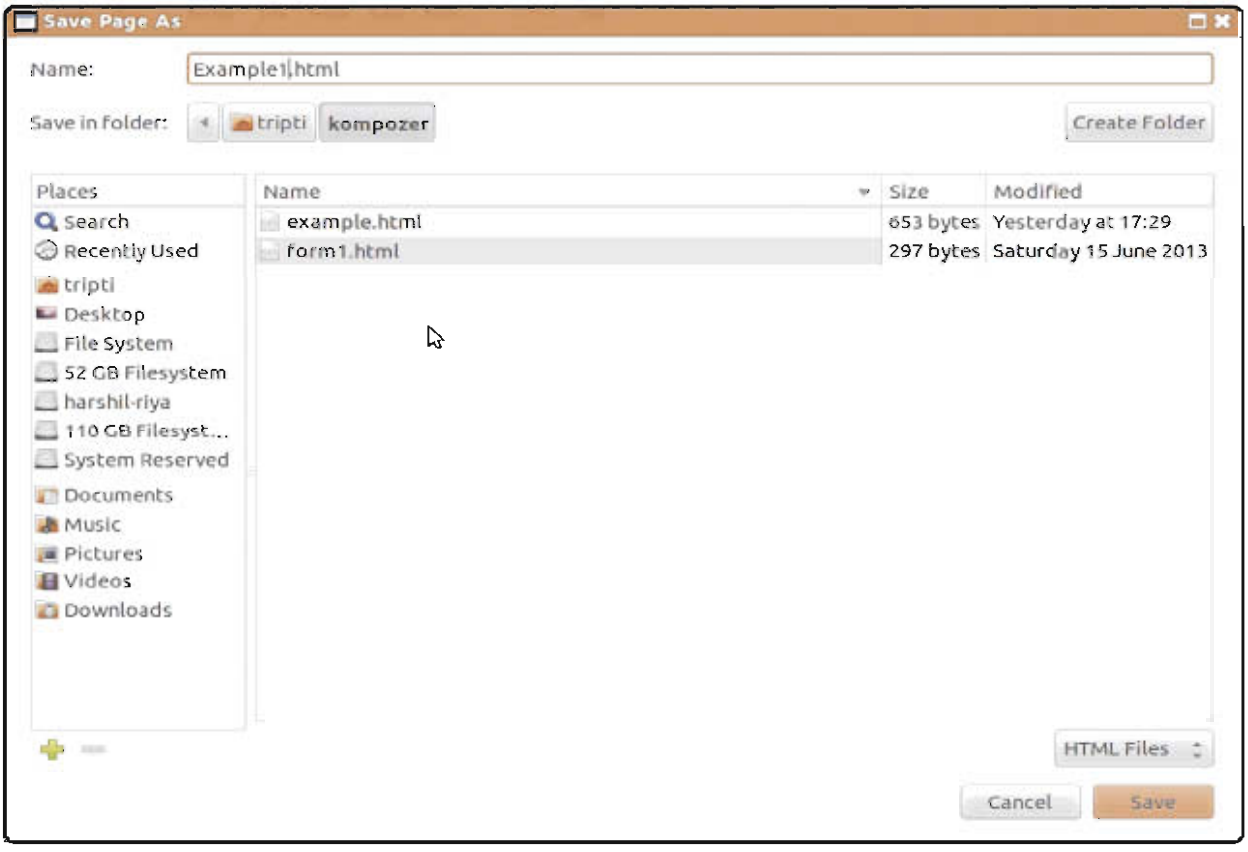


Figure 1.15 : Save Page As Dialog box

Remember to save the file with ***html*** or ***htm*** as an extension. Click on Save button. This will take us back to the main window.

Note : If you are creating a website and this page is the home page that will open when you type the website's URL, then save the page with the name index.html.

Having learnt how to create, open, save and create a simple form in KompoZer, let us now create the registration form, which we had created earlier using the HTML tags. Follow the steps given to create the registration form.

- Create a new file.
- From the menu bar, select **Insert → Form → Define Form.** In the Form Properties dialog box enter the details as shown in figure 1.16.

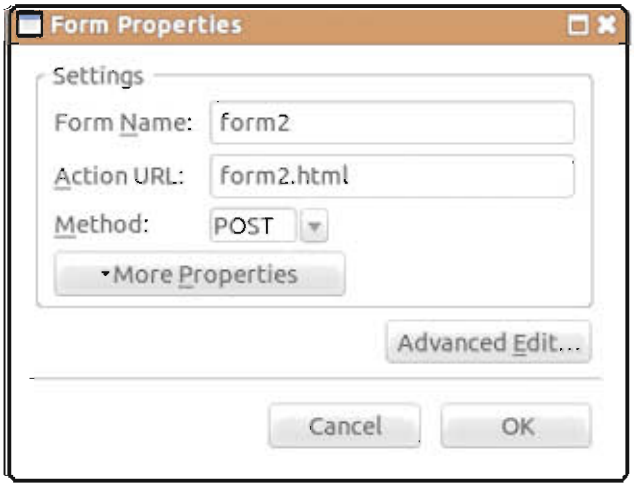


Figure 1.16 : Form Properties dialog box

- Press OK. A form will be displayed showing the light blue colored outline. Press the Enter key to create space in the form.
- To give the heading to the form, select Heading1 from Format Toolbar1. In Format Toolbar2, select centre align icon. Enter the text "Registration Form".
- To insert the label, Click **Form → Define Label.** Type "First Name" in the Field name. Next, to insert input field for the label "First name", click **Form → Form Field.** Select Text in the Field Type menu. Figure 1.17 shows the Form Field Properties dialog box. Observe that we have used "firstname" as the field name. Press OK button.

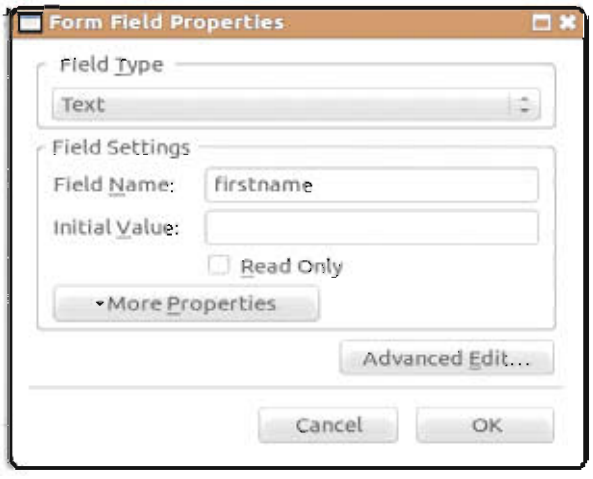


Figure 1.17: Form Field Properties dialog box for Firstname

- Similarly, insert the label "Middle Name" and "Last Name" in the form. The form after adding the fields will look similar to the one shown in figure 1.18.

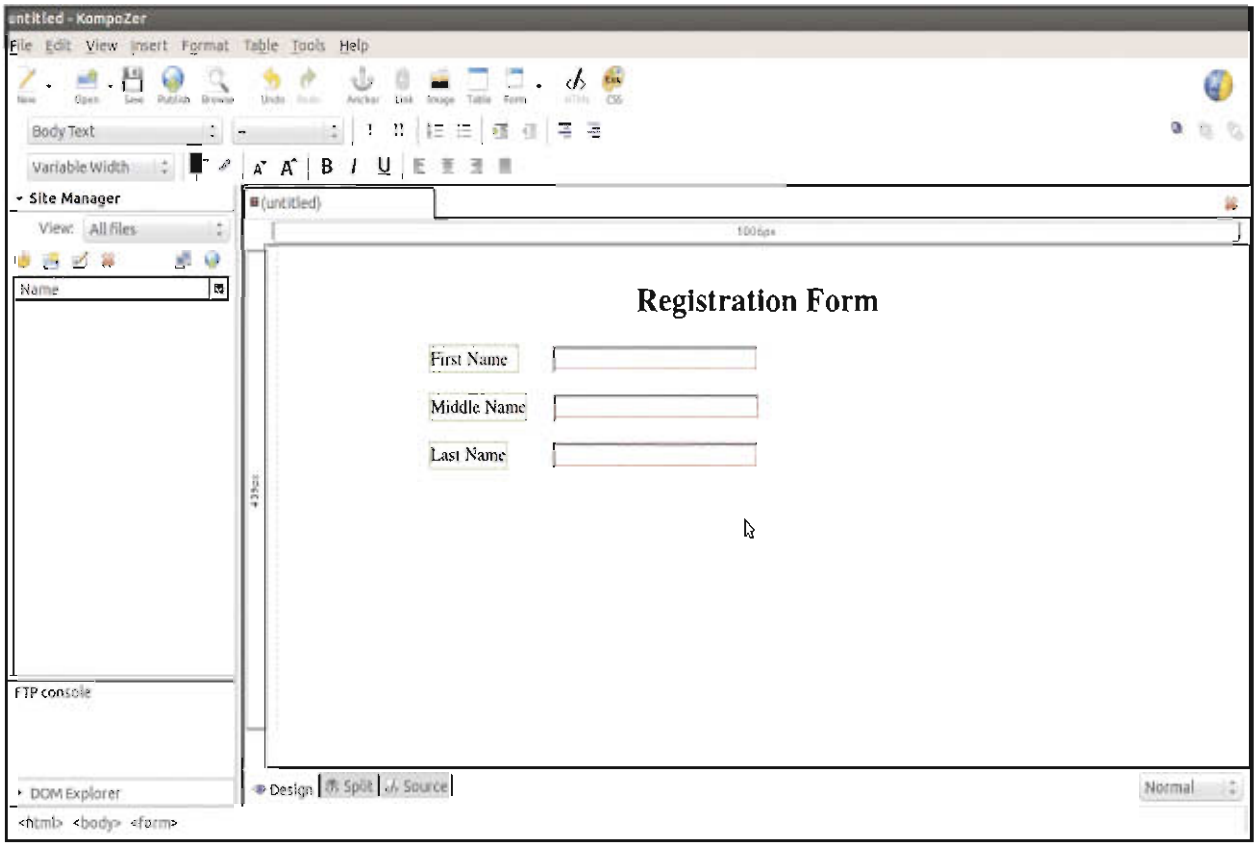


Figure 1.18 : Form displayed after adding the fields

- Now we need to create radio buttons for the Gender field. First, create a label named "Gender".
- For creating a radio button, click **Form → Form Field** and select field type as Radio Button from the drop down menu as shown in figure 1.19.

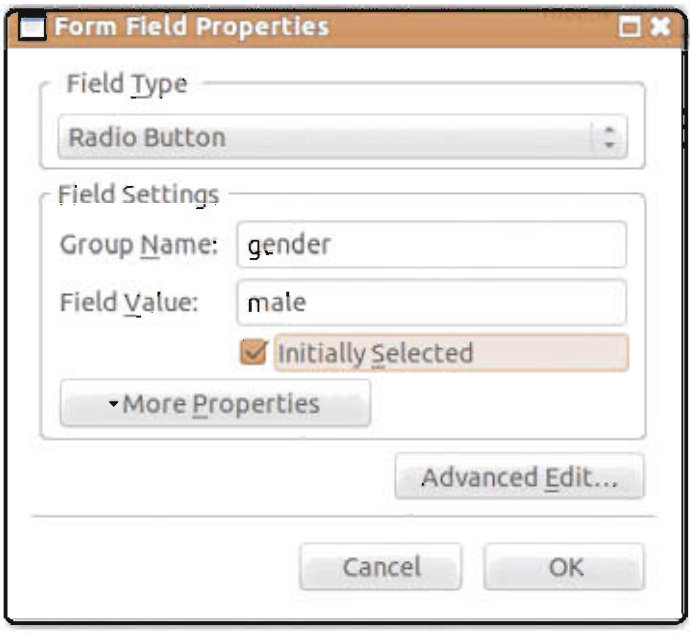


Figure 1.19 : Form field property for Radio Button

Type a name in the Group name box (note that the name should not contain spaces). Here we have entered Group Name as "gender". Similarly in the Field Value text box type "male". If we want the male option of the radio button selected when the form is loaded then, check the box in front of the text "Initially Selected" and click OK button.

- Insert a label with title "Male" near the radio button created.
- Similarly create another radio button named "Female". Remember, when we create the radio buttons within a group, the group name must be the same for all the possible answers. Hence enter the Group Name as "gender". In the Field Value text box type "female". Click OK button.
- Insert a label "Female" near the radio button created. The form after inserting the radio buttons and their labels will look as shown in figure 1.20.

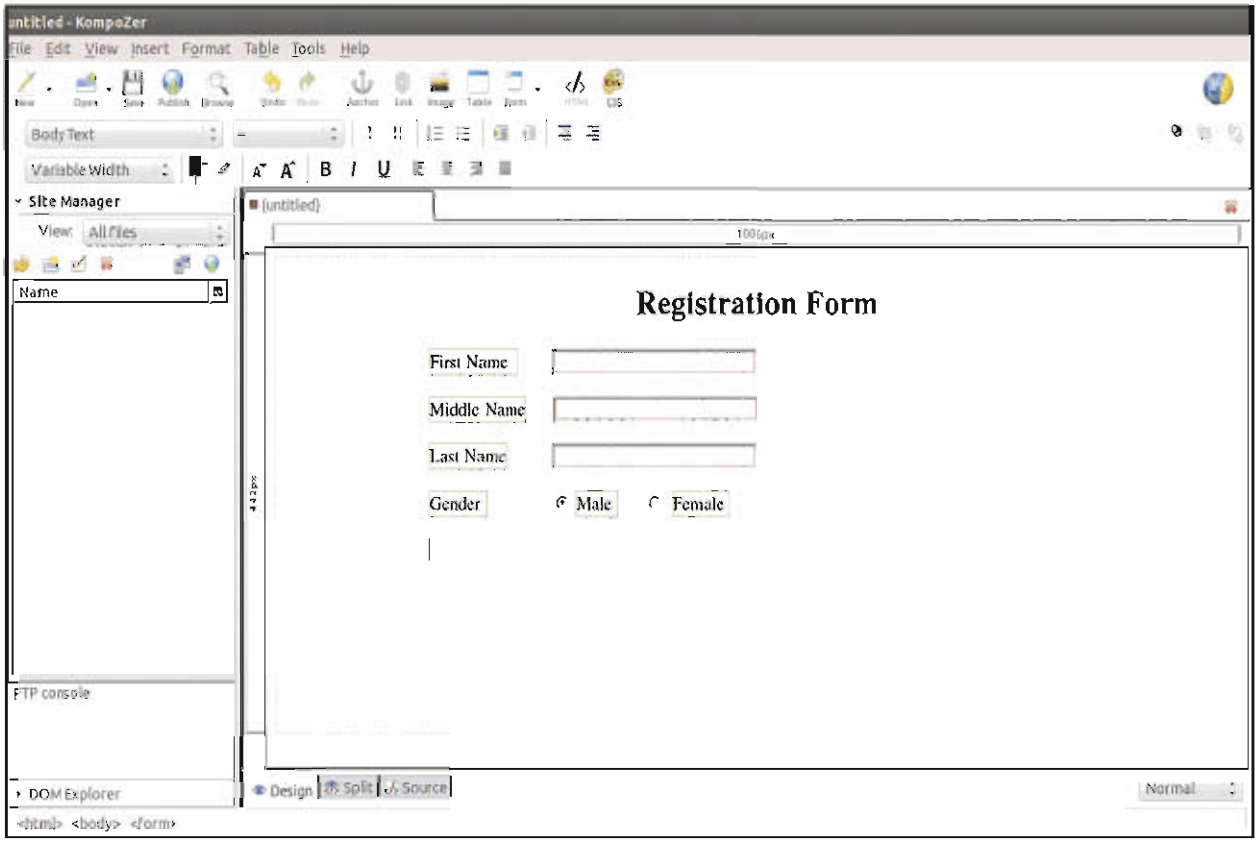


Figure 1.20 : Form displayed after radio buttons are added

- Now we need to add "Hobby" field. As a person can have more than one hobby, hence multiple selections of hobbies are possible. Thus we need to create check box for the hobby field. Create a label for hobby.
- Now, click **Form → Form Field** and from the drop down menu select Check Box field type. Enter name in the Field Name box and a value in the Field Value box as shown in figure 1.21. Check the box in front of "Initially Selected" so as to keep the option checked when the form loads. Click OK button.

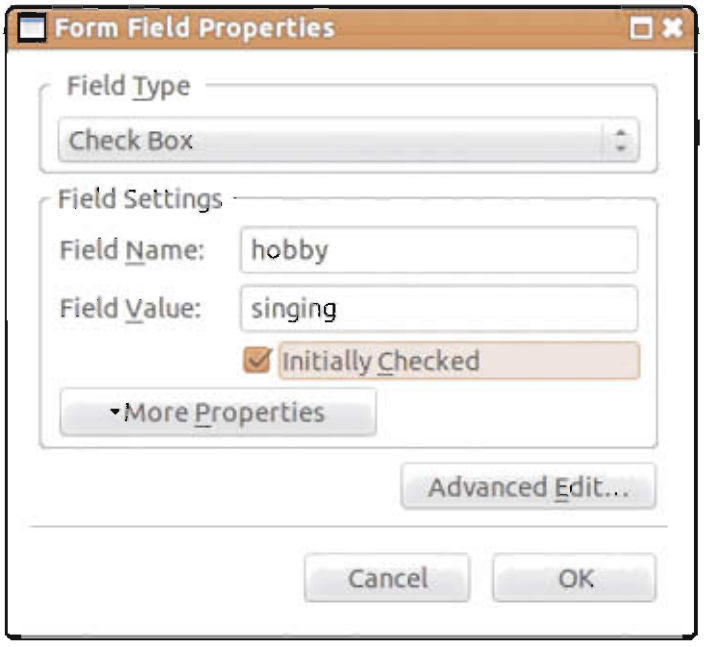


Figure 1.21 : Checkbox field type settings

- Insert a label "Singing" near the checkbox created.
- Similarly create two checkboxes with Field Values as "dancing" and "reading" respectively. Remember, to keep the Field Name same for all the options of checkbox.
- Insert the labels "Dancing" and "Reading" near the checkboxes created. The form after inserting checkboxes and their labels will look as shown in figure 1.22.

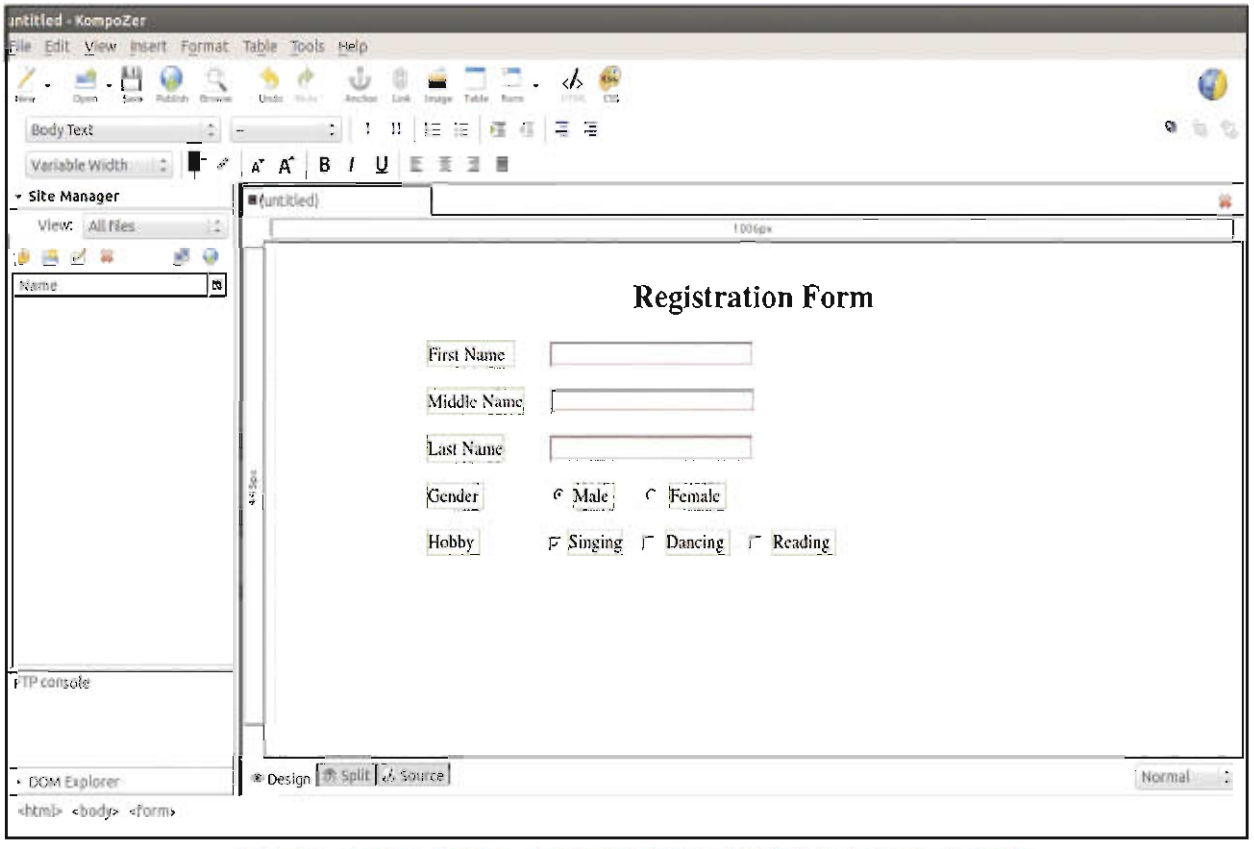


Figure 1.22 : Form display after checkboxes are added

- Next, we need to the "Address" field. As the user enters a large text in the address field, we will keep the field type as textarea. First, create a label named "Address". Now, click **Form → Text Area.** This will open a Text Area Properties dialog box as shown in figure 1.23.

Figure 1.23 : Text Area Properties dialog box

Enter the Field Name. Select the rows and the columns required for the textarea. Here we have kept the rows as 5 and columns as 70. In the Initial Text field enter a suitable text which will be displayed when the form loads. Click OK button.

- Next, we will insert the "City" field. Add a label for the city field. The user will be asked to select the city from the drop down menu. So we need to create the city field using selection list option. Click **Form → Selection List.** This will open a Selection List Properties dialog box as shown in figure 1.24.

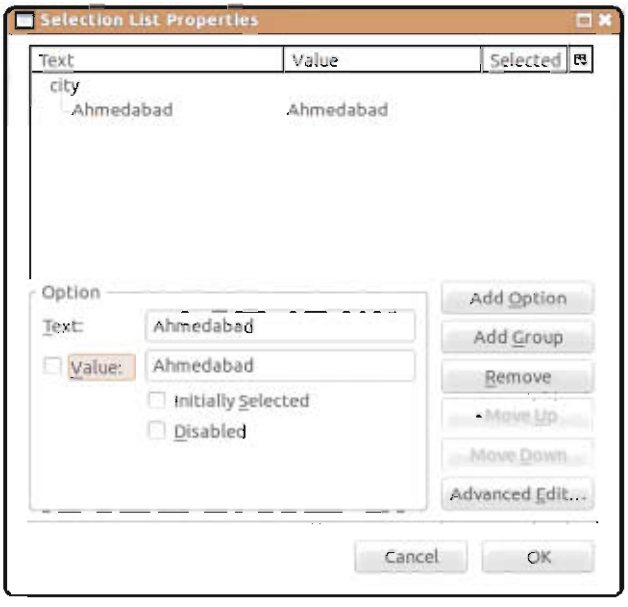


Figure 1.24: Selection List Properties dialog box

Type the name "city" in the List Name box and press Add Option button. Next, type "Ahmedabad" in the Text field. Again press Add option to add the city "Baroda". Likewise, add the city "Rajkot" and "Surat". Remember, to select the option "Initially Selected" when adding the Rajkot city. Click OK button. Figure 1.25 shows the Selection List Properties dialog for the added cities.

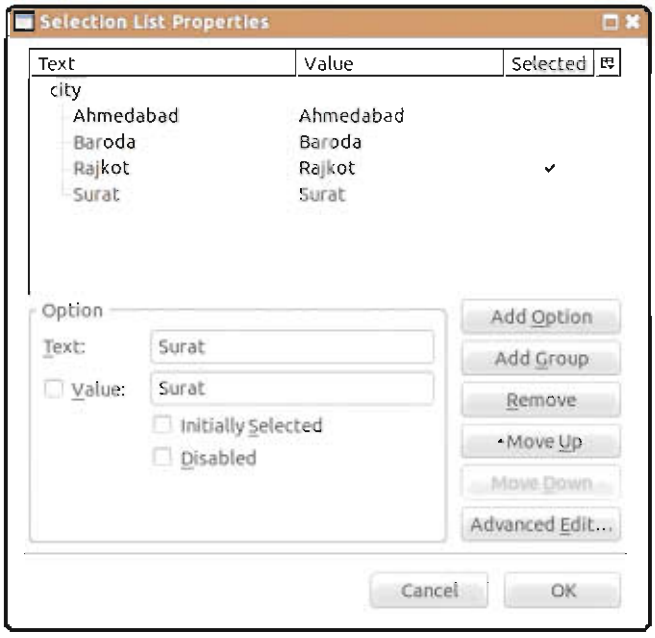


Figure 1.25 : Selection List Properties dialog box for various cities

- Next, we will add the "Submit" button. Click **Form → Form Field.** From the drop down menu select Submit Button. Enter text Submit in Field Name and Field Value text box respectively. Click OK button.
- Similarly we will add the "Reset" button. From the drop down menu select Reset Button. Enter text Reset in Field Name and Field Value text box respectively. Click OK button. The final form in the normal will be as shown in figure 1.26.

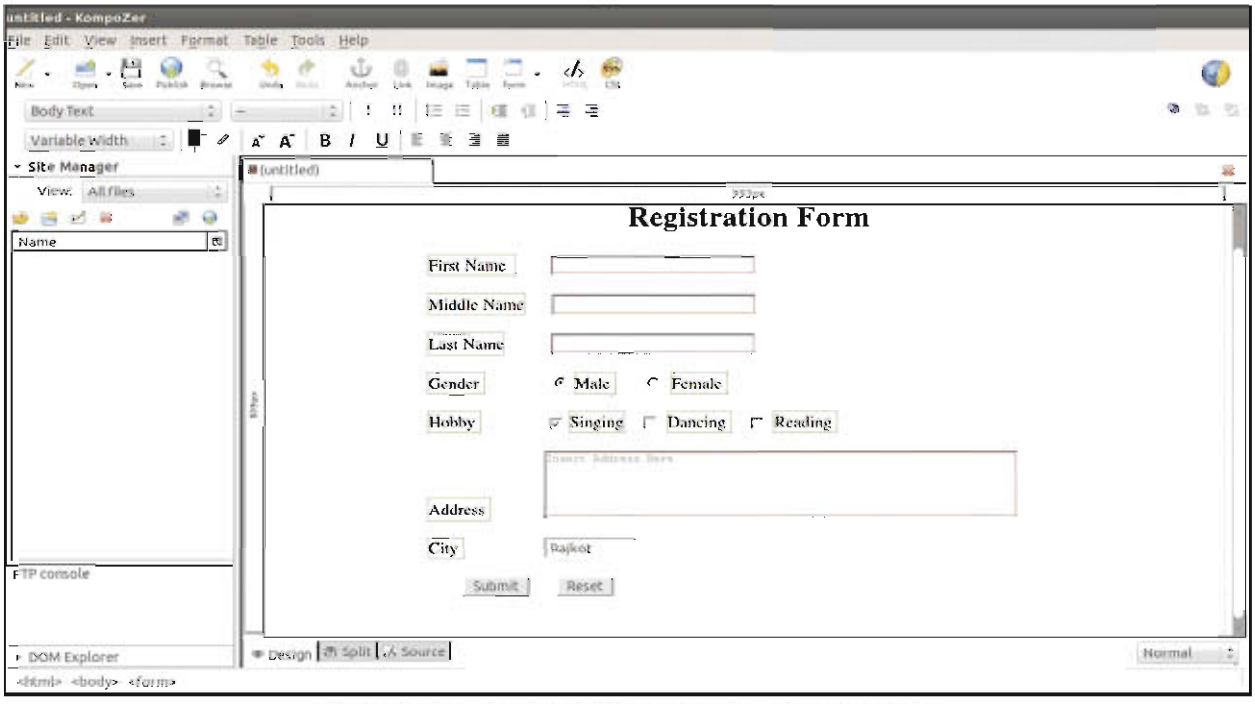


Figure 1.26 : Form displayed in normal view

Save the file with name "example2". Figure 1.27 shows the preview mode of the form.

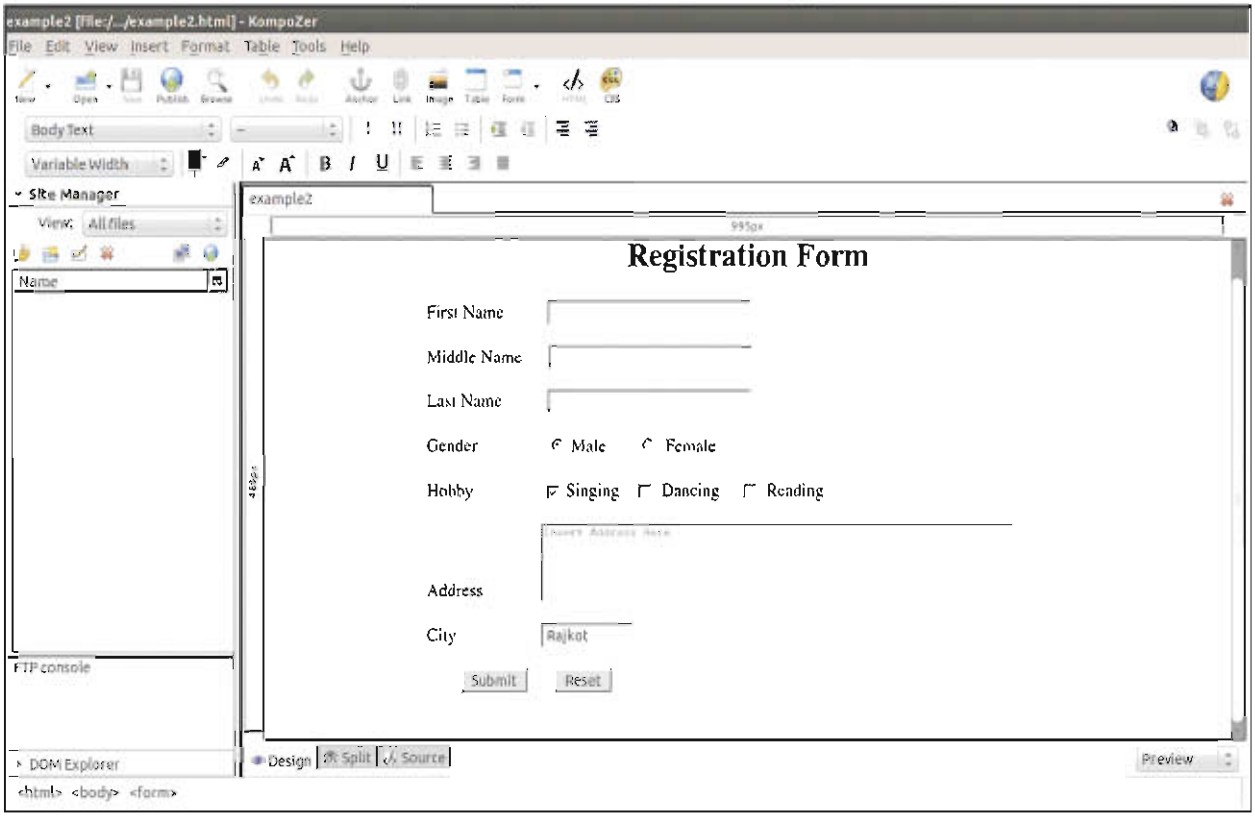


Figure 1.27 : Form displayed in preview mode

Figure 1.28 Shows the registration form as seen in the browser.

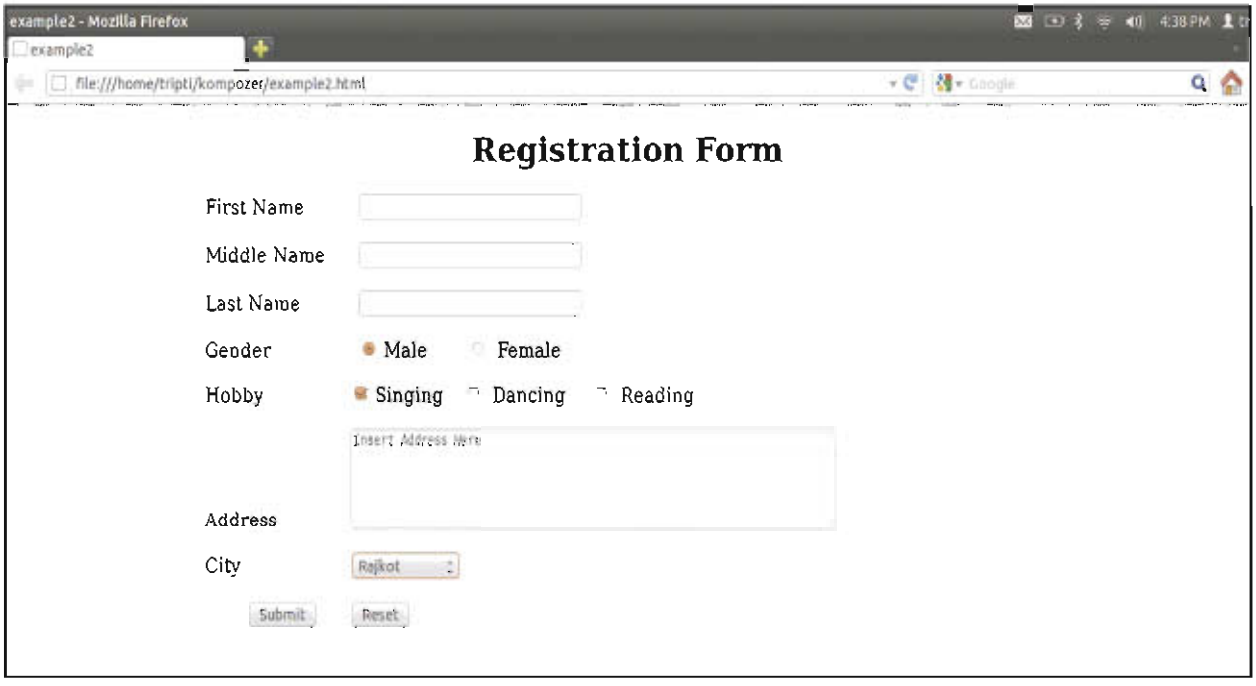


Figure 1.28 : Form displayed in Browser

Observe that the form background is white colored. If you want to give a background color to the form, go to **Format → Page Colors and Background.** This will open a dialog box as shown in figure 1.29.

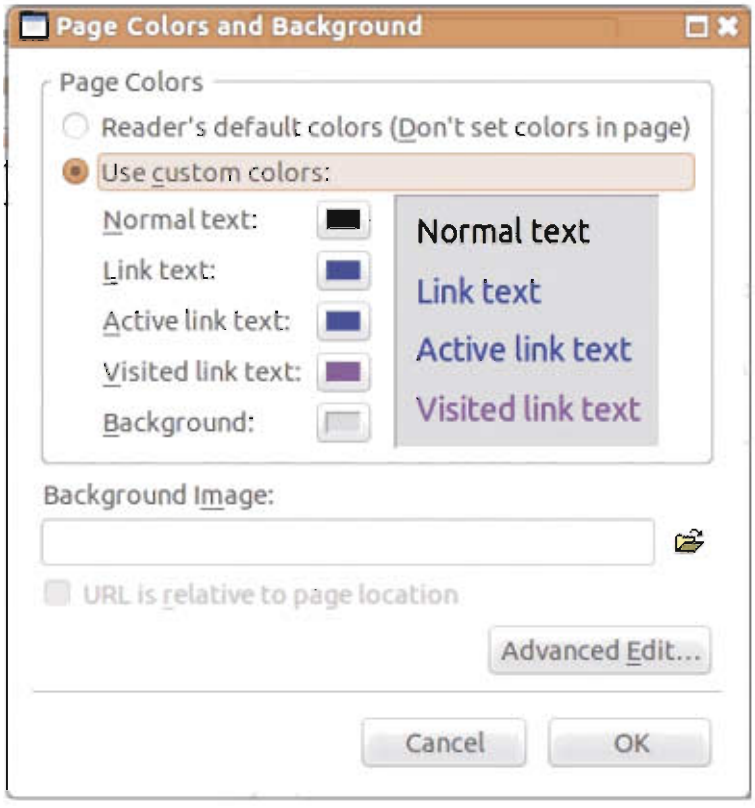


Figure 1.29 : Page Colors and Background dialog box

Select "Use custom colors" option. Click the background option and select the color of your choice from the Block Background Color dialog box as seen in figure 1.30. Click OK button. This will lead you back to the dialog box shown in figure 1.29. Again click OK button.

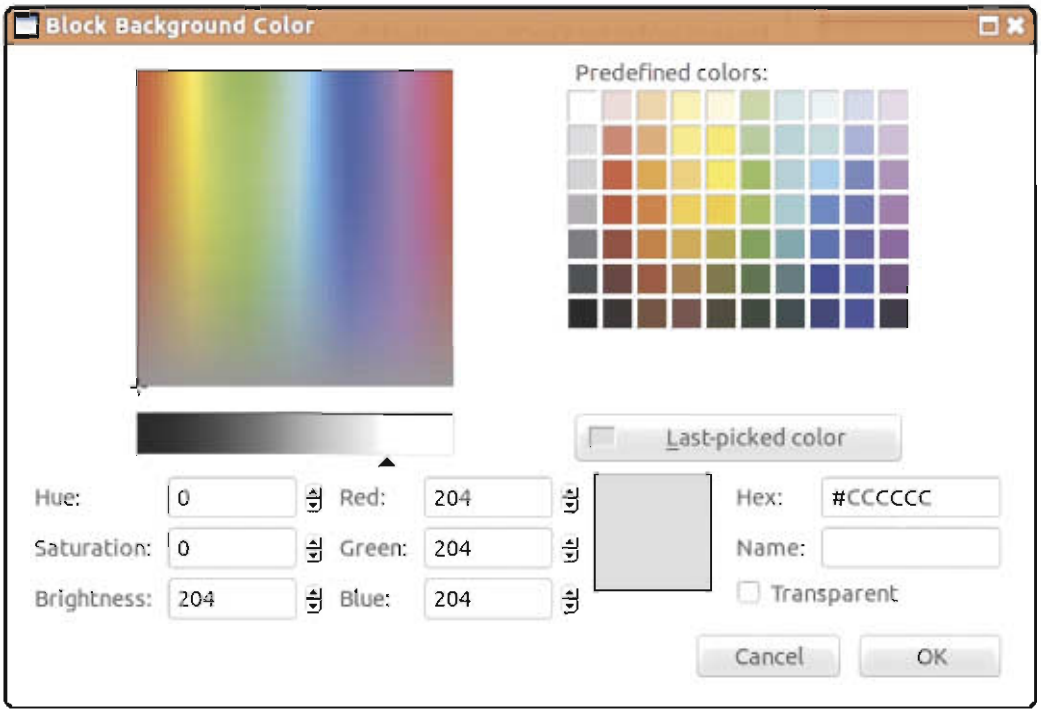


Figure 1.30 : Background color selection

After selecting the color the form will look as shown in figure 1.31.

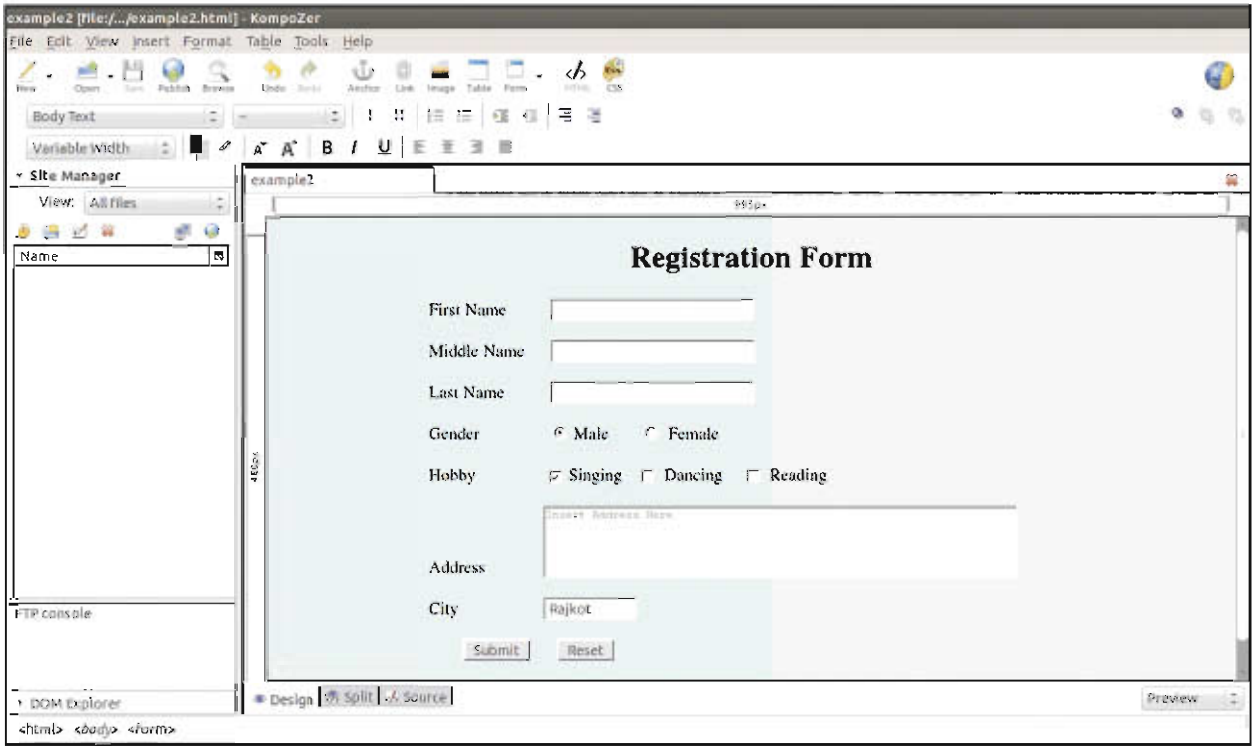


Figure 1.31 : Form displayed in preview mode after adding background color

View the form using the browser and observe the change in the background color.

Summary

Forms are used to accept the data over the web. A form in HTML is a container used to collect different kind of inputs from the user. The users enter the information in the form which can be their personal information, a feedback about a product, a survey or shipping and credit card details. KompoZer is a free open source web development IDE used to create websites. It provides a web page editor which has a simple graphical interface known as WYSIWYG "what you see is what you get". Creating forms using Kompozer is simple and fast.

EXERCISE

1. What is a Form ? List the elements used to create forms in HTML.
2. State the use of Input element in HTML forms. Write about the different attributes of input tag.
3. What is the purpose of textarea element in HTML forms ?
4. Write about select and option element.
5. List the various toolbars seen in the Kompozer interface.
6. Which are the two attributes of form? Explain.

7. Choose the most appropriate option from those given below :

(1) Which of the following is a container used to collect different kinds of inputs from the user.

(a) Form (b) Webpage (c) Text (d) Input

(2) Which of the following element is used to create an HTML form ?

(a) Textarea (b) Form (c) Select and Option (d) Input

(3) Which of the following is the tag used to implement form element ?

(a) <form>… </form> (b) <form>… <form>

(c) </form>… </form> (d) <frm>… </frm>

(4) Which of the following attribute of form is used to specify where to send the form data when the form is submitted ?

(a) method (b) action (c) submit (d) input

(5) Which of the following attribute of form specifies the HTTP method to be used when sending the data ?

(a) submit (b) action (c) method (d) input

(6) Which of the following values are used by method attribute ?

(a) GET and POST (b) GET and SET

(c) GET and PUT (d) SET and POST

(7) Which of the following method allows only a limited amount of information to be sent at a time?

(a) GET (b) POST (c) SET (d) PUT

(8) Which of the following method sends the data as a block through the HTTP transaction ?

(a) GET (b) SET (c) PUT (d) POST

(9) Which of the following attribute of the input element specifies the field that is to be created in the form ?

(a) Input (b) Type (c) Name (d) Value

(10) Which of the following element allows multi-line text input ?

(a) Textarea (b) Input (c) Select and Option (d) Form

(11) Which of the following element is used to create a drop down list or menu in a form ?

(a) Input (b) Textarea (c) Select (d) Form

(13) Which of the following is a free open source web development IDE ?

(a) HTML (b) Kompozer (c) Scite (d) Base

(14) Which of the following stands for "WYSIWYG" ?

(a) When You See Is When You Get (b) What You See Is When You Get

(c) What You See Is What You Get (d) When You See Is What You Get

LABORATORY EXERCISE

1. Create a form for student's personal details.
2. Create a feedback form for the guests who visit your school.
3. You had gone for a vacation with your parents; the tour operator has asked you to give reviews of your trip. Create a form for the same.

Cascading Style Sheets and Java script 2

Cascading Style Sheets or CSS allows you to specify styles for the visual elements of the website. It helps us to keep the information content of a document separate from the details of how to display it. This detail of how to display the document is known as style. Styles specify the appearance of particular page elements on the screen. Keeping the style separate from the content helps us to :

- Avoid duplication in coding
- Use the same content with different styles for different purposes
- Easy maintenance of code

In HTML, for example, if we want contents of some paragraph tag <p> to appear in bold, then we have to insert the bold tag <b> every time the <p> tag appears in the source code. If the website is large, repeating the tag every time becomes tedious and time-consuming. But using CSS, we can set all element types to appear in a style as per our choice. So even if we have more number of <p> tags in the website, using CSS we can set the style for all the <p> tags at once. Thus we can say that, HTML is used to describe the information content of the document and not the style, while CSS describes the style of the elements in the document and not its contents. Using CSS, we can control the font types, font and element colors, pad spaces, margins, and element positions in our website.

Syntax of CSS

The syntax of CSS consists of special symbols known as rules. A CSS rule has two main parts: a selector, and one or more declarations. The selector is the HTML element on which you want to apply the style. Declaration consists of a property associated with the HTML element used in selector and its corresponding value. The general syntax of CSS is defined as:

selector {property : value}

Figure 2.1 gives an example of one such syntax :

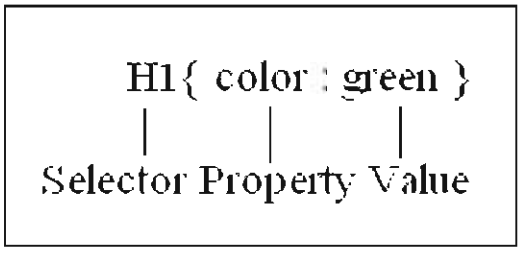


Figure 2.1 : Syntax of CSS

We can easily add CSS using KompoZer. Let us learn how to add CSS to a web page in KompoZer. To make sure that KompoZer uses CSS by default, click **Edit → Preferences.** This will open Options dialog box as shown in figure 2.2. Select the General category on the left side of the window. Select the checkbox "Use CSS styles instead of HTML elements and attributes" if it is not selected. KompoZer will now use the CSS styling instead of HTML to format the text.

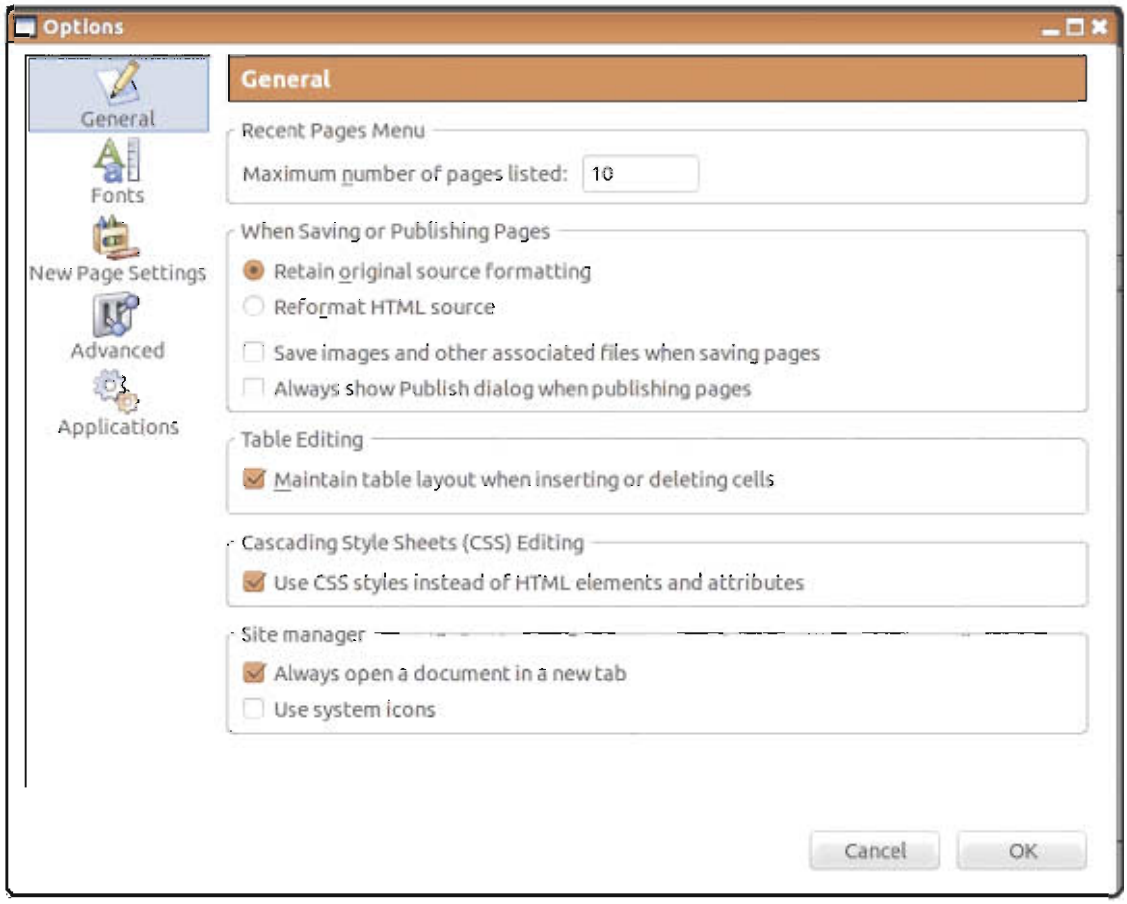


Figure 2.2 : Options dialog box

Assume that we want to design a website and the heading on all the web pages should follow a particular style. For example, all the Headings1 (h1) inserted in the web pages should follow the style as given :

- Font : Times New Roman
- Case : Uppercase
- Alignment : Centre aligned
- Background color : Light Blue
- Border : Dotted Border

Follow the steps given to create the CSS for the above heading.

- Open a new file. Give the Title name and save the file.
- In the composition toolbar, select the Cascade button . (Note: If the file is not saved, the Page title dialog box will be opened. After giving the page title, save the file). This will open the dialog box as shown in figure 2.3. Using this dialog box we will define the styles for each element. Click on the first radio button "style applied to all the elements of type".

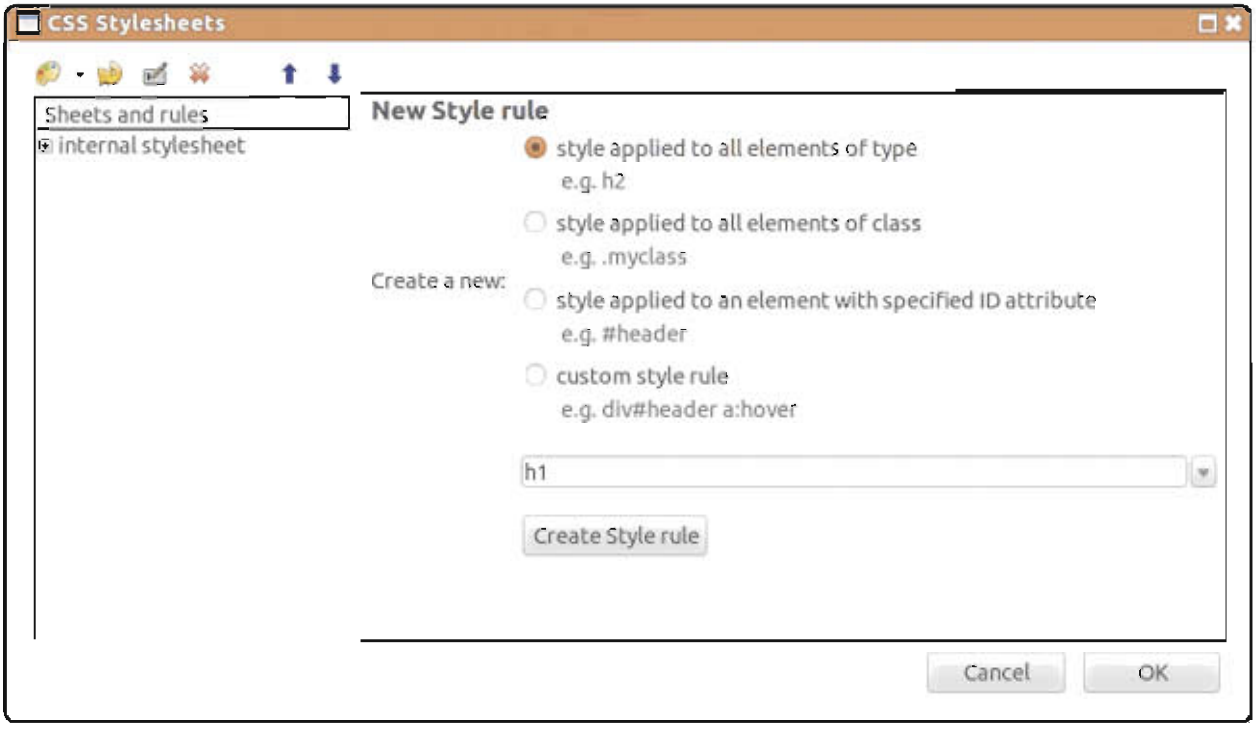


Figure 2.3: CSS Stylesheets dialog box

- From the drop down menu choose an element to create the style rule. As we want to create a style for Heading1, we have selected h1 (Heading 1) from the drop down menu as shown in figure 2.3.
- Click the Create Style rule button. You will see that the CSS Stylesheets dialog box remains open but the options will change.
- Figure 2.4 shows the h1 element just below the heading "internal stylesheet" on the left pane of the window. For each element we add, a style rule will appear in this list below the heading "internal stylesheet". On the right side of the window, you will see various tabs like General, Text, Background, Borders, Box, Lists and Aural. Each of these tabs can be used to give a specific style to the element. But all the tabs may not be applicable to the selector for which the style rule is created. Now, let us create the style rule for h1 selector.

Figure 2.4: CSS stylesheets dialog box with options for h1

- Select h1 in the left pane of the window. Click the Text tab as shown in figure 2.5. This will show various style options like Font family, Font size, Line height, Color, Case, Alignment and many more.

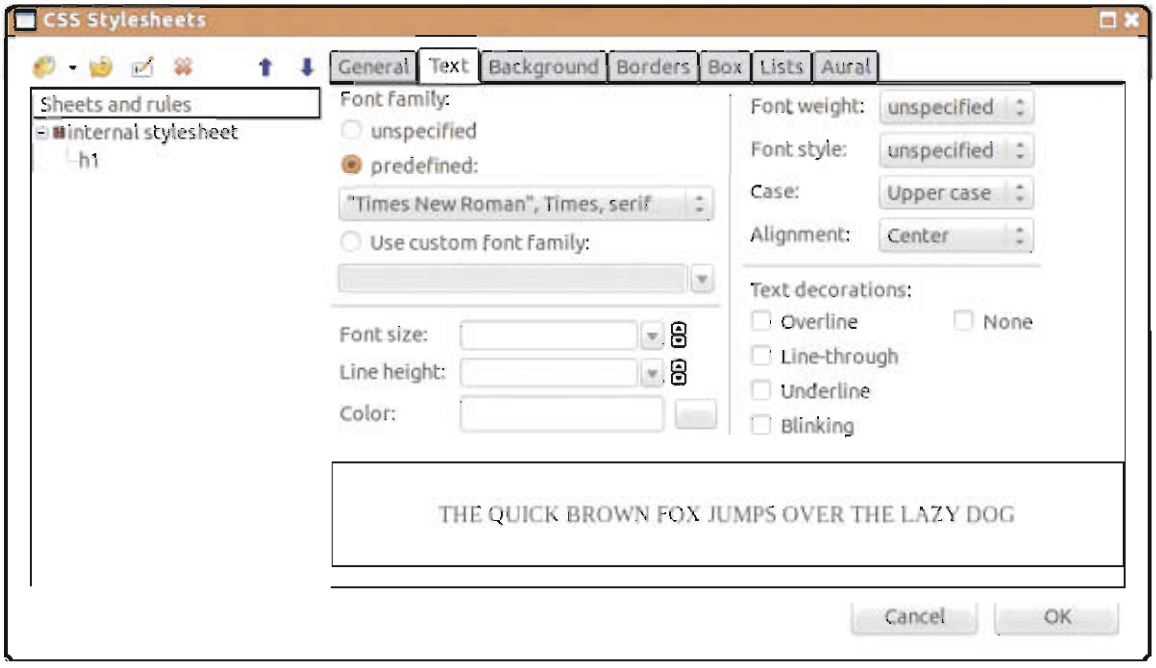


Figure 2.5 : Text tab

- We will now select the options as per the style required for h1. In the Font family section, select the "predefined" radio button and choose "Times New Roman" from the drop down menu. Choose "Uppercase" from the Case menu and "Center" from the Alignment menu. Figure 2.5 shows the options selected. As you change the options, you can see the default text in the window changes automatically according to the styles.
- To set the background color, select the Background tab. This will open the options for the background as shown in figure 2.6.

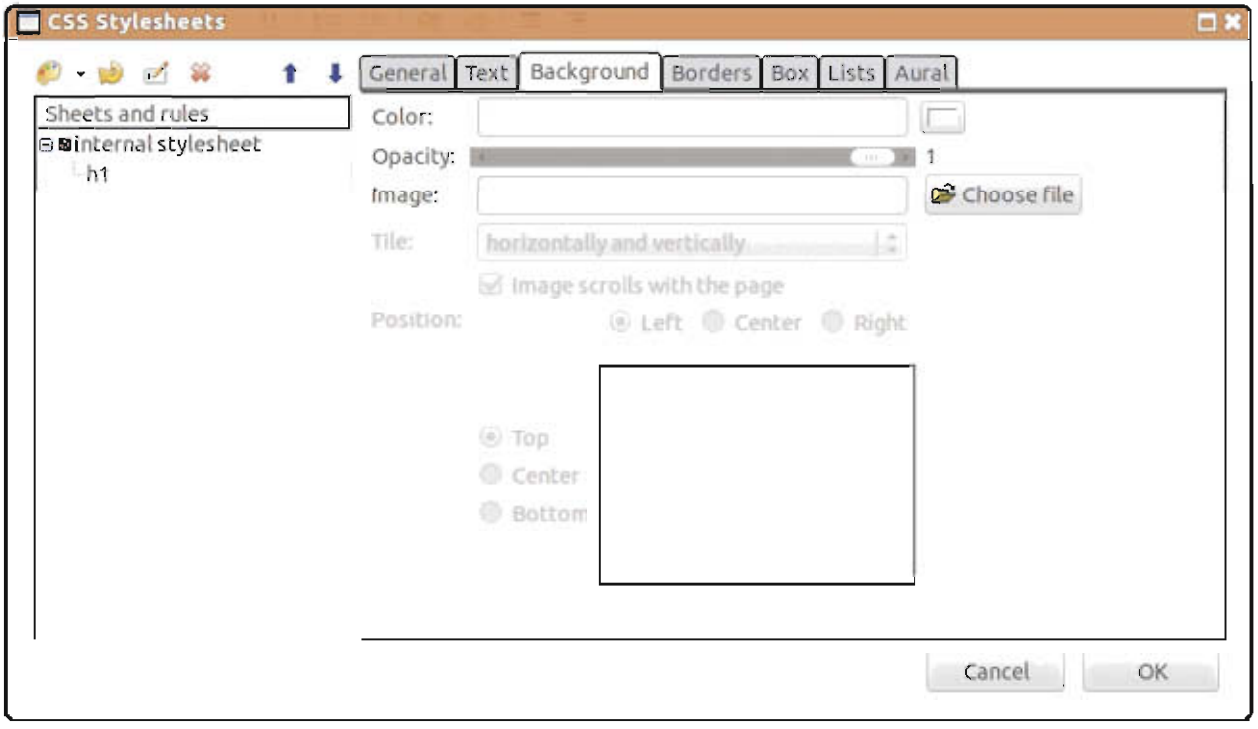


Figure 2.6 : Background tab

- In the Color option, click the color palette button. This opens Block Background Color dialog box as shown in figure 2.7. Select the color of your choice and press OK button.

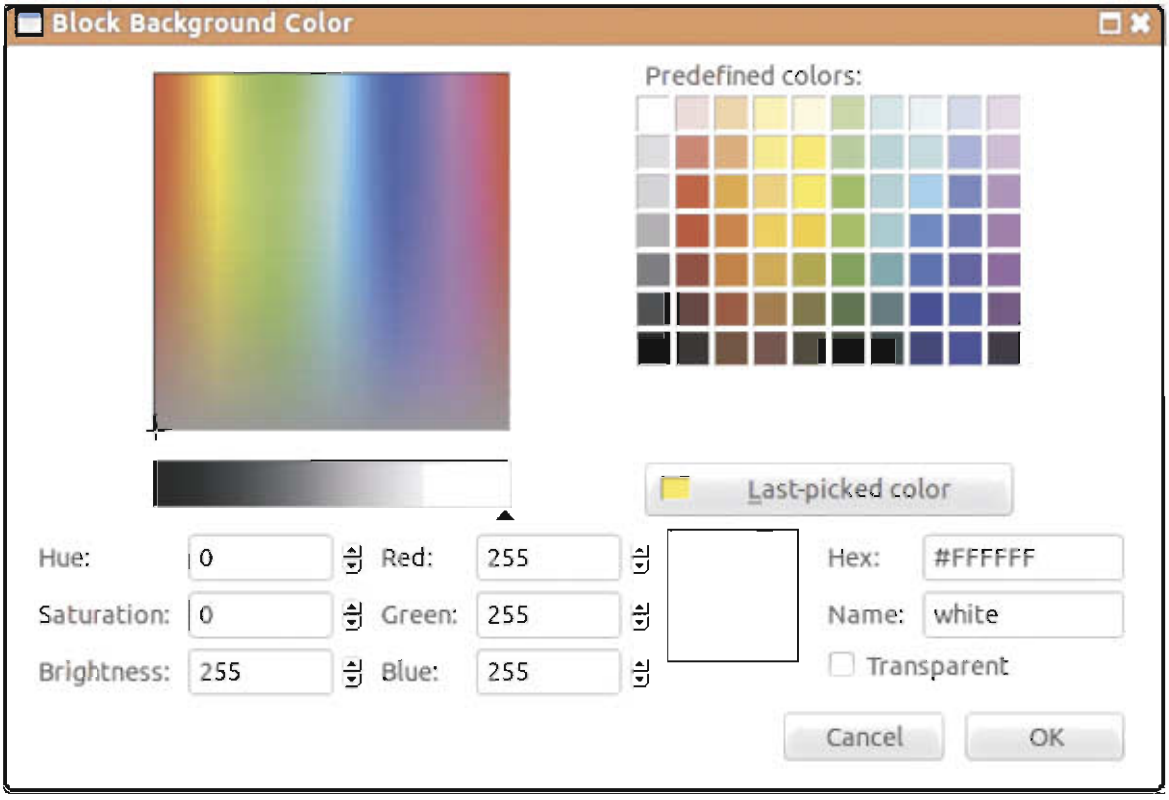


Figure 2.7 : Block Background Color dialog box

- Figure 2.8 shows the background tab after selecting the color.
  Note : If you want to keep an image in the background then in the Image option, click on "Choose file" and select the file for the background.

Figure 2.8 : Options selected in Background tab for h1

- To set the border, select the Border tab. This will open the options for the border as shown in figure 2.9. If you want to apply the same border on all the four sides, tick the checkbox in front of the text "All four sides use same border style". You will see that only one option "Top" will be enabled and all others are disabled. This is because; the effect made on a single side will be applied on all the four sides. From the drop down menu of Style, select dotted. You can also specify the width and the color of the border as per your choice. We can also see the preview of the border style used by us. Press OK button.

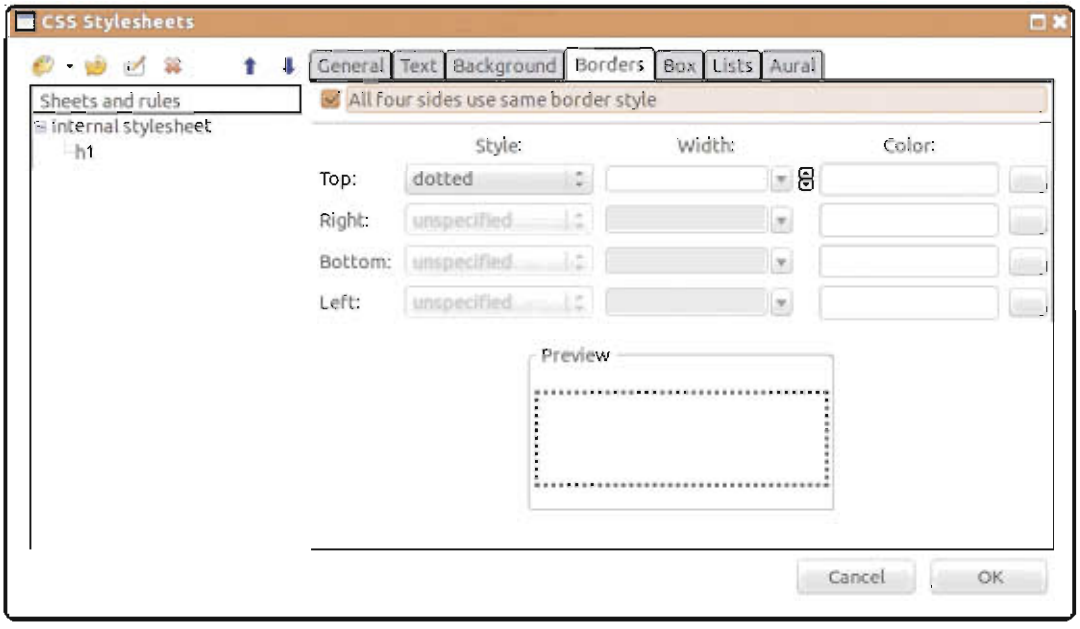


Figure 2.9 : Options selected in Border tab for h1

There are more options available if you select the other tabs in the window. But in our example, we require only the selected options for applying style to h1.

- Click on the General tab. This will show the CSS code which is generated as shown in figure 2.10. If you know HTML coding you can also edit the code.

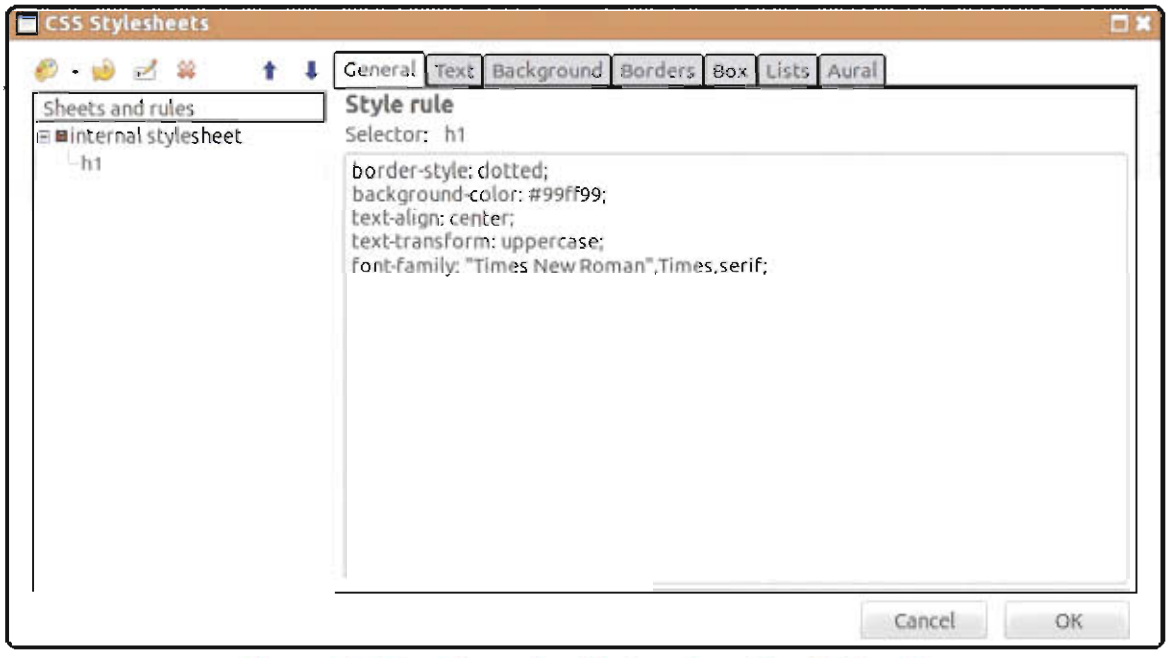


Figure 2.10 : General tab showing the CSS code

Thus, we have created the style to be applied to h1 tag. We can use the same procedure to create a style for any other element in the web page. To add other elements, click on the arrow of CSS button in the upper left corner of the CSS Stylesheets dialog box. Choose "style rule" from the drop down menu. After adding the style rule for all the elements click OK button.

Let us now use the CSS created for h1 tag in a web page. In Format toolbar1, select the Heading1 option to insert a heading in the page. When you select the Heading1 option, your cursor will automatically be placed in the center of the web page (as we had applied center alignment in CSS). Type some text. The text will appear in uppercase with font style "Times new roman". Figure 2.11 shows a web page that uses Heading1 text with CSS applied to it. So, now wherever you insert Heading1 in any of the web pages, the style will remain the same for all.

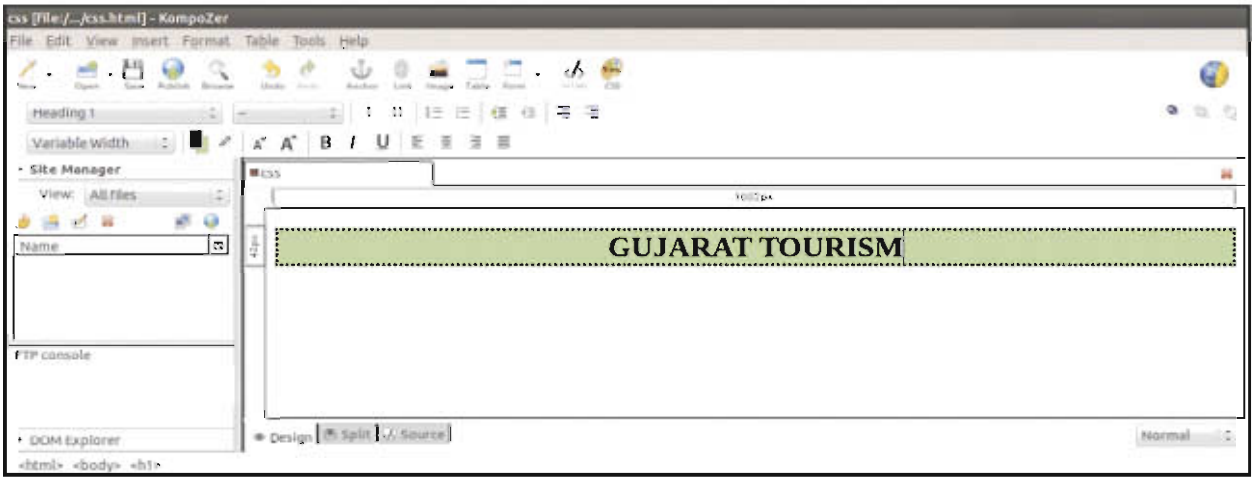


Figure 2.11 : CSS applied to the text in web page

Select the source tab at the bottom of the window to view the CSS stylesheet code. Figure 2.12 shows the CSS code in the head section of the page source code.

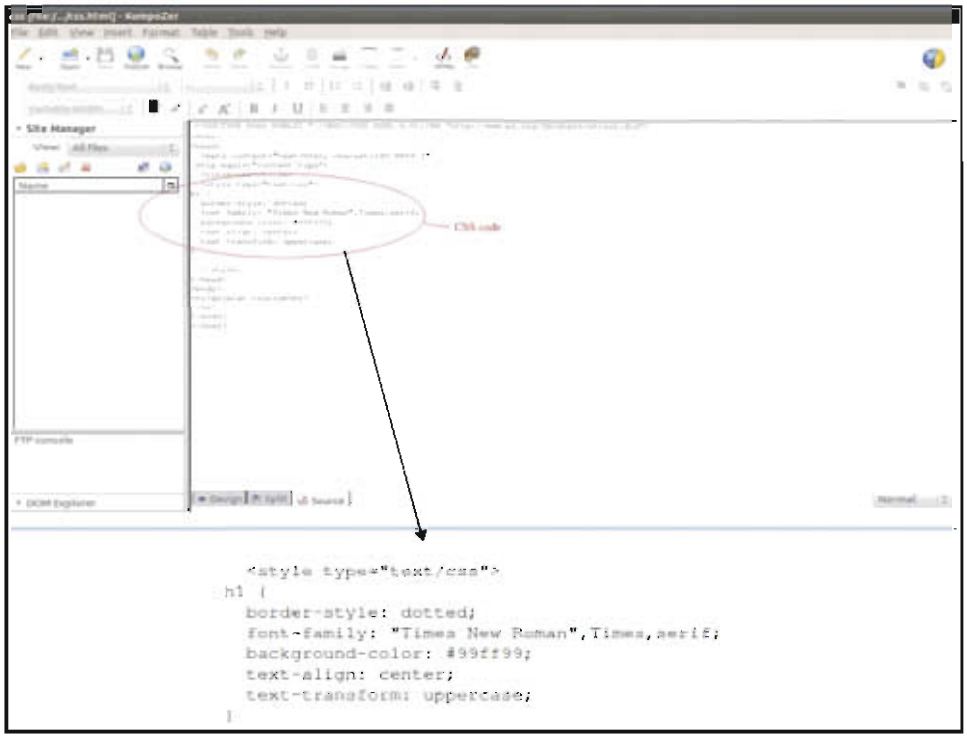


Figure 2.12 : CSS code in the Source view

Advantages of CSS

From the above discussion we can say that CSS allows us to set the format of the website without changing the underlying structure. By separating the layout and format properties of the website from its underlying logical structure, we can make changes to the website without the fear of accidently changing the data.

The web designer's job is made easier as to change the style of an element he only needs to make modifications in the CSS file. Since we have to set the style for each element only once, the CSS has less code compared to that in HTML. Thus the web pages will load faster. Using CSS makes website designing quick and efficient.

Disadvantages of CSS

The CSS compatibility varies with different browsers. This means that some of the style sheet features are not supported by the browser and the style cannot be displayed as per the users design. But now-a-days, the latest browser versions are more standard-compliant and the compatibility issue has reduced.

JavaScript

HTML was originally used to control the appearance of web pages. The web pages designed using HTML, were static and could not be changed after the browser rendered them. However, with the growth of Internet, people demanded the websites to have greater interactivity and better visual design. But, HTML could only provide static web pages. Thus, the demand of more interactivity created the need for a new web programming language. Hence, Netscape developed JavaScript. JavaScript is a scripting language that allows you to add programming aspects to your web pages.

A scripting language is a simple, lightweight programming language that does not contain the advanced programming functionalities of languages like C and Java. JavaScript is used in web pages to improve the design and validate the forms. It adds interactivity to HTML pages and is inserted directly into the HTML code. Today, mostly all the web browsers like Mozilla Firefox, Chrome, Safari and Internet Explorer support JavaScript. Using JavaScript, a web page no longer remains static, but can include coding that allows interactivity with the user, control the browser, and dynamically create HTML content.

We learnt how to create forms using KompoZer. When the user enters data in the form, he might leave some of the important fields empty or may enter the data in a wrong format inside the field. In such cases there must be some type of validation provided. This validation will restrict users from making mistakes. When the user enters wrong data or leaves a field empty, an error message should be generated and the form should not be submitted.

The most common form of JavaScript application today is client side script which runs inside a web browser. It is used to validate the data entered in the HTML forms on the client side before sending it to the server. Using JavaScript, the form is generally checked for the following things:

- Has the user left any required field empty ?
- Has the user entered a valid E-mail address ?

- Check whether contents of two fields are same or not ?
- Has the user entered a valid date ?
- Has the user entered text in the numeric data field ? Say for example; in the quantity field instead of numeric data, user entered text.

Before starting to learn about how to validate a form, let us first learn how to write a JavaScript code.

JavaScript code runs from within the HTML web page. Thus, the JavaScript code can be put directly inside the web page code as a separate section. The JavaScript code is inserted into an HTML page, using the <script>… </script> tag. The lines between the <script> and </script> tag contains the JavaScript code. Remember, JavaScript is a case sensitive language.

JavaScript can be placed inside the <body> or <head> section of an HTML page. But generally it is preferred that the code be placed in the <head> tag. The <script> tag tells the browser to interpret all the text between the <script> tag as a script. Let us write a simple JavaScript code using the source view in KompoZer. Figure 2.13 shows the source view with the JavaScript code. Save the file and open it in the browser.

```
 1. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
 2. <html>
 3. <head>
 4.   <meta content="text/html; charset=ISO-8859-1"
 5.  http-equiv="content-type">
 6.   <title>js1</title>
 7.   <script>
 8. document.write("HELLO STUDENTS");
 9.   </script>
10. </head>
11. <body>
12. <br>
13. </body>
14. </html>
```

Figure 2.13 : JavaScript in source view

You can see that inside the <script> tag we have written a single statement "document.write("HELLO STUDENTS")". Here write is a method which is a part of the document object. The string to be displayed is passed as variable to the method document.write().

The semicolon seen in figure 2.13 at the end of the statement is optional in JavaScript. The semicolon is used as a mark of separation rather than termination. Thus, if you place each statement on a new line then you need not put a semicolon. In our example, even if we remove the semicolon after the document.write statement, the result would be same. However, putting a semicolon is a good programming practice. The output of the code as seen in the browser is shown in figure 2.14.

Figure 2.14 : Output in Browser

Note : JavaScript statements can be grouped together in a block. A block starts and ends with the curly bracket.

Let us now learn how to write a simple JavaScript for form validation. First, we will create a simple form with two fields: a name field and submit button. For form validation, if the user leaves the name field empty and clicks the submit button then a message should be displayed that asks the user to enter the name. Follow the steps given to perform the operation.

- Create a new file and save it. In the example we have saved the file with name js1.html.
- Create a new form using **Form → Define Form.** Give a name to the form. In our example, we have named the form as js1. As shown in figure 2.15, insert a label and input text field for the displaying name, also insert a Submit button.

  Note : The names given to the form, input field or any other fields created in the form should be kept simple and easy to remember, as they will be referenced in the script frequently.

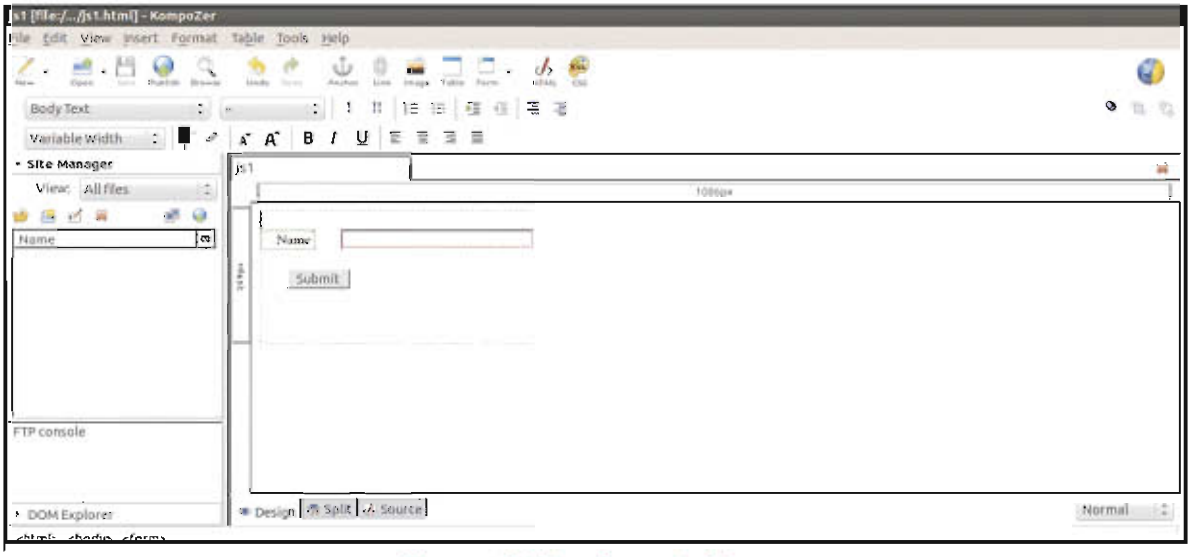


Figure 2.15 : Sample Form

- Go to **Form → Define Form.** This will open the Form Properties dialog box as shown in figure 2.16.

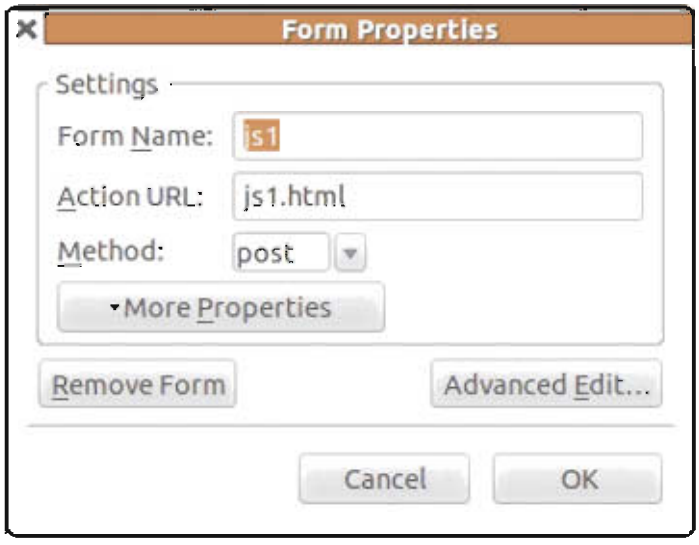


Figure 2.16 : Form Properties dialog box

- Click on Advanced Edit button. This will open Advanced Property Editor dialog box as shown in figure 2.17. Select the JavaScript Events tab. In the Attribute option select "onsubmit" and in the Value write "return validateForm()" as shown in figure 2.17. Press OK button.

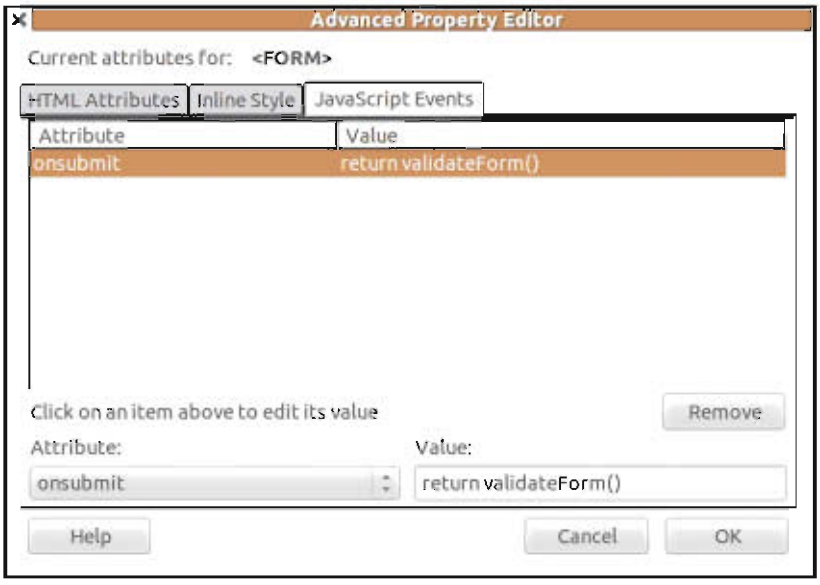


Figure 2.17 : Advanced Property Editor dialog box

- Now, open the source view. Add the script inside the <head> tag as shown in figure 2.18.

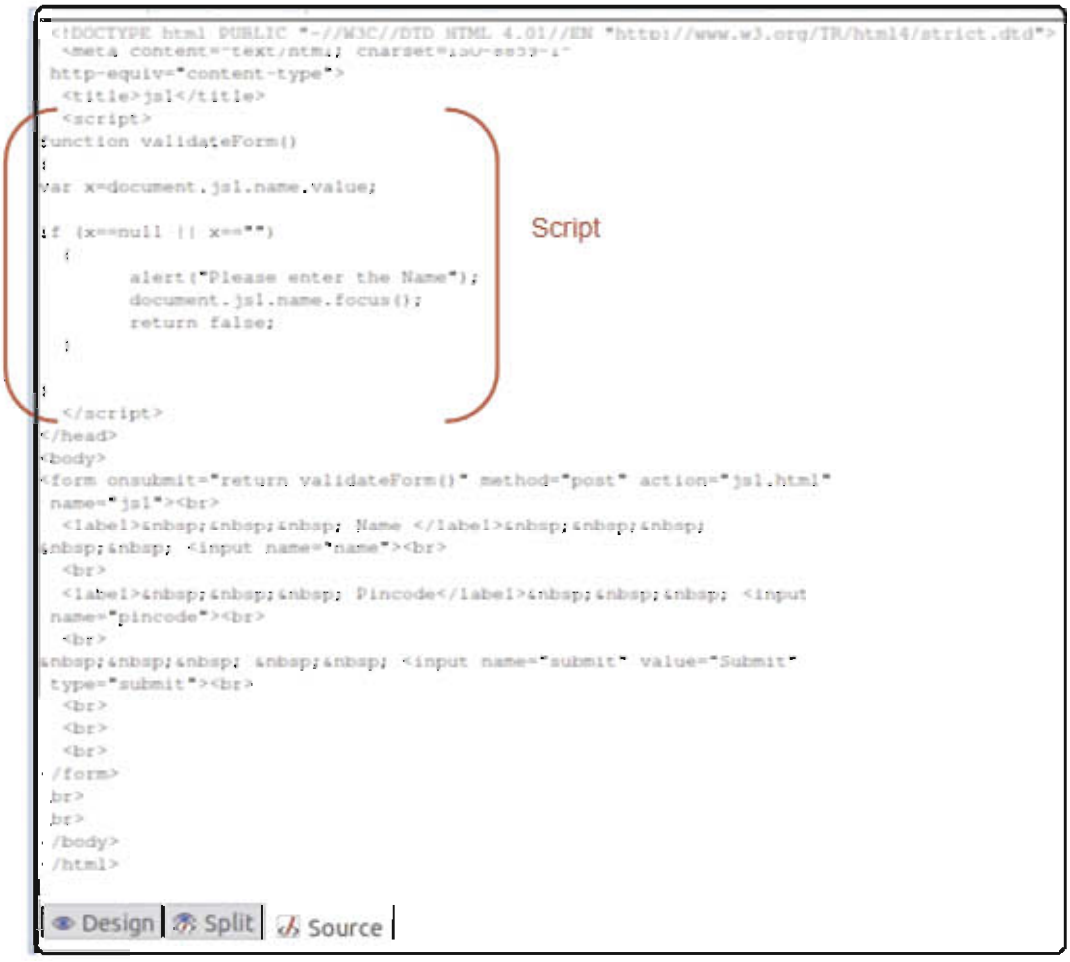


Figure 2.18 : JavaScript in source view

As seen in the script tag of figure 2.18, we have defined :

- a function validateForm()
- a variable x
- a conditional statement if
- built-in function alert()
- an object named document
- a method named focus

Let us discuss each of them in brief.

Function validateForm()

A function (also called routine) is a reusable block of code that performs a particular task. A function is defined by the keyword function and the block of code is written inside the curly braces "{ }". Here, validateForm() is a function.

JavaScript provides some inbuilt functions such as alert(). The alert() function takes the text and displays it in an alert box.

For example,

```
function hello()
{
 alert("Hello Students");
}
```

Here the function "hello()" when called, will display an alert box with the message "Hello Students".

The function will be executed by an event or when it is called inside a source code. A function can be called anywhere from within the source code. We can define the function in the head or the body section of the HTML code. A function is called by its name. It can also take a value inside the parenthesis. This value is passed as a parameter to the function. Sometimes you want the function to return a value. This is possible by using the return statement. The use of return statement, will stop the function from executing, and return the specified value.

In our example, the function validateForm() is called on the click event of the submit button. Thus, when the user clicks the submit button, which is known as an event, the function is loaded and the statements inside the function will be executed. The function returns a false value.

Events

JavaScript is useful in creating interactive web pages which responds to the action performed by the user. The interaction between the user and the web page causes the browser to generate an event. In other words, when the user does something an event takes place. Table 2.1 gives a list of JavaScript events.

Event	Description
abort	Loading of image is cancelled
blur	Element such as a radio button becomes inactive
click	User clicks on a form element
change	Value of a form field is changed by the user
error	Error occurs during loading of a document or image
focus	Element such as button becomes active
load	Document or image is loaded
mouseout	Mouse moves off the element
mouseover	Mouse moves over the element
reset	Form fields are reset to default values
select	User selects a form field
submit	User submits a form
unload	User leaves a page

Table 2.1 : JavaScript Events

When an event occurs, a specific JavaScript code is executed in response to a given situation. This JavaScript code is known as event handler. Event handler names are kept the same as the name of the event. For example, the click event will have an event handler as onclick(). Likewise, submit event will have an event handler as onsubmit().

In our example, we want the validation to be done when the user clicks the submit button i.e. on the "submit" event. So, we need to use the onsubmit() event handler. Therefore, in figure 2.18 we had added an event handler onsubmit(). This event handler will call the function validateForm(). The function will return a value either True or False depending on whether the name field has a value or null (no value).

Variable

In our example, inside the function validateForm() we have a statement

```
var x=document.js1.name.value
```

We have declared a variable named x. A variable is a container for storing data. Remember, that the variables in JavaScript are also case sensitive. So, a variable "x" is not the same as variable "X". The variables can store numbers, strings or text. In JavaScript, we can declare a variable using var keyword. For example,

```
var x=3;
var y="Hello";
var z="Hello Students";
```

If statement

You must be familiar with conditional statement if. The conditional statement if is used to change the flow of the program. The if statement evaluates an expression to validate specific condition. If the condition is true, then the program enters if block and executes the statements inside it. In our example, we have used the if statement to check whether the variable x is having null value or not. If the variable x is having null value then the statements inside the if block will be executed.

Document Object Model (document.js1.name.value)

Sometimes, you may want to use JavaScript to control the web browser. For example, you may want to change the web page or control the elements of the web page. To control the web browsers window or the web page we use the Browser Object Model (BOM).

All browsers are split into different parts or objects that can be accessed using JavaScript. These parts are known as the Browser Object Model, or BOM. On the top of this browser hierarchy is the Window object as shown in figure 2.19.

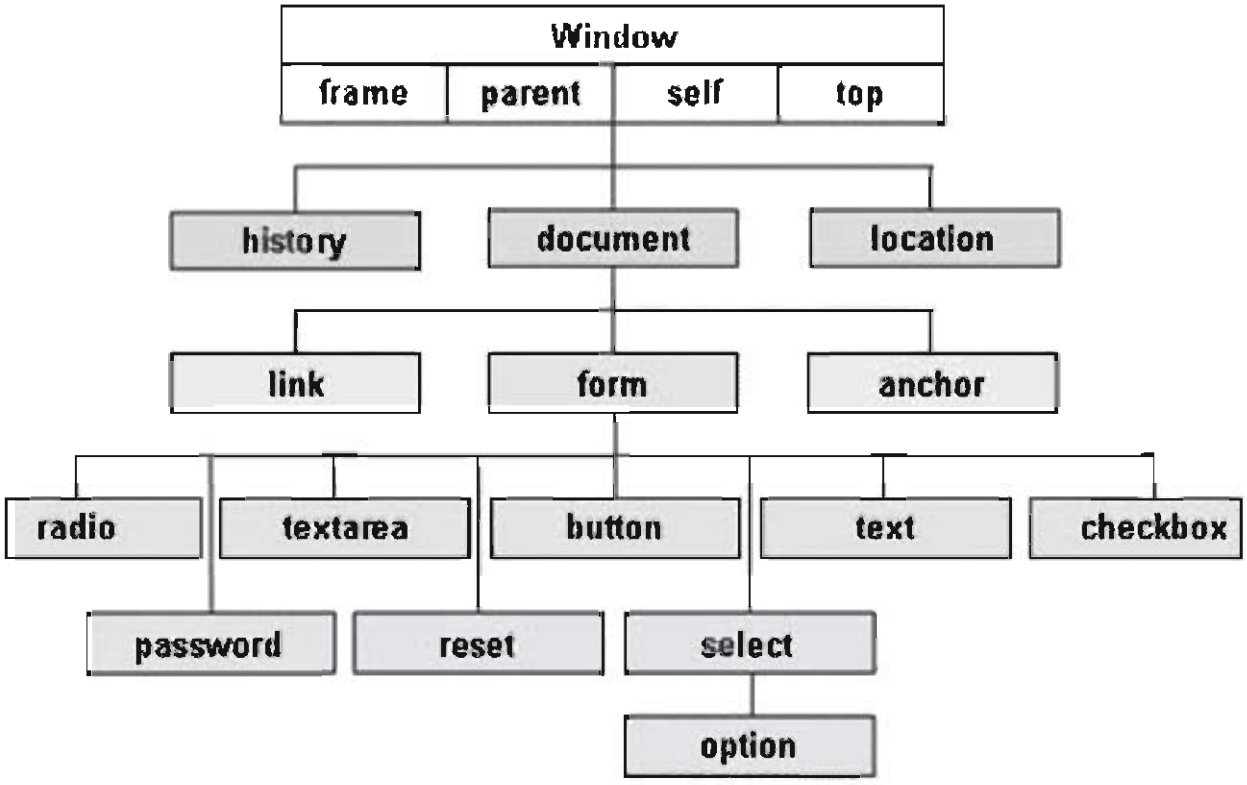


Figure 2.19 : Hierarchy of browser object

It represents the hierarchy of entire browser along with the toolbar, menu, status bar, web page and many more. We can use the properties and the methods of objects in the browser object model to change the window or elements displayed in the browser. We do not have to create an object in a browser object model; they are created automatically when the browser opens the web page.

The top level object in the browser object model is Window object. Window object represents the browser window or individual frame within the window. It is created automatically by the browser. The window object is the global object as it contains all the other objects of the browser object model within it. For example, window object contains the document object. We use the methods and properties of the window object to control the web browser window, while the methods and properties of document object are used to control the web page.

The document object is the most important object in the browser object model. It is used to represent the web page displayed in the browser. All the elements of the web page like forms, images, links and others are contained within the document object. For example, the form object which is used by JavaScript to represent forms created using the <form> element is within the document object. Just as the document object contains a form object, similarly the form object contains element object. The element object is used to reference each element in a form. The element objects can be radio, text, checkbox or any other object.

The focus Method

In the script, we have also used a method named focus. It is used to apply focus on a particular element of the form. Thus, after the user gets an alert message, calling the focus method, the cursor will be placed inside the element and the particular element will be highlighted. Here the name element will be focussed.

Let us now open the file in the browser. The output as shown in figure 2.20 will be visible to us.

Figure 2.20 : Output in Browser

Keep the Name field empty and press the Submit button. You will see an alert message as shown in figure 2.21 appear on the screen.

Figure 2.21 : Alert message for the empty name field

The alert message is the result of the script that we had written in the source view. This is the

validation that we have provided in the form using JavaScript. Thus, a form having more fields can be validated to check if the user has kept any of the fields empty.

You must have observed while filling forms on Internet, some fields are marked with a red color asterisk mark "*". This is an indication to the user that it is a compulsory field which should not be kept empty. Using JavaScript, we can check the entry in such fields and provide the alert message to the user.

Let us now add another field in the form named "Pincode". The validations which we will apply on this field are :

- User should not keep the field empty.
- Only numbers are allowed (characters are not allowed).
- The Pincode should be 6 digits long.

If any of the validation is violated, the user will be displaced an alert message. Follow the steps given to create such a validation.

- In the same form insert a label and a input field for Pincode.
- In the source view, add the script as shown in figure 2.22.

```
<script>
function validateForm()
{
 var x=document.js1.name.value;
 var y=document.js1.pincode.value;
 if (x==null || x==“”)
 {
  alert(“Please enter the Name”);
  document.js1.name.focus();
  return false;
 }

 if (y==“” || isNaN(y) || y.length>6 || y.length<6)
 {
  alert(“Please enter the Pincode properly”);
  document.js1.pincode.focus();
  return false;
 }
}
</script>
```

Figure 2.22 : JavaScript for validating name and Pincode

As seen in the script, we have declared another variable named y. The variable y will hold the value of the Pincode field. In the if condition, we have used a function named isNaN(). Let us understand this function.

isNaN()

To work with numeric values, JavaScript uses built-in functions. One of most commonly used numeric function is isNaN(). NaN stands for "Not a Number". The function isNaN(value) determines whether the value is NaN (Not-a-Number). It returns true if the value is not-a-number and false if the value is a number. For example,

isNaN(123) returns false as the value "123" is a number

isNaN("hello") returns true as the value "hello" is not a number

Thus, using if statement we check if y is; empty, not a number or not equal to six digits. If any of the condition is violated then the alert message will be displayed to the user.

Open the file in the browser. The output will be as shown in figure 2.23. Enter the name in the name field (Do not keep the name field empty, as this will again generate the alert message for it).

Figure 2.23 : Output in the Browser

Keep the Pincode field empty and press the Submit button. This will show an alert message as shown in figure 2.24.

Figure 2.24 : Alert message for empty field

Now, enter some characters or numeric value less than 6 digits length in the Pincode field. This will generate the same alert message as shown in figure 2.24.

Here, we have generated the same alert message for all the three validations in the Pincode field. If you want that the alert message should be displayed appropriately as per the validation, then we need to modify the script. In the example, we used a single if condition for all the three validations and gave the alert message. Now, to display the alert message as per the validation, we need to add three if conditions. Figure 2.25 shows the script with three if conditions.

```
<script>
function validateForm()
{
 var x=document.js1.name.value;
 var y=document.js1.pincode.value;
 if (x==null || x==="")
 {
  alert("Please enter the Name");
  document.js1.name.focus();
  return false;
 }
if (y==="")
 {
  alert("please enter the pincode properly");
  document.js1.pincode.focus();
  return false;
 }
if(isNaN(y))
{
 alert("please enter a number");
 document.js1.pincode.focus();
 return false;
}
if(y.length>6 || y.length<6)
{
 alert("please enter a six digits");
 document.js1.pincode.focus();
 return false;
}
}
</script>
```

Figure 2.25 : JavaScript to display appropriate message

Open the file in the browser. Enter some characters in the Pincode field and press the submit button. This will show an alert message as shown in figure 2.26.

Figure 2.26 : Alert message to check numeric entry

Enter some numeric value less than 6 digits length in the Pincode field and press the submit button. This will show an alert message as shown in figure 2.27.

Figure 2.27 : Alert message to check length

Thus, you learnt how using JavaScript we have added interactivity to our web pages. Now, let us take the "Registration Form" that we had created in the Chapter 1 and add JavaScript to it.

To apply validations, we will make some simple changes in the form that are listed below :

- Gender and Hobby field will not be kept initially selected.
- Address field has no initial text.
- City field has an option "choose the city" whose value is -1. All the other options in the city field like Ahmedabad, Rajkot, Surat have values 1, 2, 3 respectively.

To make the above changes, Open the form and select the respective field in the form. Go to **Form → Form Field.** This will show the properties of the respective field and make the changes. The form after making the changes will look as shown in figure 2.28.

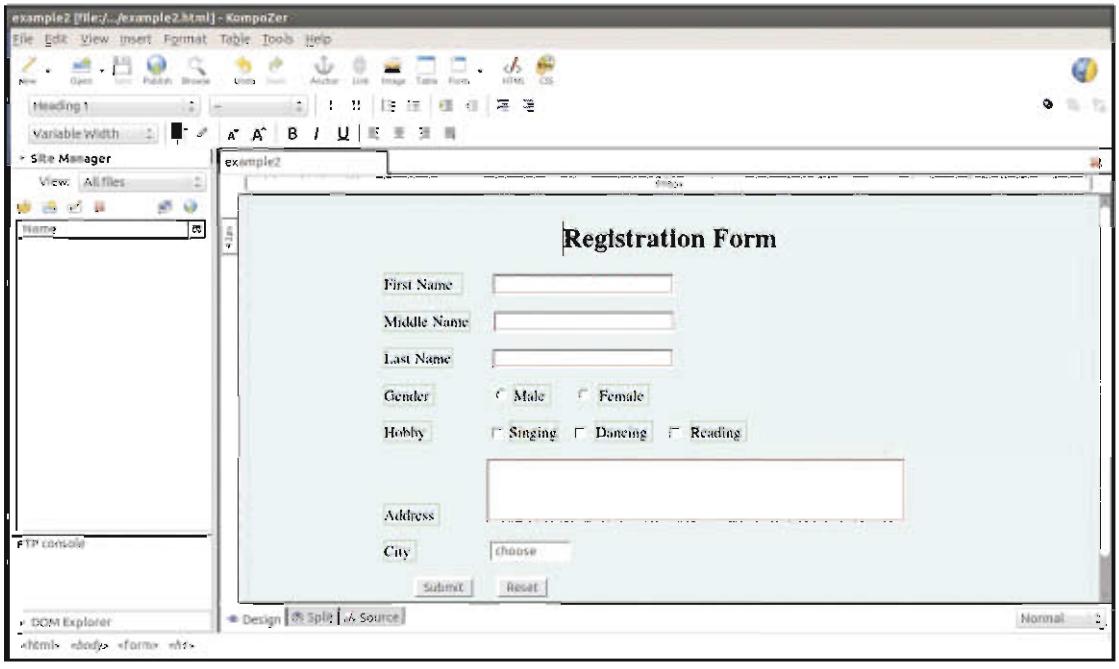


Figure 2.28 : Modified Registration Form

- Go to **Form → Define Form.** Alternatively, you can right click in the form and select properties option. This will open the Form Properties dialog box as shown in figure 2.29.

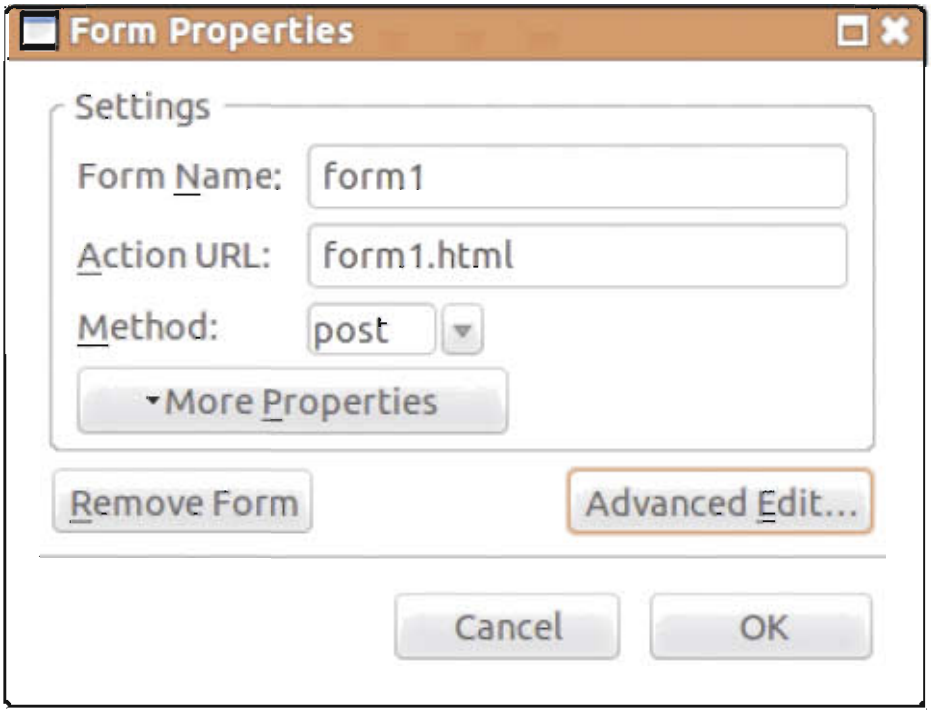


Figure 2.29 : Form Properties dialog box

- Click on Advanced Edit button. This will open Advanced Property Editor dialog box as shown in figure 2.30. Select the JavaScript Events tab. In the Attribute option select "onsubmit" and under the Value option write "return validateForm(). Press OK button.

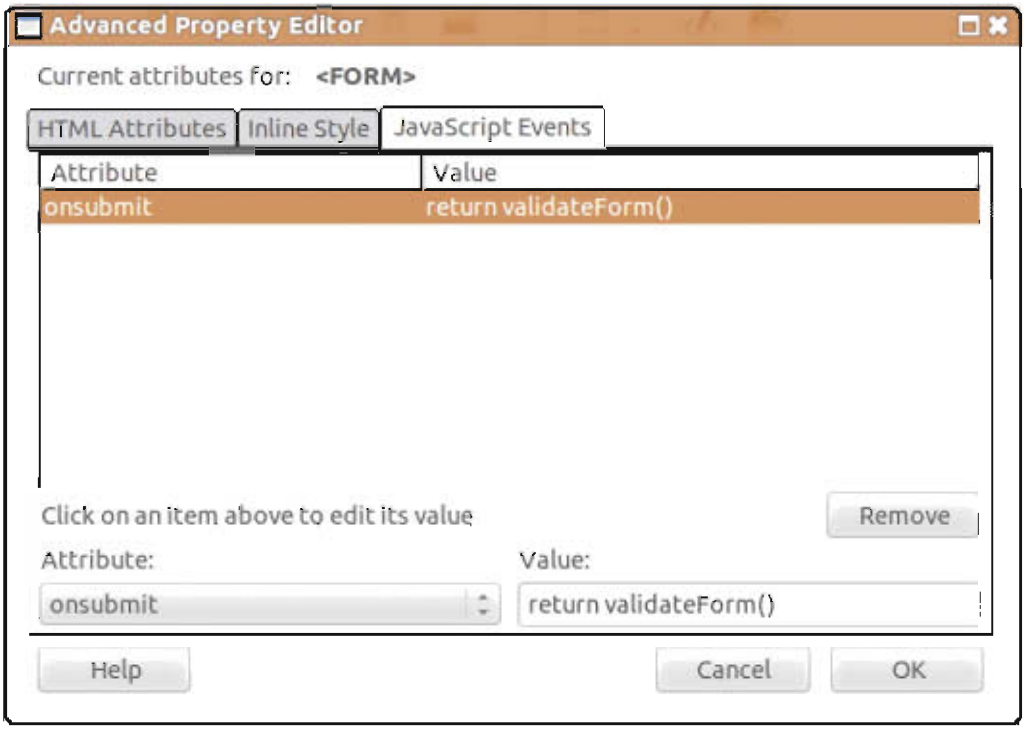


Figure 2.30 : Advanced Property Editor

- Add the script shown in figure 2.31 inside the <head> section as of the HTML.

```
<script>
function validateForm()
{
var x=document.form1.firstname.value;
var y=document.form1.middlename.value;
var z=document.form1.lname.value;
var r=document.form1.address.value;
if (x==null || x==="")
{
alert("Please enter the first name properly");
document.form1.firstname.focus();
return false;
}

if (y==""||y==null)
{
alert("Please enter the middle name properly");
document.form1.middlename.focus();
return false;
}
if(z==""||z==null)
{
alert("Please enter a last name properly");
document.form1.lname.focus();
return false;
}
if(r==="")
{
alert("Please enter address");
document.form1.address.focus();
return false;
}
if (( document.form1.gender[0].checked == false ) && ( document.form1.gender[1].checked ==
false ) )
{
alert ( "Please choose your Gender: Male or Female" );
document.form1.gender[0].focus();
return false;
}
if ((document.form1.hobby[0].checked == false) && (document.form1.hobby[1].checked ==
false ) && (document.form1.hobby[2].checked == false))
{
alert ( "Please choose a hobby" );
document.form1.hobby[0].focus();
return false;
}
if( document.form1.city.value == "-1" )
{
alert( "Please provide your city!" );
document.form1.city.focus();
return false;
}
}
</script>
```

Figure 2.31 : Validation JavaScript

As seen in the script, we have initialised four variables x, y, z and r.

In the statement var x=document.form1.firstname.value. The term form1 refers to the form name; term firstname refers to the element name (the name given to the input field "first name"). Thus,

- variable x stores the value of first name
- variable y stores the value of middle name
- variable z stores the value of last name
- variable r stores the value of address

In the script, using the if condition,

- variable x checks, if the first name is empty or not.
- variable y checks, if the middle name is empty or not.
- variable z checks, if the last name is empty or not.
- variable r checks, if the address is empty or not.

The radio buttons for gender field are grouped by the name "gender". Thus, gender is an array with two elements. The first element of the gender array is checked using the if condition with the statement document.form1.gender[0].checked and the second element of the array is checked using the statement document.form1.gender[1].checked. If both of them are false then no option is selected by the user. Thus, alert message is displayed and the focus is placed on radio button.

Similarly, the check boxes for the hobby field are grouped by the name "hobby". Thus, hobby is an array with three elements. Just as we checked the gender using the if condition, we check the hobby and if all the three elements are false then no option is selected and the user is displayed the alert message to select a hobby and the focus is placed on the check box.

For the city field, we check if the value is equal to -1 (Note: -1 is kept as the value for option "choose the city"), in which case no option is selected and the user is given an alert message. Figure 2.32 to 2.35 shows some of the alert messages displayed after the fields are left empty.

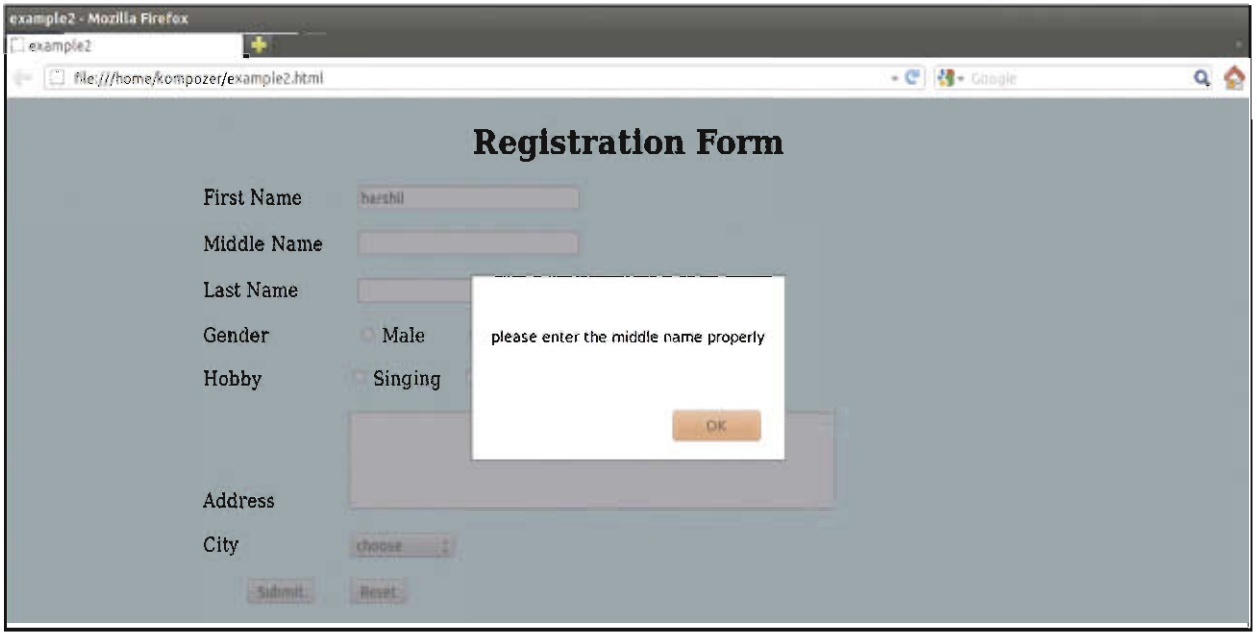


Figure 2.32 : Alert message when middle name left empty

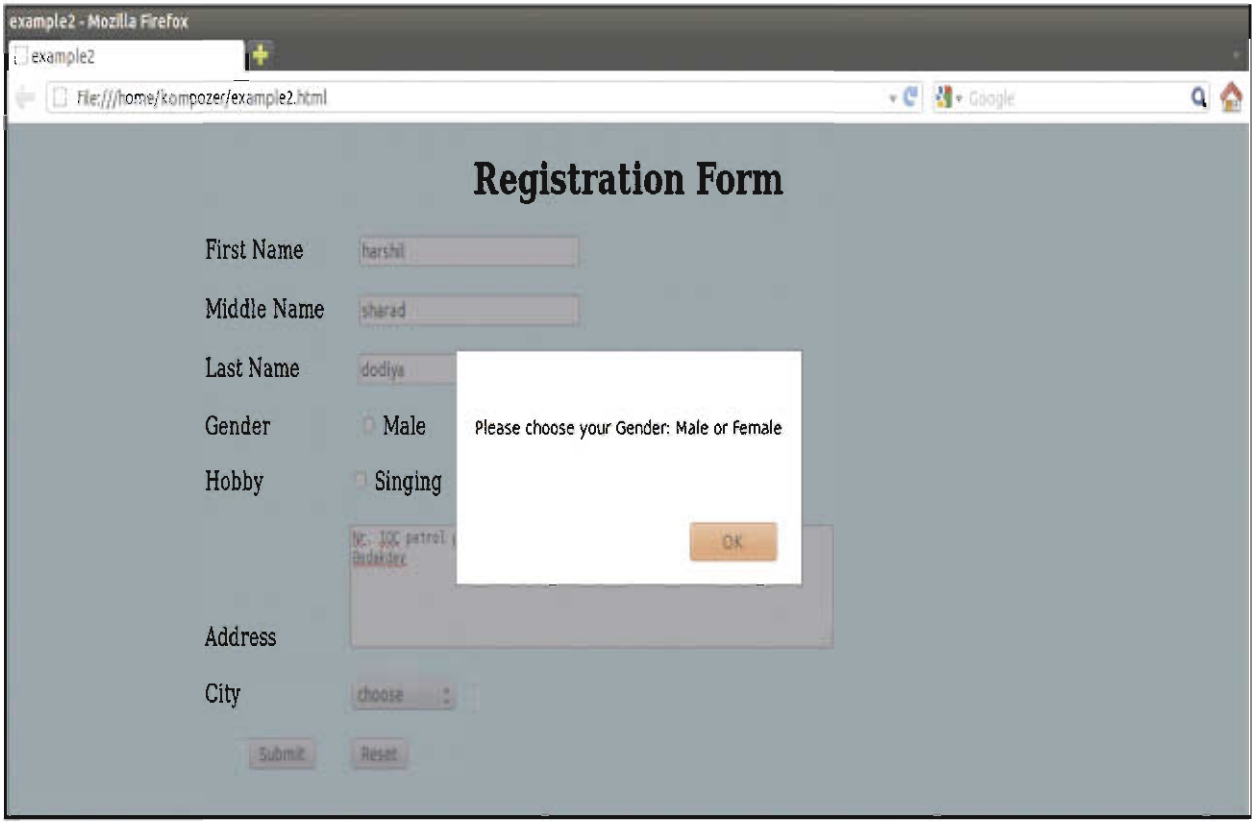


Figure 2.33 : Alert message when gender left empty

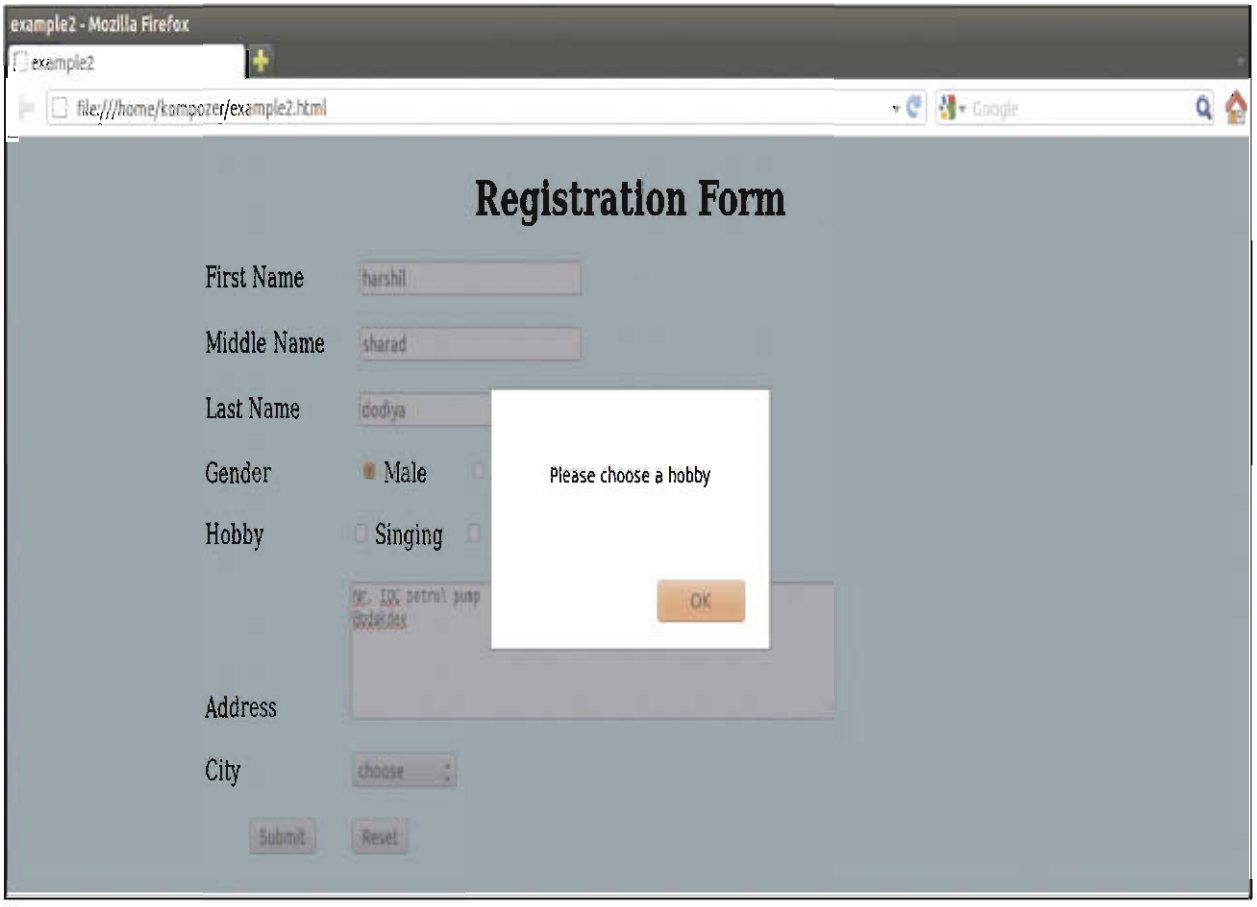


Figure 2.34 : Alert message when hobby left empty

Figure 2.35 shows the final form with all the fields filled as seen in the browser.

Figure 2.35 : Final form

Thus, we learnt how to use JavaScript for form validation and make the web pages more interactive.

Summary

In this chapter we learnt about Cascading Style Sheets and JavaScript. CSS allows you to specify styles for the visual elements of the website. It helps us to keep the information content of a document separate from the details of how to display it. We discussed about the advantages and disadvantages of CSS. JavaScript is a scripting language that allows us to add logical aspects to our web pages. A scripting language is a simple, lightweight programming language that does not contain the advanced programming functionalities. It is used in web pages to improve the design and validate forms. It adds interactivity to HTML pages and is inserted directly into the HTML code. The JavaScript code is inserted into an HTML page, using the <script>… </script> tag.

EXERCISE

1. State the purpose of Cascading style sheets.
2. Why should we keep the style and the content of a webpage separate ?

3. Explain the syntax of CSS.

4. State the advantages of CSS.

5. List the disadvantages of CSS.

6. Why do we use JavaScript in HTML pages ?

7. How does an HTML page identify the JavaScript code ? Give an example.

8. Choose the most appropriate option from those given below :

(1) Which of the following allows specifying styles for the visual elements of the website ?

(a) Cascading Style Sheets (b) Webpage

(c) Form (d) Animation

(2) Which of the following is known as special symbol in the syntax of CSS ?

(a) Rules (b) Selector (c) Declaration (d) Input

(3) Which of the following are two main parts of CSS rule ?

(a) Selector, declaration (b) Select, declaration

(c) Selector, declare (d) Selection, declaration

(4) Which of the following is an HTML element on which style can be applied ?

(a) declaration (b) selector (c) select (d) declare

(5) Which of the following is the syntax of CSS ?

(a) select {property : value} (b) selector {value : property}

(c) selector {property : value} (d) selection {property : value}

(6) Which of the following has developed JavaScript ?

(a) Yahoo (b) Google (c) Wikipedia (d) Netscape

(7) Which of the following is a scripting language that allows adding programming to web pages ?

(a) Action script (b) JavaScript (c) HTML (d) CSS

(8) Which of the following is a scripting language that is simple, lightweight programming language that does not contain advanced programming functionalities ?

(a) JavaScript (b) HTML (c) C (d) Java

(9) Which of the following tag is used to insert JavaScript code into an HTML page ?

(a) <script>... <script> (b) <script>... </script>

(c) <script>... <//script> (d) </script>... </script>

(10) Which of the following symbol signifies the start and end of a JavaScript block ?

(a) semicolon (b) square bracket

(c) curly bracket (d) round bracket

(11) Which of the following is a reusable block of code that performs a particular task ?

(a) Array (b) Code (c) Program (d) Function

(12) Which of the following statement is used to return a value in a function ?

(a) return (b) function (c) select (d) send

(13) Which of the following is generated by the browser due to interaction between the user and the web page ?

(a) Function (b) Response (c) Event (d) Value

(14) Which of the following is not an event ?

(a) Abort (b) Mouseover (c) Set (d) Load

(15) Which of the following is a container for storing data ?

(a) Variable (b) Integer (c) Event (d) Event handler

(16) Which of the following stands for BOM ?

(a) Browser Of Model (b) Browser Object Model

(c) Browser Object Modelling (d) Browse Object Model

(17) Which of the following is the top level object in the browser object model ?

(a) Window (b) Document (c) Page (d) Location

(18) Which of the following stands for NaN ?

(a) Not a Numeric (b) Not a Number

(c) Not a Noun (d) Not an Numeric

LABORATORY EXERCISE

1. Create a CSS for H1 as per the given rules and apply it on the text.

- Font: Times New Roman
- Color: red

- Case: Lowercase
- Font style: Italic
- Alignment: Centre
- Text decoration: Underline
- Background color : Light Grey
- Border: Dotted

2. Create a form with fields: name, email address, phone number and submit button. Give validation as given below :
   - Fields should not be empty.
   - Only numbers are allowed in the phone field.
   - The Phone number should be 10 digits long.
3. Give validation to the student's personal details form created in chapter 1.
4. Give validation to the feedback form created in chapter 1.

Designing simple website using KompoZer 3

A website plays a very important role in a business now-a-days. It helps in presenting the business to the world. It helps in promoting the business, selling the products and attracting a large number of customers. Therefore it is important that the website should have good design techniques which will enable the company to get the maximum amount of traffic, therefore making the maximum amount of profit.

A website is a collection of interlinked web pages for a specific purpose. The challenge however is in creating a website that is interactive, user friendly and providing accurate and useful information. The website should be designed in such a way that the users find the website informative and visit it repeatedly. It should increase customer base if the site is commercial. This objective can be achieved if appropriate care is taken at the time of website designing. This chapter discusses the general points to be considered while planning for a website. We will create a website using KompoZer and publish it on Internet.

Planning for the website

Developing a good website is a project similar to any other project and requires detailed planning. The better the planning, the chances of success in terms of usefulness of the website is higher. In other words we can say that, the desired goals for developing the website can be achieved. Following are the important points which should be considered for developing a good website as part of planning process.

Purpose

The purpose of the website should be clearly identified. Before creating a website, we should be clear with the definition and goal of the website. A properly designed website will be helpful to the organization. The purpose for creating the website may be to provide information to a group of people, or it may be to attract the new customers or it may be to sell the products online. Having decided the purpose, the content and layout of the website can be properly developed.

Audience

Before we start with the designing part, we should know the expected users of the website. The website should contain both general and detailed information. It should also contain specific and non-specific data. The proportion of the different type and extent of detailed and specific information depends on the nature of the expected audience. The user expectations from the website are important to know, as it helps to develop the appropriate content and layout of the site. This also ensures that the website is useful and successful.

Another consideration is the speed of connection that the expected users will have. In case, large graphical files are kept on the website, they will take a long time to download. The users will become impatient while waiting for large files and leave the site. Thus, it is desirable that the website should contain minimal required graphic and multimedia files.

Content

The website must contain complete and relevant information. Excessive use of irrelevant information can cause frustration to the users as they are unable to seek the specific data. Also if the information provided is incomplete, the users can leave the website. Placing appropriate and relevant content will satisfy users and sustain their interest in the website.

The website content should be classified into general and detailed categories.

The general content provides an overview of the site, organization, products and services, and other items. Brief descriptive information helps user in determine what he is searching. It directs to the appropriate specific data and informs about other available items at the site.

The detailed content provides users with the detailed information on the site like products and services description. The layout of the site must ensure easy access to the detailed information, and specific information they seek.

Text paragraphs should always be kept at reasonable lengths. If there is a lot of content to be displayed on a webpage, then try to split it into small text blocks, this way visitors will be easily able to pick out the content they need to know. The navigation in the website should also be made as easy as possible. Moving from one page to another should be easy so that the user knows where he is and how to get back to home page.

Ensuring that appropriate information (both general and detailed) is available and easily accessible ensures that site can serve its intended purpose. A web site is a valuable tool for businesses and other organizations when composed of the right material (content). As with other tools, the web site can perform an outstanding job.

Medium

More and more people are using Internet through smart phones and tablets. The website design should scale for all the devices like computers, smart phones, and tablets. Also the website should be designed keeping in mind that it is well displayed in all the popular web browsers like Mozilla Firefox, Chrome, Opera or Internet Explorer.

Creating a simple website using KompoZer

In earlier chapters we discussed how to create web pages using KompoZer and use JavaScript for interactivity. We have also seen the general considerations while designing a website. Now we will see how to design and develop a small E-commerce website in KompoZer.

Let us create a simple Ecommerce website named "School Plaza", which caters to the needs of the school going students. On our website, all the school related items like books, stationary, school bags, water bottles, lunch box, files, folders and other such related items are available and can be easily ordered. We will also create a shopping cart which will show us all the products and the total amount of the items that we have purchased. Figure 3.1 and 3.2 shows the look of the home page and shopping cart of the website that we plan to design.

Figure 3.1 : Homepage of website

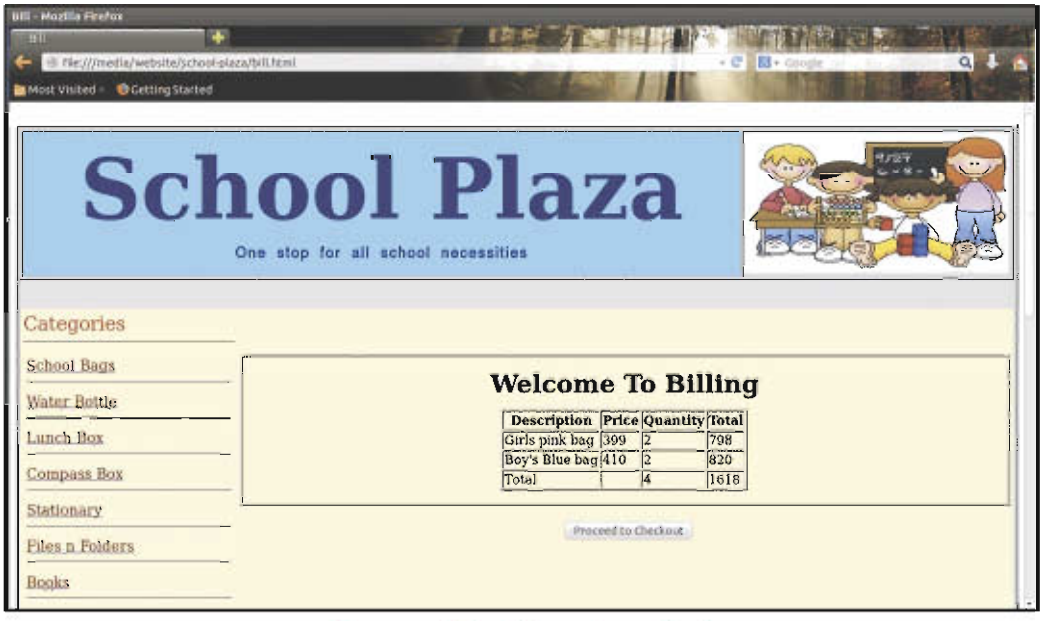


Figure 3.2 : Shopping Cart

Let us learn to create our website step by step. To start with, we should first design the home page of our website. The home page of the website is the first page that opens when the user enters the URL address in the address bar of the browser. The home page of our website should list all the categories of the items that are available. Clicking on any one of the categories will take us to another webpage showing the items of that particular category. To create the website, various designs which are known as templates are freely available on Internet. We can easily download the templates and can customize them as per our needs.

To start with the designing of home page, open KompoZer and create a new webpage. To organize all the contents and the images related to the theme we will insert a table with single cell as shown in the figure 3.3. The table will contain all the text and the images related to the website. To insert the table go to **Insert → Table.** Select single cell table and press OK button.

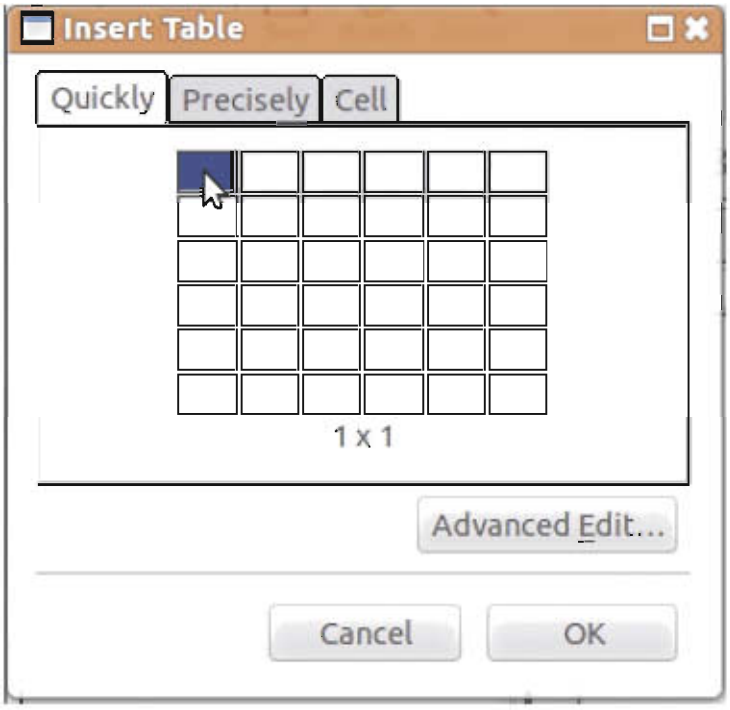


Figure 3.3: Insert 1×1 Table

Save the webpage with a suitable title. The title will be displayed on the title bar of the web browser. Here, we have given the title as "School Plaza". Save the file with the name "index.html". Generally, whenever we create the home page of the website it is named as index.html. Every website is built inside directories on a web server and each webpage is a separate file on that web server. Whenever the user types the URL of the website in the browser, say for example, www.schoolplaza.com, he/she does not specify the file to be opened. But the web server needs a default file name in order to display some content. This default file is the page named index.html. Thus, when we type a URL without the filename, the server looks for the default file and displays it automatically.

Now, we will give a background color to the inserted table. Select the table and double click it. Alternatively, we can also right-click the table and select 'Table Cell properties'. This will open Table Properties dialog box as shown in figure 3.4.

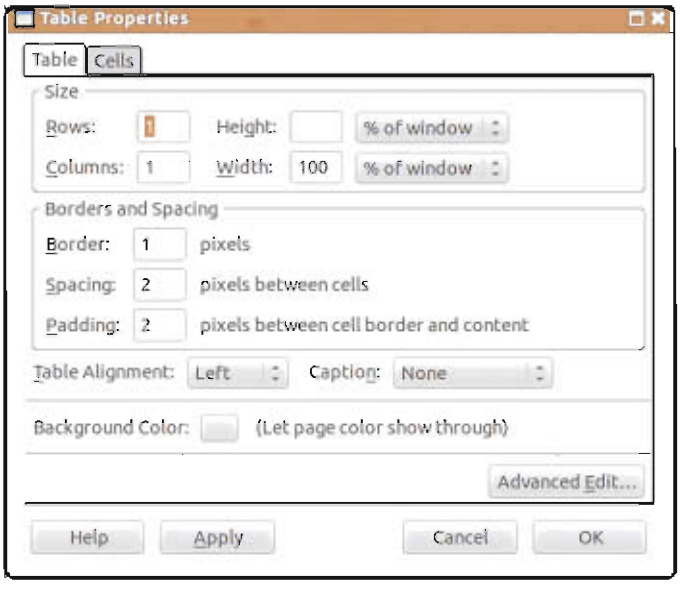


Figure 3.4 : Table Properties dialog box

You can see two tabs 'Table' and 'Cells' which allows user to control several aspects of either the table or individual cells. You can see the cells related to a particular table. The options seen in the table tab are :

1. Size : it shows the number of rows, columns and height and width of the table. Height and width can be specified in pixels or % of the window.
2. Borders and Spacing: it shows the border option in case you want to give a border to the table. Spacing is used to specify gap between cells. Padding gives a gap between the edge of the cell and the text within it.
3. Table Alignment : to align the table as left, centre or right.
4. Caption : Used to give caption to table if require.
5. Background Color : to give a background color to the table.

Now select a background color of your choice as shown in figure 3.5.

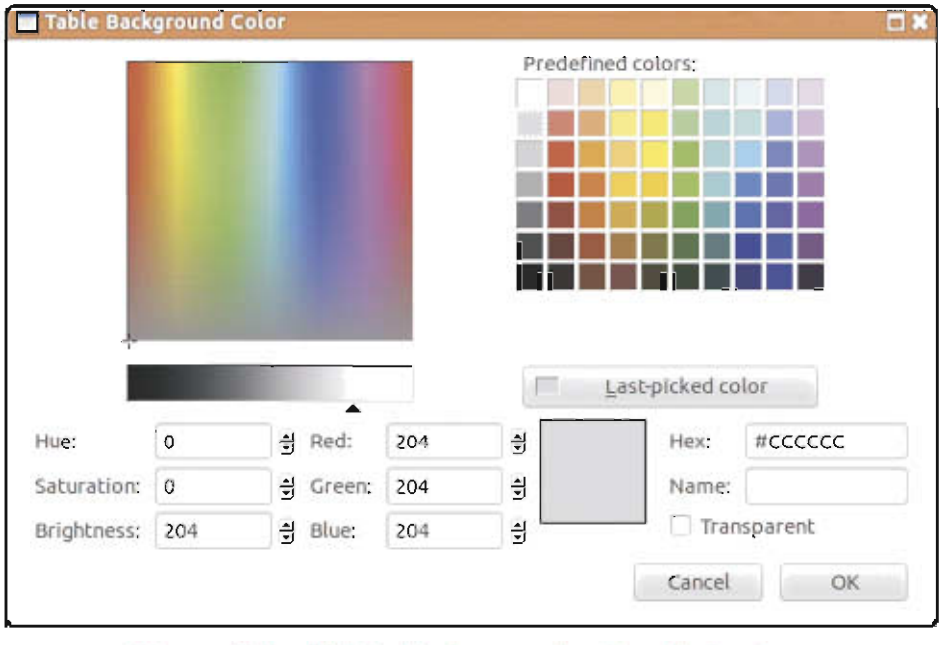


Figure 3.5 : Table Background color dialog box

Now, to display the text "School Plaza" and the related image, we will insert another table with one row and two columns. One of the columns will display the text and the other displays the image. Go to **Insert → Table.** Select 1 × 2 table as shown in figure 3.6 and press OK button.

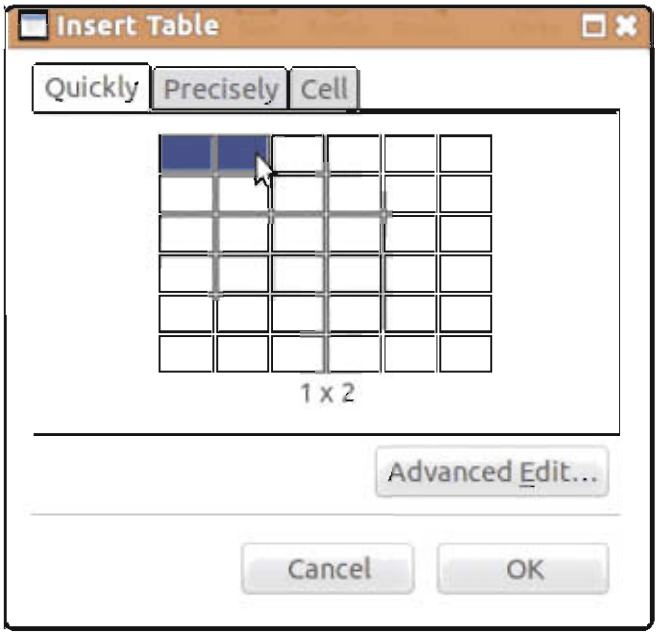


Figure 3.6 : Insert 1×2 table

Give a background color to the table. Now in the left cell we will insert the text and in the right cell the image. To insert image, select **Insert → Image.** Alternatively, you can click the image icon on the toolbar. This will open the dialog box as shown in figure 3.7. Specify the location of the image. Specify the alternate text in the input box otherwise select the "Don't use alternate text" radio button in case you do not want to give alternate text. Giving alternate text will display the text in case the image is not displayed in the browser.

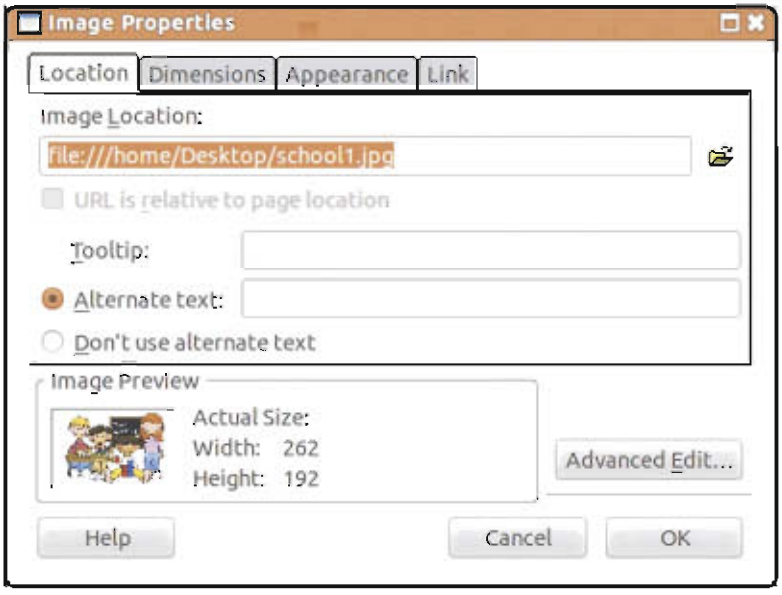


Figure 3.7 : Image Properties dialog box

Note : We have used an image named school1.jpg. Students may design their own image or use any other alternative image of their choice.

Now, select the Dimensions tab and select the Custom Size option. As we want to display the image in the whole cell we will specify the height and the width as 100% of the cell as shown in figure 3.8. Insert the text as shown in figure 3.9 in another cell.

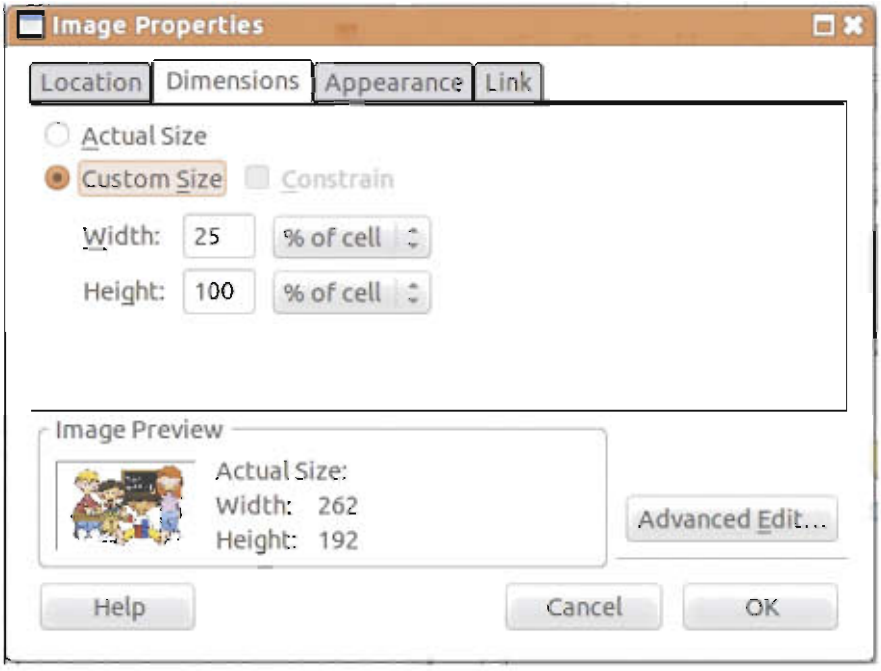


Figure 3.8 : Dimensions tab of Image Properties dialog box

Figure 3.9 shows the inserted text and the image.

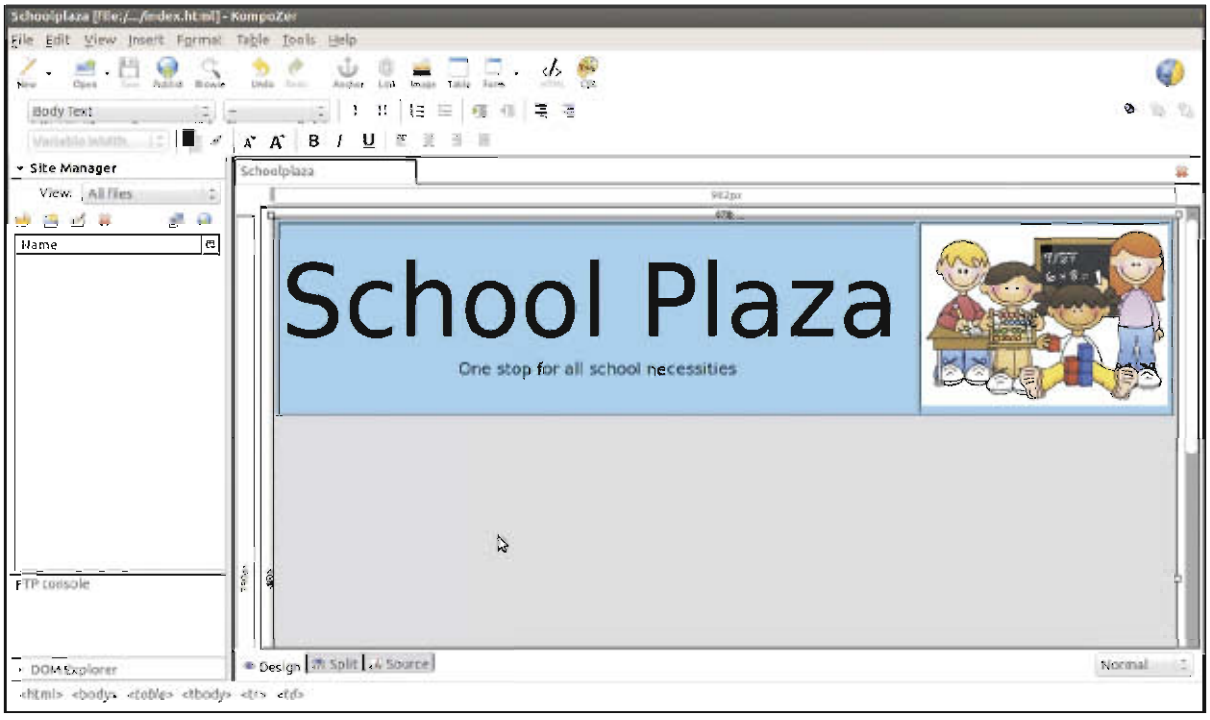


Figure 3.9 : Table with image and text

Now, we will insert another table with 2 rows and 2 columns to display the categories and New Products. In the first row cells, type the text "Categories" and "New Products". Give a suitable color to the text by going to **Format → Text color.** Figure 3.10 shows the row with the text.

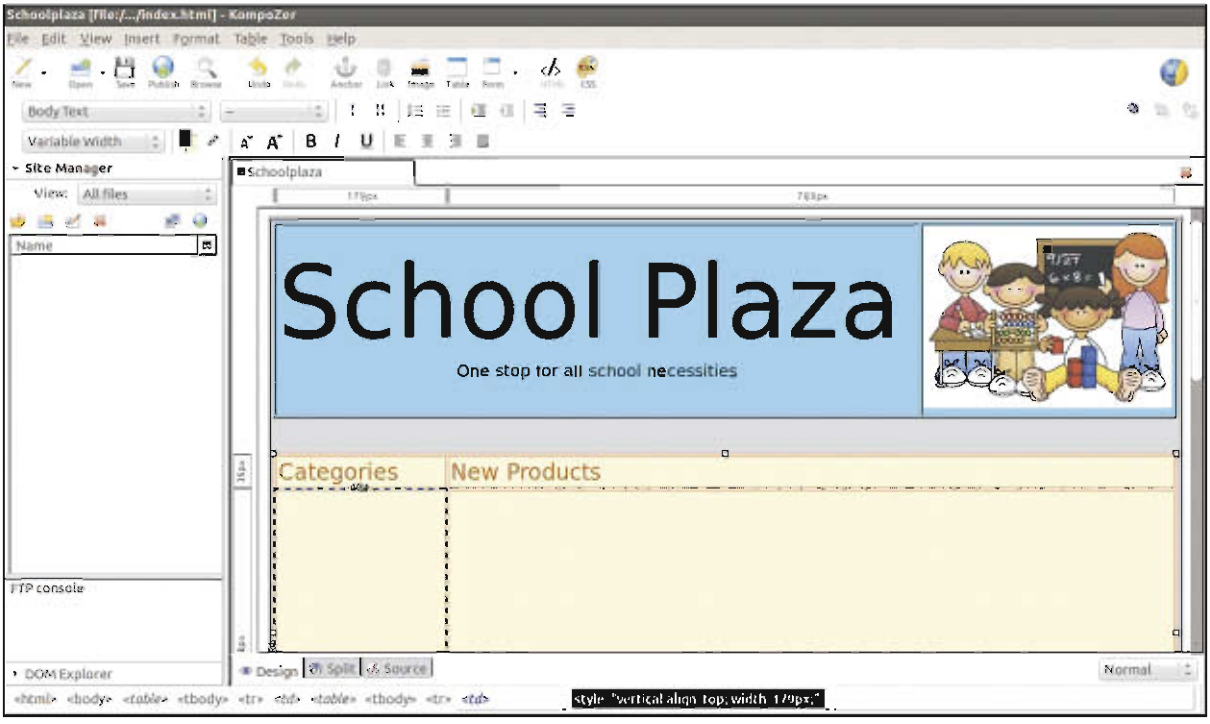


Figure 3.10 : Text inserted in the table row

Next, in the second row just below the categories we will insert another table with single column and multiple rows. In this table, we will insert different categories in each row like school bag, water bottle and others. Give a suitable color to the text. Here, we have inserted a horizontal line to differentiate the categories. To insert a horizontal line, go to **Insert → Horizontal line.** Also, the categories will be a link which takes us to the related page of the selected category. We will make the link on the categories later on. Let us now create table to add the images of the items and the related details in the cell below the new products. Insert the table depending on the items we want to display on the page. Here we have inserted a table with 3 rows and 3 columns. Figure 3.11 shows the added tables in the web page.

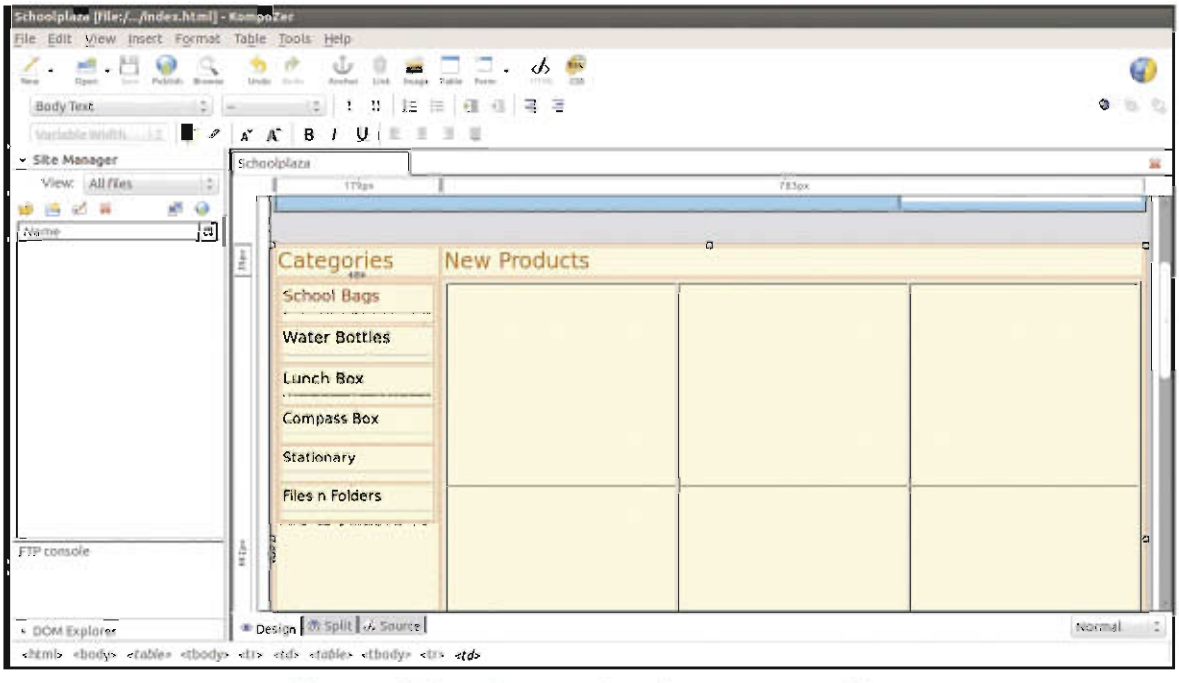


Figure 3.11 : Screen showing category list

Next, we will insert another table at the bottom of the page for the details like "About school plaza", "Feedback", "Contact us" and "Site map". Create the table and insert the text. This text will also be a link.

We have designed most of the home page. Whenever we click on any of the categories, the design of the page remains the same, but the images related to the category are changed. Thus we need not create other web pages from scratch but make multiple copies of our home page which can then be customized as per our need.

Go to Save As option and save the file with different names. We have saved the file as waterbottles.html, school-bags.html and lucnchbox.html. Now when the link of a particular category is clicked the corresponding web page will be opened. You must have observed that the web pages thus created have the same title as that of the home page. Open the web pages one by one and change the page title by going to **Format → Page Title and Properties.** Now, open the respective web pages and put the images as per the category. We will add the images on the home page and the same way they can be added to other web pages also.

Before inserting images of the items and their details on the home page, let us create link on the different categories. Clicking on a category will open the related web page.

Open the index.html. To create link, select the text (School Bags), go to **Insert → Link.** This will open a dialog box as shown in figure 3.12. Under the Link Location option, select the file to be opened (school-bags.html). Press OK button.

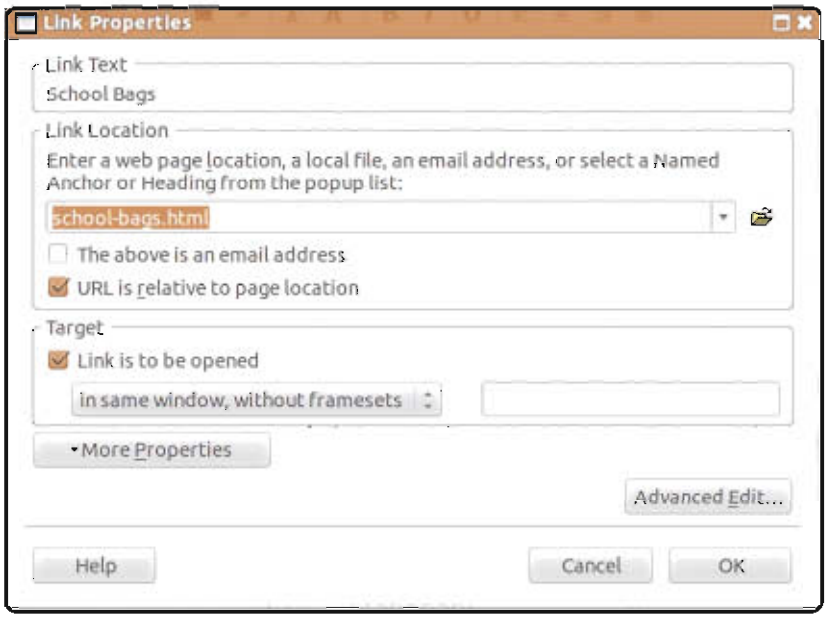


Figure 3.12 : Link Properties dialog box

Now, let us insert the details of the new products in the home page. In each cell of the table, we will insert an image of the item, name of the item, price and add to cart checkbox and quantity textbox. Insert the image as discussed earlier. Add the product name and price. Now, insert a checkbox by going to form field. We will add an HTML attribute "id" named b1 to the checkbox which will be used in JavaScript later on. Add the label "Add to cart". Then, insert textbox with initial value 1 and an id "q1". Add the label "Quantity". When the user selects add to cart checkbox, the item and its quantity are saved in the shopping cart. Figure 3.13 shows the details of the products added.

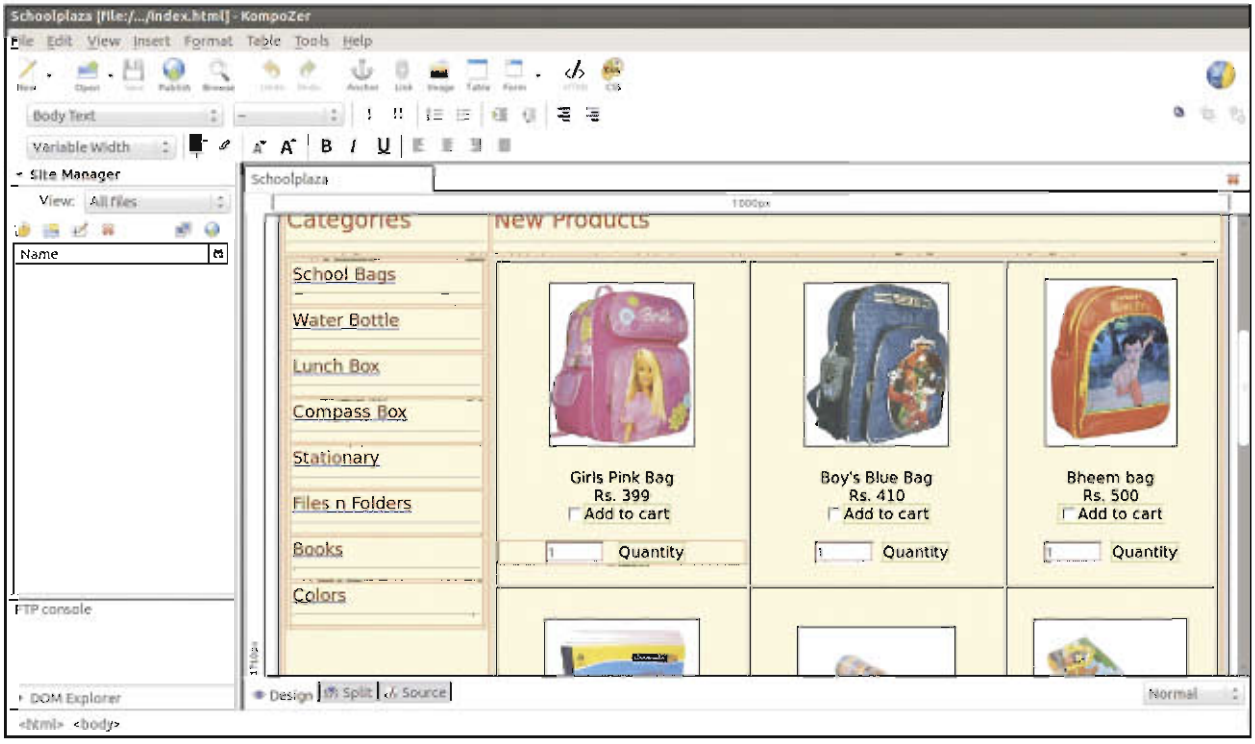


Figure 3.13 : Product Details

Note: Add the images related to the category on the web pages that we have created. Rest of the features remains the same. We have not shown the other web pages here.

We will add a button "Purchase" at the bottom of the page which will take us to the bill details web page. This page displays the total items purchased and the total amount. On the buttons onclick event we will add a Javascript shown in code listing 3.1. It calls the function oncart(), the function oncart() will then call the bill web page.

```
<script>
function oncart()
{
var check=false;
var cookievalue="";

if(document.getElementById("b1").checked)
{
check=true;
var quantity=document.getElementById("q1").value;
var price = 399;
var total=(price*quantity);
cookievalue+=":Girls pink bag,"+ price + "," + quantity + "," + total;
}

if(document.getElementById("b2").checked)
{
```

```
check=true;
var quantity=document.getElementById("q2").value;
var price = 410;
var total=(price*quantity);
cookievalue+=":Boy's Blue bag,"+ price + "," + quantity + "," + total;
}

if(document.getElementById("b3").checked)
{
check=true;
var quantity=document.getElementById("q3").value;
var price=500;
var total=(500*quantity);
cookievalue+=":Bheem bag,"+ price + "," + quantity + "," + total;
}

if(document.getElementById("nb1").checked)
{
check=true;
var quantity=document.getElementById("q4").value;
var total=(50*quantity);
cookievalue+=":NoteBook (set of 3),"+total;
}

if(document.getElementById("cp1").checked)
{
check=true;
var quantity=document.getElementById("q5").value;
var total=(45*quantity);
cookievalue+=":Color pencils,"+total;
}

if(document.getElementById("p1").checked)
{
check=true;
var quantity=document.getElementById("q6").value;
var total=(80*quantity);
cookievalue+=":Pencil Box,"+total;
}

if(document.getElementById("l1").checked)
{
```

```
check=true;
var quantity=document.getElementById("q7").value;
var total=(110*quantity);
cookievalue+=":Angry Bird Lunch box,"+total;
}

if(document.getElementById("l2").checked)
{
check=true;
var quantity=document.getElementById("q8").value;
var total=(120*quantity);
cookievalue+=":cartoon lunch box,"+total;
}

if(document.getElementById("e1").checked)
{
check=true;
var quantity=document.getElementById("q9").value;
var total=(50*quantity);
cookievalue+=":Eraser (set of 4),"+total;
}

if(!check)
{
alert("No item selected");
}
else
{
document.cookie=cookievalue;
window.location="bill.html";
}
}
  </script>
```

Code listing 3.1 : Validation JavaScript

As seen in the code listing 3.1 we have a function oncart(). The first if statement block checks whether the checkbox for item1 is checked or not. Earlier we have given an id named "b1" to the checkbox of item1. Using document.getElementById("b1").checked, we will check whether the user has checked the item. If it is checked then we will save the quantity into a variable and calculate the total price. We finally save the product name, price, quantity and amount as a string in a variable. Each of the details are separated with a delimiter "," (comma). Individual product description is separated with delimiter ":" (colon). The script continues for each item listed.

If the user has not selected any of the items then pressing the purchase button will show a message with text "No item is selected". Otherwise, the items are saved as cookie. A cookie is a variable that is stored on the user's computer. They are useful as they provide feature to remember and track preferences, purchases, commissions, and other information required for better visitor experience or site statistics. JavaScript can manipulate cookies using the cookie property of the document object. Cookies can be read, created, modified and deleted.

When the user clicks on the Purchase button, the script calls the web page "bill.html". The JavaScript in bill.html will calculate the total of all the items selected along with the total amount to be paid. Code listing 3.2 shows the JavaScript that calculates the total amount.

Here we have defined a function myfun(). The function is accepting data from cookie and storing it in an array. Then we are converting the data into number by using function parseFloat() and add it to total. Finally we print the values of the total.

```
function myfun()
{
var cookiearr=new Array();
cookiearr=document.cookie.split(":");
document.writeln("<center><h1>Welcome To Billing</h1></center>");
document.write("<center><table
border=3><thead><tr><th>Description</th><th>Price</th><th>Quantity</th><th>Total</th></tr></
thead>");
var total2=0;
var total3=0;

for(var i=1;i<cookiearr.length;i++)
{
var cookiedata=cookiearr[i].split(",");
document.writeln("<tr><td>"+cookiedata[0]+"</td><td>"+cookiedata[1]+"</td><td>"+cookiedata[2]
+"</td><td>"+cookiedata[3]+"</td></tr>");

total2+=parseFloat(cookiedata[2]);
total3+=parseFloat(cookiedata[3]);
}
document.write("<tr><td>Total</td><td>"+" "
+"</td><td>"+total2+"</td><td>"+total3+"</td></tr>");
document.write("</tbody></table></center>");
}
 </script>
```

Code Listing 3.2 : JavaScript to calculate total amount

Finally, the user clicks on the Proceed to Checkout button on the bill webpage. This will ask the user to login in case he/she has not logged in or Sign-up to create an account if he/she is not a registered user. The registration form can be created as discussed in the earlier chapters. Later the payment details can be entered as decided by the owner. Generally, to handle such websites we need to create database which stores the details of the product, registered user, login name, password and other relevant details.

Publishing a Website

Till now we learnt how to design a simple website using KompoZer. Now, let us see how to publish our website. To publish a website means to transfer the web pages, images and stylesheets related to the site, to a web server from which they may be accessed by the users. This process is generally called 'Uploading'.

We can upload the website if we have an account on the web server. The Internet Service Provider (ISP) offers limited free space on the web server. We can also buy space from professional hosting providers. Alternatively, to upload the website, there are many web hosts that provide limited free space to the users. The free space can also be for a limited time period. We can register on these sites and can avail the free space. In our sample website example we have opted one of these free space provider and uploaded the website. To publish a site, we need to know the settings for the space to set up the system. Open the index.html page and go to **File → Publish.** This will open a Publish Page dialog box as shown in figure 3.14.

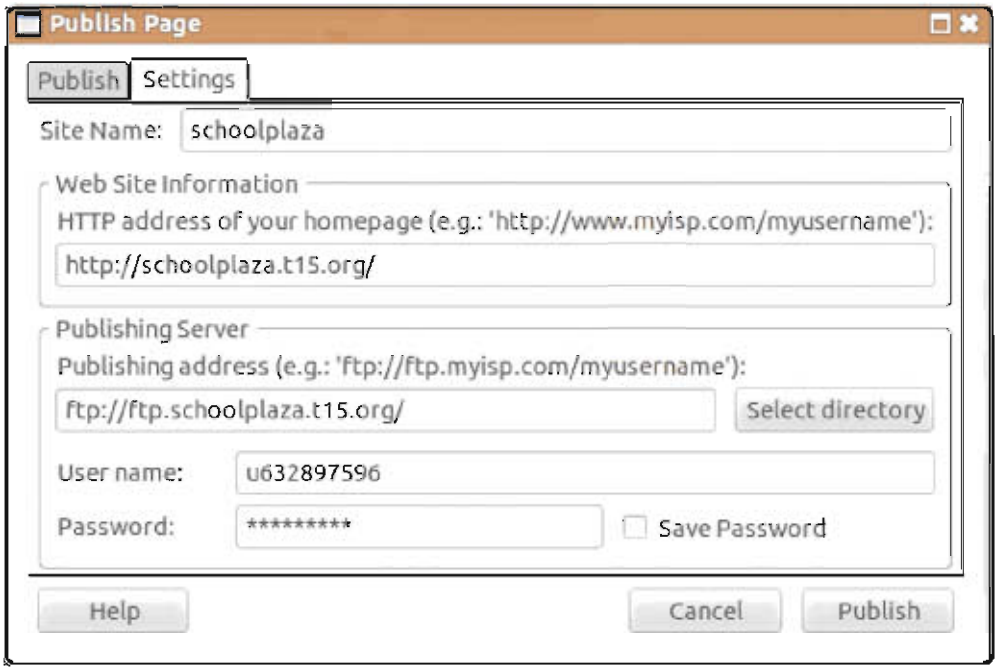


Figure 3.14 : Publish Page dialog box

The following details are to be provided in settings tab of Publish Page dialog box.

1. In the site name field, enter the name of the website. This name is only used for internal purpose by KompoZer to refer to the website.

2. In the website information, "HTTP address of your homepage" field specify the actual web address or URL of the website.

   Note : We can register a domain on Internet. There are many websites on Internet wherein we can register a domain name, for example: www.schoolplaza.com, for our website by paying nominal cost.

3. Publishing server details are provided by the ISP or the web host from where we have purchased the space. Type the publishing address. If connecting via FTP then type in FTP address. File

Transfer Protocol (FTP) is the means by which we can transfer the web pages from our computer to the web host. As discussed earlier, this is known as uploading or publishing. Provide the FTP username and password.

Note : The detail shown in figure 3.14 has been provided by the web host providing the free space.

Now click on Publish tab of figure 3.14. This will open a dialog box as shown in figure 3.15.

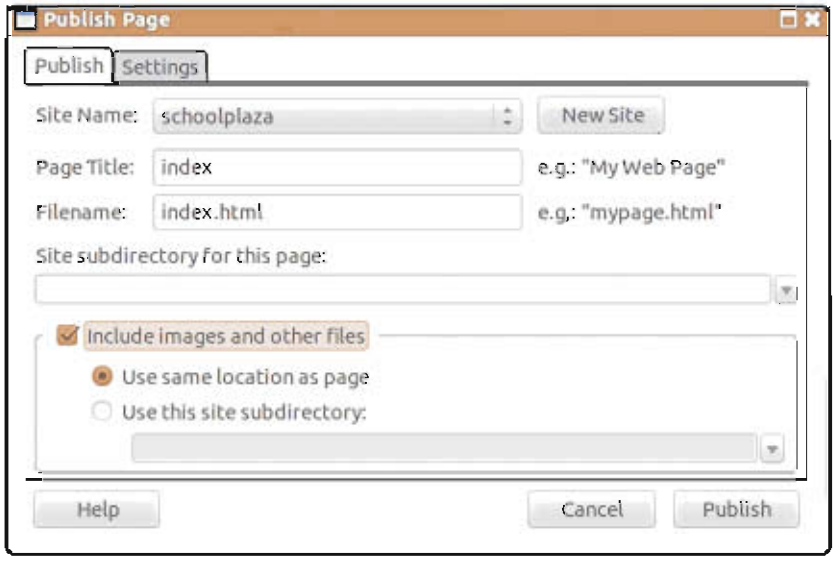


Figure 3.15 : Publish tab of Publish Page dialog box

Perform the steps mentioned below :

1. Select the site name from the drop down box if it is not selected.
2. Enter Page Title, it refers to the title of our index.html page.
3. Provide the filename. Here, as index.html file is open, it will take the filename automatically.
4. If this web page needs to be in a sub-directory of your site, enter in the "Site subdirectory" field. Remember that some hosting providers require that you keep the web pages in a separate directory like public_html.
5. Specify the directory for the images. Generally, keep the images in a folder in the same directory.
6. Click Publish button. This will open Publishing dialog box as shown in figure 3.16.

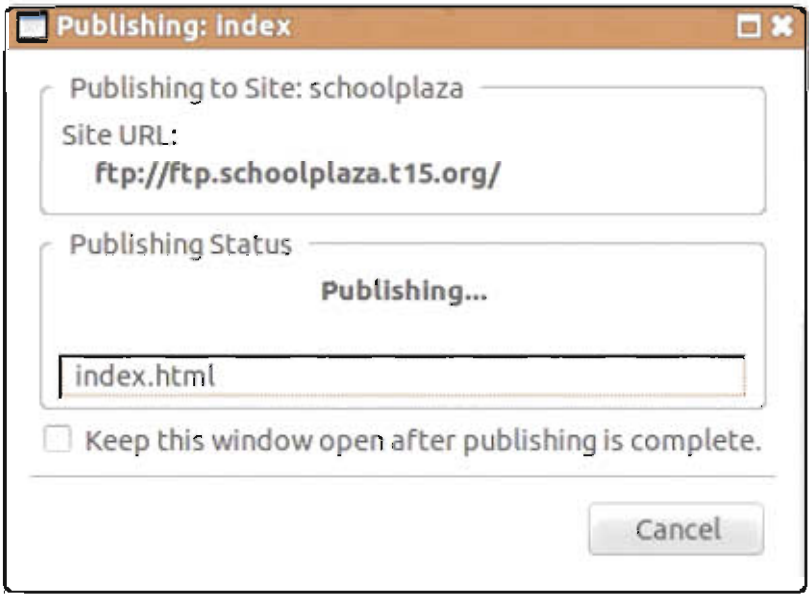


Figure 3.16 : Publishing dialog box

KompoZer will now upload the files and a popup message will be shown when finished. After uploading, type the URL of the website in the browser and check the website.

Other Open source web development tools

We have learnt how easy and fast it is to design web pages using KompoZer. As we have discussed earlier, KompoZer is a free open source web development IDE. Just like KompoZer, we have many other open source web development tools which are freely available on Internet.

Let us discuss about some of the open source web development tools that are available easily on the net and are used by web developers to develop websites with ease.

Aptana studio

Aptana studio is a powerful open source IDE (Integrated Development Environment) for building web applications. It is a complete web development environment which provides language support for HTML, CSS, JavaScript, Ruby, Rails, PHP, Python and many others. It also comes along with a large number of additional plugins. Aptana Studio 2.0.5 (also known as Aptana studio 2) is currently used to develop web applications using HTML, CSS and JavaScript. It can be easily downloaded from its website www.aptana.com. Figure 3.17 shows the Aptana studio interface. As seen in the figure it can create projects in PHP, Rails, Ruby or Web. Figure 3.18 shows the HTML code written in this interface.

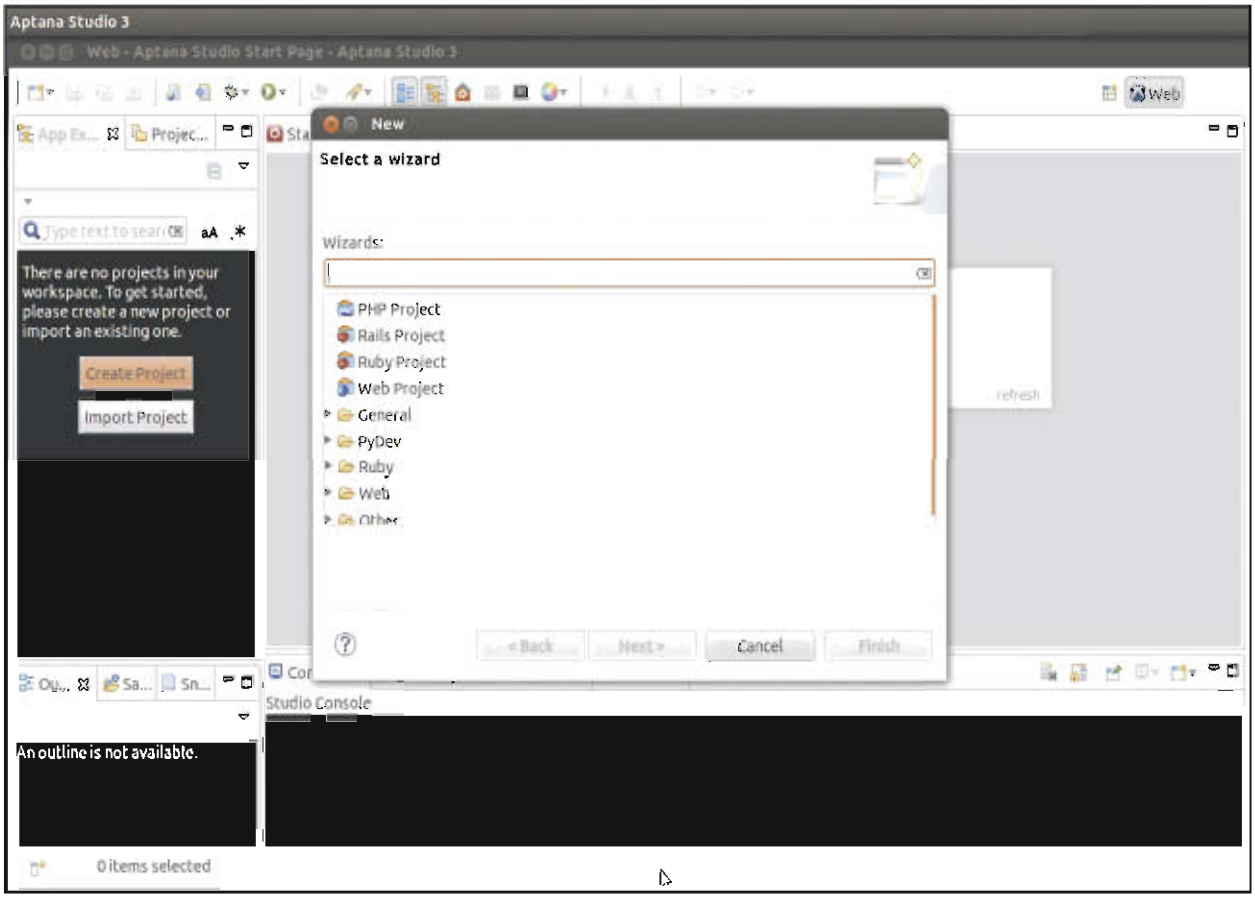


Figure 3.17 : Project creation using Aptana Studio

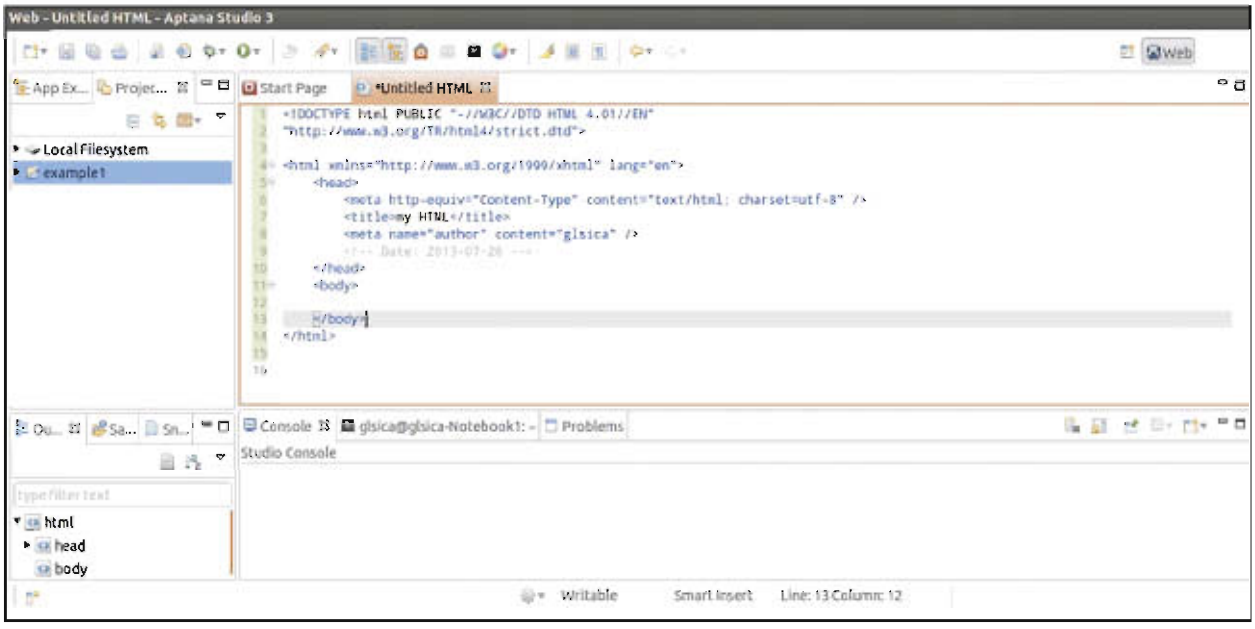


Figure 3.18 : Aptana Studio Interface

BlueGriffon

BlueGriffon is another free open source WYSIWYG HTML editor. It can be easily downloaded from www.bluegriffon.org. It supports languages like English, Dutch, German, Chinese and many more. It is an intuitive application that provides web developers with a simple user interface which allows them to create attractive websites without requiring in depth technical knowledge about web standards. Figure 3.19 shows the interface of BlueGriffon version 1.6.2. We have created a simple form using the tools.

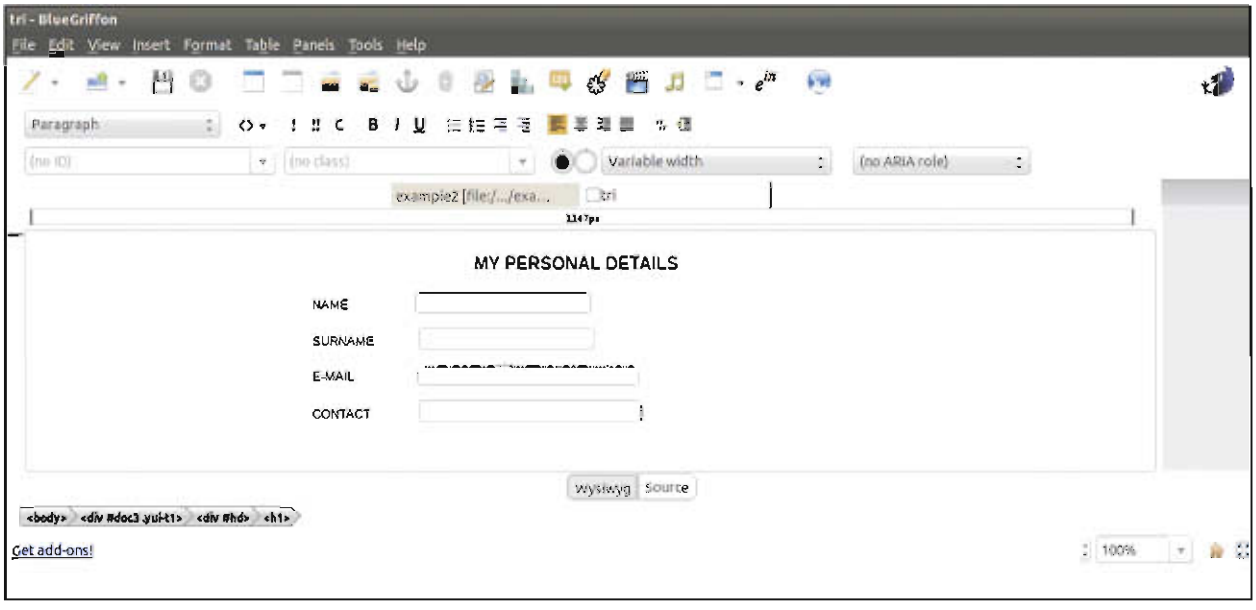


Figure 3.19 : Webpage created using BlueGriffon

Amaya

Amaya is another free open source WYSIWYG web editor developed by the World Wide Web Consortium (W3C). It was initially started as an HTML/CSS editor and has now grown into an

editor for many XML-based systems. It can be easily downloaded from www.w3.org/Amaya. Figure 3.20 shows a web page created in Amaya. The source for the web page can be seen in the window just below the design of the web page.

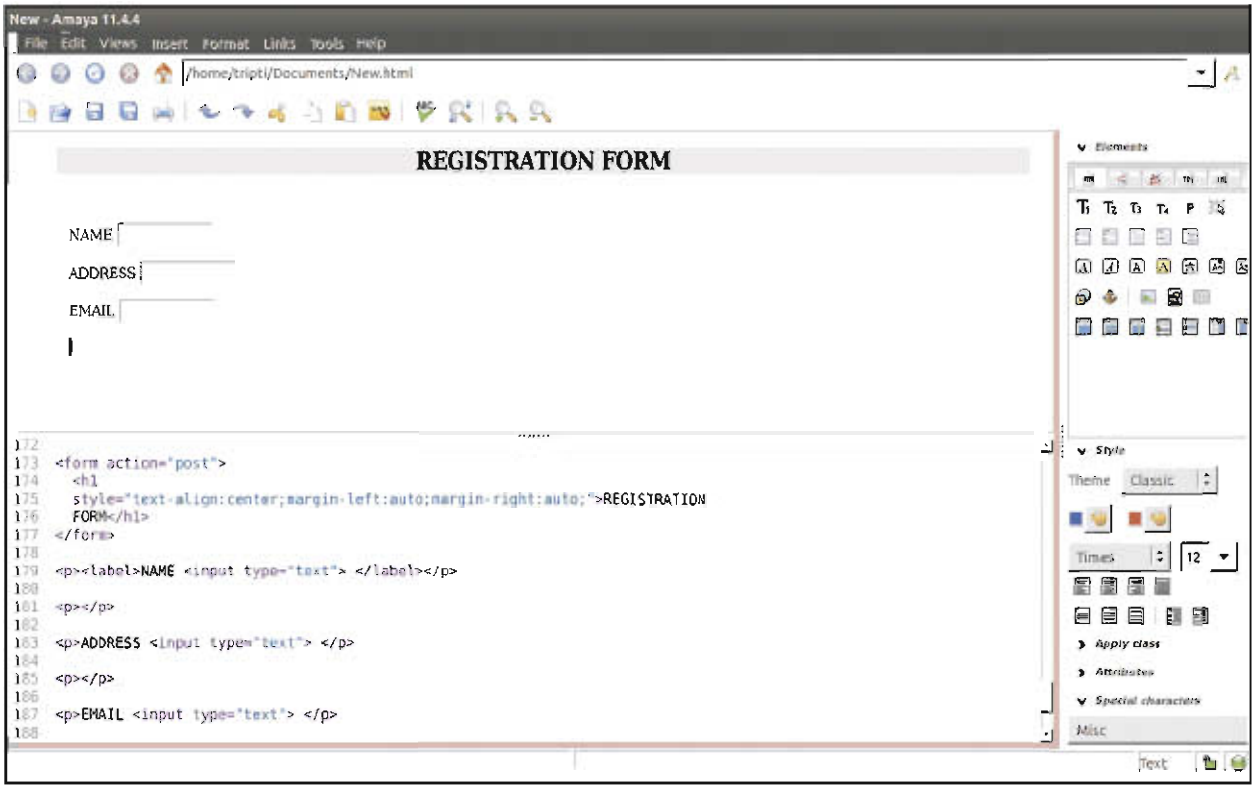


Figure 3.20 : Web page created using Amaya

Summary

In this chapter we discussed how we can create a simple website using KompoZer. After creating the website we need to upload it on the Internet. Publishing a website means transferring the site, i.e. the pages, images and stylesheets, to a web server from which they can be accessed by the users. This process is called 'Uploading'. We can upload the website if we have an account on the web server. The Internet service providers (ISP) offer limited free space on the web server. We can also buy space from professional hosting providers. We also discussed about the different free open source web development IDE available on Internet like Aptana studio, Blue Griffon and Amaya.

EXERCISE

1. How does a website play an important role in business ?
2. List the important points to be considered for developing a good website as part of planning process.

3. Why should the purpose of the website be clearly defined ?

4. How does knowing the expected audience help in the design of the website ?

5. What content should be placed in the website while designing it ?

6. Why is the homepage of the website named as index.html ?

7. Which are the options available to set the table properties ?

8. What is a cookie ?

9. What do you mean by publishing a website ?

10. Name some of the open source web development tools available in the market.

11. Choose the most appropriate option from those given below :

(1) Which of the following helps in promoting the business, selling the products and attracting a large number of customers ?

(a) Website (b) Webpage (c) Form (d) CSS

(2) Which of the following is not an important point to be considered for developing a good website as part of planning process ?

(a) Purpose (b) Audience (c) Content (d) Input

(3) Which of the following information should a website contain ?

(a) Complete, Relevant (b) Complete, Irrelevant

(c) Incomplete, Irrelevant (d) Incomplete, Relevant

(4) Which of the following content provides an overview of the site, organization, products and services, and other items ?

(a) Detailed (b) Long (c) General (d) Short

(5) Which of the following is a collection of interlinked web pages ?

(a) Webpage (b) Form (c) Kompozer (d) Website

(6) Which of the following is the first page that opens when the user enters the URL address in the address bar of the browser ?

(a) Home page (b) Last page (c) Web page (d) First page

(7) Which of the following filename is the home page of the website saved as ?

(a) first.html (b) index.html (c) home.html (d) one.html

(8) Which of the following is a variable that is stored on the user's computer ?

(a) Integer (b) HTML (c) Cookie (d) Java

(9) Which of the following stands for FTP ?

(a) File Truncate Protocol (b) File Transfer Process

(c) Fine Tune Protocol (d) File Transfer Protocol

LABORATORY EXERCISE

1. Using Kompozer create a simple website on your school.
2. Using Kompozer create a website on the theme "Garvi Gujarat".
3. Using Kompozer create a website on "Gujarat Tourism".
4. Using Kompozer create Ecommerce website on "Sports Shop".

Introduction to E-Commerce 4

We are living in the era of information. Information is available from sources like radio, television, newspaper and Internet. The availability of Internet and mobile devices has changed the way we used to access information. People have started using Internet almost every hour of the day with mobile devices. Earlier websites were used mainly to disseminate information about the product or organization. For example, website of an educational institute provides information regarding various courses, course content, students and its faculties. Similarly, business organizations provide information regarding their products, features of the product, their suppliers, how to order the products and so on.

Last few decades, due to tremendous development in telecommunication infrastructure and software technologies, Internet has become popular among the individuals as well business organizations. Internet has revolutionized the way business is conducted nowadays. Today, Internet is being used by people for various purposes like paying bills, banking and even shopping. Business organizations conduct activities like marketing of the product, selling of products, providing catalogue, trading of the stocks and customer service. The use of Internet for conducting such business activities is known as E-commerce.

E-commerce can also be defined as buying and selling of products, services and information using electronic media like Internet. It is a modern business methodology that addresses to the needs of organizations, merchants and consumers in cutting costs while improving the quality of goods and services. It also increases the speed of delivery. It allows people to rise above the barriers of time and distance and take the benefit of global markets and business opportunities.

Selling the products through websites is the fastest growing method of trading worldwide. Many different types of products and services such as books, electronic gadgets, cars, holiday packages and many more are traded online. Many businesses have now set up their own websites for E-commerce related activities. E-commerce creates a whole world of global village from where anyone can buy anything, anytime and from anywhere.

Applications of E-commerce

The major areas of business and commercial activities where E-commerce is widely used today includes trading of goods which comprises of marketing and selling, auctions of goods, financial services like banking and insurance. Let us discuss these areas briefly. E-commerce is not limited to the areas discussed here but spreads across many other developing areas for doing business and other activities.

Internet bookshops

This was one of the first applications of E-commerce on Internet. Customers prefer purchasing books on Internet as they do not require to be physically checked and can be easily described. Books can be easily shipped to the customers place little or no damage to them. Online bookstores need

a good website displaying all the books category wise, picture of the cover page, description of the books including the number of pages, price of the book, discounts and reviews of other customers. The search of a book can be made by title of the book, author's name or publication name. The online bookstores may also record the customer's interests and inform them about new arrivals to attract and retain their interest. Figure 4.1 shows the home page of one of the first online bookstore www.amazon.com.

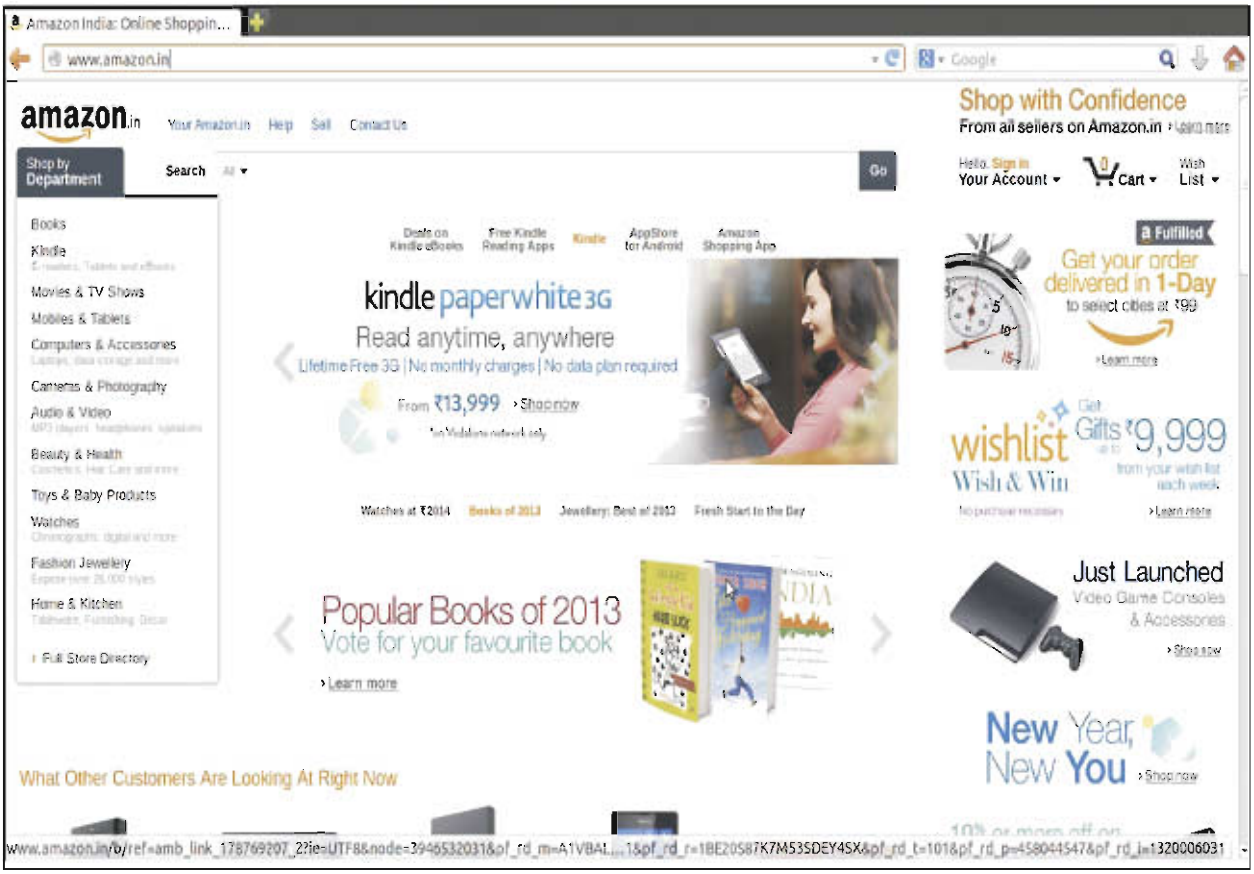


Figure 4.1 : Online bookstore

Some of the large online bookstore websites are as mentioned :

- www.amazon.com
- shopping.indiatimes.com
- www.buybooksindia.com
- www.bookshopofindia.com

Electronic newspaper

An electronic newspaper also known as E-newspaper is a newspaper that exists on the Internet in digital form. It has advantages over the printed newspaper and the news broadcasted on television and radios. It can give us up-to-date news on the issues that are happening worldwide. With the advent of digital technologies, the browser can be set to select the news as per the interest of the reader. It also removes the hassle of printing process and further help in reducing the costs. Majority of the leading newspaper now provide E-newspaper to the readers. Figure 4.2 shows the home page of E-newspaper Washington Post.

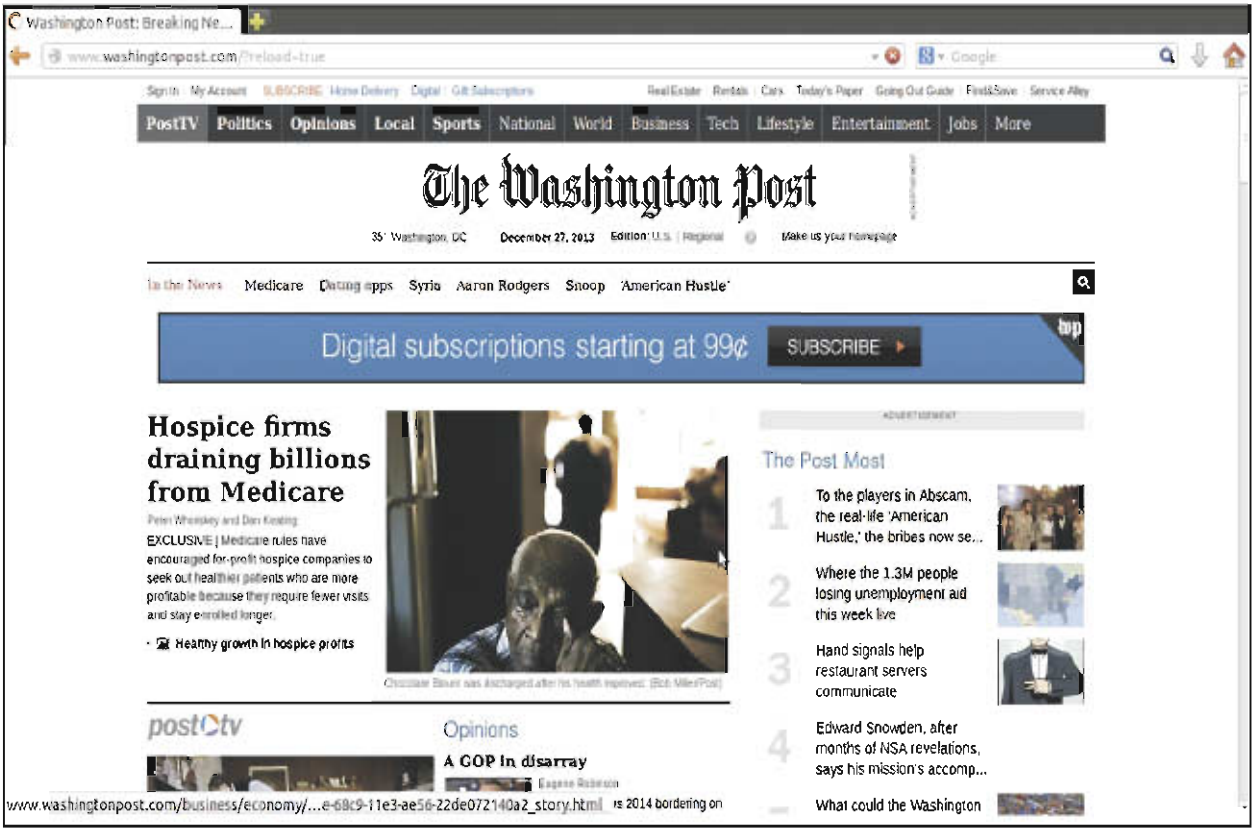


Figure 4.2 : E-newspaper

Online Auctions

An auction is the process of buying and selling products or services by offering the customers to bid the price and selling the product to the highest bidder. The traditional auctions have limited participation of people. Today, the same auction mechanism can be implemented using E-commerce technologies which allow the people to bid on Internet. This is known as Online Auction. There are many websites providing live auctions of goods. These websites provide the platform for both the seller and the bidders. When you place a product for auction on these sites, you are a seller. At the same time you can also bid for a product which is placed by other sellers on the site, in this case you are a bidder.

To sell an item through an online auction site, you need to first register with the site. This is required to track the items you sell or bid on, determine the winning bids and build a database of seller and bidder feedback. Members are also required to provide their basic contact information before they are allowed to sell. Then, the member must go through the site's steps to sell each item that they want to put up for auction. The member generally puts a digital picture of the item and writes a brief description about the item.

By placing goods for auction, a person can get a good price through bidding and once the target price is reached or the time limit is over, the item is transferred to the bidder. Various payment options are also provided to the bidder. The sellers can get the advantage of getting best price for their products and the bidders save the time to get the product of their choice. Figure 4.3 shows the home page of online auction site www.ebay.com.

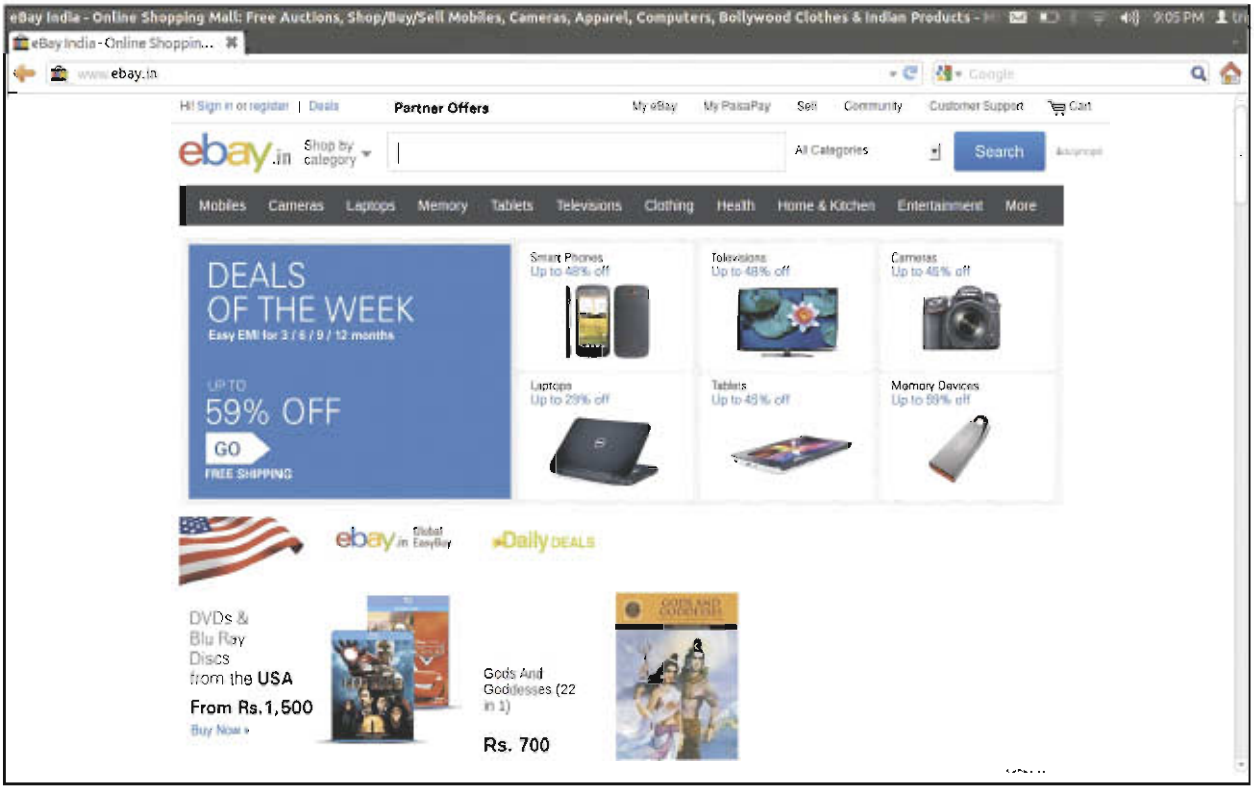


Figure 4.3 : Online auction site

Some popular websites for online auctions are listed below :

- www.ebay.com
- www.onlineauction.com
- www.mybids.in
- www.ubid.com

Marketing and Selling

Many companies now-a-days conduct their business of marketing and selling of goods and services by their websites. They provide their product catalogue online over Internet for better marketing. The catalogue displays different categories of products with images, videos in some cases, brief description and features of the product. The customers can view the catalogue and select the products of their choice by adding them to the shopping cart. An online shopping cart is similar to the original store shopping cart. At the store, the customer selects the products and puts it in the shopping cart. When finished, the final billing is done for all the products in the shopping cart purchased by the customer. Similarly, in online shopping cart the customer can add the product, review what he/she has selected, make necessary modification or addition and finally order the products while checking out. The user needs to provide shipping detail. The payment of the purchase can also be made through Internet.

Today you can purchase groceries, toys, computer accessories, mobiles and many more products from Internet. Some websites provides holiday packages along with airline tickets. We can easily buy an airline or railway tickets online. For example, the website of Indian railways www.irctc.co.in gives all the information related to various trains and provides facilities for booking tickets and payment for it online. Once the ticket is booked the E-copy of the ticket is send on your mail as well as sent as an SMS on your mobile. Figure 4.4 shows the online booking of railway ticket.

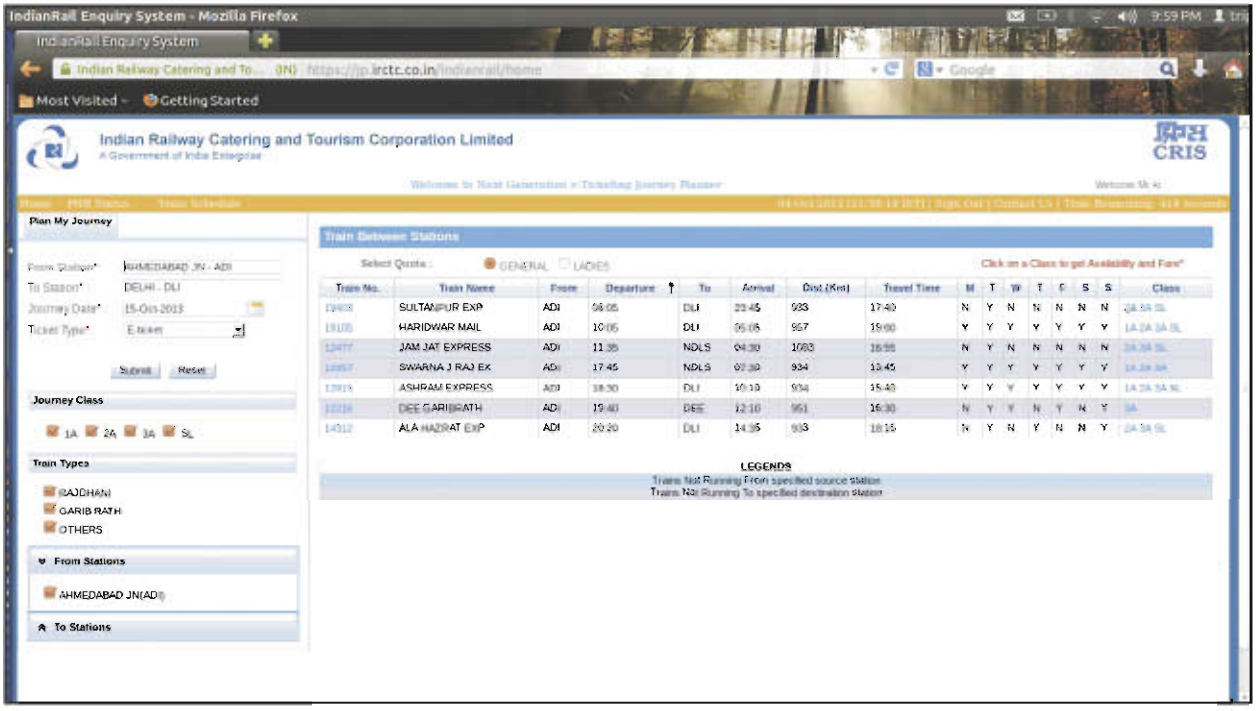


Figure 4.4 : Online railway ticket booking facility

Some popular sites used for marketing and selling are :

- www.homeshop18.com
- www.flipkart.com
- www.myntra.com
- www.makemytrip.com

Online billing

In online billing, companies send their bills to customers through E-mail. Once the customer receives the bills, he/she can pay online on the company's website using credit card or net banking facility. Companies who need to send the bills to a large number of customers periodically can use these facilities. For example, BSNL sends it customers online bills and the customers can also do online payment using Internet.

Information services

Many organizations use the Internet to provide latest information to their users or members. This includes educational institutes and universities, which provides the examination results, online enrolment forms, examination schedule and seating arrangement and important notices. Students can view their results using the website from anywhere. Another example of information services is the notices and reminders sent to the customer by the companies or banks.

Many organizations provide various forms to be downloaded so that the customer can easily download and use it. Companies can also get the information or details about the customers by filling the forms online.

Support services

Support services have become increasingly important due to the huge technological changes that have taken place in the last decade. Today, even the simplest products utilise sophisticated electronics

which requires specialised knowledge and technical ability for maintenance and support when any problem arises. After selling the products, companies are providing online support to the customers. For example, a company selling electronic product provides online complaint registration to their customers, which is then forwarded to the support engineer. Customers can also track the status of their complaint placed online. Software companies provide online support to their customers for any problem in installation, configuration or use. Software vendors also allow their licensed customers to download the recent updates of the software. Hardware vendors put software drivers for their devices so that the customer can download them by choosing proper product type and model.

Net Banking

Net banking or electronic banking is getting more popular day by day. Sometimes a customer may want to make an urgent payment or check his account balance without visiting the bank due to some problems. Online banking can help the customers solve these problems. Online banking is the process of conducting the banking transactions over the Internet. Today majority of the leading banks have started providing online banking facility to its customers. With the help of online banking customers can avail the following services :

- Check account balance at any time.
- Transfer the money from one account to other.
- Obtain statements for any credit or debit.
- Find status of transactions.
- Pay various bills online like telephone, electricity and many more without going to the bank.

The customer is provided a password for online banking services with which he logs in to the bank site and performs all the banking activities from his computer or mobile. Figure 4.5 shows the home page of State Bank of India website. The URL of the same is www.onlinesbi.com.

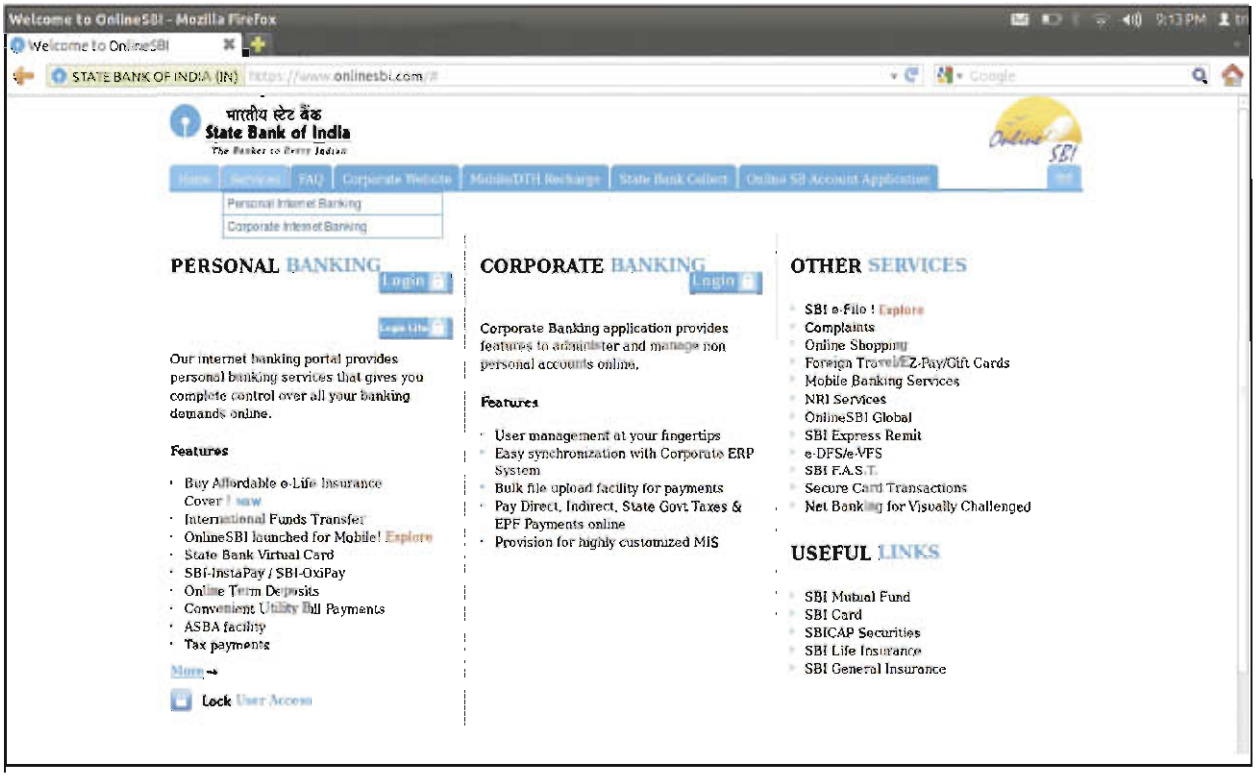


Figure 4.5 : Online banking facility

Most of the banks today provide online banking; a list of few of them is given below :

- www.centralbankofindia.co.in
- www.bankofbaroda.co.in
- www.iob.in
- www.pnbindia.in
- www.denabank.co.in

Traditional commerce vs. E-commerce

Due to the growth of Internet, the nature of competition in traditional way of doing business and using E-commerce has changed significantly. In traditional commerce, the businesses have to compete within a single industry and limited geographical area. In many cases business processes use traditional commerce activities very effectively and they cannot be improved through technology. The products that customers prefer to touch, smell or examine precisely are difficult to sell using E-commerce. For example, buyers will hesitate in buying perishable food items, expensive jewellery and high-fashion cloth as they cannot be examined closely before purchase. In traditional commerce, the retail merchants with their years of experience create the store design that helps to convince the customer to purchase a product. This arrangement of products, store design and layout is called merchandising. The merchant develops the skills to identify customer needs and find products and services that meet those needs. But this art of merchandising can be difficult to implement over Internet. Some of the common features of traditional commerce are :

- Operates within a certain period of time or during business hours.
- No sharing of information with competitors.
- Hiring of sales persons, sales manager and many more.
- Location renting or purchasing, advertising, inventory, shipping of products.

However, in today's fast paced world it is very important to break through these conventional rules. We need to adapt to the new ways of doing business worldwide using Internet. Some of the features of E-commerce are :

- Advertising of the product is done electronically.
- Customers can browse through products catalogue and available offers.
- E-payments systems are used for receiving payment. However, in many cases payment on delivery option is also available.
- Goods are delivered to the customer within few days.
- Reduces the per transaction cost.
- Reduces the time taken to perform an overall transaction.

E-commerce in India

The Internet users in India are growing day by day. E-commerce in India is experiencing a remarkable growth and successfully changing the way people transact. From high end designer apparel to second hand books, everything is now available to Indian online buyers at the click of a button. It has opened up new horizons and opportunities for millions of people.

A greater variety of goods and services that can be bought over the internet is making online buying more attractive and convenient. As internet has reached the rural parts of India, a drastic change is observed in their lifestyles, the quantity and quality of consumption is helping change mindsets in the rural areas.

Many factors are playing role for the growth of E-commerce in India. Some of the key factors are listed below :

- The growth of technology facilitators such as Internet connections, broadband and third generation (3G) services, laptops, Smartphones, tablets and dongles.
- Increase in use of mobiles devices.
- Availability of much wider product range.
- Busy lifestyles, traffic congestion and lack of time for traditional shopping.
- Lower prices compared to brick and mortar retail driven by disintermediation and reduced inventory and real estate costs
- Increased usage of online classified sites, with more consumers buying and selling second-hand goods.
- Evolution of the online marketplace model with sites like eBay, Flipkart, Snapdeal and many more.

Though the penetration of E-commerce is low in India as compared to other countries, but it is growing at a much faster rate with a large number of new entrants. There are a lot of Online stores in India which are becoming quite popular like Flipkart, eBay India, Snapdeal, Amazon India, Myntra, Dominos, PayTM, Jabong and many more.

Retailers have also started offering a cash-on-delivery option, it is the most preferred payment method in India. Almost 80% of Indian E-commerce business uses cash-on-delivery mechanism. Though, from the online retailer point-of-view this is an expensive proposition, as they have to finance the sale till the order is delivered. For the consumer, cash-on-delivery makes it easy to reject products at the point of delivery.

Industry experts are optimistic about the future potential of E-commerce industry in India. Though there are challenges to be faced, they can be easily overcome with well-planned and pre-decided strategies which will give a boost to the growth of E-commerce in India.

Advantages of E-commerce

With the growth of Internet, companies are finding new and exciting ways to expand their business. Today we can hardly find any successful company which do not use computers in their everyday business activities. E-commerce provides multiple benefits to the customers in the form of availability of goods at lower cost, wider choice and saves time. E-commerce sales would rise in the coming years with the availability of faster broadband services and new applications. Some advantages of E-commerce are listed below :

- **Conduct business 24×7**

  Using E-commerce a business can operate anywhere anytime. The business activities do not remain time bound. A customer can purchase a product any hour of the day; hence an order can also be accepted at any hour of the day.

- **Lower cost**

  Using Internet for business activities can reduce the cost of marketing, distribution, phone, postage, printing and many other such activities. In E-commerce requirement of physical store space or infrastructure investment is minimal. The cost of opening a physical store is very high as compared to doing business on Internet. Customers are also benefited by getting a wide range of products for selection at lower cost due to competition amongst the vendors and elimination of distributors or intermediaries for selling the products.

- **No boundaries or geographical limitations**

  A physical store is limited by its geographical area of service. With E-commerce, businesses can reach out to millions of customers in an instant which is not possible in any conventional mode of marketing. It has the ability to reach potential buyers easily further creating new markets. Global coverage can be reached through creating a website and uploading it on Internet.

- **Improved and better customer service**

  As there is a direct communication with the customer, it is possible to solve their queries related to the price, quality, features of the product and others thus resulting in a better customer service. For companies that do business with other companies, adding customer service online is a competitive advantage. Online customer service makes customers happier. E-commerce gives better and quicker customer service. Information can be shared easily on Internet. A merchant can email its customers about the new product and can also solve the product related problems.

- **Teamwork**

  E-commerce helps organizations to work together. One good example is the use of e-mail which helps people to exchange information. It has changed the way organizations interact with suppliers, vendors, business partners, and customers. More the interaction better is the result.

- **Eliminate Travel Time and Cost**

  The customers do not have to travel long distances to reach their preferred physical store. E-commerce allows them to visit the same store virtually, with a few mouse clicks.

- **Speed**

  Doing business electronically is much faster than using the traditional methods. The speed of business transactions increases significantly as delays for communications is avoided by conducting transactions online. Business organizations can generate purchase orders and send it to suppliers online without delay. Suppliers can receive and process the order quickly. This also helps in reducing the shipping time. Information appearing on the Internet can be changed rapidly. This gives business organizations the ability to inform the customers of any changes in the product or services offered by the company.

E-commerce also provides the following benefits to the society :

- Buy from home, office or any place.
- Less travelling to purchase a product which further reduces pollution and traffic.
- Health care services.
- Distance learning and education.

Limitations of E-commerce

Although E-commerce has a list of advantages, it also has some limitations and disadvantages. However, with passage of time many of the limitations may disappear. Following are some of the limitations of E-commerce.

- **Resistance to change**

  Business organizations need to change from traditional business to E-commerce which is faced by a lot of resistance. People need to get used to paper-less and faceless business transactions.

- **Initial Cost**

  E-commerce requires significant initial investment in new technologies which can change the company's core business processes. It also requires investment in hardware and software technologies. The staff training is needed to understand the way of conducting the business using E-commerce.

- **Security**

  A primary concern in E-commerce is security. The data or information travelling on the Internet related to the company or an individual should be protected by unauthorised access. Internet provides universal access but companies should protect their assets from malicious or accidental misuse.

- **Privacy**

  The privacy of the customer's information is a serious issue. Customers hesitate to share their personal information on the Internet due to fear of misuse. Many times, the companies sell their database information to marketing companies and they in turn send unwanted (spam) mails to the customers. The hackers also might intercept the information on the Internet and misuse the data. Credit card frauds result into financial loss to the customer. The protection should be provided from hackers, viruses, transfer of data and transaction risks.

- **Lack of trust**

  Sometimes, business frauds like non-delivery of products, incorrect information of the product, lack of security for payment transactions creates dissatisfaction amongst the customers. Also returning defective goods purchased online can be difficult issue. The common concerns here are who pays the return postage, will full refund be made by the merchant and whether the product will reach to their original source.

- **Time for delivery of products**

  In a physical store, we select the product and leave with the product in our hands. In E-commerce we often buy products that are not available locally, which means the product needs to be delivered which takes time and cost money. The cost of shipping charges is an overhead in case the product is ordered from a distant place.

- Perishable products like fruits, vegetables and others are not preferred to be purchased online. Online store doesn't provide the opportunity to touch, wear or feel the product. Therefore online selling of such products like clothes and furniture might be tricky.

- Payments on E-commerce are most often conducted using credit card facilities and as a result very small and very large transactions are not preferred to be conducted online. Timely updates

of information and services are often a problem in case of small organizations trying to establish their business online. E-commerce is dependent on Internet and any problem in the Internet connectivity causes business risk.

E-commerce Business Models

The business models of E-commerce are defined based on the parties involved and the type of business activities or services provided. Many business models have been proposed for E-commerce, but the following are the most popular amongst all. Generally, the classification is done on the basis of who is selling to whom.

- Business to Consumer (B2C)
- Business to Business (B2B)
- Consumer to Consumer (C2C)
- Consumer to Business (C2B)

Let us understand each of them in detail.

Business to Consumer

Business to Consumer (B2C) refers to business and organizations that sells products or services to consumers over the Internet using websites. Consumers from anywhere can browse and order the products or services anytime. The sellers can sell products directly to the consumers. And the buyers are individual customers. This is the type of E-commerce that the consumers are most likely to see on Internet. Lately, it has gained a high popularity amongst consumers due to simplified and fast way to buy products. Apart from online retailing, B2C also includes services like online banking, real estate services, travel services and many more. Some examples of B2C websites are: amazon.com, rediff.com, fabmart.com, flipkart.com etc. Figure 4.6 shows the home page of Flipkart website. It is a popular website for purchasing variety of goods.

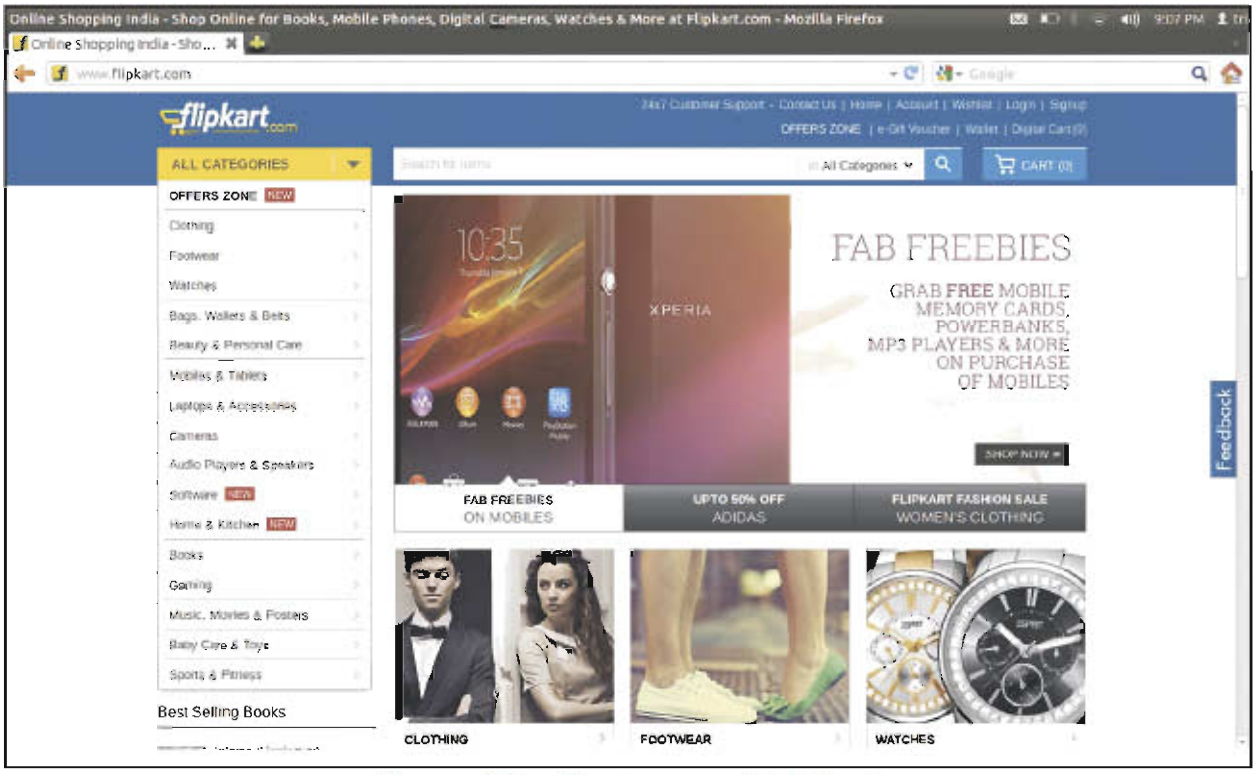


Figure 4.6 : Home page of Flipkart

As seen in the figure 4.6, the products are displayed category wise on the left side of the window. Select the category as per your choice. Here we have selected the eBooks category as shown in the figure 4.7. This will display all the eBooks in the catalog.

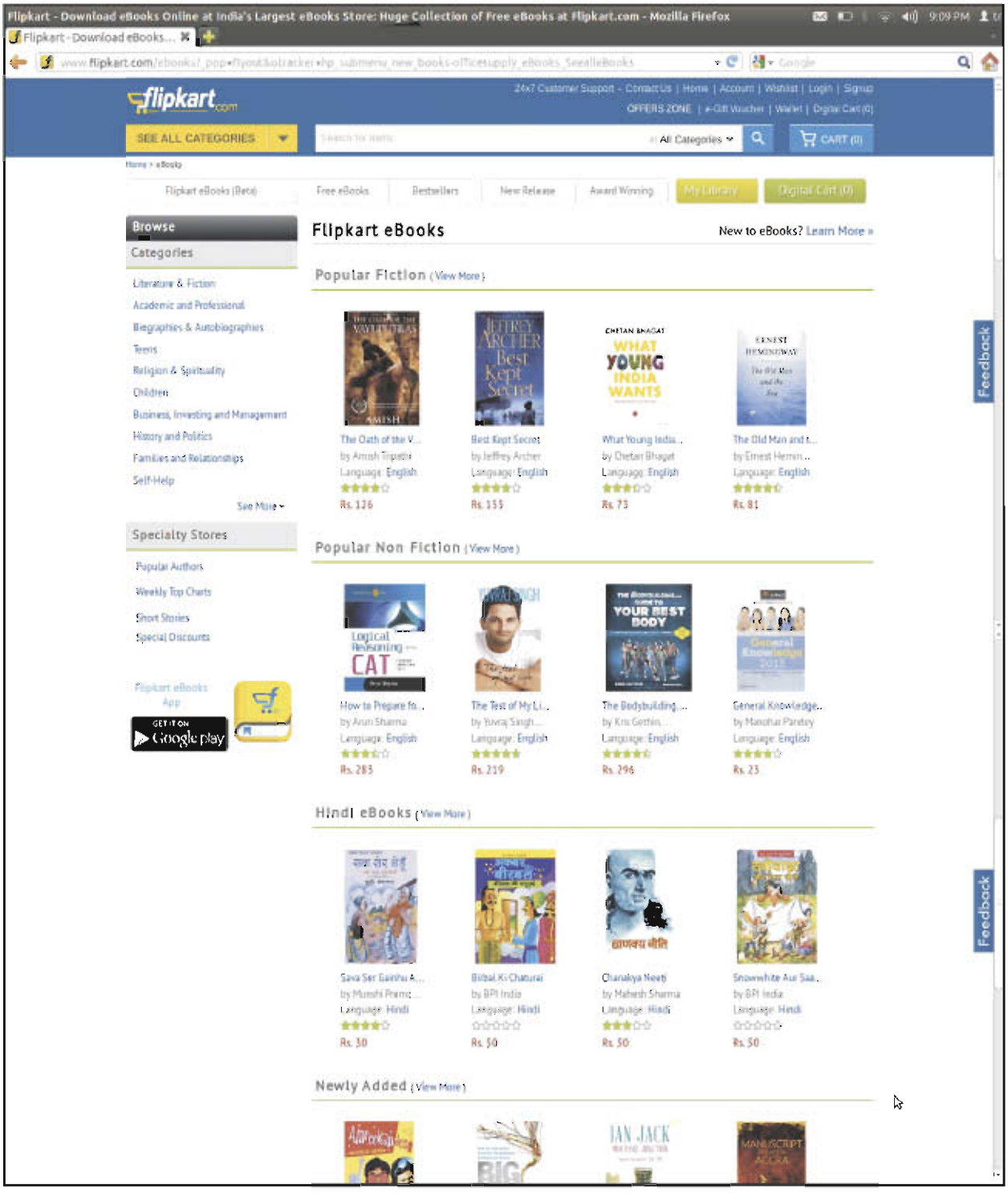


Figure 4.7 : Selecting eBooks category in Flipkart.com

When we click on icon of any of the eBook it shows the details like authors name, price, readers rating, book review, publisher and ISBN number as shown in the figure 4.8. We can also search a particular book from the different categories using the search tool provided at the topmost portion of the window.

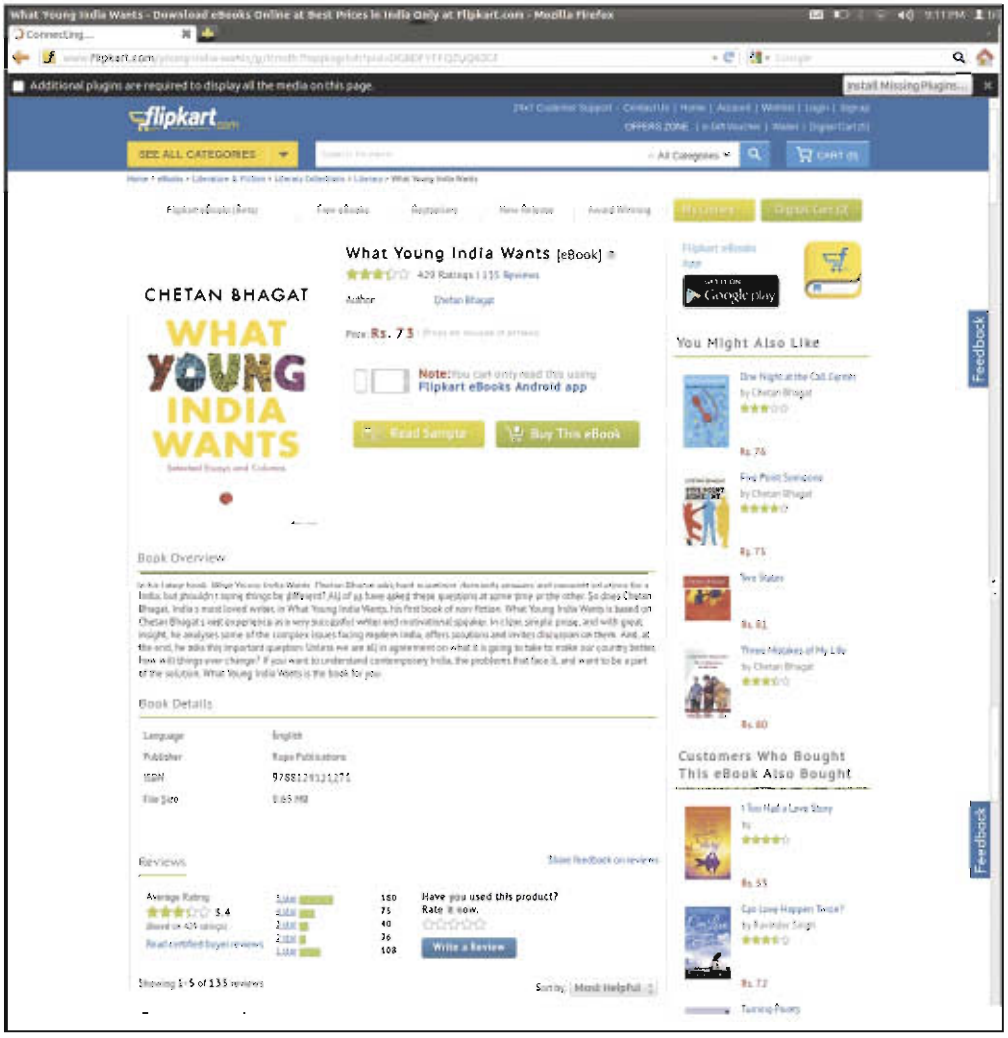


Figure 4.8 : Selecting an eBook

Pressing the button "Buy this eBook" adds the book in the shopping cart as shown in the figure 4.9.

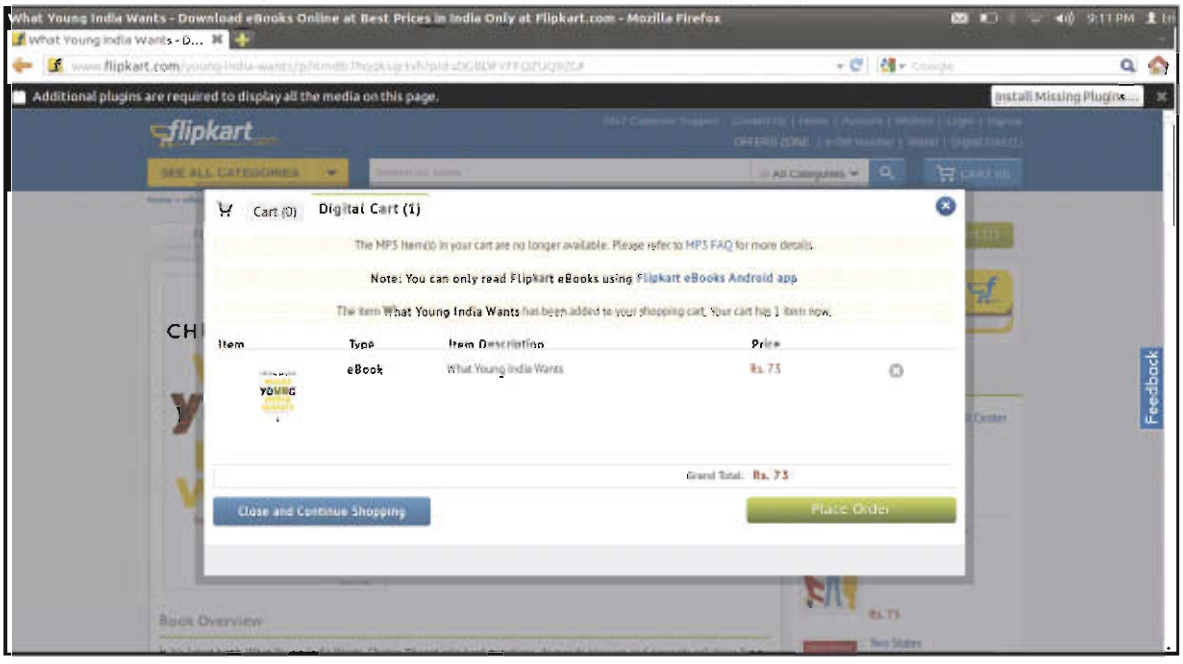


Figure 4.9 : Shopping cart showing the selected eBook

In similar way, you can add any number of products to the shopping cart from various categories you want to purchase. You can also delete a selected product; change the quantity of the product in your shopping cart.

To confirm the order, press the button "Place Order" as shown in figure 4.9. This will take you to the final screen where you are asked to create a login in case you have not created; provide the billing address, shows order summary and payment options. Once all this process is over, the purchase order is made and the eBooks are shipped to your address provided in the shipping details. Now-a-days you also have an option of "Payment on Delivery" option where you can pay for the product purchased when it is delivered at your address.

Business to Business

Business to Business (B2B) refers to E-commerce activities between different business partners. In B2B both the seller and the buyer are business entities. It enables the businesses to form E-relationship with their suppliers, distributors and other agents. This allows more transparency among the business entities involved and results in more efficiency. For example, a manufacturers deals with a supplier, a distributor and a wholesaler. The manufacturer using B2B can quickly communicate to the supplier about diminishing stock product, the supplier in turn can respond quickly to it.

With the help of B2B, the companies can improve the efficiency of common business activities like supplier management, inventory management, payment management and many more. It is an effective media for managing telemarketing, supply chain, procurement, just-in-time delivery, online services. Some examples of B2B websites are commodity.com and tradeindia.com.

Consumer to Consumer

Consumer to Consumer (C2C) refers to E-commerce activities involving transactions between and among the consumers. It enables the consumers to directly deal with each other through online auctions and classified advertisements without the involvement of third party. Any Internet user can become a vendor or purchaser at C2C websites. Auctions sites are a good example of C2C model. If we have any product to sell, we can get it listed at an auction site and others can bid for it. Some example of C2C websites are Ebay.com, OLX.com and Quikr.com

Consumer to Business

Consumer to Business (C2B) involves reverse auctions where the consumers determine the prices of the products or services. In this type of E-commerce, consumers have a choice of a wide range of products and services, along with the opportunity to specify the range of the prices that they can afford or are willing to pay for a particular product or service. The companies bid to offer the products or services to the consumer. This helps in reducing the bargaining time while increasing flexibility for both the consumer and the company. C2B uses Internet to reverse the normal buying process wherein the consumers decide what they are willing to pay and business decides whether to accept it or not. Some examples of C2B websites are bidstall.com, JeetLe.in

There are other variations to E-commerce business models. If we consider Government as a separate entity, then we can have :

- Government to Business (G2B)
- Government to Citizen (G2C)
- Government to Government (G2G)

Government to Business

Government to Business (G2B) refers to the services and information provided by the government to the business organizations through vast network of government websites. A business organization can get all the information related to business policies, approvals for starting the business, setup requirement and other specifications from these websites. Various forms can be filled and submitted online to the related government office. For example, income tax department of Government of India has their website www.incometaxindia.gov.in wherein all the rules related to tax; different forms and facility for submission of online tax returns are provided. This business model is a part of E-governance initiative.

Government to Citizen

Government to Citizen (G2C) is also a part of E-governance. The objective of this business model is to provide good and effective services to individual citizen. The website provides information regarding various government departments, various welfare schemes, different application forms to be used by the citizens etc. The Gujarat Government has developed its own network called Gujarat State Wide Area Network (GSWAN) for this purpose and is available online on www.gswan.gov.in. Figure 4.10 shows the home page of GSWAN.

Figure 4.10 : Home page of GSWAN website

Government to Government

Government to Government (G2G) refers to online non-commercial communication between the Government agencies, organizations and departments with other Government agencies, organizations and departments. The sharing of information helps in reducing IT costs, streamline procedures and government offices can be more efficient.

From all the E-commerce models described above, B2C and B2B are the two most widely used models. The major difference between the two models is the consumer. In B2B model, the consumers are other companies while in B2C model the consumers are individuals.

Summary

In this chapter we discussed about E-commerce, its advantages and disadvantages, applications of E-commerce and its business models. E-commerce can be defines as the use of Internet for conducting business activities. E-commerce is used for marketing and selling, purchasing of products, online newspapers, information services, support services, net banking and many more. The business models of E-commerce are classified based on the parties involved and the type of business activities or services provided. We studied various models like, Business to Consumer (B2C), Business to Business (B2B), Consumer to Consumer (C2C), Consumer to Business (C2B) and Government to Business (G2B).

EXERCISE

1. Define E-commerce. List some of the applications of E-commerce.
2. What is online auction?
3. What is electronic newspaper? List some of the features of the website.
4. List some online bookstores and discuss the features of any one.
5. What is net banking? Name some of the websites for online banking.
6. State the difference between traditional commerce and E-commerce.
7. State the advantages of E-commerce.
8. Write about the following E-commerce business models. Also give example for each.
   (1) Business to Consumer (B2C) (2) Business to Business (B2B)
   (3) Consumer to Consumer (C2C) (4) Consumer to Business (C2B)
   (5) Government to Business (G2B)
9. Choose the most appropriate option from those given below :
   (1) Which of the following is an example for online bookstore ?
   (a) Amazon (b) irctc (c) Gmail (d) yahoo
   (2) Which of the following is newspaper that exists on the Internet in digital form ?
   (a) l-newspaper (b) Internet-newspaper
   (c) www-newspaper (d) E-newspaper

(3) Which of the following is the process of buying and selling products by offering the customers to bid the price ?

(a) Marketing (b) Auction (c) Bookshop (d) Booking

(4) Which of the following is known as the process of conducting the banking transactions over the Internet ?

(a) Auction (b) Bidding (c) Net banking (d) www-banking

(5) Which of the following is a feature of traditional commerce ?

(a) Operates within a certain period of time or during business hours.

(b) Advertising of the product is done electronically.

(c) E-payments systems are used for receiving payment.

(d) Customers can browse through products and offers.

(6) Which of the following is a feature of E-commerce ?

(a) Operates within a certain period of time or during business hours.

(b) No sharing of information with competitors.

(c) Location renting or purchasing.

(d) Advertising of the product is done electronically.

(7) Which of the following is not an advantage of E-commerce ?

(a) Lower cost (b) Conduct business 24×7

(c) Security (d) No geographical limitations

(8) Which of the following is a disadvantage of E-commerce ?

(a) Privacy (b) Improved customer service

(c) Speed (d) Conduct business 24×7

(9) Which of the following E-commerce business model refers to business and organizations that sell products or services to consumers over the Internet using websites ?

(a) Business to Consumer (B2C) (b) Business to Business (B2B)

(c) Consumer to Business (C2B) (d) Government to Business (G2B)

(10) Which of the following E-commerce business model refers to activities between different business partners ?

(a) Government to Business (G2B) (b) Consumer to Business (C2B)

(c) Business to Business (B2B) (d) Business to Consumer (B2C)

(11) Which of the following is a good example of C2C model ?

(a) Auction sites (b) E-newspaper

(c) Online purchasing (d) Information services

(12) Which of the following E-commerce business model refers to E-commerce activities involving transactions between and among the consumers ?

(a) Government to Business (G2B) (b) Consumer to Consumer (C2C)

(c) Business to Business (B2B) (d) Business to Consumer (B2C)

(13) Which of the following E-commerce business model involves reverse auctions where the consumers determine the prices of the products or services ?

(a) Consumer to Business (C2B) (b) Business to Business (B2B)

(c) Consumer to Consumer (C2C) (d) Government to Business (G2B)

(14) Which of the following e-commerce business model is also a part of E-governance ?

(a) Business to Business (B2B) (b) Consumer to Business (C2B)

(c) Consumer to Consumer (C2C) (d) Government to Citizen (G2C)

(15) Which of the following E-commerce business model refers to online non-commercial communication between the Government agencies, organizations and departments with other Government agencies, organizations and departments ?

(a) Business to Business (B2B) (b) Consumer to Business (C2B)

(c) Government to Government (G2G) (d) Consumer to Consumer (C2C)

Introduction to M-Commerce 5

Mobile Commerce also known as M-commerce. It refers to buying and selling of goods or services through the use of Internet enabled wireless devices such as a Mobile phone, Personal Digital Assistant (PDAs), Smartphone, Tablet, Palmtop or any other mobile device.

M-commerce provides the user with the advantage of flexibility and ubiquity. By using the mobile phone, consumers can conduct business transactions without being fixed at a computer terminal or being physically present at the shop. These devices are carried by the user wherever he/she goes, making it possible to access the Internet from any place. It allows real time transactions while on the move. As the popularity of smart phone's and tablets is increasing day by day, more users are moving towards the use of M-commerce. Some of the examples of M-commerce are:

- Purchasing airline tickets
- Purchasing movie tickets
- Restaurant booking and reservation
- Hotel booking and reservation
- Stock market analysis

Banks and other financial institutions are increasingly using M-commerce to retain their business. They allow their customers to access account balance, stock quotes, make transactions via mobile phones. This service is known as Mobile Banking or M-Banking. The stock market services offered via mobile devices is also becoming popular and known as Mobile Brokerage. News information, sports, entertainment, shopping and reservation areas have also grown with the demand for mobile related services.

Benefits of M-Commerce

Mobiles are being used more and more on daily basis and today it is not merely used to make or receive a call. Mobile companies are coming up with new features for their smart phones, which offer consumers ease, flexibility and security at the same time. The ease of availability and faster speed has made M-commerce more popular now-a-days. The web design and development companies have also optimized the websites, such that it can be viewed correctly on mobile devices.

M-commerce is the integration of wireless networks accessed through handheld devices and Internet. The benefits of Internet and E-commerce are offered by M-commerce also. Some of the advantages of M-commerce as listed :

- It provides convenience to the user. In just a few clicks on the mobile device, the user can do shopping, banking and download media files while on the move.

- Mobile device enables the user to be contacted at virtually anytime and anywhere.
- Reduces transaction cost.
- Reduces the time to order. The user does not need to be on the PC or laptop to order.
- Streamline business processes.
- Provides global reach.
- Conduct business 24*7.
- Flexibility of accessing the information through any mobile devices.
- Payment can be done using the mobile devices itself just the same way as we do it on personal computer.
- Useful to deliver time critical and emergency information.
- Easily identifies the physical location of the handheld device. The emergence of location based applications enable the user to receive relevant information. We will discuss about location based services later in the chapter.
- Customized alerts can be easily received on the mobile device.
- Instant connectivity and availability of faster 3G services has made M-commerce more popular these days.
- Timely information can reach the user. The information like flight or train schedule, delay or cancellation can be given to the user on his mobile device on real time basis.

Limitations of M-commerce

Though the list of advantages of M-commerce is large it has a number of limitations. Some of the limitations of M-commerce are listed below :

- The handheld devices commonly used today offer a limited screen size. This further limits the types of file and data transfer. At times it is difficult to display videos.
- User interface is less convenient when compared to personal computers.
- Mobile devices have limited computing power, memory and storage capacity.
- It operates over wireless networks which are less secured as compared to wired network.
- It offers a limited bandwidth.
- High cost of establishing mobile and wireless broadband infrastructure.

Applications of M-Commerce

Applications of M-commerce are increasing rapidly. It is used today in many areas not limited to the ones that are discussed here. More and more applications will be introduced in the near future. Let us discuss some of the applications of M-commerce.

Mobile Marketing and Advertising

Today, companies are using M-commerce to expand their services from marketing to advertisement. Mobile advertising is one of the most popular ways for companies to reach a large audience. Advertising on Internet has become a major source of revenue for most of the portals. Many retailers are offering location based mobile advertising in order to target consumers and increase their sales. An advertisement placed on the mobile device of the user can thus be made on personal requirements and location-specific. It can update the users about the various discounts and schemes available in the nearby areas of the current location of the user.

Mobile Ticketing

Users can easily buy tickets for air or rail travel, movies etc. The tickets can be sent to the user's mobile device. Users can further show these tickets on their mobile devices at the respective place. Tickets can also be easily cancelled on the mobile phones using the application or accessing the portals of travel agents. This helps in reducing the traffic and the parking problems which are increasing day-by-day as the user need not travel to the place for buying tickets.

Figure 5.1 shows the Indian Railway Catering And Tourism Corporation limited (IRCTC) login page.

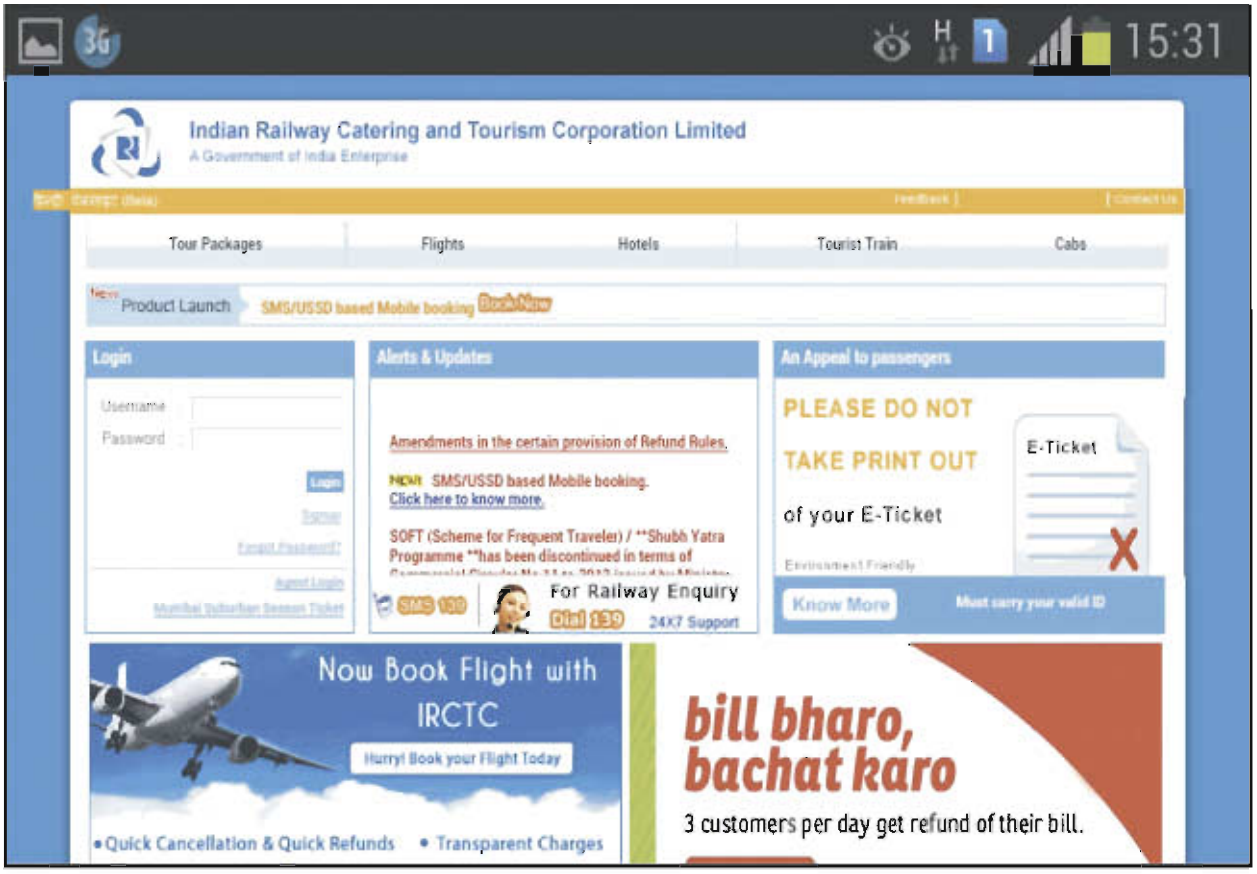


Figure 5.1 : IRCTC login page

Let us try to book an online ticket using the mobile device. After providing the username and password (note: the user already has an account), the user can give his plan travel details like source and destination name, date of travel, ticket type and quota as shown in figure 5.2.

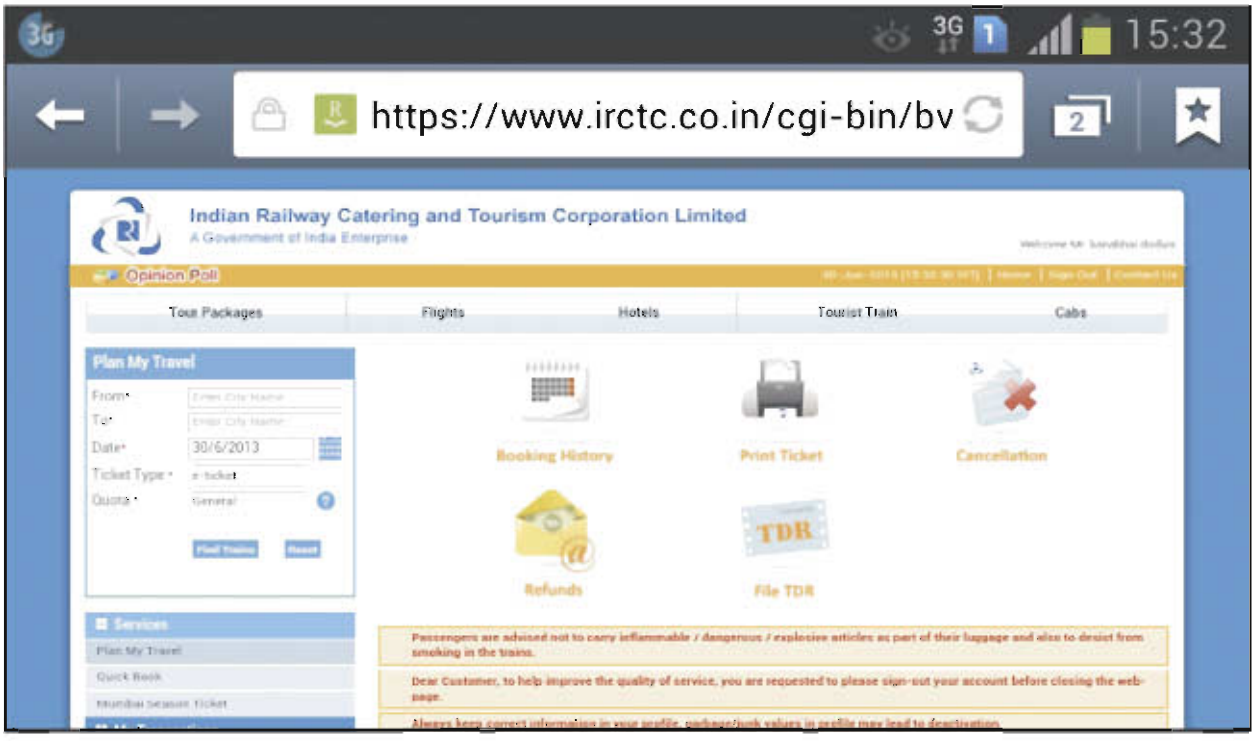


Figure 5.2 : Details of Journey

When the user clicks on Find Trains button he/she is provided with the list of trains available on the given date as per his/her requirement as shown in figure 5.3.

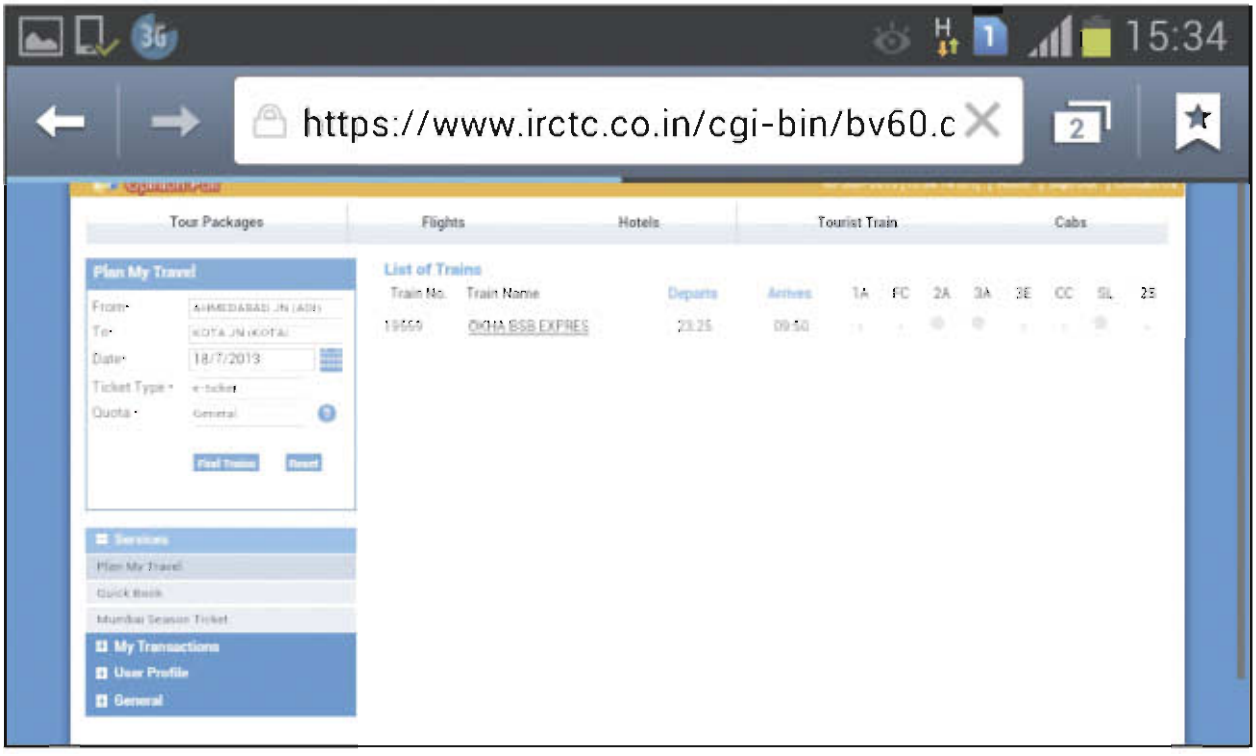


Figure 5.3 : Details of the journey with available train list

The user can click on the name of the train to get more details. In figure 5.3 we can see name of only one train. It is possible to get list of more trains also. Later, he/she can proceed to book the ticket and provide the details of the passengers travelling as shown in figure 5.4.

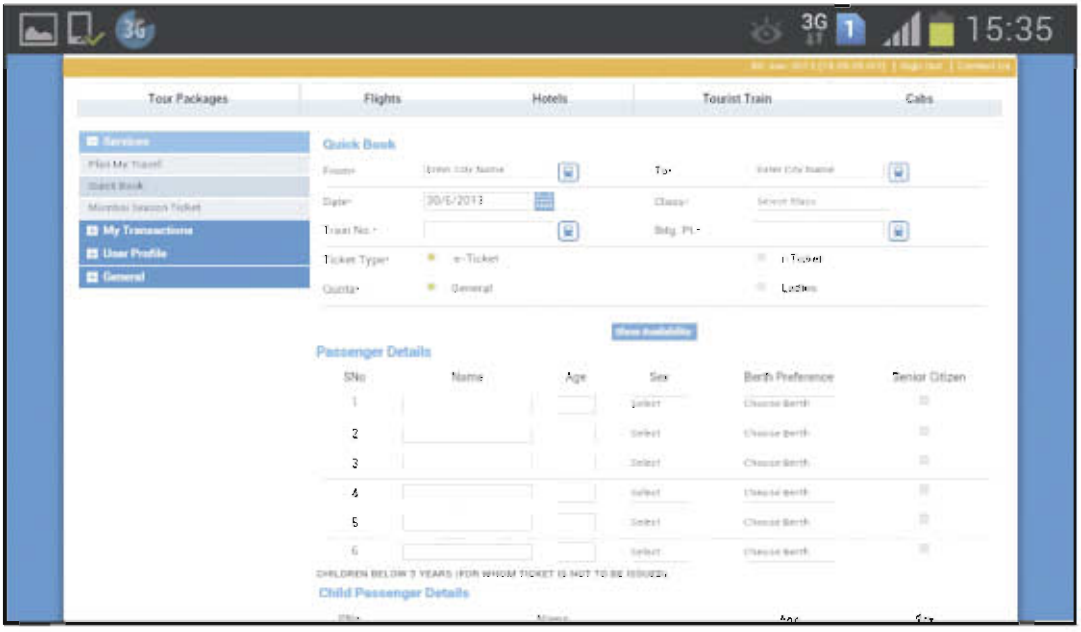


Figure 5.4 : Enter details of passenger for ticket booking

After booking, the user gets an e-ticket message from IRCTC on the mobile device which can be presented at the time of travelling. As you can see, without being fixed at the computer terminal or being physically present at the railway station, the customer can access the services using the mobile device whenever and wherever he/she goes.

Mobile Auctions

The auctions sites are becoming more popular these days. Mobile devices further help in increasing the reach of these auction sites. A user while on the move can access these sites, make a bid, monitor bids and take a timely action on the bidding process. Many of the auction sites have built gateways and interfaces to provide access to mobile devices through wireless networks. Figure 5.5 and 5.6 shows the auction site ebay as viewed on a mobile device. The user can bid for the products using the mobile device.

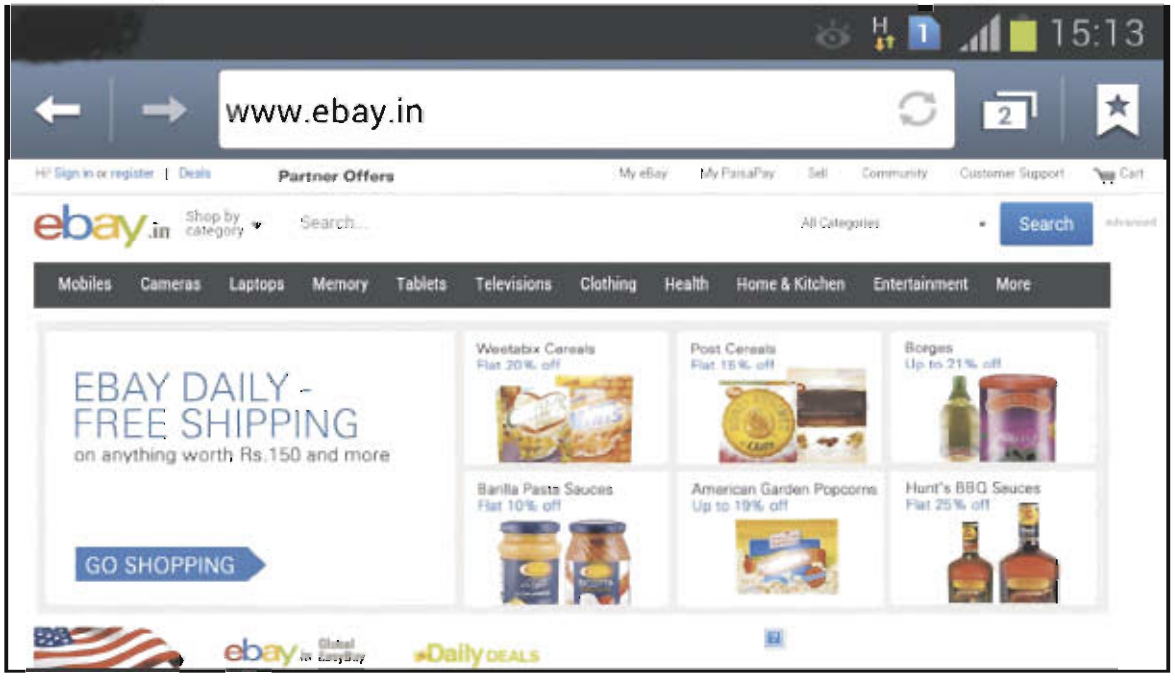


Figure 5.5 : Home page of ebay

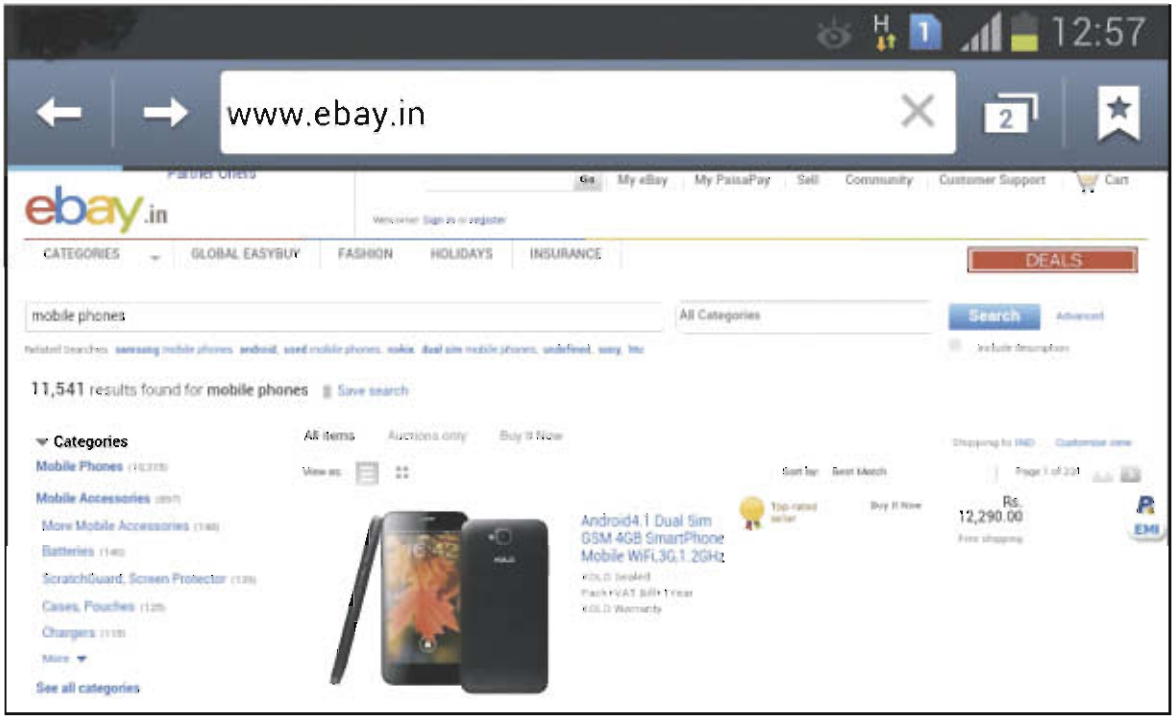


Figure 5.6 : Bid for a product

Mobile Entertainment

Mobile devices are used extensively for listening to audio, viewing video and playing games. The mobile users can subscribe to entertainment libraries where they can search for songs, videos or games and easily download them in their device for playing later. Entertainment services such as pay-per-download, pay-per-event or on subscription basis can cater to a large number of mobile users and are willing to pay for the services.

Mobile Purchase

Mobile purchase allow customers to shop online anytime anywhere. Customers can browse and order products while using a secure payment method. Instead of using paper catalogue, retailer can send a list of products that a customer would be interested in, directly to their mobile device. Alternatively, the consumers can also visit a mobile version of a retailer's E-commerce site. The retailers can also track the customers and notify them of discounts at local stores that the customer would be interested in. Figure 5.7(a) shows the home page of flipkart as seen on mobile device. Figure 5.7 (b to f) shows the purchase process from selecting a category, choosing the product, viewing its detail, placing the order and provide the shipping details on the mobile device.

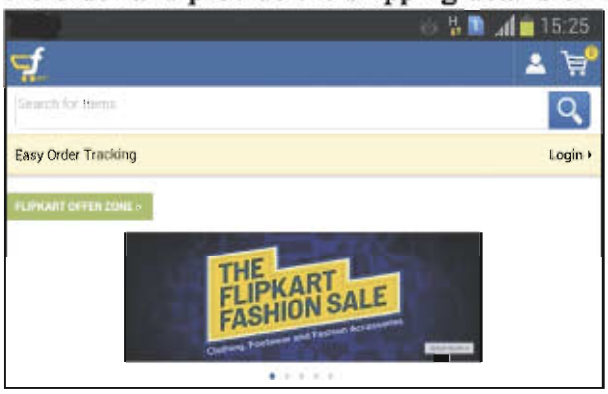


Figure 5.7(a): Flipkart home page

Figure 5.7(b): Select a category

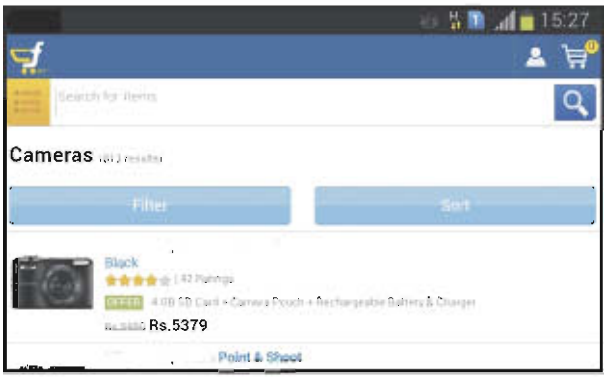


Figure 5.7(c): Products in the category

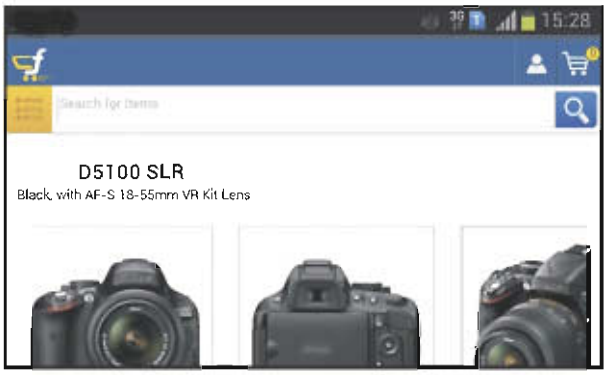


Figure 5.7(d): Select a product

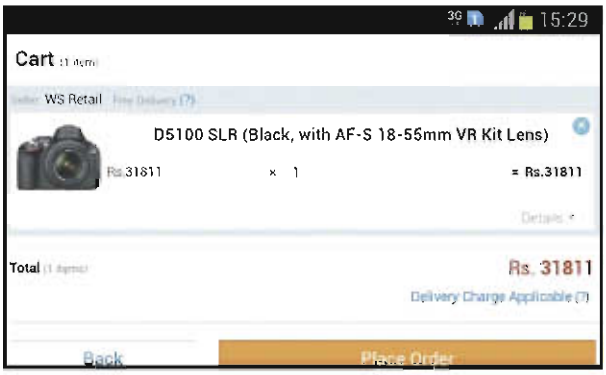


Figure 5.7(e): Place order

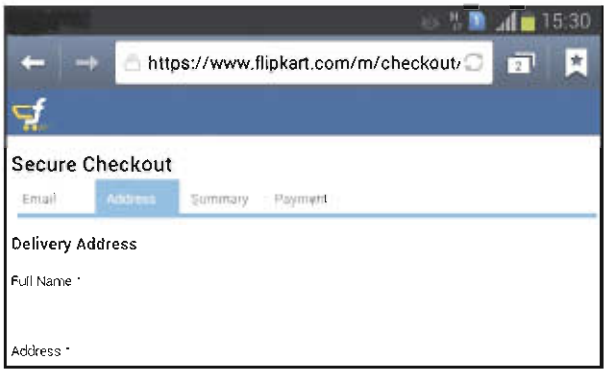


Figure 5.7(f): Provide customer details

Mobile Financial Services

Today, many popular banks and financial institutions use M-commerce. They allow their customers to access account information; perform transactions like stock purchase, remit money, via mobile phones and other mobile equipments. Figure 5.8 shows the mobile services provided by Union bank. The customer can download the application "umobile" from the android market. After getting registered, the customer can avail the services that are provided by the bank on his mobile as shown in figure 5.9. In figure 5.10, you can see the various main menu options. The customer can check his/her balance, transfer the funds to any other bank account, place request for a cheque book and can avail many other services from his/her mobile device.

Figure 5.8 : Mobile Financial services

Figure 5.9: Login using password

Figure 5.10 : List of various services

Mobile Information Services

A wide variety of information services can be delivered to mobile phone users in much the same way as it is delivered to personal computers. The services include :

- News service
- Stock market data
- Sports news
- Financial records
- Traffic information

Location and search service

The location of the mobile phone user is an important piece of information used during mobile commerce transactions. For example, a consumer wanting to buy a Smartphone within a certain price limit and specifications would be interested in knowing the nearest location of the store where he can get the desired product. In this case, it is important that the location and search service should be able to provide the list of stores in the city or nearby areas of the mobile user's current location. Knowing the location of the mobile user allows vendors to provide location based services such as local maps, local offers, local weather, people tracking and monitoring. Mobile devices can also be used to get directions to particular place, movie theatre, restaurant, hospital or other such amenities. Let us learn about location based search which is known as L-commerce.

L-Commerce

With more and more wireless handheld devices like PDA's, Cellular phone and pocket PC's there are significant opportunities for the growth of M-commerce. Although mobile commerce enables access to goods and services regardless of the location of buyer or seller, in many situations the specific location of the buyer and seller is important to the transaction. Today various location-specific applications and services are emerging. These applications track the user's location in order to deliver a service or product. The use of technologies which provide the location information for business purposes is known as L-commerce.

The technology uses the geographical location of the mobile device to determine which applications are appropriate based on that area. It enables users to log their locations, track the location of another person, and find places such as a bank or restaurant. The technology works by using signals

from GPS, cellular and Wi-Fi sources. The Global Positioning System (GPS) is the most accurate in determining a mobile device's position. It is based on a worldwide satellite tracking system where the GPS signals are generated by a group of satellite that orbits around the Earth. To locate a point, a mobile device will utilize three satellites to create an intersecting point that locates the device within 500 meters. This is known as triangulation. If the GPS signal is poor, weak or blocked, the mobile device can use the signals from cell towers and Wi-Fi hot spots. These signals do not broadcast their own locations but Smartphone companies use databases that store the locations of these sources.

Figure 5.11 shows the application which tracks the user's location on the mobile device using GPS. (Note: GPS must be enabled on the device). The advertisements seen on the mobile device in the figure 5.11 are also location specific. In the figure the user's current location is CG road in Ahmedabad. If the user wants to search the restaurants that are close to his current location, he selects the restaurant option in the figure 5.12. This will show a list of restaurants that are near to his/her current location. The user can also specify the distance as shown in figure 5.13 to search within a specific range of distance only. This will further help in filtering the search results as shown in figure 5.14.

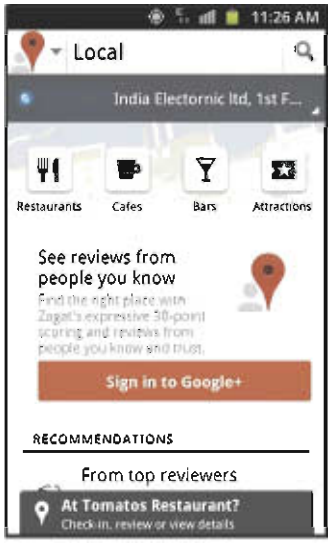


Figure 5.11 : Location specific application

Figure 5.12 : Search nearest restaurants according to user's location

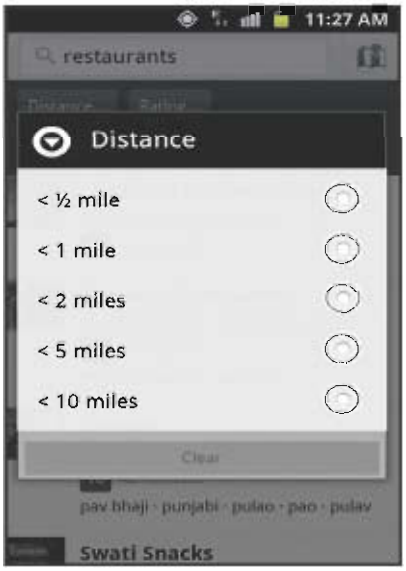


Figure 5.13 : Filtering the search result

Figure 5.14 : Search result after filtering

Let us discuss another example of location based services, where we can use maps to find the path to the destination. For example, the user wants to reach a particular destination whose path is not known to him. Using the maps, the location based services tracks the user's current location as seen in figure 5.15. The current location is indicated using blue color pointer. Now to find the destination we can either type or speak the destination name as shown in figure 5.16.

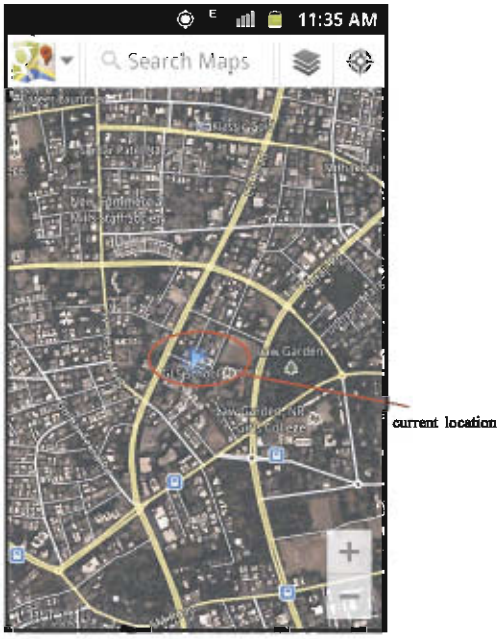


Figure 5.15 : Use of Maps to find the user location

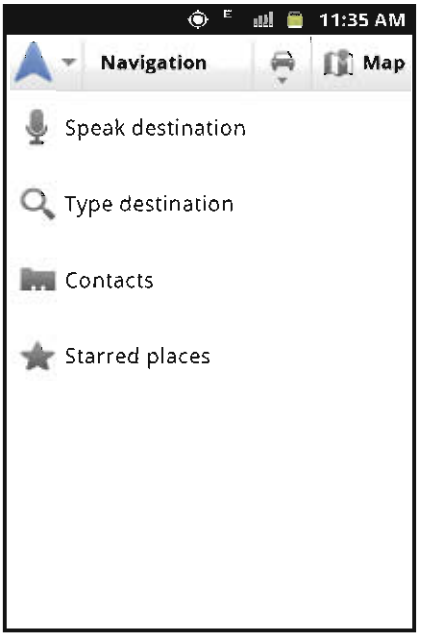


Figure 5.16 : Options to enter destination

Figure 5.17 : Select the destination

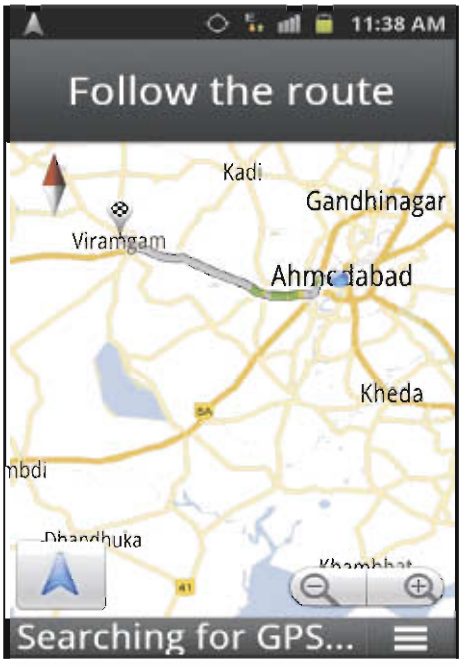


Figure 5.18 : Route to the destination

As seen in figure 5.17, the destination "Viramgam" is selected. The map shows the route from source to destination as shown in figure 5.18.

Figure 5.19 : Distance and approximate time to reach the destination

Figure 5.20 : Driving directions

The distance to the destination and the approximate time to reach the destination is also shown in figure 5.19. We can also see the driving direction to the destination as shown in figure 5.20.

Location based services offers convenience and opportunity to provide services that are more quick or precise and can meet a customer's needs. Some examples of location-based applications are:

- Information or directory services: dynamic yellow pages automatically inform the users about the nearest restaurants, parking facility, traffic updates.
- Tracking services: tracking of assets, locating friends in a geographic location, tracking stolen cars, tracking of children by parents.
- Emergency services: emergency medical ambulance, search and rescue mission, roadside assistance, police and fire response.
- Advertising promotion: targeted ads, promotional messages, customer identification in a store.
- Mapping: Creating maps of specific geographical location
- Navigation: plotting route from one place to another

Security Issues in E-commerce & M-commerce

Internet is a public network system that consists of thousands of private computer networks connected together. These private computer networks are exposed to potential threats from anywhere on the public network. Internet provides a good opportunity to the businesses but along with the convenience also come new risks. The valuable data or information that travels on the Internet may be misused, stolen, corrupted or lost. For example, while making online purchase on the E-commerce website, customer needs to provide the credit card number and the personal details. This information is transmitted to the merchant server. The merchant server sends it to the issuing bank for authorization through payment gateway. All these transmissions occur on the public network i.e. Internet. An

unauthorised user may read credit card number during the transmission and misuse it later on. Also, there are possibilities that order information might be changed in between. If the customer has ordered 10 items, and somehow the merchant receives order of 100 items, he would ask us to pay for 100 items. An intruder can steal or tamper information anywhere in the world while sitting on his computer. He can create new programs and run them on remote computers causing it to malfunction or break down in worst cases while hiding his identity.

E-commerce/M-commerce sites have to keep their online data such as customer's personal details, their bank details and many more safe. They have to be aware of all the frauds that are taking place now-a-days. As E-commerce deals with payments such as online banking, electronic transactions, using debit cards, credit cards and many others; the E-commerce/M-commerce websites have more security issues. They are at more risk of being targeted than other normal websites. Thus, it becomes very important to secure the data on Internet. The E-commerce/ M-commerce security must meet four important aspects as mentioned below:

- **Confidentiality**

  It refers to the secrecy of the information so that unauthorized user cannot read it. It is achieved by using cryptography in which all the messages transmitted are encrypted and only the receiver can read it after decrypting the message using appropriate key. This protects the data from private attacks and ensures that the message is not revealed or leaked to anyone as it travels to the destination. It helps in protecting the confidential data like credit card number.

- **Integrity**

  It ensures that the information must not be accidentally or maliciously altered or tampered in transit. The receiver should receive the same message as was sent by the sender. If the message is altered in between the transition, it should be detected. This removes the problem of modifying the order quantity in between and later creating the payment problems.

- **Authorization**

  It ensures that only authentic users are allowed to use the system. The login and password is one of the ways to achieve authentication.

- **Non-repudiation**

  It ensures that the sender of the message cannot deny that he/she has sent the message. It prevents sender or receiver from denying a transmitted message when in fact they did send it. For example, if the customer denies of sending a purchase order for any reason, then it can be proved that the customer has send the message. It is usually accomplished via digital signatures or a Trusted Third Party (TTP).

Internet Security Threats

The most common threats that are faced on Internet are :

- **Malicious code**

  Malicious code is one that causes damage to a computer or system. Malicious code can either activate itself or be like a virus requiring a user to perform an action, such as clicking on something or opening an email attachment. It can also affect a network, send messages through email and steal information or cause even more damage by deleting files.

- **Sniffing**

  A sniffer is a program that uses Internet to record information that passes through a computer or router in transit from sender to receiver. Using a sniffer program is like tapping the telephone wire and recording the conversation. Sniffer programs can read e-mail messages, user login, password and credit card numbers.

- **Denial of service attack**

  A Denial-of-Service (DoS) attack is an attack used to shut down a machine or network, making it inaccessible to its intended users. By targeting the user's computer and its network connection, or the sites which the user tries to access, an attacker may be able to prevent the user from accessing email, websites, and online accounts like banking or other services that rely on the affected computer. The users are flooded with hundreds and thousands of messages that create traffic problem on the network.

- **Cyber Vandalism**

  Cyber vandalism is the electronic defacing of an existing website page. An attacker replaces the website's original content with his/her own content. It is an example of integrity violation. It is the electronic equivalent of destroying property or placing graffiti on someone's photograph. Today, there are so many cases of cyber vandalism where the business content is replaced by offensive material.

- **Spoofing**

  Spoofing or masquerading is pretending to be someone you are not, or representing a website as authentic when it is actually a fake. It is a technique where the attacker tries to assume the identity of another person or system for transacting with victim site. For example, an attacker can create a fake website as www.gswan.co.in and substitute his IP address for the real website IP address. All the user's visiting to the real site will then be redirected to the fake website.

Security measures

To prevent the various security threats many security measures are taken. Let us discuss some of them.

- **Antivirus software**

  Antivirus software is a computer program that detects, prevents and takes action to remove the malicious codes like viruses, worms and trojan horses from the infected system. To protect your computer you need good antivirus software. A system without antivirus software can easily be targeted by malicious code within a short span of time on Internet. The problems and damage that are caused by an infection can be extremely varied. The infection may be simple as causing strange noises, pop-ups and other annoying things on the system. It may delete the files and slow down the system or also can damage the hardware or destroy the entire computer system. Once a system is infected by virus, it will spread by attaching to other programs and files within the system. Viruses not only replicates itself within the system but can also spread to other systems by taking control of the users email and sending out copies of itself to those in the users contacts list. The most common way a system is attacked is through infected attachments

to email. These attachments can be in the form of pictures, videos, sound clips or any other type of file that can be attached to an email. Infections can also be spread through Internet downloads.

To prevent harm to the system, the antivirus software scans the downloaded files and the entire computer system in order to detect the presence of an infection. Today, we have antivirus software for mobile devices also due to prevalent use of Internet on these devices. You can find a large range of antivirus software in the market. Antivirus software is critical to be installed and kept updated regularly on the computer.

- **Firewall**

Companies having their own websites have to control the access to the network services both inside and outside the company network. The most commonly used network protection barrier between the company's network and the outside world is a Firewall. As shown in Figure 5.21, firewall is a device (a computer or a router) placed between the network and the Internet to monitor and control the traffic between the company's local network and the outside world. The primary goal of a firewall is to keep intruders away from the company's E-commerce infrastructure. It ensures that the company's crucial data is kept safe and not accessed by the intruders.

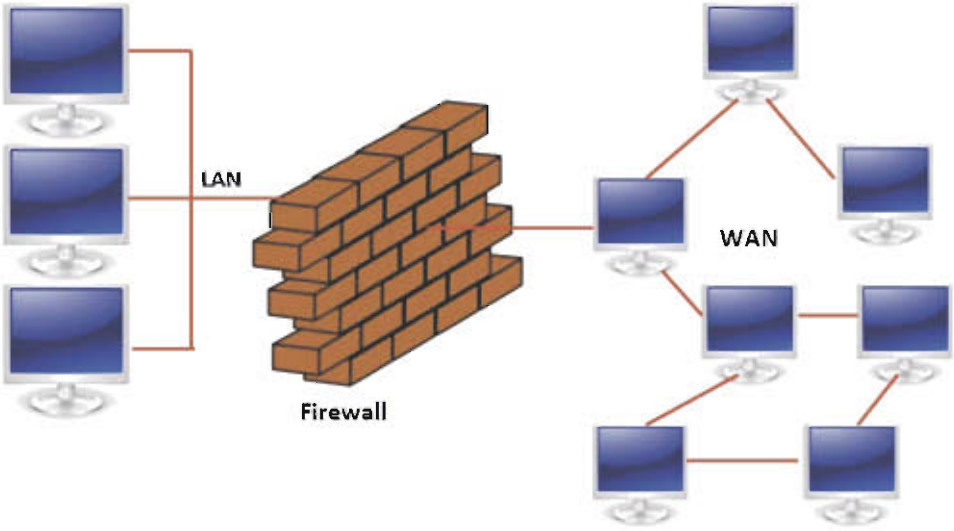


Figure 5.21 : Firewall placed between local and public network

A firewall protects the local network against the following :

- Email services that sometimes create problems.
- Undesirable material like photos, videos entering into local network.
- Unauthorized persons gaining access to local network.
- Unauthorized data or information leaving the company's network.
- Blocks the traffic from outside world to the local network.
- Protects from any type of network attack.

- **Digital Certificate**

Digital Certificates or Digital ID are used for proving our identity in electronic transactions. Just as we have a driving license or a passport to prove our identity in the real world. With a Digital Certificate, we can assure the business organisations, online services and friends that the electronic information they receive from us are authentic. Digital certificate is issued by a trusted third party to establish the identity of the holder. The third party who issues certificates is known as a Certification Authority (CA). Digital Certificate contains the holder's name, a serial number, expiration dates, a copy of the certificate holder's public key which is used for encrypting messages and digital signatures, and the digital signature of the certification authority so that a receiver can verify that the certificate is real.

- **Cryptography**

Cryptography is an art of protecting the information by transforming it into an unreadable form. Encryption is the transformation of normal text known as "plain text" into unreadable or secret text known as "cipher text" using encryption algorithm. A secret key is used to encrypt and decrypt a message. It does not hide the text but converts it into other text that does not make any meaning. Its purpose is to ensure privacy by keeping the information hidden from anyone on the Internet except the receiver of the message. Messages are encrypted just before they are sent on the Internet or network. When the encrypted message is received by the receiver, it needs to be decrypted. Decryption is the reverse of encryption. It is the transformation of encrypted text back into normal text. There are number of encryption algorithms available in the market today.

In recent years, number of cases has been reported where the data in transit was intercepted. Encryption is used to protect data in transit, for example data being transferred via networks like Internet or E-commerce, mobile telephones, Bluetooth devices and bank Automatic Teller Machines (ATMs). Assume that you want to send a message "HOW ARE YOU?" to your friend. However, to protect the message you want to encrypt it. Using encryption we create a coded message to be sent to your friend. Here the encryption mechanism substitutes each alphabet with the alphabet that comes after it. This means that "A" becomes "B," and "B" becomes "C" and so on. You have also told your friends that the code to decrypt is "Shift by 1". Your message will now be encrypted as shown in the figure 5.22 :

H	O	W	A	R	E	Y	O	U	← Plain text
I	P	X	B	S	F	Z	P	V	← Cipher text

Figure 5.22 : Encryption using key

Your friend when he gets the message can decode it. Anyone else who sees the message in transit will only see weird characters. The key kept here is simple, but usually they are quite long.

For E-commerce security various protocols are also used. Let us understand the SSL protocol.

- **Secure Socket Layer (SSL)**

Now-a-days every user sends various types of data from email to credit card details. The user

wants this data to be protected when in transit over a public network. SSL protocol is used for securing web transactions on the Internet. It was developed by Netscape. During the E-commerce transaction, all the information is exchanged in secured manner using SSL by encrypting all the messages. It also provides the authentication of the merchant or shopper using a digital certificate so that the customer knows that they are communicating with a valid owner. To identify whether a site is secured, a security logo is present on the screen. If a site is secured by VeriSign, then the security logo of VeriSign is displayed on the login screen of the secured site. On clicking, you will get the owner information and the validity of the certificate. This indicates that the communication with this site is secured and the owner of the website is valid who is certified by the authority like VeriSign. The other indication of the security is that whenever connection is made to the secured site its address starts from https:// rather than http:// as shown in figure 5.23.

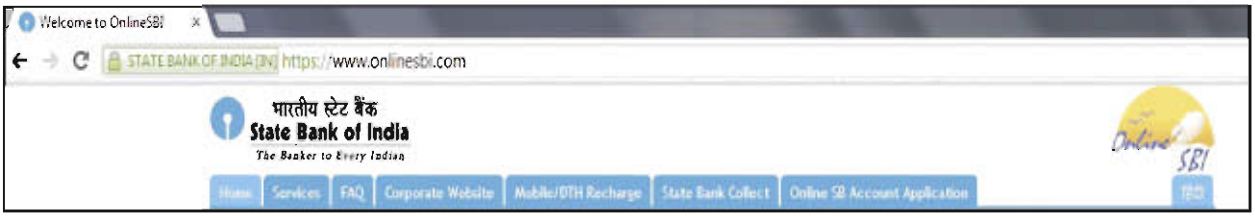


Figure 5.23: Example of https://

Legal Issues in E-commerce/M-commerce

E-commerce as well as M-commerce presents a world of opportunity for doing businesses, reaching global markets and providing facility of online purchasing. It provides opportunities to improve the business processes. However, just as any new business has some issues and risks so does E-commerce and M-commerce. Both of these pose many legal challenges, as it is the activity performed on global Internet without observing national boundaries. Every nation has its own rules and regulations. The major challenges are related to the intellectual property rights, copyrights, privacy and many times the disputes among the parties. The legal framework is necessary to resolve these issues. Many countries have already established their legal framework for electronic commerce. Indian Government has also established the IT laws (Information technology laws) under the IT act.

Enforcement of legal rules and regulations provides confidence to the customers that their personal information remains secret and will not be misused. If it is misused, responsible party can be punished. This type of security is a must in E-commerce and M-commerce as customers provide their sensitive information like credit card details. Companies making online business are getting legal support in case of misuse of their logos or any copyright materials like digital content. Any dispute between two parties can be resolved under these laws.

Securing Intellectual property

The intellectual property includes books, software's, music, video, copyrights, trademarks and web pages. Let us discuss some issues related to them.

- **Copyright**

  Copyright provides the author with a tool to protect his/her original work from being used or taken by others without permission. It is applicable to books, software programs and articles. Copyright law protects intellectual property in its various forms, and cannot be used freely.

It is very difficult to protect Intellectual property in E-Commerce or M-commerce. For example, if you buy software you have the right to use it and not the right to distribute it. The distribution rights are with the copyright holder. Also, most of the web pages are protected by copyright. In that case, copying contents from the website also violates copyright laws.

- **Trademark**

  It is a specific logo, mark, word, symbol, design, phrase or image which is used by an individual or a company to distinguish a product or service from that of others in the market. A trademark may be designated by the following symbols: TM, SM and ®.

- **Domain name disputes**

  The competition over domain names is another legal issue. Earlier, the domain names were given on first come first serve basis. Thus people would register domain names that were not in use but would be of importance. Later on such domain names were sold to concerned company at a very high price. This is known as cyber squatting. Another problem is Name changing where someone registers purposely misspelled variations of well known domain names. This can mislead the consumers who generally make typographical errors while entering a URL.

Protecting Intellectual property

Several new and improved methods are continually being developed to protect the intellectual property. Some of them are discussed here.

Steganography

Steganography is the process of hiding information within other information. The information in the files if not protected can be used for any malicious purpose. It works by replacing unused data in computer files such as images, sound or text with invisible information. This hidden information can be plain text, cipher text, or even images. Special software is needed for steganography, and there are freeware versions available on Internet which can be easily downloaded.

Digital Watermarking

The watermark is a digital code inserted into a digital image, audio or video file which can identify the file's copyright information. It also allows hiding information in a totally invisible manner. Earlier, artists used to creatively sign their paintings with a brush to claim copyright. But in digital world, artists can watermark their work by hiding their name within the image. Hence, the invisible embedded watermark helps to identify the owner of the work.

This concept is also applicable to other media such as digital video and audio. Currently the unauthorized distribution of digital audio over the Internet in the MP3 format is a big problem. In this case, digital watermarking may be useful to set up controlled audio distribution and to provide efficient means for copyright protection. In the field of data security, watermarks can be used for certification and authentication. For example, the photo identity card of a person can be protected by an identity number "123456" written on the card and hidden as a digital watermark in the identity photo. So, manipulating or changing the identity photo can be detected easily.

Digital watermarking can also link information on the documents. For example, the name of a passport owner is normally printed in clear text. But using digital watermark, the name can also be hidden

in the passport photo. If anyone tries to tamper with the passport by replacing the photo it would be possible to detect the change by scanning the passport and verifying the name hidden in the photo. A visible digital watermark can be added to any image using photo editor tools like GIMP.

Payment in E-commerce/M-commerce

Payment is one of the most important aspects of E-commerce as well as M-commerce. In traditional payment method we do the payment using cash, cheque or credit card. Electronic payment systems are becoming more important to the online business processes as companies are looking for different ways to serve the customers faster and at low cost. Electronic payment is a financial exchange that takes place online between the buyer and the seller. There are various payment options available for payment in the market today. The different types of electronic payment systems used are:

- **Payment cards**

  The payment cards can be classified as Credit cards, Debit cards and Smart cards, generally, businesses use the term payment card for all types of plastic cards that consumers use to make purchases.

  Credit card

  This is one of the most popular and widely accepted methods of payment on Internet. A credit card is issued to the customers by the banks known as issuing banks. The issuing banks provide the credit cards of the financial institutions which are established and reputed in the services of credit card business. Examples include MasterCard® or Visa®. Depending upon the customer's credit history and income level, credit limits are provided and up to that limit the customer can spend and pay to the issuing bank within the billing period.

  As the credit cards are linked to a bank account, when we use them to pay online, the merchant charges the goods to the issuing bank account and the bank shows the debit on your next statement. The customer simply pays the bank. For accepting payments on websites through credit card, merchant needs to open a merchant account with the banks known as acquiring banks, which provide the services of online authorization and payment processing. Authorization is the process of verifying whether the card is active; the credit limits are available to make purchase and verifying the other details of the customer like billing information. Credit cards are widely accepted by merchants and provide assurance to the customer as well as the merchant. Figure 5.24 shows some credit cards.

Figure 5.24 : Examples of Credit card

In credit card transactions over the Internet, the customer visits the merchant's website and selects goods to buy, all the information related to credit card is entered and then this information is transmitted to the merchant electronically. In this transaction four parties are involved :

- Customer with credit card
- Merchant accepting the credit card
- Issuing Bank: issues the credit card and guarantees the payment to the merchant. The bank collects the payment from the customer.
- Acquiring banks: Financial institution that establishes the account with the merchant validates the credit card information of the customer and authorises sale based on the customer's credit limit.

Two more entities play role in online payment. These are payment gateways and processors. Payment gateways are services provided by the third parties like PayPal which connects networks of all the parties involved and enables to perform authorization and payment in secured manner. Processors are data centres which perform the credit card transactions and settle funds to the merchant. Processors are connected to the E-commerce website of the merchant through the payment gateway.

The online payment through credit card on Internet is divided into two parts: Authorization and Settlement. During authorization the following steps are performed :

- Customer checkouts, provides credit card information on the E-commerce website, which along with the transaction details (like item detail, date of purchase and others) is transferred to the payment gateway.
- The payment gateway passes the information to the processor which contacts the issuing bank for the verification of the information.
- After verification, issuing bank sends the status of verification (or transaction result like accepted or rejected) to the processor which in turn passes it to the payment gateway.
- Finally, payment gateway sends the result of the transaction to the merchant's website. If the merchant accepts the transaction then the next step is the settlement during which it transfers the amount from the customer's account to the merchant's account.

During the settlement or payment processing the following steps are performed :

- Merchant sends the transaction request with all the details to the payment gateway which sends to the processor.
- Processor sends the payment details to the issuing bank of the customer. It also sends the payment details to the acquiring bank where the merchant has an account.
- The acquiring bank credits the amount to the merchant account. The issuing bank after including all the charges sends the bill to the customer which he needs to pay within the billing period.

Major credit card companies use the Secure Electronic Transfer (SET) security system to make online transactions secure.

The advantages of credit card are :

- Gives flexibility to the customer as they do not have to carry lot of cash. Customer can pay for goods and services both online and offline.
- Keeps record of the customer's purchase through the bank statement.
- Allows customers to purchase goods even when they do not have the cash available in the bank account. They can settle the cash by the end of the month.

The limitations of credit cards are :

- They are unsuitable for very small or very large payments. Also, due to security issues, these cards have a limit and cannot be used for excessively large transactions.
- Customers tend to overspend using credit cards.
- Problems arise in case lost or stolen credit cards.

Debit Cards

A debit cards looks like credit card but works differently. It is a kind of payment card that transfers fund directly from the consumer's bank account to the merchant. The amount is immediately deducted from the bank account of the consumer. The debit cards can keep the consumer purchases under a limit and do not allow him to exceed beyond his/her budget. But while using a debit card for a purchase, the consumer should always be aware of his account balance. Figure 5.25 shows an example of debit card.

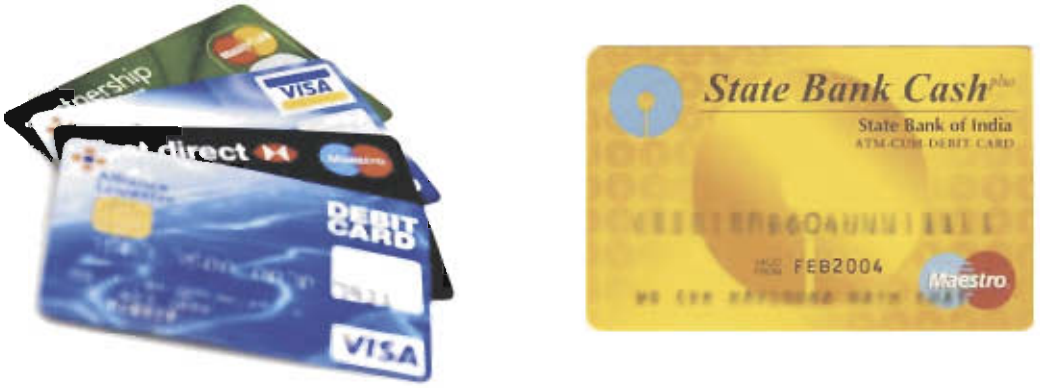


Figure 5.25 : Example of Debit card

Smart Cards

Smart cards look just like credit cards but are different as they have a microchip embedded in their surface. A smart card contains user's private information, such as account information, health insurance information, private keys etc. It can store 100 times more information than the normal cards. They are much safer than the credit or debit cards as the information stored in the smart card is encrypted. Customers can load their card with cash and then use this to pay for goods in a merchant's retail outlet or web site. Card readers are available for retail outlets as well as an attachment for PCs. This convenience gives a great advantage to smart cards. They can be used for a range of purposes like storing digital cash, storing a patient's medical records etc. Smart cards are popular in countries like U.S., Europe, Japan and some parts of Asia.

A smart card reader is required to read information or add data to it. It is small device into which the smart card is inserted. For example, when you visit your family doctor, you can give him your smart card to review the medical history and prescribe medicines. This information is also inserted into the smart card. At the medical store, you hand over your card to the pharmacist, who sees your prescription, and give you the medicine accordingly. Also, the payment can be done using smart card.

Charge cards

A charge card is another form of payment mechanism wherein the customer can pay through the card to the vendor. As compared to credit cards that have a credit limit, the charge card does not carry any spending limit. The customer has to pay the total amount used at the end of the billing period to the company that has issued the card. If the total amount is not paid back then the customer has to pay a late fee.

- **Net Banking**

  Another option which is becoming more popular is net banking or online banking. It does not involve any type of card. It can be used by customers who have bank accounts enabled with Internet banking. The bank provides the net banking password to the customer for operating the account from Internet including the payment for online purchases. Many of the E-commerce/ M-commerce websites provide the facility to make the payments using net banking. Instead of entering card details on the website, it allows one to specify the bank through which you wish to pay from. On these websites, when you proceed to make the payment, you will be asked to select your bank. Once you select the bank, you will see the screen of your bank's website where you need to log in using your account number and net banking password. You can perform the transaction and transfer the amount from your account to the account of the merchant.

  Indian railway provides the facility to book tickets online on their website www.irctc.co.in. First, the user needs to register on the site and then provide the details of the journey along with the passenger information. Once the information is filled up you can proceed to the payment where net banking option is provided. Select your bank and continue further which displays the login screen of your bank and transfer the fund to Indian railway to complete the booking. After successful payment, the user gets a message on the registered mobile number from irctc, which can be shown at the time of travel or E-copy of the ticket can also be printed. Net banking payment system is seen as being safer than using credit cards as nearly all merchant accounts in India offer it as an option.

- **Electronic Fund Transfer**

  Electronic Funds Transfer (EFT) means transferring money from one bank account to another electronically. It is safe, secure, efficient, and less expensive than paper check payments and collections. Examples of EFT are: transactions amongst various banks around the world, payment of tuition fees using an ATM, deposit of employee's salaries in their accounts, monthly bank account deductions and many more. The popularity of EFT for online bill payment is growing. The benefits of EFT include reduced administrative costs, increased efficiency, simplified bookkeeping, and greater security. However, the number of companies who send and receive bills through the Internet is still relatively small.

- **E-wallet**

  Most of the time when you make purchase on the web, you are required to fill out a form with your name, shipping address, billing address, credit information, and so on. Doing this a few times is fine, but having to do it every time you shop on the web is an annoyance. Some merchants solve the problem by having you fill out a form once, and then saving the information stored on their servers for later use. These merchants provide e-wallet services to its customers which makes the shopping or booking of tickets much easier for its customers. Today many banks, online grocery stores, telecom services etc. provide e-wallet services. E-Wallet is an electronic card for making secure online payments towards a merchant. It works just like a credit or a debit card. While making payment through e-wallet the customer is not required to provide the credit/debit card number thus reducing the risk of credit/debit card number being exposed.

  For example, IRCTC has launched e-wallet scheme for the customers to make online booking easier. The customers having an account can deposit money in advance with IRCTC which can be used in future as payment option to book the tickets online. Currently the users have to provide the credit/debit card details for booking tickets. This payment procedure takes time as the customer is transferred to the bank's server for payment. He/she can

- **RuPay**

  The term RuPay is coined from two terms Rupee and Payment. It is a new card payment mechanism launched by National Payments Corporation of India (NPCI). Figure 5.26 shows the RuPay card which can be used the same way as we use the credit and the debit cards.

Figure 5.26 : RuPay card

Currently, as there is no domestic card Indian banks have to tie with Mastercard and Visa to connect cardholders, merchants and the issuing banks around the globe. Mastercard and Visa are the world leaders in card payments and all payment transactions are processed through them. Every transaction done using a credit or debit card issued by a domestic bank is routed through their network switches which are outside the country. These transactions involve additional charges for providing the services. The banks have to pay for processing all debit and credit card payments. RuPay cards are the domestic alternative to the other payment cards. By using RuPay cards, all the transactions will be processed within India. As the transaction processing will be done domestically, the cost of each transaction clearing and settlement will be reduced. RuPay will benefit the customers and the banks by reducing the cost.

Summary

In this chapter we discussed about Mobile Commerce. M-commerce refers to buying and selling of goods and services through the use of Internet enabled wireless devices such as a Mobile phone, Personal digital assistant (PDAs), Smartphone, Tablet PCs and Palmtops. we also discussed about the advantages and limitations of M-commerce. M-commerce applications are becoming popular with the mobile users increasing day by day. Location-based commerce applications and services are emerging with significant implications for the future of E-commerce. These applications track the user's location in order to deliver a service or product. We discussed about the various the security issues and threats. Measures to be taken against the security threat are also discussed. Legal issues related to e-commerce are discussed. The various payment options provided in E-commerce like credit card, debit card, smart cards, Ru-Pay, net banking and E-wallet are discussed.

EXERCISE

1. What is M-Commerce ? List some of the examples of M-commerce.
2. Why is M-commerce helpful to the user ?
3. List the advantages of M-commerce.
4. What are the limitations of M-commerce ?
5. List some of the applications of M-commerce.
6. Give some website examples for the following M-commerce applications :
   (1) Mobile ticketing (2) Mobile Auctions
   (3) Mobile purchase (4) Mobile information services
   (5) Mobile financial services
7. What is L-commerce ?
8. What is GPS ? How does it locate a device ?
9. List some applications of location based services.
10. What are the four important aspects of E-commerce security ?
11. Explain the following Internet security threats :
    (1) Malicious code (2) Sniffer (3) Denial of service attack
12. What is cyber vandalism ?
13. Explain the term spoofing ?
14. List the various security measures taken for security threats.
15. What does a digital certificate contain ?
16. What is cryptography ? Using encryption mechanism of each alphabet with the alphabet that comes before it, encrypt the message "Gandhi Ashram".

17. What is the purpose of SSL ?

18. List the issues related to intellectual property.

19. Which are the different ways of protecting Intellectual property ?

20. List the different types of electronic payment systems.

21. List the advantages and limitations of credit card.

22. How is a smart card different from credit cards ?

23. What is electronic fund transfer ?

24. Choose the correct option from the following :

(1) Which of the following refers to buying and selling of goods or services through the use of Internet enabled wireless devices ?

(a) Internet (b) M-commerce (c) M-banking (d) WWW

(2) Which of the following is the use of technologies which provide the location information for business purpose ?

(a) E-commerce (b) M-commerce

(c) L-commerce (d) Traditional commerce

(3) Which of the following stands for GPS ?

(a) Global Positioning System (b) Global Postal System

(c) Grand Positioning System (d) Google Positioning System

(4) Which of the following security aspect refers to the secrecy of the information so that unauthorized user cannot read it ?

(a) Confidentiality (b) Integrity

(c) Non-repudiation (d) Authorization

(5) Which of the following security aspect ensures that the information must not be accidentally or maliciously altered or tampered in transit ?

(a) Confidentiality (b) Integrity

(c) Non-repudiation (d) Authorization

(6) Which of the following security aspect ensures that only authentic users are allowed to use the system ?

(a) Authorization (b) Confidentiality (c) Non-repudiation (d) Integrity

(7) Which of the following security aspect ensures that the sender of the message cannot deny that he/she has sent the message ?

(a) Authorization (b) Confidentiality (c) Non-repudiation (d) Integrity

(8) Which of the following is a program that uses Internet to record information that passes through a computer or router in transit from sender to receiver ?

(a) Sniffer (b) Denial of service attack

(c) Malicious code (d) Spoofing

(9) Which of the following is an attack used to shut down a machine or network, making it inaccessible to its intended users ?

(a) Malicious code (b) Denial-of-Service

(c) Spoofing (d) Cyber vandalism

(10) Which of the following is known as electronic defacing of an existing website page ?

(a) Cyber vandalism (b) Denial-of-Service

(c) Spoofing (d) Malicious code

(11) Which of the following is pretending to be someone you are not, or representing a website as authentic when it is actually a fake ?

(a) Cyber vandalism (b) Malicious code

(c) Denial-of-Service (d) Spoofing

(12) Which of the following is a computer program that detects, prevents and takes action to remove the malicious codes like viruses, worms and trojan horses from the infected system ?

(a) Antivirus software (b) Digital certificate

(c) Firewall (d) Cryptography

(13) Which of the following is the transformation of normal text known as "plain text" into unreadable or secret text known as "cipher text" using encryption algorithm ?

(a) Firewall (b) Encryption

(c) Antivirus software (d) Digital certificate

(14) Which of the following is the transformation of encrypted text back into normal text ?

(a) Firewall (b) Digital certificate

(c) Decryption (d) Virus

(15) Which of the following is a protocol used for securing web transactions on the Internet ?

(a) TCP/IP (b) HTTP (c) Bluetooth (d) SSL

(16) Who developed SSL protocol ?

(a) Google (b) Netscape (c) Yahoo (d) Firefox

(17) Which of the following starting address indicates that site is secured by SSL protocol?

(a) http:// (b) ssl:// (c) https:// (d) http-ssl://

(18) Which of the following provides the author with a tool to protect his/her original work from being used or taken by others without permission ?

(a) Trademark (b) Copyright (c) Digital watermarking (d) Steganography

(19) Which of the following is a specific logo, mark, word, symbol, design, phrase or image which is used by an individual or a company to distinguish a product or service from that of others in the market ?

(a) Trademark (b) Copyright

(c) Digital watermarking (d) Steganography

(20) Which of the following is the symbol for trademark ?

(a) TM, MT and © (b) TM, MS and ®

(c) TM, SM and ® (d) TM, SM and (C)

(21) Which of the following is the process of hiding information within other information ?

(a) Squatting (b) Steganography (c) Name changing (d) Copyright

(22) Which of the following is a digital code inserted into a digital image, audio or video file which can identify the file's copyright information ?

(a) Image mark (b) Digital mark

(c) Code mark (d) Watermark

(23) Who issues the credit card and guarantees the payment to the merchant ?

(a) Merchant (b) Issuing bank (c) Acquiring bank (d) Customer

(24) Which of the following payment card has a microchip embedded in its surface ?

(a) Smart card (b) Debit card (c) Credit card (d) Charge card

Object-Oriented concepts 6

Today we are in the era of Internet, Websites and Web based operations, where rapid application development and reusability of source code is very important. Object-oriented techniques as methodology or as paradigm is playing significant role in analysis, design and implementation of software system. Software developed using object-oriented methodology are proclaimed to be more reliable, easier to maintain, reuse and enhance. This chapter explains basic object-oriented concepts. Implementation of this concepts using Java programming language is covered in the later chapters.

Introduction

Object-oriented programming concepts started originating in the 1960s. Since mid 1980s, it had become the main programming paradigm used in the creation of new software. It was developed as a way to handle the rapidly increasing size and complexity of software systems and to make it easier to modify these large and complex systems over time. Some of the popular programming languages that support object-oriented programming are C++, Java, C#, VB.net, ASP.net and PHP.

The way of programming can be divided into two categories namely structure/procedural and object-oriented. In procedural programming; the focus is on writing functions or procedures which operate on data. For example, for library application software, we will think of all processes of library application and focus on the modules like student registration, book issue, book return, fine calculation.

In object-oriented paradigm, the focus is on **objects** which contain both data and functionality together. For library application, our focus is on the objects involved in the application. Here we think of objects like student, book and librarian. We also need to think of association between such objects. For example, student returns book.

The power of object-oriented programming language enables the programmer to create modular, reusable and extendable code. As a result, programmer can formulate a program by composition and modification of existing modules. Flexibility is gained by being able to change or replace modules without disturbing other parts of code. Software development speed is gained by reusing and enhancing the existing code.

Object-oriented programming uses object as its fundamental building block. Similar objects are classified using a concept of class. A computer language is object-oriented if they support four specific object properties called abstraction, encapsulation, polymorphism and inheritance.

Object

In the "real" world, objects are the entities of which the world is comprised. Some objects can be concrete things like person, car or a coffee cup. Other objects may be abstract that do not represent things which can be touched or seen; for example, concepts like date and time.

All objects have unique identity and are distinguishable from each other. For example, every person has characteristics like name, city, gender, birth-date, profession. In object-oriented terminology, such

characteristics are known as **properties** or **attributes**. To uniquely distinguish one person from other, we use the value of these characteristics. The names 'Ram' and 'Shyam' specify two different persons. When two persons have same name, they may be distinguished using other attribute like birth-date. In this way, to identify the objects, we use the value of these attributes. These values are called **state**. Additionally, there is a **behaviour** associated with objects. For example, person takes birth, gets name, changes location. The behaviour is also known as **method**. State of the object can change due to its behaviour. Thus, any real world object can be described in terms of what it is called (identity), what it is (its state) and what it does (its behaviour).

In object-oriented programming, attributes that describe the object are also referred to as data fields. The data attributes and behavioral methods associated with an object are collectively referred to as its **members *or* features.**

When we design for any software application, objects to be considered should be meaningful to the application. For example, in railway reservation application, meaningful objects are train, passenger, ticket or station. Objects like car, computer and watch may be irrelevant here.

Class

From the example of the object person, it can be easily seen that various objects possessing same characteristics and behavior differ from each other only in the state (value of the data) that they hold. Object-oriented system uses the concept of class that enables to express the set of objects that are abstractly equivalent, differing only in the values of their attributes. Class can be considered as a blueprint for various objects.

A class is a template for multiple objects with similar features. It describes a group of objects with similar attributes and common behavior. Objects in the same class share a common semantic purpose. Thus, class is a general concept used to embody all the common features of a particular set of objects.

For example, we have a class named 'Person' describing common attributes and behavior of all persons. Individual persons are then objects and are identified by the state; that is the value of their attributes. Figure 6.1 shows difference between the class and its objects.

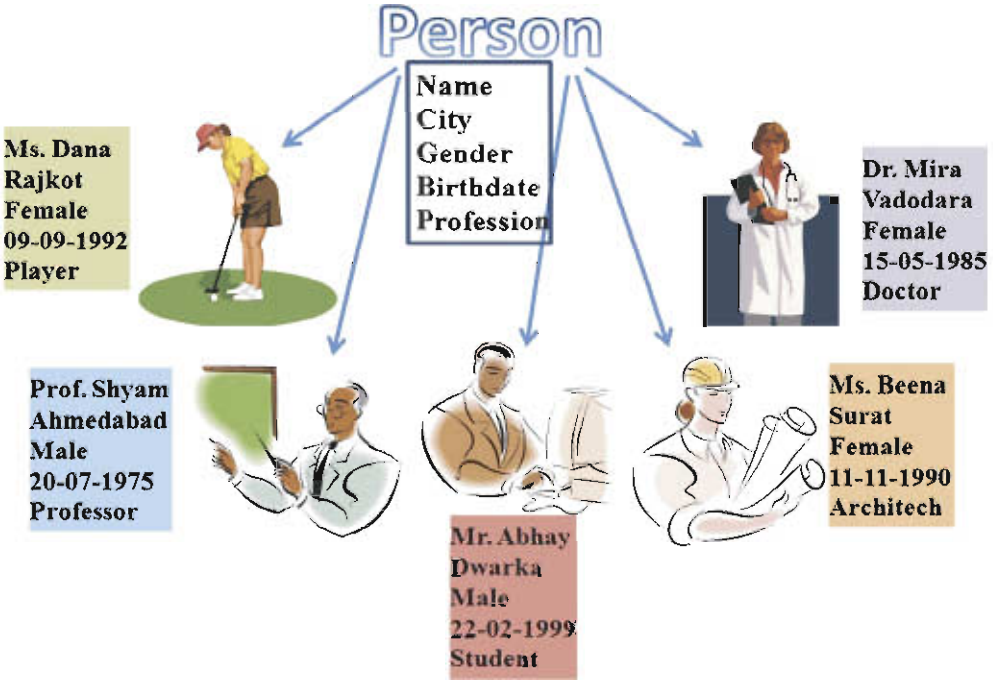


Figure 6.1 : Class 'Person' and its Objects

Before we proceed ahead to learn other object-oriented concepts, let us have brief introduction to class diagram.

Introduction to Class Diagram

The class diagram presents a collection of classes, constraints and relationship among classes.

Unified Modelling Language (UML) can be used to create models of object-oriented software to help with design of an application. UML is a visual modelling language defined and maintained by the Object Management Group (OMG). UML specifies several diagrams for representing different aspects of a software application.

The purpose of the class diagram is to model the static view of an application. The class diagrams are the only diagrams which can be directly mapped with object oriented languages and thus widely used among software developers.

In class diagram, a class is represented with an icon using a rectangle split into three sections to present name, attributes and behavior as follows :

1. Name of the class in the top section
2. Attributes or properties of the class in the middle section
3. Behavior or operations or methods of the class in the bottom section

Figure 6.2 shows the pictorial representation of a class using UML convention, while figure 6.3 shows the class diagram of Person.

Class Name
Visibility attribute : data type = initial value
Visibility operation (argument list) : return type

Figure 6.2 : Class representation in UML convention

Person
name : string city: string gender : char = 'M' -birthdate : date profession : string
setBirthdate (d:int,m:int,y:int): date changeCity (newCity:string) :string display ( )

Figure 6.3 : Diagram of class 'Person'

The class Person is intended to provide information that is useful in understanding the role of the class in the context of class diagram. It does not have to contain every attribute and operation of the class.

As seen in figure 6.2, in UML notation, an attribute is declared using following syntax :

[<visibility>] <attribute name> [: <attribute data type> [= <initial value>]]

Here, items written in pair of square brackets [ ] are optional and user is supposed to specify the values for items enclosed in angle brackets < >. Visibility can be private, protected, public or package represented using symbols -, #, + and ~ respectively. We will study about visibility in later chapters. Attribute generally refers to a variable. Data type and initial value identify the type of data stored and its value at the start of the program. As can be seen here, only attribute-name is mandatory and all other items are optional.

Some examples of declaring attribute are as follows :

name : string

- balance : float = 0.0

As seen in figure 6.2, In UML notation, an operation is declared using following syntax :

[<visibility>] <method name> (parameter list separated by comma) : <return data type>

An example of a method in the previous example is setBirthdate(d:int, m:int, y:int) : date. Here the parameter can be specified with data type and optional default value; for example gender : char = 'M'. Parameters can also be specified as input or output depending on whether it is read only or not.

UML diagrams are independent of the programming language used for coding an application. Some software developers prefer to specify attributes and operations in the more familiar format of a programming language instead of using the standard UML notation. That's fine, as long as it makes sense to everyone concerned.

Objects are presented using their state during execution of an application. Thus, objects are dynamic. Corresponding to figure 6.1, an object (also called an instance) of class person is represented as shown in figure 6.4.

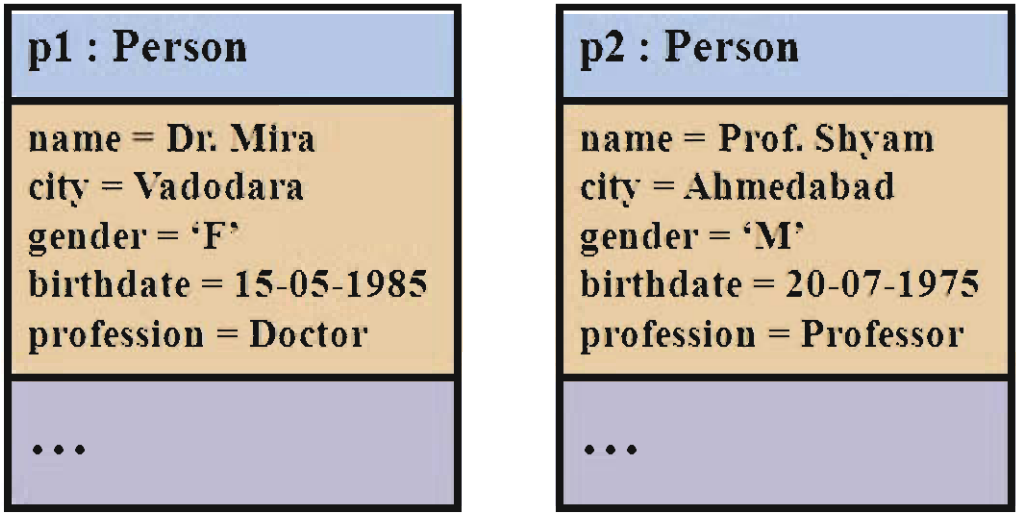


Figure 6.4 : Objects p1 and p2 of Class 'Person'

Encapsulation

For any computer program, two core elements are data and functions. Structured/Procedural programming views these two core elements as two separate entities; whereas object-oriented programming views them as single entity.

In procedural programming, data can be altered by any component of the program. It is not protected from modification.

In object-oriented programming, this problem can be solved using encapsulation. Here, data and the methods that manipulate data are guarded against modification or misuse by other components of the program. This mechanism of providing protection to data and methods of a program is called **encapsulation.** This is possible by wrapping data and methods into a single unit known as class and declaring them as private. Private members of the class are not available directly to outside world. If necessary, the data is made available via public methods. Thus, encapsulation provides data hiding capability. We will study about class and private/public data and methods in later chapters.

Encapsulation keeps the data safe from unintended actions and inadvertent access by outside objects. Unlike procedural programming where common data areas are often used for sharing information, object-oriented programming discourages direct access to common data (other than the use of global variables) by other programs. Only the object that "owns" the data can change its content. Other objects can view or change this data by sending message to the "owner".

Data Abstraction

Data abstraction is a process of representing the essential features of the objects without including implementation detail. Abstraction is a concept that hides the complexity; it says what it does, but not how it is done.

Let's take one real life example of a television set. We can turn television set on and off, change the channel, adjust the volume. But, we do not know its internal details like how it receives signals over the air or through a cable, how it translates them and finally displays them on the screen. Thus there is a clear separation of internal implementation of television set from its external interface. We can play with it via external interfaces like the power button, channel changer and volume control without having any knowledge of its internals.

Data abstraction thus is a technique that relies on the separation of interface and implementation. It is not a new concept in programming. Most people already practice it up-to some degree, probably without realizing it. For example, when we use functions like sqrt(25) and printf("Hello world"), in C programming we never thought of how they are implemented.

A user-defined function with necessary input data parameters also provides data abstraction. Let us consider a C program using structure for declaring 'date' type of data. Here date is composed of three elements; day, month and year. Let us use primitive integer data type for these elements. Consider a function named 'dateDiff' that takes two dates as parameters and returns the number of days between two dates. User of this function is not supposed to know how this difference is calculated. It is necessary to know only the signature of the function; that is the name of the function, number and type of parameters and the return type of parameters. If there is any change in the process of finding difference, it has no effect on the part of the program using this function.

Thus data abstraction provides the skeleton or templates for our use. The system hides certain details of how data is stored, created and maintained. All that is visible to the rest of the world is the

abstract behaviour of the data type; details of how that behaviour is implemented are hidden, so that they can be changed as per need without impacting others.

Abstract Data Types (ADT) or structures (struct) in C/C++, classes in C++/Java are examples for data abstraction. In ADT, we simply define a data type and a set of operations on it. We do not show the implementation of operations.

The basic difference between data encapsulation and data abstraction is: Encapsulation protects data by making them inaccessible from outside and abstraction enables to represent data in which the implementation details are hidden (abstracted).

Messaging

In object-oriented terminology, a call to a method is referred to as a message. Due to encapsulation, all method calls are handled by objects that recognize the method.

Different classes may have same methods with same name. For example, class 'date' has method named 'display' that displays the date object in specific format. Suppose there are other classes 'time' and 'person'; both having method with same name 'display'; to display time in specific format and to display the details of person respectively. When method 'display' is invoked in the program, how to know which method is to be executed? This is determined using the object that invokes the method. For example, if 'display' is called by an object of 'person' class, it executes the display method defined in 'person' class.

Polymorphism

Polymorphism means 'many forms'. There may be different forms of single method or operation.

Assume that we have written a function named 'max' that finds maximum of two numbers. It takes two integers as parameters and returns maximum as integer value. Now suppose we want to add one more function named 'max' that finds maximum of integer elements stored in an array. It takes array and its size as parameters and return maximum integer. Is it possible to define more than one function with the same name ? In some programming languages, the answer to this question is 'no'. But in object-oriented programming, the answer is 'yes' as long as the methods differs in signatures (number and type of parameters).

Object-oriented programming allows defining more than one method having same name but different signatures in a single class. This feature is known as **function** or **method overloading.**

Object-oriented programming also allows writing expression using operators on objects. For example, we can use expression 'date1-date2' where both operands are objects of class 'date'. Here operation '-' is performed in a way different than performing subtraction on two numbers; say n1-n2. Here same operation is given different meanings depending upon the data type of operands used. This type of polymorphism is achieved through **operator overloading.**

Thus, polymorphism is achieved using two types of overloading: function overloading and operator overloading. In general, the capability of using same names to mean different things in different contexts is called overloading.

Aggregation and Composition

When objects of one class are composed of objects of other class, it is called aggregation or composition. It represents 'has-a' or 'a-part-of' relationship between classes. For example, motherboard is a part of computer.

As we know, computer is composed of many parts like motherboard, screen, keyboard and mouse. A screen itself may be a class with attributes like length, width and model. Similarly motherboard may be defined as a class with attributes like model and company. When we define class 'computer', it will contain attributes that are objects of class 'motherboard' and 'screen'. Thus it implies containment.

Let us modify class 'Person' discussed before. First of all, we will define two new classes 'Name' and 'Address' as shown in figure 6.5. Class 'Name' has attributes first name, middle name and last name. Class 'Address' has attributes house (that refers to details of house), street, city, state and pin code. Now let us modify class 'Person' and change the name of the attributes name and address to nm and addr respectively. The data type of attributes nm and addr is class 'Name' and 'Address' respectively. Thus class 'Person' contains objects of class 'Name' and 'Address'.

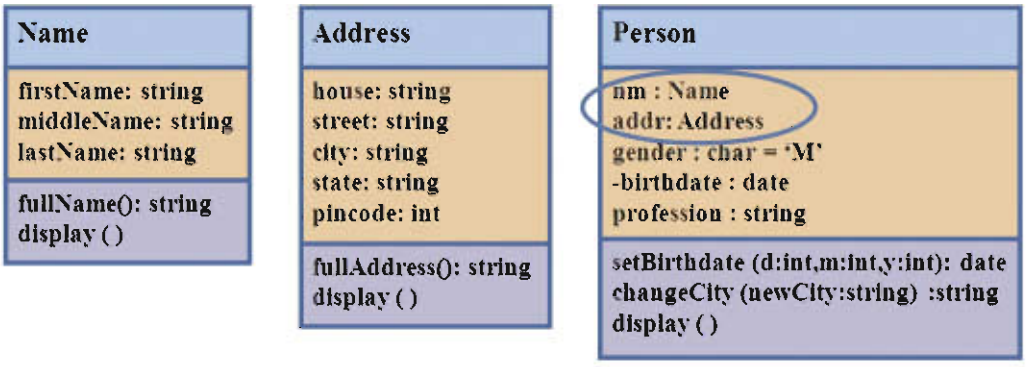


Figure 6.5 : Class 'Person' with attributes of class 'Name' and 'Address'

Aggregation vs. Composition

Aggregation represents non-exclusive relationship between two classes. In aggregation, the class that forms part of the owner class can exist independently. The life of an object of the part class is not determined by the owner class. For example, although the motherboard is part of the computer, it can exist as a separate item independent of the computer. Thus motherboard is not exclusively associated with computer. Similarly object address may be shared by two or more persons as can be seen in figure 6.6. So, address is not exclusive to any one person.

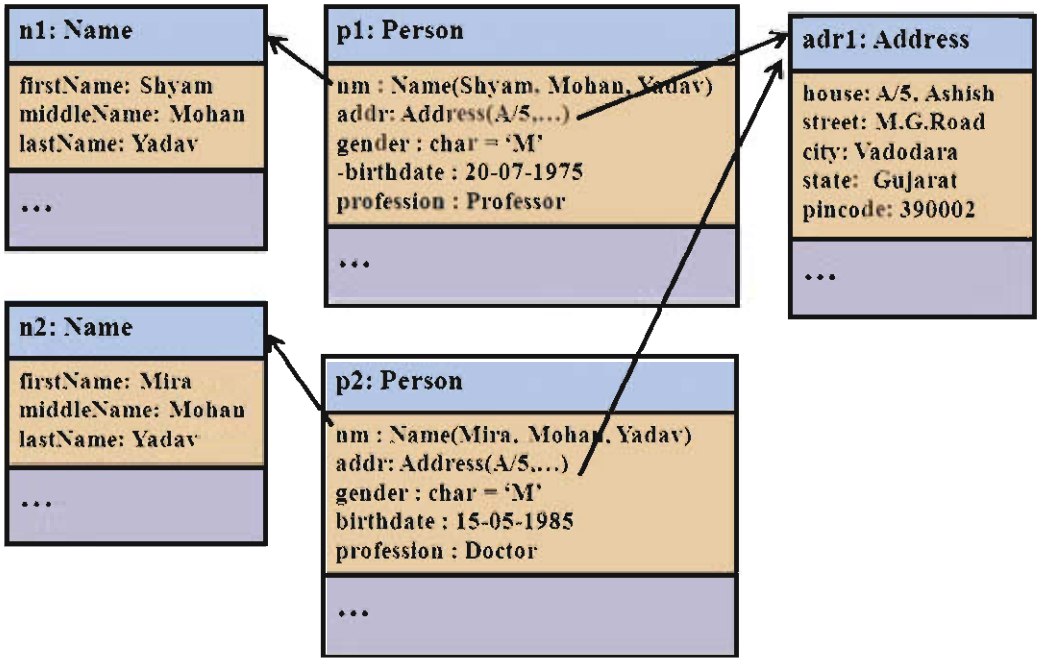


Figure 6.6: Two 'Person' objects having different names but same address

Basic aggregation is represented using an empty diamond symbol next to the whole class as shown in figure 6.7.

Composition represents exclusive relationship between two classes. Composition is a strong type of aggregation where the lifetime of the part class depends on the existence of the owner class. If an object of aggregating class is deleted, its part class object also will get deleted. For example, when an object of class Person is deleted, the object of class Name is also deleted. Name is associated exclusively with single person as shown in figure 6.6. Composition relationships are represented using a filled diamond symbol next to the whole class as shown in figure 6.7.

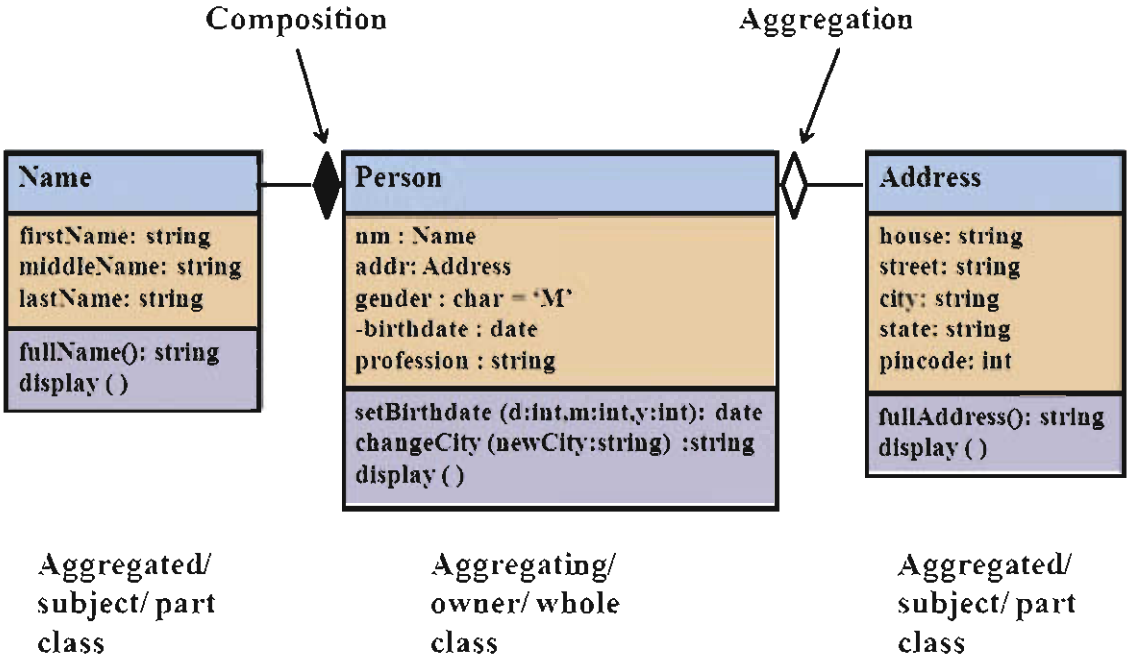


Figure 6.7 : Composition and Aggregation

In the given example, a relationship between class 'Person' and class 'Name' is composition relationship; whereas a relationship between class 'Person' and class 'Address' is aggregation relationship. Address may be shared by more than one person. So, when a person is deleted, the corresponding name object is deleted but address cannot be deleted.

Note : The class that contains objects of other class is known as owner class or whole class or aggregating class. For example, the class 'Person' shown in figure 6.7 is an aggregating class.

The class that is contained in owner class is known as subject class or part class or aggregated class. For example, the classes 'Name' and 'Address' shown in figure 6.7 are aggregated class.

Inheritance

Inheritance is generally referred to as 'is-a-kind-of' relationship between two classes. It is appropriate when one class is 'a kind of' other class. For example, teacher is a kind of person. So, all the attributes and methods of class 'Person' are applicable to class 'Teacher' also. In other words, class 'Teacher' inherits all attributes and behavior of class 'Person'. Class 'Teacher' may have additional attributes like subject and methods like taking lectures of the subject. In such scenario, class 'Teacher' can be defined using class 'Person'.

Inheritance refers to the capability of defining a new class of objects that inherits the characteristics of another existing class. In object-oriented terminology, new class is called sub class or child class or derived class; whereas the existing class is called super class or parent class or base class. The data attributes and methods of the super class are available to objects in the sub class without rewriting their declarations. This feature provides reusability where existing methods can be reused without redefining. Additionally new data and method members can be added to the sub class as an extension. It also allows the methods to be redefined in subclass as per need.

In class diagram, inheritance is represented using an arrow pointing to superclass as shown in figure 6.8. In this example, 'Person' is a superclass and 'Teacher' is a subclass.

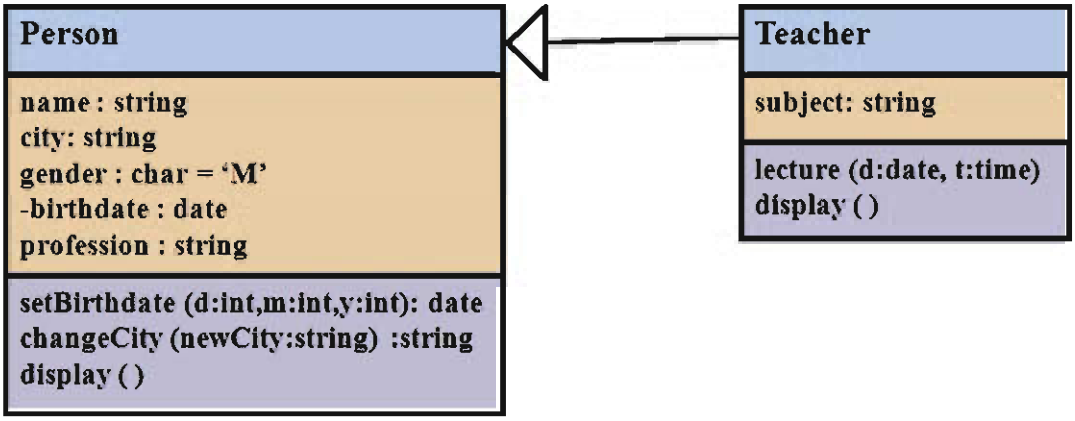


Figure 6.8 : Example of Inheritance

Generalization is another name for inheritance or "is a" relationship. It refers to a relationship between two classes where one class is a specialized version of another. Common attributes and methods are defined in superclass. Subclass is a specialized version with additional attributes and methods.

Sometimes, there may be a classical hierarchy of inheritance between classes. For example, class 'Employee' can be derived from class 'Person', then class 'Teacher' can be derived from 'Employee'. Here employee is a kind of person and teacher is a kind of employee. Such type of inheritance is known as multilevel inheritance. An example of multilevel inheritance is shown in figure 6.9.

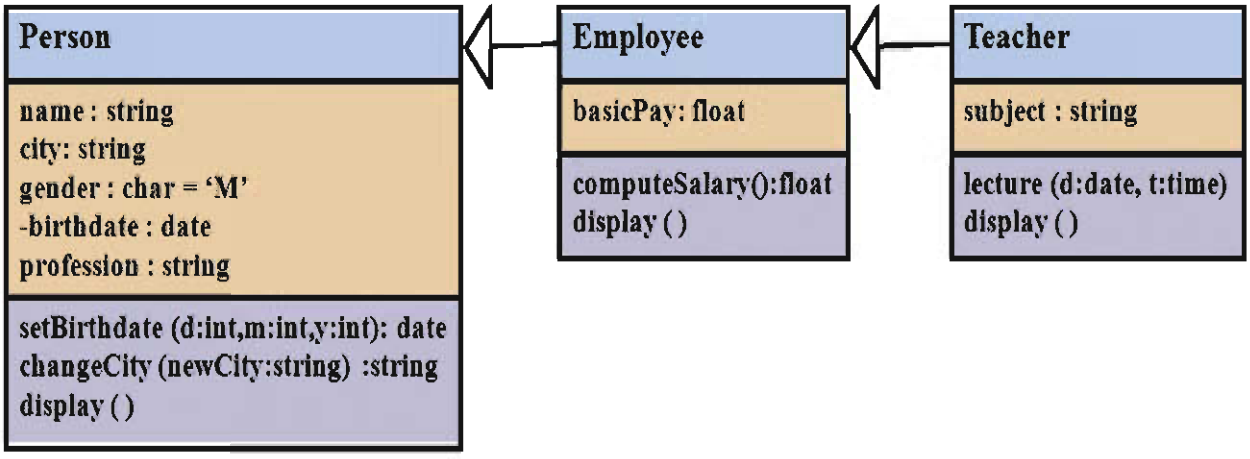


Figure 6.9 : Example of multilevel inheritance

A class can also be derived using more than parent classes. For example, a child inherits the characteristics of both mother and father; airplane is a kind of vehicle as well as flying object. When a class is derived from two or more classes, it is known as multiple inheritance.

Composition vs. Inheritance

In inheritance, class inherits from other classes in order to share, reuse or extend functionality. Here there exists 'a kind of' relationship between super class and subclass.

In composition, classes do not inherit from other classes, but are 'composed of' other classes. Class contains the attributes where some attributes are of objects of other class types.

Note that there other types of relationships and also constraints that can be represented in class diagram. The study of all the concepts is out of scope of this book.

Summary

Object-oriented as methodology is playing significant role in analysis, design and implementation of software system. In this paradigm, the focus is on objects which contain both data and functionality together. A class encapsulates attributes (data) and methods (behavior or functionality) together as a template that can be shared by all objects. Objects are then distinguished by their state; that is value of these attributes. More than one class may be associated with each other. When there is 'has-a' or 'a-part-of' relationship between two classes, the relationship is called aggregation or composition. When a class contains objects of other class, the container class is called owner class or whole class or aggregating class. The class that is contained in owner class is known as subject class or part class or aggregated class. Aggregation represents non-exclusive relationship between two classes. Composition represents exclusive relationship between two classes. When there is 'is-a' or 'a-kind-of' relationship between two classes, there is inheritance relationship. General or common features of two classes are implemented in the superclass and special features are implemented in subclass.

EXERCISE

1. List the features supported by object-oriented programming languages.
2. Distinguish between class and object.
3. Differentiate between 'Encapsulation' and 'Data Abstraction'.
4. What do you mean by polymorphism ? Name two type of overloading that may be supported to achieve polymorphism.
5. Explain the use of aggregation and composition.
6. When should one use inheritance ? Give example.
7. Explain different types of inheritance.
8. Choose the most appropriate option from those given below :

   (1) In Object-oriented methodology, the focus is on which of the following entities ?

   (a) Data (b) Functions (c) Objects (d) All of the above

(2) Which of the following best suits to Java ?

(a) A procedural programming language

(b) An Object-oriented programming language

(c) A Query language

(d) All of the above

(3) Which of the following is used to distinguish objects from each other ?

(a) Attributes (b) State (c) Behavior (d) All of the above

(4) Which of the following is used to define common features of similar objects ?

(a) Class (b) Object (c) Methods (d) All of the above

(5) Which of the following is not a visibility symbol ?

(a) ~ (b) * (c) # (d) -

(6) Which of the following is provided using encapsulation ?

(a) Data protection (b) Data sharing

(c) Separation of data and methods (d) All of these

(7) Which of the following is enabled by data abstraction ?

(a) Data protection (b) Data hiding

(c) To hide implementation details of method manipulating the data

(d) All of these

(8) With which of the following options polymorphism cannot be achieved ?

(a) Method overloading (b) Operator overloading

(c) Data hiding (d) All of these

(9) An aggregation model refers to which of the following relationships ?

(a) 'is-a' relationship (b) 'is-like' relationship

(c) 'a-part-of' relationship (d) All of these

(10) An inheritance model refers to which of the following relationships ?

(a) 'is-a' relationship (b) 'has-a' relationship

(c) 'a-part-of' relationship (d) All of these

(11) In class diagram, composition is represented using which of the following symbols ?

(a) Empty diamond symbol (b) Filled diamond symbol

(c) Empty triangle symbol (d) All of these

Java Basics 7

Java is an object-oriented programming language developed by Sun Microsystems, a company best known for its high-end Unix workstations. Modelled after C++, the Java language was designed to be small, simple, and portable across platforms and operating systems, both at the source and at the binary level. In this chapter, we will learn to start programming with Java. We will discuss simple Java statements, the basic things we can do in Java within a method main(). As we have already learnt C language, the basic constructs in Java will be very easy to understand. Most of the Java syntax is very similar to C language.

Introduction to Java

Java language was developed at Sun Microsystems in 1991. Java is small, fast, efficient, and easily portable to a wide range of hardware devices. It is considered as one of the ideal language for distributing executable programs via the World Wide Web, and also a general-purpose programming language for developing programs that are easily usable and portable across different platforms.

Java is an object-oriented language and here it differs from C. Using Java, we can take full advantage of object oriented methodology and its capabilities of creating flexible, modular and reusable code. It includes a set of class libraries that provide basic data types, system input and output capabilities, and other utility functions. These basic classes are part of the Java Development Kit (JDK). JDK has classes to support networking, common Internet protocols and user interface toolkit functions. Because these class libraries are written in Java, they are portable across platforms as all Java applications are.

Java is platform-independent at both the source and the binary level. Platform-independence is a program's capability of being moved easily from one computer system to another. At the source level, Java's primitive data types have consistent sizes across all development platforms. At binary level, platform-independence is possible due to bytecode interpreter. The designers of Java chose to use a combination of compilation and interpretation. The scenario of platform independence is shown in figure 7.1.

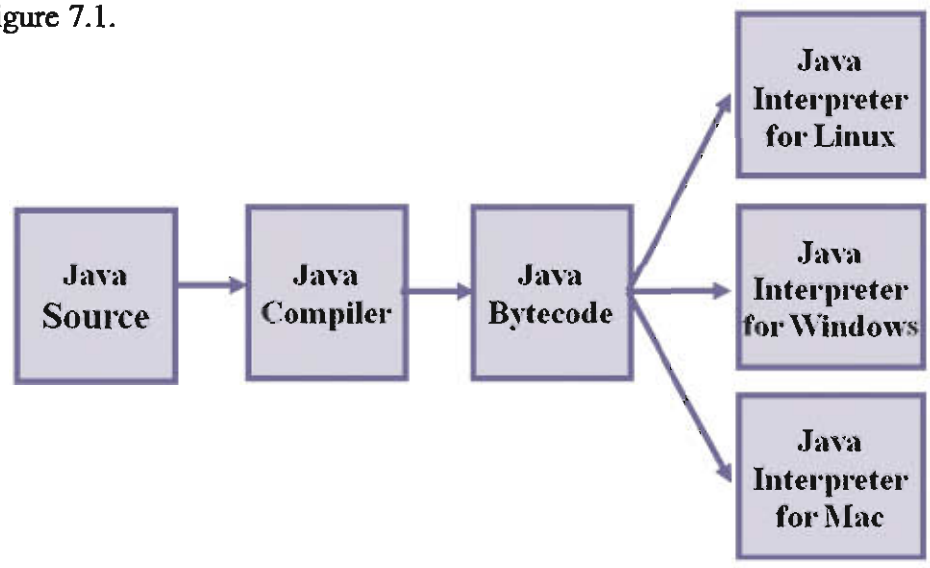


Figure 7.1 : Java : platform-independent

Programs written in Java are compiled into machine language for a computer that doesn't really exist. This so-called "virtual" computer is known as the Java Virtual Machine (JVM). The machine language for the Java Virtual Machine is called Java bytecode. Different Java bytecode interpreter is needed for each type of computer. Java binary files are actually in a form called bytecodes that is not specific to any one processor or any operating system. The only disadvantage of using bytecodes is its slow execution speed. There are tools available tools to convert Java bytecodes into native code. Native code is faster to execute, but then it does not remain machine independent.

Creating Simple Java Application

Before we learn the basics of Java programming language, let us begin exploring Java with a simple program that computes phone call charges and update the pre-paid balance amount. Here, we will learn how to create, compile and run Java programs.

A Java program is composed of classes. It should have at least one class and it must have main method in it. C programmers can think of a class as a sort of creating a new composite data type by using struct and typedef. Classes, however, can provide much more than just a collection of data. Note that typedef is not available in Java.

Let us give a name 'CallCost' to this Java application that computes call charges and update balance. To create and execute 'CallCost' application, we need to perform the following steps:

1. Create Java source file using any plain ASCII text editor.
   - Choose any text editor and type the program as given in code listing 7.1.
   - Save the source file with name 'CallCost.java'. Conventionally, Java source files are given the same name as the class they define, with an extension of .java. Note that the class name and filename are case sensitive. So, if class name is CallCost, filename should be CallCost.java (Applicable to SciTE editors).
2. Compile the source file using the Java compiler.
   - To compile the Java program, type javac followed by the name of your source file: javac CallCost.java
   - If the compiler show any errors, go back and make sure that you've typed the program exactly as it appears in code listing 7.1.
   - When the program gets compiled without errors, compiler creates a file with extension .class in the same directory as your source file. See that compiler has created file 'CallCost.class'. This is our Java bytecode file that will be executed.
3. Run the application using Java interpreter.
   - In the JDK, the Java interpreter is called simply using java. Type java CallCost.
   - To execute the program, we only need the compiled class file, not the source code.
   - Interpreter java uses the bytecode CallCost.class and executes it.

```
/**
* This class implements a simple program that
* will compute the cost of phone call and update balance
*/
public class CallCost
{
public static void main(String[] args)
{
/* declare variables */
double balance;        // balance amount in rupees
double rate;           // call rate; rupees per second
double duration;       // call duration in seconds
double cost;           // cost of last call

/* computations. */
balance = 170;
rate = 1.02;
duration = 37;
cost = duration * rate;              // compute the cost
balance = balance - cost;     // update balance amount

/* display results */
System.out.print("Call Duration: ");
System.out.print(duration);
System.out.println(" Seconds");
System.out.println("Balance: " + balance + "Rupees ");

} // end of main()
} // end of class CallCost
```

Code Listing 7.1: Sample Java Program

The process of compiling and executing a Java application is shown in figure 7.2.

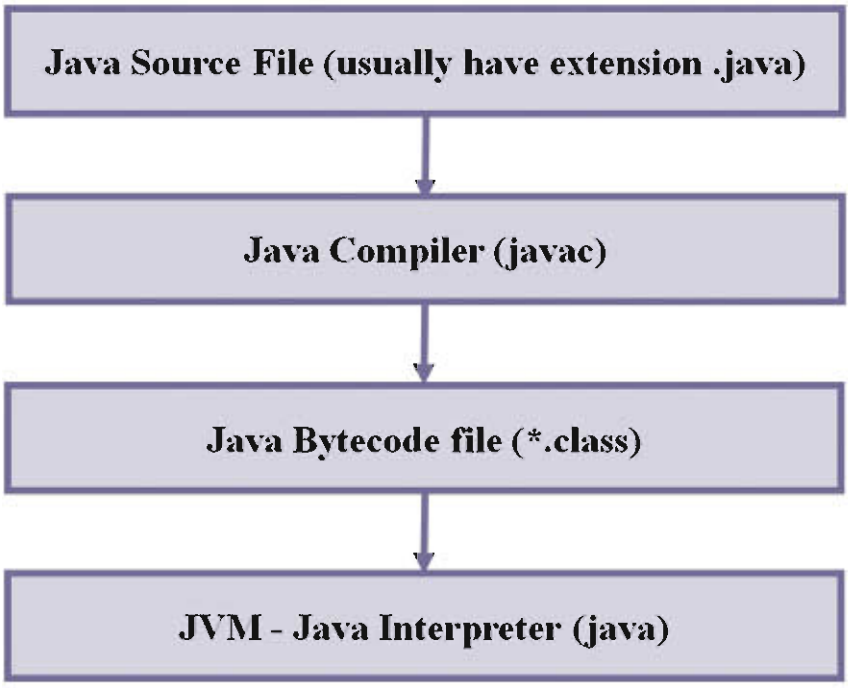


Figure 7.2 : Compilation Process

Java source program file should have extension .java. It's name should be same as that of class when class is public. Note that the name is case sensitive. The compiler 'javac' compiles source file and creates Bytecode file. The name of bytecode file is same as the class name containing main method and it has an extension .class. An interpreter 'java' interprets bytecode and executes it.

Figure 7.3 shows how to compile and execute the program in linux environment using a terminal.

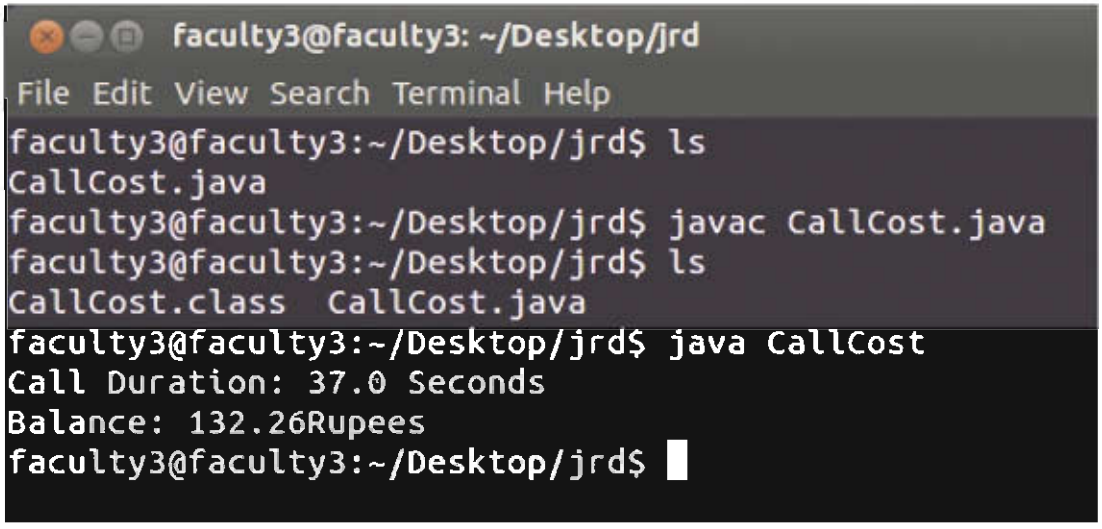


Figure 7.3 : Compilation and Execution of Java program in linux terminal

Notice two main parts in code listing 7.1.

1. The program is enclosed in a class definition; here, a class called 'CallCost'.
2. The body of the program is contained in a routine called main(). In Java applications, main() is the first routine that is run when the program is executed.

Explanation of Code Listing 7.1

The text written after // and enclosed within /* and */ are comments. As we know, comments are not compiled or interpreted.

- Variables are declared using data type followed by variable name.
- Computation part contains expressions including assignment statements.
- Here, several subroutine (also called function or method in Java) call statements are used to display information to the user of the program.
  - Methods used to display results : System.out.print and System.out.println. Both these methods take a value to be displayed as an argument.
  - Method System.out.println adds a linefeed after the end of the information that it displays, while System.out.print does not.
  - Call duration is displayed using three calls. First call displays label 'Call Duration:', second call displays the value of variable 'duration', third call displays label 'Seconds' and then bring the cursor in the next line. Note that string literal is enclosed in double quotes.
  - Balance is displayed using single call. Notice the use of + operator in Java in the parameter passed. It first evaluates the expression given as parameter and then display.

When we run the program, the Java interpreter calls the main() method and the statements that it contains are executed. These statements tell the computer exactly what to do when the program is executed. The main() routine can call other subroutines that are defined in the same class or even in other classes, but it is the main() routine that determines how and in what order the other subroutines are used.

The word "public" in the first line of main() means that this routine can be called from outside the program. This is essential because the main() routine is called by the Java interpreter, which is external to the program. The remainder of the first line of the routine is harder to explain at the moment; so for now, just think of it as part of the required syntax.

Using SciTE

Let us create one more Java application using SciTE editor. Here our application will compute simple interest and display the results. Figure 7.4 shows the code listing and its output. Perform following steps :

- Start SciTE application. Select **File → New**.
- Type the Java program as per code listing given in figure 7.4 and save this source program in a file with name Interest.java. To save the file, select **File → Save** command.
- Compile source program using **Tools → Compile** command.
- If the program is compiled without any error, execute it using **Tools → Go** command.

Interest.java * SciTE

File Edit Search View Tools Options Language Buffers Help

1 Interest.java *

```
// compute simple interest

public class Interest
{
    public static void main(String[] args)
    {
        /* declare variables */
        double principal;   // principal amount in rupees
        double rate;        // interest rate in percentage
        double duration;    // number of years
        double maturity;    // maturity amount
        double interest;    // interest amount

        /* computations. */
        principal = 17000;
        rate = 9.50;
        duration = 3;
        interest = principal * duration * rate / 100; // compute interest amount
        maturity = principal + interest;              // compute maturity amount

        /* display results */
        System.out.println("Principal amount: " + principal + " Rupees");
        System.out.println("Deposit for duration of " + duration + " years");
        System.out.println("Interest Rate: " + rate + " %");
        System.out.println("Interest amount: " + interest + " Rupees");
        System.out.println("Maturity amount: " + maturity + " Rupees");
    } // end of main()
} // end of class Interest
```

```
>javac Interest.java
>Exit code: 0
>java -cp . Interest
Principal amount: 17000.0 Rupees
Deposit for duration of 3.0 years
Interest Rate: 9.5 %
Interest amount: 4845.0 Rupees
Maturity amount: 21845.0 Rupees
>Exit code: 0
```

Figure 7.4 : Executing Java Program using SciTE

Structure of a Java program

Programming languages differ from ordinary human languages in being completely unambiguous and very strict about what is and is not allowed in a program. The rules that determine what is allowed are called the syntax of the language.

Syntax rules specify the basic vocabulary of the language and how programs can be constructed using things like variables, expressions, statements, branches, loops and methods. A syntactically correct program is one that can be successfully compiled or interpreted.

A structure of Java program is shown in figure 7.5. Here, text in angle bracket < and > is used as a placeholder that describes something actual we need to type while writing actual program. The definition of the method (function) in Java consists of function header and the sequence of statements enclosed between braces { and }.

As Java is an object-oriented language. Here, everything is defined as part of class. Thus, a method can't exist by itself. It has to be part of a class.

```
public class <class-name>
{
        <optional-variable-declarations-and-methods>
        public static void main(String[] args)
        {
                <statements >
        }
        <optional-variable-declarations-and-methods>
}
```

Figure 7.5 : Structure of a Java Program

- <class-name> in the first line is the name of the class having main method in it.

  If the name of the class is CallCost, then the program should conventionally be saved in a Java source file with a name CallCost.java. When this file is compiled, another file named CallCost.class is generated. This class file is given a name using class name. Class file, CallCost.class, contains the translation of the program into Java bytecode, which can be executed by a Java interpreter.

- Variable and method declaration after and before main() method is optional.
- Each program must have one class that contains public method main().
- Java is a free-format language. The layout (such as the use of blank lines and indentation) of the program in code listing 7.1 is not a part of the syntax or semantics of the language. The computer doesn't care about the program layout. We can write the entire program together on single line as far as it is concerned. However, layout is important to human readers.
- A program can contain other methods besides main(), as well as other variables. We will learn more about these later.

Now, let us start examining simple Java statements; the basic things we can do in Java within a method definition such as main(). We will learn the following with syntax :

- Data types
- Variables
- Literals
- Comments
- Java statements and expressions
- Arithmetic operators
- Comparisons
- Logical operators

Data Types

Data type determines the required memory size, type of values, range of values and type of operations that can be performed. Java supports eight primitive data types that handle common types for integers, floating-point numbers, characters, and boolean values (true or false).

The primitive data types are named **byte, short, int, long, float, double, char, boolean.** The first four types hold integers (whole numbers such as 17, -38477, and 0), next two hold real numbers (such as 5.8, -129.35), char holds a single character from the Unicode character set and boolean holds one of the two logical values true or false.

These data types are called primitive as they are built into the system. Note that these data types are machine-independent, which means that we can rely on their sizes and characteristics to be consistent across all Java programs on all machines. Table 7.1 lists the details of data types.

Data Type	Storage Space	Type of Value	Range of Values	Default Value
byte	1 byte	Integer	-128 and 127	0
short	2 bytes	Integer	-32768 to 32767	0
int	4 bytes	Integer	-2147483648 to 2147483647	0
long	8 bytes	Integer	-9223372036854775808 to 9223372036854775807	0
float	4 bytes	Real	10 ±38 with about 7 significant digits	0
double	8 bytes	Real	10 ±308 with about 15 significant digits	0
char	2 bytes	Character	16-bit Unicode character	0
boolean	1 byte	Boolean	true, false	false

Table 7.1 : Data Types in Java

Integer numbers with b bits precision store signed values in the range of $(-2^{b-1}-1, 2^{b-1})$. When they are preceded with keyword unsigned, the values are in the range of $(0, 2^{b-1})$.

Real numbers in Java are compliant with IEEE 754 (an international standard for defining floating-point numbers and arithmetic). Java uses the Unicode character set. The char type has 16 bits of precision and is unsigned. This allows thousands of characters from many different languages and different alphabets to be used in Java. Data type boolean is not a number, nor can it be treated as one.

Variable

If we want anything to be remembered by the computer during program execution, then it needs to be stored in the memory of a computer. Programs manipulate the data that are stored in memory. In machine language, data can only be referred to by giving the numerical address of the location in memory where it is stored. In a high-level language such as Java, names are used instead of numeric address of memory location to refer to data. The programmer has to remember only the name. A name used to refer to the data stored in memory is called a variable.

A variable can take different data values at different times during the execution of the program, but it always refers to the same memory location. A variable can be used in a java program only if it has first been declared.

One or more variables can be declared in Java using declaration statement with following syntax:

<type-name> {variable-names};

The conventions used here in the syntax are as follows:

- angle brackets < > denote the item to be specified by user
- curly brackets { } denote the list of items separated by commas

Here {variable-names} denote the list of variable names. When the list contains more than one item, items should be separated by commas.

<type-name> is to be replaced with the keyword denoting the data type of the variables. It is used to determine the size of variable, the values it can hold and the operations that can be performed on it. When the computer executes a variable declaration statement, it sets aside memory for the variable and associates the variable's name with that memory.

Some examples of variables are as mentioned :

int marks;

double amount, interest;

float rate;

char grade;

boolean isPass;

To define variable name, we need to follow certain rules as mentioned :

- It must begin with an alphabet, underscore (_) or dollar sign ($). After first character, it may contain digits, alphabets, $ and underscore.

  Some legal names are: birth_date, result, CallCost, top5students, date, amount$, $price

  Some illegal names are: 4me, %discount, birth date

- No spaces are allowed in variables. Thus, birth date is invalid variable name.
- It cannot be a reserved word. Reserved words have special use in Java and cannot be used by the programmer for other purposes. Examples of some reserved words are class, public, static, if, else and while.

Guidelines for naming variables :

- Choose meaningful variable names. Java allows variable name of any length.
- When a name includes several words, such as 'balance amount', follow one of the conventions mentioned below :
  - Capitalize the first alphabet of each word, except for the first word. This is sometimes referred to as camel case, since the upper case letters in the middle of a name are supposed to look like the humps on a camel's back. Example: balanceAmount, birthDate

- Separate words with underscore. Example: balance_amount, birth_date

- Note that Java is case-sensitive. So, upper case and lower case letters are considered to be different. Thus, variable names balance and Balance are different.
- It is customary for names of classes to begin with upper case letters, while names of variables and of methods begin with lower case letters.

Good programming style for declaring variables :

- Declare only one variable in a declaration statement.
- Include a comment with each variable declaration to explain its purpose in the program.
- Declare important variables at the beginning of the function. Declare variables which are not important to the overall logic of the function at the point where they are first used.

In Java, there are three kinds of variables: instance variables, class variables, and local variables. Function parameters and variables declared in the function are considered as local variables. We will study about instance and class variables later.

We can also give each variable an initial value while declaring. For example,

```
int marksObtained, totalMarks=100, counter=0;

boolean isPass = true;
```

In general, syntax of declaring variables is as follows: <type name> {variable [= <value>,]};

Here items in square bracket [ ] denotes optional item.

Note that local variables are not initialized with default values. It is programmer's responsibility to assign value to such variables before their first use. Figure 7.6 shows the use of variable in java program.

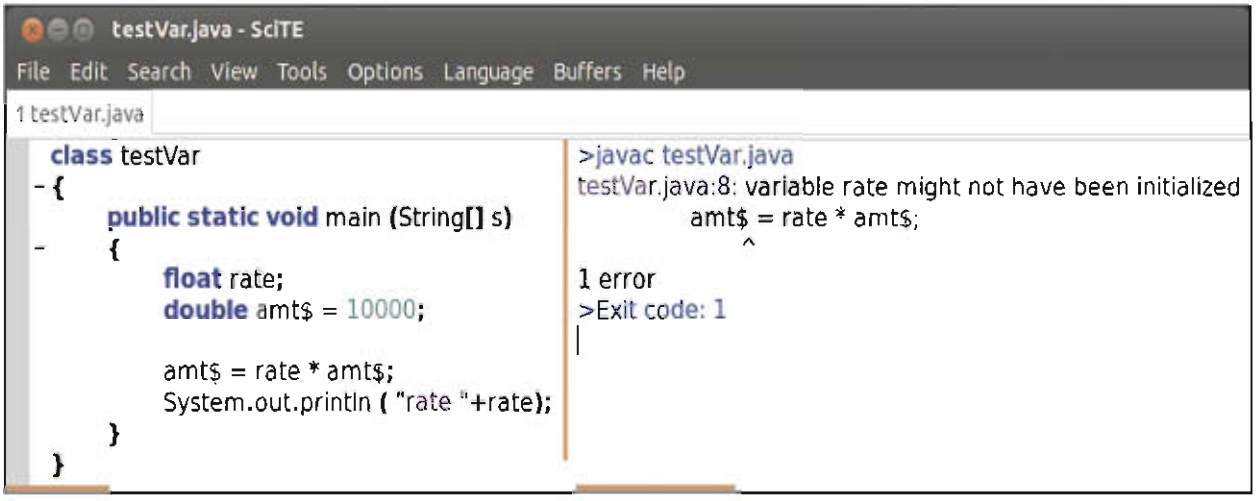


Figure 7.6 : Local Variables in Java

Observe that the local variable 'rate' is not assigned any value before its use in statement 'amt$ = rate * amt$'. When we try to compile the program the compiler will give an error as can be seen in figure 7.6.

Literals

A name used for a constant value is known as literal. There are some different kinds of literals in Java for number, character, string and boolean values.

Numeric Literals

Numeric literals are used to represent integer or real numbers for example, 157 and 17.42 are literals.

Integer literals are literals that are whole numbers. Java allows decimal (base 10), octal (base-8) and hexadecimal (base-16) and Unicode integer literals.

Ordinary integers such as 4, 157, 17777 and -32 are decimal integer literals of type byte, short or int depending on their size. A decimal integer literal larger than int is automatically of type long. We can force a smaller number to be a long by appending an L or l (upper or lower case letter l) as a suffix to that number (for example, 4L is a long integer having value 4). Negative integers are preceded by a minus (-) sign, for example, -45.

Octal numbers use only the digits 0 through 7. In Java, a numeric literal with a leading 0 (zero) is interpreted as an octal number. For example, literal 045 represents octal integer whose decimal number value is 37.

Hexadecimal numbers use 16 digits, the usual digits 0 through 9 and the letters A, B, C, D, E, and F. Upper case and lower case letters can be used interchangeably in this context. The letters A to F represent the numbers 10 to 15 respectively. In Java, a hexadecimal literal begins with 0x or 0X. Examples of hexadecimal literals are 0x45 or 0xFF7A. Hexadecimal numbers are also used in character literals to represent arbitrary Unicode characters.

A Unicode literal consists of \u followed by four hexadecimal digits. For example, the character literal '\u00E9' represents the Unicode character that is an "e" with an acute accent.

Java 7 also supports binary numbers, using the digits 0 and 1 and the prefix 0b (or 0B). For example: 0b10110 or 0b101011001011.

Real number literals are called floating point literals. These numbers can be represented using two types of notations: standard and scientific.

In standard notations, the integer part and the fractional part are separated with decimal point (.), for example 12.37.

In scientific notation, a number is followed by letter e (or E) and a signed integer exponent, for example 1.3e12 and 12.3737e-108. The "e12" and "e-108" represent powers of 10. Hence 1.3e12 means 1.3 times 1012 and 12.3737e-108 means 12.3737 times 10^{-108}. Scientific format can be used to express very large and very small numbers.

In Java, floating point literal by default is of the type double. To make a literal of type float, we have to append an "F" or "f" as a suffix to the number. For example, "1.2F" specifies literal 1.2 to be considered as a value of type float.

```
testVar.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 testVar.java

class testVar
{
        public static void main (String[] s)
        {
                float rate = 10.2;
                double amt$ = 10000;

                amt$ = rate * amt$;
                System.out.println ( "rate "+rate);
        }
}
```

```
>javac testVar.java
testVar.java:5: possible loss of precision
found   : double
required: float
                float rate = 10.2;
                             ^
1 error
>Exit code: 1
```

Figure 7.7 : Compilation error

Observe that in figure 7.7 we get a compilation error as we have assigned a literal 10.2 to variable of float type. To make the program work we have to write this statement as "float rate = 10.2f;". The suffix F can be written in either lower or upper case.

Boolean literals

For the type boolean, there are precisely two literals: true and false. These literals are to be typed without quotes. They represent values, not variables. Boolean values occur most often as the values of conditional expressions. In C, 0 is treated as false and non-zero value is treated as true. In Java, literals true and false are not associated with any numeric value.

Character literals

Character literals are expressed by a single character surrounded by single quotes: 'a', '#', '3'and so on. Characters are stored as 16-bit Unicode characters. Certain special characters have special literals that use a backslash (\) as an "escape character". Table 7.2 lists the special codes that can represent nonprintable characters, as well as characters from the Unicode character set.

Escape code	Meaning
\n	New line
\t	Tab
\b	Backspace
\r	Carriage return
\f	Form feed (New page)
\\	Back slash character
\'	Single quote character
\"	Double quote character
\ddd	Character represented by three octal digits (d: 0 to 7)
\xdd	Character represented by two hexadecimal digits (d: 0 to 9, a to f)
\udddd	Character represented by Unicode number dddd (d: hexadecimal digit)

Table 7.2 : Escape Codes

String literals

In Java all the other data types represent objects rather than "primitive" data. We are not concerned with objects for the time being. However, we will consider here one predefined object type that is very important: the type String. A String is a sequence of characters.

In code listing 7.1, we have used a string literal: "Balance:". String literal is a sequence of characters enclosed in double quotes. Within a string, special characters can be represented using the backslash notation as given in Table 7.2.

For example, to represent the string value: Many "Congratulations!", we need to type string literal: "Many, \"Congratulations!\""

Similarly in string "This string brought to you by Java\u2122", the Unicode code sequence \u2122 produces a trademark symbol (™).

Comments

Comments in a program are for human readers only; they are entirely ignored by the computer. Comments are important to make the program easy to understand for everyone.

Java supports following types of comments :

- Single-line comment: It begins with double slashes (//) and extends till the end of a line. The computer ignores the // and everything that follows it on the same line.
- Multi-line comment: It begins with /* and ends with */. This type of comment is usually used to have more than one line as comments. Actually, one may embed phrase as well as one or more entire lines as comments between /* and */. Any text between two delimiters /* and */ is considered as a comment. Comments cannot be nested; that is, we cannot have a comment inside a comment.
- Documentation comment: These type of comments begin with /** and end with */. They are used for creating API documentation from the code. These are special comments that are used for the javadoc system. Discussion of javadoc is out of scope of this book.

Other than comments, everything else in the program is required to follow the rules of Java syntax.

Expressions

Expressions are an essential part of programming. The basic building blocks of expressions are literals (such as 674, 3.14, true, and 'X'), variables, and function calls. Recall that a function is a subroutine that returns a value.

Simple expression can be a literal, a variable, a function call. More complex expressions can be built up by using operators to combine simpler expressions.

Operators include arithmetic operators (+, -, *, /, %), comparison operators (<, >, =, ...), logical operators (and, or, not...). When several operators appear in an expression, there is a question of precedence, which determines how the operators are grouped for evaluation. For example, in the expression "A + B * C", B*C is computed first and then the result is added to A. We say that multiplication (*) has higher precedence than addition (+). If the default precedence is not what

we want, we can use parentheses to explicitly specify the grouping. For example, in expression "(A + B) * C", it adds A and B first then multiplies the result by C.

Operators

Operators are special symbols used to build an expression. Java supports many types of different operators. In this chapter, we will discuss the following operators :

- Arithmetic operators
- Comparison operators
- Logical operators
- Conditional operator
- Assignment operator

Arithmetic operators

In Java, basic arithmetic operators are addition (+), subtraction (-), multiplication (*), division (/) and modulus (%). All these operators are binary, they take two operands. Operators + and - can be used as unary (taking only one operand) also. All operators can be applied on any type of numeric data: byte, short, int, long, float, or double. They can also be used with values of type char, which are treated as integers in this context. With char type of data, its Unicode code number is considered as its value when it is used with an arithmetic operator. Arithmetic operators are described in table 7.3.

Operator	Meaning	Example	Result
+	Addition	2 + 8	10
-	Subtraction	2 - 8	-6
*	Multiplication	2 * 8	16
/	Division	8 / 2	4
%	Modulus (Gives remainder after division where quotient is an integer value)	8 % 3 25.8 % 7	2 4.8

Table 7.3 : Arithmetic operators

Data type of the result after binary arithmetic operation :

- When both operands are of same data type, the data type of the result is same as the type of operands.
  - If both operands are integers, result is an integer. For example, result of 9/2 is 4 and not 4.5. Integer division results into integer value and remainder is discarded.
  - If both operands are float, result is float. Example : 9f/2f results in 4.5.

- When both operands are of different data types,
  - First of all, lower range data type is implicitly converted to higher data type to have the same types of operands. This type of conversion is also known as promotion.
  - Now, the result of an expression will be same as higher range operand.
  - Example: 4 + 3.5 results in 7.5, 9/2.0 results in 4.5, 9f/2 results in 4.5.

Points to remember :

- With modulus operator %, if first operand is negative, the result is negative.
- In Java, % operator can be used with floating point data types also. The result is the remainder after integer quotient. Thus for 25.8 % 7, integer quotient is 3 and remainder is 25.8 (3*7) = 4.8.
- Operator + can also be used to concatenate a string.

When operator + is applied with one of the operand of type String, other operand is automatically converted into type String. This is also an example of implicit type conversion. This type of conversion is seen in code listing 7.1 in an expression ' "Balance: " + balance ' while printing balance amount. It is also used in code listing shown in figure 7.8.

ArithOperators.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 ArithOperators.java

```
class ArithOperators
{
    public static void main (String[] args)
    {
        short x = 6;
        int y = 4;
        float a = 12.5f;  // suffix f to explicitly specify floating-point literal
        float b = 7.2f;   // suffix f for floating-point literal

        System.out.println ("x is " + x + ", y is " + y);
        System.out.println ("x + y = " + (x + y));
        System.out.println ("x - y = " + (x - y));
        System.out.println ("x / y = " + (x / y));
        System.out.println ("x % y = " + (x % y));
        x = -6;
        System.out.println ("x % y = " + (x % y));
        y=-4;
        System.out.println ("x % y = " + (x % y));
        x=6; y=-4;
        System.out.println ("x % y = " + (x % y));

        System.out.println ("a is " + a + ", b is " + b);
        System.out.println ("a / b = " + (a / b));
        System.out.println ("a / x = " + (a / x));
        System.out.println ("a % x = " + (a%x));
        System.out.println ("a % b = " + (a%b));
    } // end main()
} // end class ArithOperators
```

```
>javac ArithOperators.java
>Exit code: 0
>java -cp . ArithOperators
x is 6, y is 4
x + y = 10
x - y = 2
x / y = 1
x % y = 2
x % y = -2
x % y = -2
x % y = 2
a is 12.5, b is 7.2
a / b = 1.7361112
a / x = 2.0833333
a % x = 0.5
a % b = 5.3
>Exit code: 0
```

Figure 7.8 : Java program showing use of arithmetic operators

Explanation of program shown in figure 7.8 :

- We initially define four variables in main() method: x and y, which are integers (type short and int); a and b, which are floating point real numbers (type float).

- Remember that default type for floating-point literals (such as 12.5) is double. So, f is suffixed with literals 12.5 and 7 to treat them as float type.
- The System.out.println() method merely prints a message to the standard output device. This method takes a single argument, a string, but we can use + to concatenate values into a string. As pointed before, Java converts numbers into string and then concatenates the two.
- The result of x%y is 2 when x=6 and y=-4 because first operand is positive.
- The result of a%x is 0.5 where a=12.5 and x is 6. Here, integer quotient is 2 and the remainder is 0.5. Similarly in case of a%b where a=12.5 and b=7.2; the integer quotient is 1 and the remainder is 5.3.
- See the mixed type of operands in the call: System.out.println ("a / x = " + (a / x));. Here variable a is float and x is int, the result after division is float. While evaluating an expression, values are promoted to the higher data type operand.

Experiment with the same program after removing parenthesis of expression x+y in 2nd call of System.out.println method (i.e. in line 11 of code listing in figure 7.8, replace an expression "x + y = " + (x + y) with an expression "x + y = " + x + y) and analyse the results shown in figure 7.9.

ArithOperators.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 ArithOperators.java

```
class ArithOperators
{
    public static void main (String[] args)
    {
        short x = 6;
        int y = 4;
        float a = 12.5f;  // suffix f to explicitly specify floating-point literal
        float b = 7.2f;   // suffix f for floating-point literal

        System.out.println ("x is " + x + ", y is " + y);
        System.out.println ("x + y = " + x + y);
        System.out.println ("x - y = " + (x - y));
        System.out.println ("x / y = " + (x / y));
        System.out.println ("x % y = " + (x % y));
        x = -6;
        System.out.println ("x % y = " + (x % y));
        y=-4;
        System.out.println ("x % y = " + (x % y));
        x=6; y=-4;
        System.out.println ("x % y = " + (x % y));

        System.out.println ("a is " + a + ", b is " + b);
        System.out.println ("a / b = " + (a / b));
        System.out.println ("a / x = " + (a / x));
        System.out.println ("a % x = " + (a%x));
        System.out.println ("a % b = " + (a%b));
    } // end main()
} // end class ArithOperators
```

```
>javac ArithOperators.java
>Exit code: 0
>java -cp . ArithOperators
x is 6, y is 4
x + y = 64
x - y = 2
x / y = 1
x % y = 2
x % y = -2
x % y = -2
x % y = 2
a is 12.5, b is 7.2
a / b = 1.7361112
a / x = 2.0833333
a % x = 0.5
a % b = 5.3
>Exit code: 0
```

Figure 7.9 : Effect of removing ( ) from (x+y) in line 11of code listing in figure 7.8

Increment and Decrement operators

Unary operators ++ and -- are called the increment operator and the decrement operator respectively. Operator ++ adds 1 to a variable and - - subtracts 1 from the variable.

These operators can be used on variables belonging to any of the integer types and also on variables of type char.If x is an integer variable, we can use x++, ++x, x--, --x as expressions, or as parts of larger expressions.

When ++ or -- operator is used after variable name, it is known as post-increment or post-decrement operator. The old value of variable is used while evaluating the expression and thereafter the value of variable is incremented or decremented. For example, let the value of variable x be 3 before executing statement y = 4 + x++;. Here old value of x is used while evaluating expression on right-hand side and thereafter x is incremented. As a result, value of x will be 4 and y will be 7.

When ++ or -- operator is used before variable name, it is known as pre-increment or pre-decrement operator. Here the value of variable is incremented or decremented first and then this new value is used in expression. For example, in statement y = 4 + ++x;, if old value of x is 3, then value of x is incremented first and then this new value is used in evaluating an right-hand side expression. So value of x will be 4 and that of y will be 8.

When this operator is used in a standalone statement, use of pre or post does not make any difference. For example, statements x++; and ++x; are standalone statements.

Comparison operators

Comparison operators are also known as relational operators. The comparison operators in Java are: ==, !=, <, >, <=, and >=. The meanings of these operators are:

A == B	Is A "equal to" B ?
A != B	Is A "not equal to" B ?
A < B	Is A "less than" B ?
A > B	Is A "greater than" B ?
A <= B	Is A "less than or equal to" B ?
A >= B	Is A "greater than or equal to" B ?

These operators can be used to compare values of any of the numeric types as well as of char type. For characters, their unicode numeric values are used in comparison.

After applying comparison operator, the result of expression is boolean; either true or false. So, such expressions are also called boolean-valued expression.

We can also assign boolean-valued expressions to boolean variables, just as assigning numeric values to numeric variables.

Operators == and != can also be used to compare boolean values. For example :

```
boolean bothPositive;  bothPositive = ((x > 0) == (y > 0));
```

Usually, relational operators are used in if statements and loops.

Logical operators

Logical operators are also called boolean operators, as they operate on Boolean operands. In Java,

logical operations AND, OR, XOR and NOT are performed using operators &&, ||, ^ and ! respectively.

Boolean operator && is used to combine two boolean values for logical AND operation. The result of expression A && B is true only if both the operands A and B are true. For example, (x == 0) && (y == 0) is true if and only if both x is equal to 0 and y is equal to 0.

Boolean operator for logical OR is || (two vertical line characters). Expression A || B is true if either A is true or B is true, or if both are true. The result is false only if both A and B are false. For example, (x == 0) && (y == 0) is false only if both x and y are not equal to 0.

Boolean operator for logical NOT is denoted by ! and it is a unary operator. It results in complemented result. If the operand is true, result is false and vice versa.

In addition, there is operator ^ to perform logical XOR (exclusive OR). It returns true only if its operands are different (one true and one false) and false otherwise. Thus result is false when both operands have same boolean value. For example, (x == 0) ^ (y == 0) is true if only one of the x or y is zero.

Short circuiting

When using the conditional AND and OR operators (&& and ||), Java does not evaluate the second operand unless it is necessary to resolve the result. If first operand is false in case of &&, there is no need to evaluate second operand. Similarly, if first operand is true in ||, there is no need to evaluate second operand.

For example, consider expression (x != 0) && (y/x > 1). If the value of x is zero, first sub-expression (x !=0) results in false. As logical operator && results in false when any one of the operands is false, there is no need to evaluate second sub-expression (y/x > 1). Here, the evaluation has been short-circuited and the division by zero is avoided. Without the short-circuiting, there would have been a 'division by zero' error at runtime.

Conditional operator

The conditional operator in Java is a ternary operator using three operands. It uses two symbols ? and : in the expression to delimit three operands. It takes the following form:

<boolean-expression> ? <expression1> : <expression2>

Here, the first operand is a boolean expression and is evaluated first. If its value is true, value of the entire expression is the value of second operand expression1; otherwise it evaluates to third operand expression2.

For example, consider statement: next = (N % 2 == 0) ? (N/2) : (3*N+1);

Suppose value of N is 8. The value of first operand (N % 2 == 0) is true, so value of right-hand side expression is the value of expression (N/2), i.e. 4. If N is odd, first expression results in false and it will assign the value (3*N+1) to next. Note that the parentheses in this example are not required, but they do make the expression easier to read.

Assignment

In Java, an expression containing assignment (=) operator is generally referred to as assignment statement.

Once a variable has been declared, we can assign a value to that variable by using the assignment operator '='. In Java, one of the ways to get data into a variable is with an assignment statement. An assignment statement takes the form :

< variable > = < expression >;

where <expression> represents anything that refers to or computes a data value.

When an assignment statement is executed, it first evaluates the expression on right side of = sign and then put the resulting data value into the variable on left side of = sign.

For example, consider the simple assignment statement: rate = 10.02f;

Here the variable in this assignment statement is 'rate', and the expression is the number '10.02f'. Execution of this statement replaces the value of float variable 'rate' by 10.02.

Now, consider another assignment statement: balance = balance - cost;

Here expression 'balance-cost' is evaluated first using the existing values of variables 'balance' and 'cost' in memory. There after the resulting value is placed in variable 'balance' replacing its older value.

When a variable is used in an expression on right-hand side, it refers to the value stored in the variable. When a variable is used on the left-hand side of an assignment statement, it refers to memory location that is named by the variable to place the value. So, variable on left-hand side of expression is called lvalue, referring to location in memory.

In general, the type of the expression on the right-hand side of an assignment statement should be the same as the type of the variable on the left-hand side. However, if they are not matching, the value of expression is automatically converted to match the type of the variable. If the data type of expression is larger than the variable on left-hand side, it may result in an error due to precision problem. For example, there may not be automatic conversion from int to short or double to float.

Shorthand assignment operators

Java also support shorthand version of assignment. It saves the typing time. It takes the form <variable> <operator> = <expression>. Its effect is same as <variable> = <variable> <operator> <expression>. Here the operator should be a binary operator using two operands.

Some examples of shorthand assignment operators are; a += b and q &&= p. Here a += b is same as a = a + b, while expression q &&= p is same as expression q = q && p.

Type cast

In some cases, we may want to force a conversion that wouldn't be done automatically. For this, we can use what is called a type cast. A type cast is indicated by putting a type name in parentheses before the value we want to convert. Thus it takes a form :

(<data-type>) <expression>

For example,

int a; short b;

a = 17; b = (short)a;

Here variable a is explicitly converted using type cast to a value of type short

Precedence and associativity of Java operators

Java has well-defined rules for specifying the order in which the operators in an expression are evaluated when the expression has several operators. The operators are evaluated as per their priority (or precedence).

Precedence Order

When two operators are having different priority, then an operator with the higher precedence is operated first. For example, in expression a + b * c, multiplication is having higher precedence than addition, so b*c is evaluated first and then the result is added to a. Thus it is as good as writing a + (b*c). Precedence rules can be overridden by explicit parentheses. So, instead of creating confusion, use parenthesis freely.

Associativity

When two operators with the same precedence appear in the expression, the expression is evaluated according to its associativity. Associativity determines the direction (left-to-right or right-to-left) in which operations are performed. In most of the cases, associativity is from left to right. For unary operations and assignment, it is from right-to-left.

Table 7.4 gives the list of operators discussed in this chapter, listed in order from highest precedence (evaluated first) to lowest precedence (evaluated last):

Operations	Operators	Associativity
Unary operations	++, --, !, unary - and +, type-cast	Right-to-left
Multiplication, division, modulus	*, ?, %	Left-to-right
Addition and subtraction	+, -	Left-to-right
Relational operators	<, >, <=, >=	Left-to-right
Relational operators (Equality and inequality)	==, !=	Left-to-right
Logical AND	&&	Left-to-right
Logical OR	||	Left-to-right
Conditional operator	? :	Right-to-left
Assignment operators	=, +=, -=, *=, /=, %=	Right-to-left

Table 7.4 : Operators and their precedence

For example x = y = z = 7 is treated as x = (y = (z = 7)), leaving all three variables with the value 7. This is due to right-to-left associativity of assignment (=) operator. Remember that an assignment is an operator, so assignment statement evaluates to the value on the right hand side expression. Thus, in x=y=z=7, it evaluates z=7 first. This assigns value 7 to variable z and value of expression z=7 is also 7, which is making the expression as x=y=7.

On the other hand, 72 / 2 / 3 is treated as (72 / 2) / 3 since the / operator has left-to-right associativity. It is to be noted that there is no explicit operator precedence table in the Java Language Specification and different tables on the Web and in textbooks disagree in some minor ways.

Control Structures

In general, the statements are executed sequentially, one by one. Sometimes, program logic needs to change the flow of this sequence. The statements that enable to control the flow of execution are considered as control structures.

There are two types of control structures: loops and branches. Loops are used to repeat a sequence of statements over and over until some condition occurs. Branches are used to choose among two or more possible courses of action, so also called selective structure.

In Java, control structures that are used to determine the normal flow of control in a program are: if statement, switch statement, while loop, do..while loop and for loop. Each of these structures is considered to be a single statement, may be a block statement.

Block

A block statement is a group of statements enclosed between a pair of braces, "{" and "}".

The format of a block is:

```
{
        <statements>
}
```

Block can be used for various purposes as follows :

- To group a sequence of statements into a unit that is to be treated as a single statement, usually in control structures. (will be discussed soon)
- To group logically related statements.
- To create variables with local scope for statements within a block.

For example,

```
{ // This block exchanges the values of x and y
        int temp;       // temporary variable for use in this block only
        temp = x;       // save a copy of x in temp
        x = y;          // copy y into x
        y = temp;       // copy temp into y
}
```

When we declare variables inside a block, they are local in that block and such variables will cease to exist after the block. We cannot use temp outside the block in which it is declared.

A variable declared inside a block is completely inaccessible and invisible from outside that block. When the variable declaration statement is executed, memory is allocated to hold the value of the variable. When the block ends, that memory is released and is made available for reuse. The variable is said to be local to the block.

There is a general concept called the "scope" of a variable. The scope of a variable is the part of the program in which that variable is valid. The scope of a variable defined inside a block is limited to that block only.

When we try to declare a variable with the same name as of the variable in scope, there will be an error. See code listing and the result shown in figure 7.10. Here, we have used labelled block only for better understanding. It is not must to use in the given example. We will see the use of labelled block later in this chapter.

In code listing shown in figure 7.10, we have tried to declare variable x in block labelled blk2. Block blk2 is in the scope of main method block. Variable x that is declared in the block of main method is also having its scope in blk2. Thus, declaring variable x in blk2 conflicts with variable x in the scope of main method block and compiler shows an error.

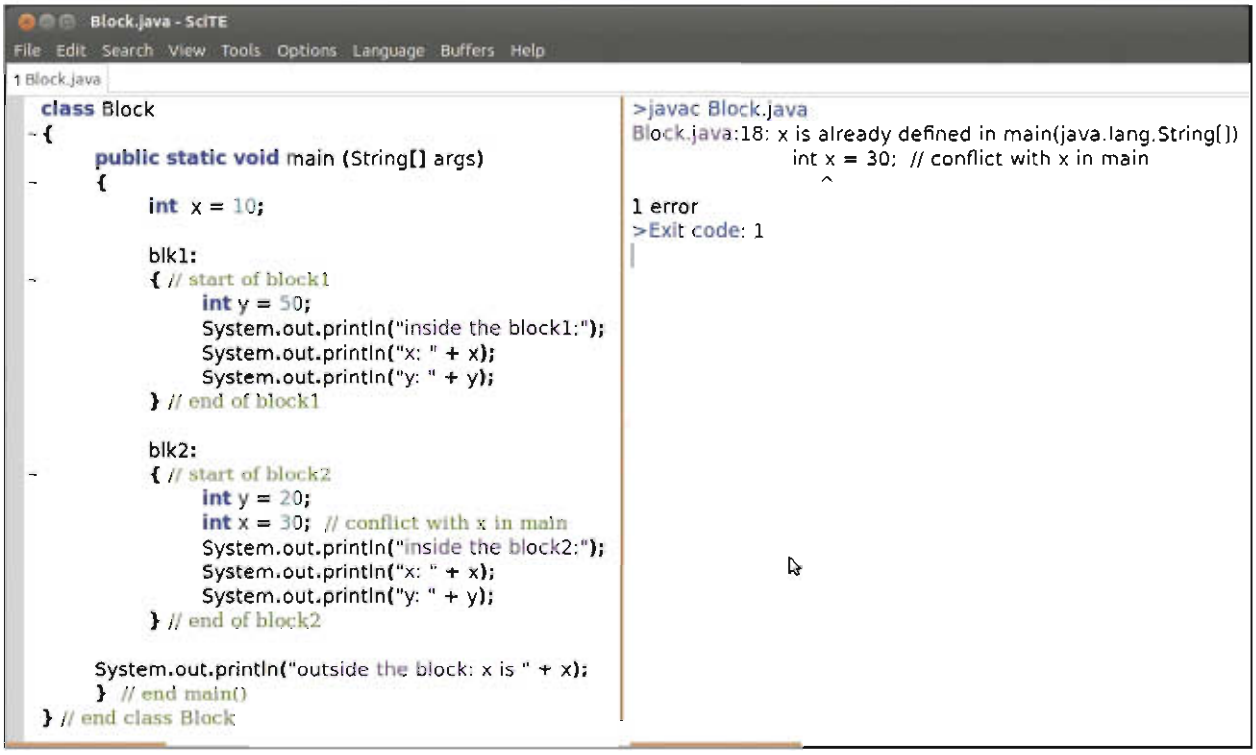


Figure 7.10 : Conflicting variable names in the same scope

Figure 7.11 shows the modified code. Here variable x is declared in the scope of entire main method. As blk1 and blk2 are in the same block, x is available in blk1 and blk2 as well. Variable y declared in labelled block blk1 ceases to exist outside that block. After execution of blk1, variable y is discarded. In blk2, when variable y is declared, it may be allocated any memory location from available memory. If we do not initialize value of variable y in blk2, it will return an error when we refer it in blk2.

This means that variable y of blk2 has nothing to do with variable y of blk1. Actually variable y of blk1 is not accessible in blk2.

```
Block.Java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 Block.java

class Block
{
        public static void main (String[] args)
        {
                int  x = 10;

                blk1:
                { // start of block1
                        int y = 50;
                        System.out.println("inside the block1:");
                        System.out.println("x: " + x);
                        System.out.println("y: " + y);
                } // end of block1

                blk2:
                { // start of block2
                        int y = 20;
                        //int x = 30;  // conflict with x in main
                        System.out.println("inside the block2:");
                        System.out.println("x: " + x);
                        System.out.println("y: " + y);
                } // end of block2

        System.out.println("outside the block: x is " + x);
        }  // end main()
} // end class Block
```

```
>javac Block.java
>Exit code: 0
>java -cp . Block
inside the block1:
x: 10
y: 50
inside the block2:
x: 10
y: 20
outside the block: x is 10
>Exit code: 0
```

Figure 7.11 : Program showing scope of variable

if Statement

The if statement when used in a program enables to take one of two alternative courses of action, depending on whether the value of a given boolean-valued expression is true or false. It is an example of a "branching" or "decision" or "selective" control structure. The form if statement is as shown in figure 7.12.

```
if ( <boolean-expression> )
{
        <statement1>
}
else
{
        <statement2 >
}
```

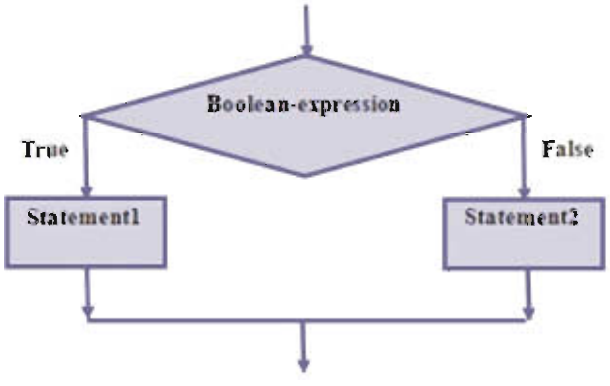


Figure 7.12 : Structure of if statement

When if statement is executed, it first evaluates boolean expression. If its value is true, it executes statement1 (all statements in a block); otherwise it executes a block of statements written after keyword else. <statement> in if statement is usually a block statement. It may be any single statement, but

it is advisable to use block to make it easy to insert other statements later. Refer following code snippet that uses if statement to determine whether an integer is even or odd.

```
if ( x%2 == 0 ) // assume int x
  { // divisible by 2
        System.out.print(x);
        System.out.println (" is even");
  }
else
  { // not divisible by 2
System.out.println (x + " is odd");
  }
```

Keyword else and a block after else are optional. So, if statement can be without else part as shown in the following form:

```
if ( <boolean-expression> )
{
        <statement1>
}
```

When an if statement is used in another if statement, it is called nested-if statement. See following example which determines grade based on marks obtained.

```
if ( marks >= 70 ) // int marks
  { grade = 'A';       // char grade
  }
else
  {
   if (marks >= 60)
        grade = 'B';
   else if (marks >= 50)
                grade = 'C';
        else
         grade = 'F';
 }
```

Let us consider one more example of nested-if statement.

```
if ( x > 0 )
        if (y > 0)
                System.out.println("both x and y are greater than zero");
else
        System.out.println("x <= 0");
```

Here, it seems that the else part is corresponding to "if (x > 0)" statement, but actually it is attached to "if (y > 0)", which is closer. Thus it is executing the true part of (x>0).

If we want to attach else to "if (x>0)", we should enclose the nested if in a block as shown here.

```
if ( x > 0 )
{
        if (y > 0)
                System.out.println("both x and y are greater than zero");
}
else
System.out.println("x <= 0");
```

Switch Statement

A switch statement is used when there are many alternative actions to be taken depending upon the value of a variable or expression. Here, the result of the test expression must be of the type taking discrete values. The form of switch statement is shown below :

```
switch (<expression>)
{
        case <constant-1>:
                <statements-1>
                break;
        case <constant-2>:
                <statements-2>
                break;

        // more such cases

        case <constant-n>:
                <statements-n>
                break;

        default:
                <statements-n1>
}
```

In the switch statement, the test expression should be of the type byte, char, short or int. It can also be of enum data type (not discussed here).

When executing switch statement, the value of the test expression is compared with each of the case values in turn from case1 onwards. If a match is found, the respective case statements (not necessarily a block) are executed. If no match is found, the default statement is executed. The default

is optional, so if there is no match in any of the cases and default doesn't exist, the switch statement completes without doing anything.

Note that break statement used after each case is not mandatory. Use of break statement is to break the switch statement, i.e. to jump at the first statement after the end of switch.

Repetitive control structures

Java supports three types of looping constructs: for, while and do...while. Figure 7.13 shows the syntax and structure of all these loops.

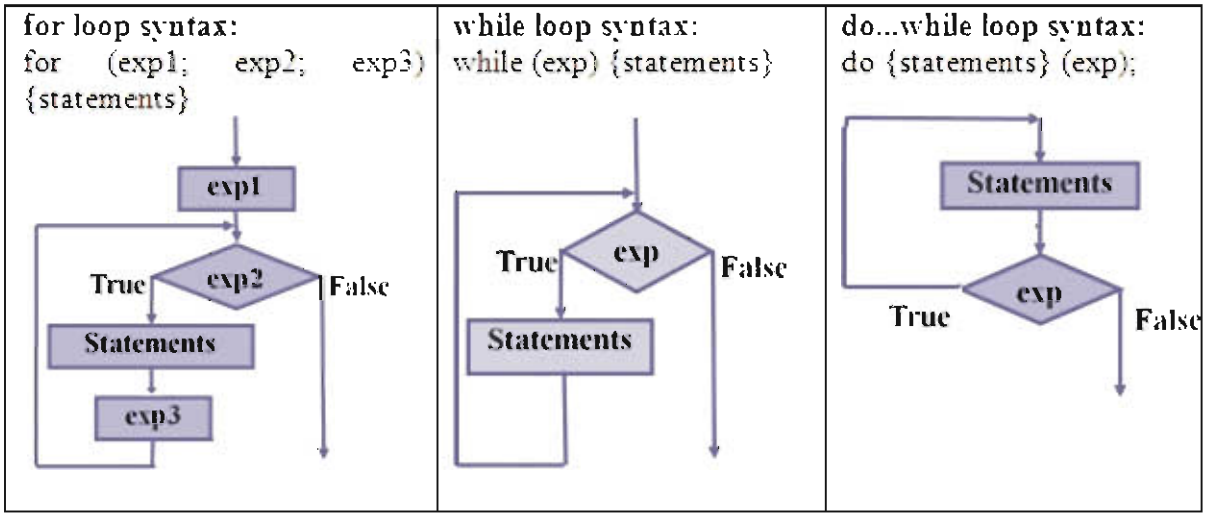


Figure 7.13 : Repetitive Control Structures

In for and while loop, test expression is evaluated first and the statements in the loop are executed if condition is true. These loops are entry-controlled or pre-test loop constructs. Here, it is possible that statements in a loop are not executed at all.

When number of iterations are pre-determined, usually for loop is used. In this loop, all the three expressions are optional. First expression is initializer, second expression is condition and third expression is an iterator. Thus for(;;) is valid; but requires some control statements to break the loop. If break statement is not executed, it will result in infinite loop.

In do...while loop, it evaluates the test expression after executing the statements in a loop. It repeats the loop if the test condition is true. Thus, do...while loop is exit-controlled or post-test loop construct. Here, statements of loop are executed at least once.

Following examples print integer values from 0 to 9 using all the three loop constructs.

Use of for loop :

```
for (int i = 0; i < 10; i++)
{
        System.out.println(i);
}
```

Use of while loop :

```
int i = 0;
```

```
while(i < 10)
{
        System.out.println(i++); //prints i before applying i++
}
```

Use of do...while loop :

```
int i = 0;
do
{
System.out.println(i++);
} while(i < 10);
```

Use of break and continue statement

The break statement is used to transfer the control outside switch or loop structure.

When break is used in switch structure, it skips all the following statements and control is transferred at the first statement after the end of switch statement.

In a loop, break statement is used to exit the loop. When break is executed in a loop, all the following statements in a body of the loop are skipped and no further iteration takes place. The control is transferred at the first statement after the end of loop structure. Remember that break jumps outside the nearest loop containing this statement.

Use of continue statement is used to skip the following statements in a loop and continue with the next iteration.

Both the statements break and continue are used the same way as in C language.

Nested loops

Loops of same or different types can be nested in Java. Figure 7.14 shows the use of nested loops, break and continue statement.

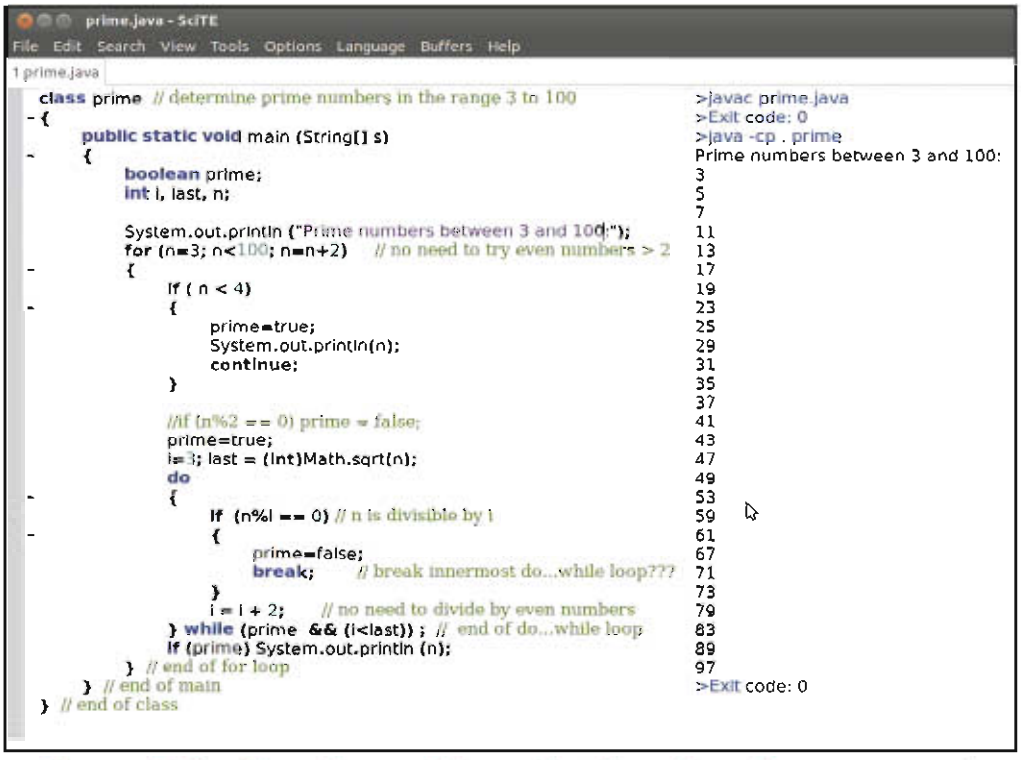


Figure 7.14 : Use of nested loops, break and continue statements

In figure 7.14 we have written a program that prints prime numbers between 3 and 100. Additionally, it also shows the use of sqrt function. The Java class libraries include a class called Math. The Math class defines a whole set of math operations. Function sqrt() is one of the static method member of the class Math and is invoked as Math.sqrt().

Labelled loops and labelled break

When we use nested loops, break statement breaks the nearest enclosing loop and transfers the control outside the loop. Similarly continue also restart the enclosing loop.

If we want to control which loop to break and which loop to reiterate, we can use labelled loop. To use a labelled loop, add the label followed by colon (:) before the loop. Then, add the name of the label after the keyword break or continue to transfer control elsewhere other than enclosing loop. Figure 7.15 shows a program that uses labelled loops.

When code in figure 7.15 is executed, the value of i*x is 350 when i=5 and x = 70. As the if condition in while loop is evaluated to true, it executes 'break out;' statement. This will terminate the outer for loop labelled as 'out'. If only break statement would have used, it would have exited an enclosing while loop and continued with next iteration in for loop. Here, it does not execute outer for loop for i=6 onwards.

LblBlock.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 LblBlock.java

```
class LblBlock  // Use of labelled block to break the outer loop
{
    public static void main (String[] s)
    {
        int x=0;
        out: for (int i = 4; i <10; i++)
        {
            x=10;
            while (x < 100)
            {
                System.out.println ( "Inside while loop: i is "
                                + i + ", x is " + x);
                if ( i * x == 350 )
                    break out;
                x = x+20;
            } // end while
            System.out.println ( "\nOutside while loop: i is "
                                + i + ", x is " + x + "\n");
        } // end out for loop
        System.out.println ( "\nOutside for loop: x is " + x);
    } // end of main
} // end of class
```

```
>javac LblBlock.java
>Exit code: 0
>java -cp . LblBlock
Inside while loop: i is 4, x is 10
Inside while loop: i is 4, x is 30
Inside while loop: i is 4, x is 50
Inside while loop: i is 4, x is 70
Inside while loop: i is 4, x is 90

Outside while loop: i is 4, x is 110

Inside while loop: i is 5, x is 10
Inside while loop: i is 5, x is 30
Inside while loop: i is 5, x is 50
Inside while loop: i is 5, x is 70

Outside for loop: x is 70
>Exit code: 0
```

Figure 7.15 : Use of labelled loop and labelled break

Let us consider another example of labelled loop and labelled break in a program that stops execution when first odd non-prime number in the range 40 to 100 appears. See figure 7.16.

LblLoop.Java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 LblLoop.java

```
class LblLoop  // determine first non-prime odd number in the range 41 to 100
{
    public static void main (String[] s)
    {
        boolean prime=true;
        int i, last, n;

        // break the loops as soon as first non-prime odd number is found
        forLoop: for (n=41; n<100; n=n+2) // no need to try even numbers >2
        {
            if ( n < 4)
            {
                prime=true;
                System.out.println(n);
                continue;
            }

            prime=true;
            i=3; last = (int)Math.sqrt(n);
            do
            {
                if  (n%i == 0) // n is divisible by i
                {
                    prime=false;
                    break forLoop;      // break for loop labelled 'forLoop'
                }
                i = i + 2;      // no need to divide by even numbers
            } while (prime  && (i<last)) ;  //  end of do...while loop
            if (prime) System.out.println (n + " is prime");
        }  // end of for loop
        if (!prime) System.out.println (n + " is not prime");
    }  // end of main
}  // end of class
```

```
>javac LblLoop.java
>Exit code: 0
>java -cp . LblLoop
41 is prime
43 is prime
45 is not prime
>Exit code: 0
```

Figure 7.16 : Program to print prime number between 40 and 100

Summary

In this chapter, we have discussed about the basics of Java. There are eight basic data types supported in Java. Character in Java is a 2-byte Unicode character. Various types of literals can be used in Java. By default, numeric literal is assumed to be of the type double. Control structures like if, switch, for loop, while loop and do...while loop are very similar as in C language. Block can be specified by enclosing the statements in a pair of curly braces { }. Scope of variables is in the block where they are defined.

EXERCISE

1. What is the size of character data type in Java ?
2. Write the number of bytes taken by int and long data types in Java.
3. Explain if and switch statement available in Java.
4. Discuss about various repetitive structures available in Java.
5. How can one use labels for a block or loop in Java ?

6. Choose the most appropriate option from those given below :

(1) How many basic (primitive) data types are supported in Java ?

(a) 2 (b) 4 (c) 8 (d) 16

(2) What is the default data type of floating point literal ?

(a) int (b) long (c) float (d) double

(3) Which character set is used for char data type in Java ?

(a) Unicode (b) ASCII (c) EBCIDIC (d) All of these

(4) Which of the following is compiled error free ?

(a) for(;;){int i=7}; (b) while (1){int i=7};

(c) while (True){int i=7}; (d) All of these

(5) Which of the following is not allowed as first character in valid variable name ?

(a) Underscore (_) (b) Digit (c) Letter (d) Dollar ($)

(6) Which of the following is not a basic data type in Java ?

(a) char (b) long (c) byte (d) String

(7) What is the default value of boolean type data ?

(a) null (b) true (c) false (d) 0

(8) What will be the result of arithmetic expression 7/2 ?

(a) 3 (b) 3.5 (c) 1 (d) 0

(9) What will be the result of arithmetic expression -7%2 ?

(a) -3 (b) -1 (c) 1 (d) -3.5

(10) What will be the result of arithmetic expression -7.5%2 ?

(a) -3 (b) -1.5 (c) 1.5 (d) Error

LABORATORY EXERCISE

Write a Java program for the following :

1. During a sale at a store, a 10% discount is applied to purchases over Rs. 5000. Write a program that assigns any value to variable 'purchase' and then calculates the discounted price. Display purchase amount and discount offered.

2. Write a program that determines the price of a movie ticket based on customer's age and the time of the show (normal or matinee). The normal show and matinee show ticket price for adult is Rs. 100 and Rs. 50 respectively. Adults are those over 13 years. The children's ticket price is Rs. 60 and Rs. 40 for normal show and matinee show respectively. Declare variables of suitable data types for age and show time, assign the values to these variables and print age, show time and the ticket price.

3. Write a program to display the grade based on percentage of marks using switch statement.

4. A bank gives loan to customers at a simple interest of 12% per annum. Interest for the whole term is charged at the beginning. Compute the monthly installments considering the term of 36 months for loan amount of Rs. 10000, 20000, 30000, 40000, 50000, 60000, 70000, 80000, 90000 and 100000.

5. Write a program that prints square root of integer numbers starting from 5 till the square root is at 50 or less.

Classes and objects in Java 8

As learnt earlier object and class are fundamental parts of object-oriented programming. Java is an object-oriented Language. This chapter explains how to create a class and object in Java programming. After studying this chapter, we will be able to get a clear picture as to what are objects and what are classes in Java.

Introduction

A class contains both data (referred to as attributes), and program code (functions referred to as methods).

In the previous chapter, we wrote programs using a class that contained a single method named 'main'. When the class was executed, the main method was called, and the application ran as a normal program. We have not used any data members in those examples.

While it is possible to use only a single class in a Java project, this is not a good practice for large applications. When designing software, we should divide the entire application into simpler components that perform logically related tasks. For each such component or module we may create a class.

Let us understand the concept of class and object in Java programming through an example of 'Room'. A room of a house, hostel, hotel or school possess common characteristics. Each room can be uniquely identified by its properties like length, width, height, number of windows, number of doors and direction. For simplicity, let us consider here length, width, height and number of windows.

We will now write a Java code as shown in code listing 8.1 to create a class named 'Room' with the attributes length, width, height, nWindows and three methods. The first method will be used to assign values to attributes. The second method will calculate the area, while the third will display its attributes. Thereafter, we will write code to use this class.

```
/* Class Room */
class Room
{
        float length, width, height;
        byte nWindows;

        void setAttr (float l, float w, float h, byte n)
        {
                        length = l; width = w; height = h;
                        nWindows = n;
        }  // end setAttr () method

      double area ( )  // area =  length * width
```

```
        {
                return (length * width);
        } // end area() method

      void display ( )
        {
                System.out.println ("\nLength: " + length);
                System.out.println ("Width: " + width);
                System.out.println ("Height: " + height);
                System.out.println ("Number of Windows: " + nWindows);
        } // end display() method
}  // end Room class

/* using Room class to create objects and run application */
class RoomDemo
{
      public static void main (String args[])
        {
                // Create a room object, assigned default values to attributes
                Room r1;  // reference variable with null value by default
                r1 = new Room();
                // both declare and create in one statement
                 Room r2 = new Room();

            // Display two room objects with initial default values
                r1.display();
                r2.display();

                // Assign values of attributes of objects
                r1.setAttr (18, 12.5f, 10, (byte)2);
                r2.setAttr (14, 11, 10, (byte)1);

                // Display updated contents
                r1.display();
                r2.display();

                // Display area
                System.out.println ("\nArea of room with length " + r1.length
                   + " width " + r1.width +  " is " + r1.area());
                System.out.println ("\nArea of room with length " + r2.length
                   + " width " + r2.width +  " is " + r2.area());
        }  // end main()
}  // end RoomDemo
```

Code Listing 8.1 : Creating and using class and objects

Here, first of all, code is written to create a class named 'Room'. Thereafter, code is written to create objects of class 'Room' and use its methods. To do so, we have created another class named 'RoomDemo' containing main() method. In a source file, these classes may appear in any order.

Here, we have separated creating 'Room' class performing logically related tasks and using this class in application class 'RoomDemo'.

When a program contains two or more classes, only one class can contain the main() method. Figure 8.1 shows the execution of the code in SciTE editor.

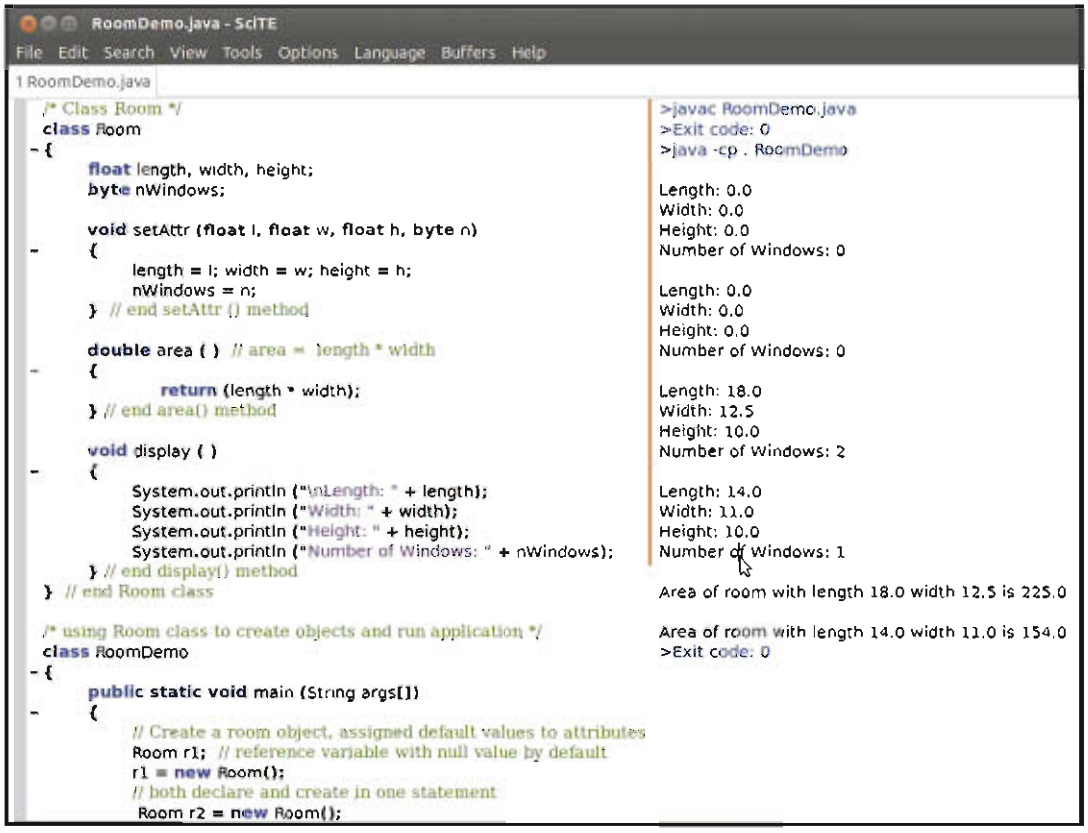


Figure 8.1 : Java Program to demonstrate use of multiple classes

In the given example of RoomDemo application as shown in figure 8.1, following operations are performed.

- Two objects r1 and r2 of class Room are created first. They are initialized with default values (Numeric values are zero by default).
- Contents of these two objects are displayed using display method.
- Invoking setAttr() method with numeric literals as parameters, attributes of room objects are modified for both the objects r1 and r2.
- The updated contents of two objects are displayed.

Finally area of both Room objects is displayed. Here area() method is invoked to get area.

Class in Java

We have already written one program that uses class. Let us now learn the syntax and other details of how to create class, objects and use them in an application.

A class is a template for multiple objects with similar features. Classes embody all the features of a particular set of objects. For example, class 'Room' is a template for all rooms with their common properties.

In Java, a class is defined using **class** keyword as follows :

```
class <ClassName >
{
        <Variables>
        <Methods>
}
```

Every class we write in Java is generally made up of two components: attributes and behaviour. Attributes are defined by variables in a class. Behaviour is defined by methods in a class. Methods are used to access or modify attributes.

In code listing 8.1, we have defined a class named 'Room' with attributes defined by variables named length, width, height and nWindows; and behaviour by methods named setAttr(), display() and area(). Another class is created just to create an application using 'Room' class.

Creating objects

Creating an object from a class requires the following steps :

- **Declaration :** A variable (reference variable) name of type class is declared with syntax <class name> <variable name>
- **Instantiation :** Keyword **new** is used to create the object by allocating memory
- **Initialization :** Constructor (a special type of method) is called to initialize the newly created object

In Java, a class is a type, similar to the built-in types such as int and boolean. So, a class name can be used to specify the type of a variable in a declaration statement, the type of a formal parameter, or the return type of a function.

To declare an object of class 'Room', use following statement :

```
Room r1;
```

Declaring a variable does not create an object. This is an important point to be remembered. In Java, no variable can ever store an object. A variable declared using class type can only store a reference to an object. So variables of class type are also referred to as reference variables. Here r1 is a reference variable.

Next step is to create an object. Using new keyword, we can create an object. Operator new allocates the memory for an object and returns the address of the object for later use. Reference is the address of the memory location where the object is stored. In fact, there is a special portion of memory called the heap where the objects live.

When an object is created, in addition to allocating memory, a special method called 'constructor' is executed to perform initial task. We will learn about constructors later in this chapter.

Let us create an object of type Room and assign its address to variable r1 as follows :

```
r1 = new Room();
```

Here, parentheses are important; don't leave them off. With empty parentheses without arguments, a default constructor is called. It initializes the attributes (variables) of the object using default values. The parentheses can contain arguments that determine the initial values of variables. This is possible by using user-defined constructor.

When statement "r1=new Room();" is executed, remember that object is not stored in variable r1. Variable r1 contains only address of an object.

Both the above steps used to declare and create an object can be combined into a single statement as follows :

```
Room r2 = new Room();
```

Variable r2 contains a reference or address of memory location where a new object is created.

It is also to be noted that the class determines only the types of the variables. The actual data is contained inside the individual objects and not in the class. Thus, every object has its own set of data. In code listing 8.1, we have created two objects r1 and r2 using new operator. These objects are allocated different memory space to hold their data values as seen in figure 8.2 .

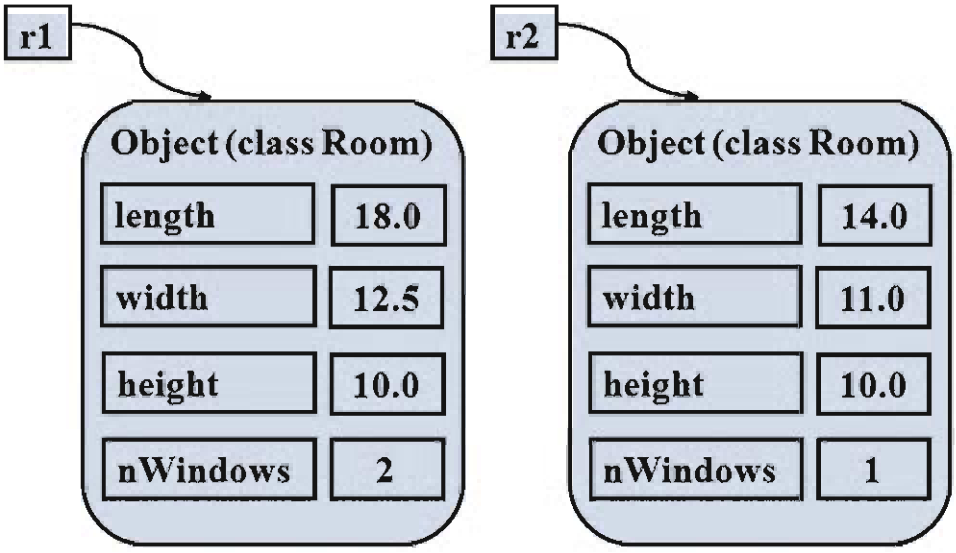


Figure 8.2 : Two instances of class Room

In Java, when objects are no more needed, the memory is claimed back for reuse. Java has a garbage collector that looks for unused objects and reclaims the memory that those objects are using. We do not need to do any explicit freeing of memory.

In object-oriented programming (OOP) languages, creating an object is also called object instantiation. An instance for an object is created by allocating memory to store the data for that object. An object that belongs to a class is said to be an instance of that class.

Thus, an instance of a class is another word for an actual object. Class is an abstract representation of an object; whereas an instance is its concrete representation. In fact, the terms instance and object are often used interchangeably in OOP language.

Each instance of a class can hold different values for its attributes in variables declared in a class. Such variables are referred to as instance variables. Instance variables are created at the time of creating an object and stay throughout the life of the object.

Instance variables define the attributes of an object. The class defines the kind of attribute. Each instance stores its own value for that attribute.

To define an object's behaviour, we create methods. In Java, methods can be defined inside a class only. These methods can be invoked using the objects to access or modify the instance variables. Such methods are known as instance methods.

Instance methods are used to define behaviour of an object. Invoking a method is to ask the object to perform some task.

In code listing 8.1, we have defined a class named Room with instance variables length, width, height and nWindows; and instance methods setAttr(), display() and area().

To summarize,

- Objects are created with the new keyword.
- The new keyword returns a reference to an object that represents an instance of the class.
- All instances of class are allocated memory in data structure called heap.
- Each object instance has its own set of data.

Accessing instance variables and calling instance methods

Instance variables and instance methods are accessed via objects. They can be referred by using dot (.) operator as follows :

<object refernce>.<instance variable or method>

For example, we can refer to length of room r1 using r1.length in a program and invoke method r1.display() that displays attribute values of room r1 as can be seen in code listing 8.1. Note that here dot (.) is an operator and associativity of dot operator is from left to right.

Remember that data should be protected from such direct access from anywhere in the program. Such protection is possible with the use of access modifiers that we will see later.

When the instance variables are referred within the methods of the same class, there is no need to use dot (.) operator. For example, in code listing 8.1, in method display, instance variables are accessed without using dot operator. This is possible because method is invoked using reference variable r1 and thus referred instance variables are considered to be of the corresponding object stored at r1.

As studied, when we only declare an object using class, object is not created. In this case, reference variable does not refer to any object and its initial value is null by default. Use of such null reference

or null pointer is illegal and may raise an exception. So, referring instance variable or invoking method with null reference will give an error. Try following code:

```
Room r1; // null value assigned to reference variable by default

System.out.println (r1.length); // illegal

r1.display();  // illegal
```

Class variables and class methods

As discussed earlier, when an object is created using new keyword, memory is allocated from heap area to store the value of its attributes. Thus every object has its own instance variables occupying different space in memory.

Now suppose we want to have a total number of windows of all Room objects created so far. To store this value, it requires only one variable per class. It is meaningless to keep this variable as an attribute of each object. Actually, it is not an attribute of an object; it is an attribute of a class.

Thus, when there is a need to have some variable that is shared by all object instances of the same class, the variable should be allocated the memory only once per class. It means that variable belongs to a class and not to an object. Such variables can be declared within a class using static keyword before data type and these static variables are called class variables.

Values of instance variables are stored in instance (memory allocated for the object); whereas values of class variables are stored in class itself.

Let us declare class variable totWindows by adding following statement in class with instance variables:

```
static int totWindows;
```

Static variables can be accessed without creating an instance of a class. For example, if we try to display the value of totWindows without creating any object of the class 'Room', it will display 0.

Just like class variable, class method can be defined using static keyword in front of the method definition. Try following method in 'Room' class to display total windows.

```
static void displayTotalWindows()
{
    System.out.println("Total Windows: " +totWindows);
}
```

Class variables and class methods can be accessed outside the class using

<classname>.< class variable/method name>

For example : Room.totWindows, Room.displayTotalWindows()

Note that class members (variables and methods) can be referred without class name by the methods of the same class as shown in the code listing of figure 8.3.

RoomDemoStatic.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 RoomDemoStatic.java

```
/* Class Room */
class Room
{
        float length, width, height;
        byte nWindows;
        static int totWindows;  // class variable

        void setAttr (float l, float w, float h, byte n)
        {
                setWindows(n);
                length = l; width = w; height = h;
        }  // end setAttr () method

        void setWindows ( byte n)
        {
                totWindows = totWindows - nWindows + n;
                nWindows = n;
        }
        double area ( )  // area =  length * width
        {     return (length * width);      } // end area() method

        void display ( )
        {
                System.out.println ("\nLength: " + length + "\nWidth: " + width);
                System.out.println ("Height: " + height);
                System.out.println ("Number of Windows: " + nWindows);
        } // end display() method
}  // end Room class

/* using Room class to create objects and run application */
class RoomDemoStatic
{
        public static void main (String args[])
        {
                Room r1 = new Room();   Room r2 = new Room();

                r1.setAttr (18, 12.5f, 10, (byte)2);   r1.display();
                r2.setAttr (14, 11, 10, (byte)1);   r2.display();
                System.out.println("\nRoom2 Number of windows modified from 1 to 2");
                r2.setWindows((byte)2);
                System.out.println ("\nTotal number of Windows: " + Room.totWindows);
        }  // end main()
}  // end RoomDemo
```

Figure 8.3 : Use of class variable totWindows

The output of the code shown in figure 8.3 is given in figure 8.4.

RoomDemoStatic.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 RoomDemoStatic.java

```
>javac RoomDemoStatic.java
>Exit code: 0
>java -cp . RoomDemoStatic

Length: 18.0
Width: 12.5
Height: 10.0
Number of Windows: 2

Length: 14.0
Width: 11.0
Height: 10.0
Number of Windows: 1

Room2 Number of windows modified from 1 to 2

Total number of Windows: 4
>Exit code: 0
```

Figure 8.4 : Output of code listing given in figure 8.3

Here, we have written an additional instance method setWindows() that updates total windows by subtracting the old number of windows and adding new number of windows in a room. Method setWindows() is defined in the same class. So it can access class variable totWindows without class name. In main() method (defined in RoomDemoStatic class) that is defined outside 'Room' class, class variable totWindows is to be accessed as Room.totWindows using class name.

Class methods are global to the class itself and available to any other classes or objects. Therefore, class methods can be used anywhere regardless of whether an instance of the class exists or not.

Now, when should we use class methods ? The methods that operate on a particular object, or affect that object, should be defined as instance methods. Methods that provide some general utility but do not directly affect an instance of the class are better declared as class methods.

For example, consider a function that determines whether a given number is prime or not. This function should be defined as class method. The code listing and its output is shown in figure 8.5.

primeClassMethod.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 primeClassMethod.java

```
// Static method (class method)
// isPrime (int) returns true if given integer is prime
class Prime
{
    static boolean isPrime (int n)
    {   //n>1 is prime if it is not divisible by any number except 1 and itself
        int i, last;

        If (n <= 1) return false;
        if (n < 4) return true;
        //if (n%2==0) return false;   // divisible by 2, so not prime

        last = (int) Math.sqrt(n);
        i=3;
        do
        {
            if (n%i == 0) return false;  // n is divisible by i
            i = i + 2;      // no need to divide by even numbers
        } while (i<last) ;  // end of do...while loop

        return true;
    } //end of method isPrime
} // end class primeFunc

class primeClassMethod
{
    public static void main (String[] s)
    {
        int i, n;

        System.out.println ("Prime numbers between 3 and 100:");
        for (n=3; n<100; n=n+2)
        {
            if (Prime.isPrime(n)) System.out.println(n);
        }
    } // end of main
} // end classs ClassMethodDemo
```

```
>javac primeClassMethod.java
>Exit code: 0
>java -cp . primeClassMethod
Prime numbers between 3 and 100:
3
5
7
11
13
17
19
23
25
29
31
35
37
41
43
47
49
53
59
61
67
71
73
79
83
89
97
>Exit code: 0
```

Figure 8.5 : Using static method to determine whether a given integer is prime or not

The designers of Java have already provided a large number of built-in classes with such class methods. In chapter 7 we have used static method 'sqrt()' without creating any object of class Math.

It is to be noted here that the main() method that we have defined till now is also a class method. This is the reason we use keyword static while defining main() method.

As seen here, the static and the non-static portions of a class serve different purposes. The static definitions in the source code specify the things that are part of the class itself; whereas the non-static definitions in the source code specify the things that will become part of every instance object belonging to a class.

Points to Remember :

- Class variables and class methods can be accessed using a class name or reference variable. To increase readability, it is advised to access with class name.
- Class variables and class methods can be accessed from instance methods also.
- Instance variables and instance methods can't be accessed from class methods, as class methods do not belong to any object.

Classification of variables declared in a class

- **Local variables :** Variables defined inside methods or blocks are called local variables. Formal parameters of the methods are also local variables. They are created when the method or block is started and destroyed when the method or block has completed. Local variables are not initialized by default values.
- **Instance variables :** Instance variables are variables defined within a class but outside any method. These variables are allocated memory from heap area when an object is instantiated (created). Instance variables are initialized by default values.
- **Class variables :** Class variables are variables defined with in a class, outside any method, with the static keyword. These variables are allocated memory only once per class and is shared by all its objects. Class variables are initialized with default values.

Polymorphism (Method Overloading)

We have seen first two steps of creating objects: Declaration and Instantiation. The third step is Initialization. It requires the use of constructors. Before we learn about constructors, let us see the way to implement polymorphism in Java.

The word polymorphism means "many forms"; different forms of methods with same name. In Java, we can have different methods that have the same name but a different signature. This is called 'method overloading'. The method's signature is a combination of the method name, the type of return value (object or base type), a list of parameters.

For example, to find maximum of two integers, maximum of three integers, maximum of three double precision real numbers and so on one requires to perform similar task but on different set of numbers. In such scenario, Java facilitates to create methods with same name but different parameters. Finding maximum does not need creation of any object, so it can be defined as static class method. For example :

```
static int max(int x, int y) {...}

static int max(int x, int y, int z) {...}

static double max(double x, double y, double z) {...}
```

You may try to define above forms of max methods and use in the application after looking the example of code listing given in figure 8.6. Here it uses polymorphism to print a line as per specified parameters. Call printline() without any parameter prints 40 times '=' character, printline(int n) prints n times '#' character, whereas printline(int n, char ch) prints n times specified character ch.

polyDemo.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 polyDemo.java

```
// polymorphism: method printline
class PrintLine
{
    static void printline()
    {
        for (int i=0; i<40; i++)
            System.out.print('=');
        System.out.println();
    }

    static void printline(int n)
    {
        for (int i=0; i<n; i++)
            System.out.print('#');
        System.out.println();
    }

    static void printline(char ch, int n)
    {
        for (int i=0; i<n; i++)
            System.out.print(ch);
        System.out.println();
    }
} // end class PrintLine

public class polyDemo
{
    public static void main(String[] s)
    {
        PrintLine.printline();
        PrintLine.printline(30);
        PrintLine.printline('+',20);
    } // end main
} // end PolyDemo class
```

```
>javac polyDemo.java
>Exit code: 0
>java -cp . polyDemo
========================================
##############################
++++++++++++++++++++
>Exit code: 0
```

Figure 8.6 : Method Overloading

Constructors

Constructor is a special kind of method that is invoked when a new object is created. Constructor can perform any action; but it is mainly designed to perform initializing actions.

Till now, we have used class without user-defined constructors. Every class is having its default constructor; sometimes referred to as no-argument constructor. Default constructor does not take any argument. It initializes the attributes of a newly created object using default values based on their data types.

Constructor differs from general methods in the following ways :

- Constructor must have the same name as class name.
- Constructor does not have return type.

- Constructor is invoked implicitly only when an object is constructed using new operator.
- Constructor cannot be invoked explicitly elsewhere in the program.

Like other methods, constructor can also be overloaded. It is possible with varying list of parameters. For example, we may write our own constructors to initialize the attributes of 'Room' object with different parameters as follows:

Room(float l, float w, float h, byte n) : To initialize length to l, width to w, height to h and nWindows to n

Room(float l, float w) : To initialize length to l, width to w, height to 10, nWindows to 1

Program seen in figure 8.7 shows the use of constructors.

RoomConstructorDemoA.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 RoomConstructorDemoA.java

```
/* Using Constructors */
class Room
{
    float length, width, height;
    byte nWindows;
    static int totWindows;  // class variable

    Room (float l, float w, float h, byte n)
    {
        length = l; width = w; height = h;
        nWindows = n;   totWindows+=n;
    }
    Room (float l, float w)
    {
        length = l; width = w; height = 10;
        nWindows = 1;     totWindows++;
    }

    double area ( ) // area =  length * width
    {   return (length * width);     } // end area() method
    void display ( )
    {
        System.out.println ("\nLength: " + length + "\nWidth: " + width);
        System.out.println ("Height: " + height);
        System.out.println ( "Windows: " + nWindows);
    } // end display() method
} // end Room class

class RoomConstructorDemoA  /* Application using Room */
{
    public static void main (String args[])
    {
        Room r1 = new Room(16, 12.5f);
        Room r2 = new Room(20, 14, 12, (byte)2);
        r1.display();  r2.display();
        System.out.println ("\nTotal number of Windows: " + Room.totWindows);
    } // end main()
} // end RoomConstructorDemoA
```

```
>javac RoomConstructorDemoA.java
>Exit code: 0
>java -cp . RoomConstructorDemoA

Length: 16.0
Width: 12.5
Height: 10.0
Windows: 1

Length: 20.0
Width: 14.0
Height: 12.0
Windows: 2

Total number of Windows: 3
>Exit code: 0
```

Figure 8.7 : Using constructors

In absence of user-defined constructors in a class, objects are constructed using default no-argument constructor. It initializes the attributes using default values.

In presence of user-defined constructors in a class, default constructor is no more available. An attempt to create an object using constructor without arguments, compiler returns an error. To solve this problem, we need to provide a user-defined no-argument constructor as follows: <classname> ( ) { };

In code listing given in figure 8.7, try to create a Room object in main method as follows and observe an error occurred during compilation: Room r3 = new Room();

To solve this error, add user-defined no-argument constructor 'Room () {};' and execute. See the successful execution as given in figure 8.8.

RoomConstructorDemoB.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 RoomConstructorDemoB.java

```
/* Using Constructors */
class Room
{
    float length, width, height;
    byte nWindows;
    static int totWindows;  // class variable

    Room ( ) { }; // user-defined no-argument constructor
    Room (float l, float w, float h, byte n)
    {
        length = l; width = w; height = h;
        nWindows = n;   totWindows+=n;
    }
    Room (float l, float w)
    {
        length = l; width = w; height = 10;
        nWindows = 1;     totWindows++;
    }

    double area ( )  // area =  length * width
    {    return (length * width);     } // end area() method
    void display ( )
    {
        System.out.println ("\nLength: " + length + "\nWidth: " + width);
        System.out.println ("Height: " + height);
        System.out.println ( "Windows: " + nWindows);
    } // end display() method
} // end Room class

class RoomConstructorDemoB  /* Application using Room */
{
    public static void main (String args[])
    {
        Room r1 = new Room( );
        Room r2 = new Room(20, 14, 12, (byte)2);
        r1.display();   r2.display();
        System.out.println ("\nTotal number of Windows: " + Room.totWindows);
    } // end main()
} // end RoomConstructorDemoB
```

```
>javac RoomConstructorDemoB.java
>Exit code: 0
>java -cp . RoomConstructorDemoB

Length: 0.0
Width: 0.0
Height: 0.0
Windows: 0

Length: 20.0
Width: 14.0
Height: 12.0
Windows: 2

Total number of Windows: 2
>Exit code: 0
```

Figure 8.8 : Using user-defined no-argument constructor

Visibility Modifiers for Access Control

Access control is about controlling visibility. So, access modifiers are also known as visibility modifiers. If a method or variable is visible to another class, then only it can be referred in another class. To protect a method or variable from such references, we use the four levels of visibility to provide necessary protection.

The Four P's of Protection are public, package (default protection), protected, and private. Access modifiers public, protected and private are used before the type of variable or method. When no modifier is used, it is the default one having visibility only within a package that contains the class.

Package is used to organize classes. To do so, package statement should be added as the first non-comment or non-blank line in the source file. When a file does not have package statement, the classes defined in the file are placed in default package. Package statement has following syntax: package <packageName>;

In this book, we will use default package. In all our programs, we have used the default access modifier till now. Table 8.1 shows the type of access modifier and its visibility.

		Type		
Access Modifier	public	(default: package)	protected	private
Visibility	widest	→ → →	→ →	narrowest

Table 8.1 : Type of access modifier and its visibility

We will now see examples of default and private modifier.

public

Any method or variable is visible to the class in which it is defined. If we want to make it visible to all the classes outside this class, declare the method or variable to have public access. This is the widest possible access. It provides visibility to classes defined in other package also. To provide public access, use access modifier public before type of variable or method. For example,

```
public float length

public double area ( )
```

Note that public variables and methods are visible anywhere and thus can be accessed from other source files and packages also.

We have used public keyword with main() method to make it available to everyone.

```
public static void main(String[] args) { ... }
```

Package (without any modifier)

This is the next level of access that has no precise name. It is indicated by the lack of any access modifier keyword in a declaration. This is the default level of protection. The scope is narrower than public variables. The variable or method can be accessed from anywhere in the package that contains the class, but not from outside that package. Note that a source file without package statement is considered as a package by default. So, in our programs till now, it is as good as public.

Refer the program shown in figure 8.9. Here 'Rectangle' class has two attributes: length and width. It also has various methods. We have not used any modifier keyword, so they have package protection by default. Due to this reason, they are directly accessible in another class 'RectangleDemo' defined in the same source file (default package).

RectangleDemo.Java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 RectangleDemo.Java

```
class Rectangle
{
    double length, width;

    void setAttributes(double x, double y)
    {
        length = x;  width = y;
    }

    double area ()
    {
        return length * width;
    }

    void display()
    {
        System.out.println ("Rectangle with length = " + length
                            + " width = " + width );
    }
}// end class Rectangle

class RectangleDemo
{
    public static void main (String[] s)
    {
        Rectangle rect1;
        rect1 = new Rectangle();
        Rectangle rect2 = new Rectangle();

        rect1.setAttributes (10.5, 20);
        rect1.display();
        System.out.println ("Area of rectangle is " + rect1.area());
        rect2.setAttributes(10,15);
        System.out.println ("Area of rectangle with length " +
                            rect2.length + ", width = " + rect2.width
                            + " is " + rect2.area());
    } //end main()
} // end class RectangleDemo
```

```
>javac RectangleDemo.java
>Exit code: 0
>java -cp . RectangleDemo
Rectangle with length = 10.5 width = 20.0
Area of rectangle is 210.0
Area of rectangle with length 10.0, width = 15.0 is 150.0
>Exit code: 0
```

Figure 8.9 : Default visibility modifier, available everywhere in a package

protected

This level of protection is used to allow the access only to subclasses or to share with the methods declared as "friend". Thus the visibility is narrower than previous two levels; but wider than full privacy provided by fourth level "private".

Use of protected protection will become more relevant when we use inheritance concept of object-oriented programming.

private

Highest level of protection can be achieved by using "private" protection level. This provides the narrowest visibility. The private methods and variables are directly accessible only by the methods defined within a class. They cannot be seen by any other class.

This may seem extremely restrictive, but it is, in fact, a commonly used level of protection. It provides data encapsulation; hiding data from the world's sight and limiting its manipulation. Anything that shouldn't be directly shared with anyone including its subclass is private.

The best way to provide data encapsulation is to make as much data as private as possible. It separates design from implementation, minimizes the amount of information one class needs to know about another to get its job done.

Let us modify code listing given in figure 8.9 and declare length and width as private. As private members are directly available only within the same class, it does not give any error when accessed in area() or display() method. When they are accesses in main method() of RectangleDemo class, it will show an error.

The modified code is shown in figure 8.10. Here, we have used constructors in addition to private instance variables. Note the error in the output pane that says 'length has private access in Rectangle'. This means that it is not accessible in class 'RectangleVisibility1'.

```
RectangleVisibility1.java * SciTE
File Edit Search View Tools Options Language Buffers Help
1 RectangleVisibility1.java *

class Rectangle
{
    private double length, width;

    Rectangle (double x, double y)
    {
        length = x;  width = y;
    }

    Rectangle ( ) { };

    double area () {  return length * width; }

    void display()
    {
        System.out.println ("Rectangle with length = " + length
                            + " width = " + width );
    }
}// end class

class RectangleVisibility1
{
    public static void main (String[] s)
    {
        Rectangle rect1;
        rect1 = new Rectangle();
        Rectangle rect2 = new Rectangle(10, 15);

        rect1.display(); rect2.display();
        System.out.println ("Area of rectangle with length " +
                            rect2.length + ", width = " + rect2.width
                            + " is " + rect2.area());
    }  //end main()
}// end class

>javac RectangleVisibility1.java
RectangleVisibility1.java:31: length has private access in Rectangle
                        rect2.length + ", width = " + rect2.width
                                 ^
RectangleVisibility1.java:31: width has private access in Rectangle
                        rect2.length + ", width = " + rect2.width
                                                 ^
2 errors
>Exit code: 1
```

Figure 8.10 : Error while accessing private instance variable from another class

The problem now is how to access private variables from another class? It can be made available indirectly by the methods that are accessible by another class. See code listing given in figure 8.11.

Here we have added two methods getLength() and getWidth(). As no modifier is used, they have package visibility and so directly available in another class, 'VisibilityPrivateB' here. Through these methods, we can get the values of private 'length' and 'width' data fields (instance variables). See the use of getLength() and getWidth() method calls in a last output statement in main() method of class 'VisibilityPrivateB'.

```
VisibilityPrivateB.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 VisibilityPrivateB.java

class Rectangle
{
        private double length, width;

        Rectangle (double x, double y)
        {
                length = x;  width = y;
        }
        Rectangle ( ) { };

        double area () {  return length * width;   }
        void display()
        {
                System.out.println ("Rectangle with length = " + length
                                + " width = " + width );
        }
        double getLength() { return length; }
        double getWidth() { return width; }
}// end class

class VisibilityPrivateB
{
        public static void main (String[] s)
        {
                Rectangle rect1;
                rect1 = new Rectangle();
                Rectangle rect2 = new Rectangle(10, 15);

                rect1.display(); rect2.display();
                System.out.println ("Area of rectangle with length " +
                        rect2.getLength() + ", width = " + rect2.getWidth();
                                + " is " + rect2.area());
        }  //end main()
}// end class
```

```
>javac VisibilityPrivateB.java
>Exit code: 0
>java -cp . VisibilityPrivateB
Rectangle with length = 0.0 width = 0.0
Rectangle with length = 10.0 width = 15.0
Area of rectangle with length 10.0, width = 15.0 is 150.0
>Exit code: 0
```

Figure 8.11 : Accessing private variables through public or package methods

Accessor and Mutator Methods

When we restrict access to data by declaring them as private, our purpose is to protect them from getting directly accessed or modified by methods of other class. If we want to allow such data to be used by others, then we write "accessor" methods. If we want to allow such data to be modified by others, then we write "mutator" methods.

Conventionally, naming of accessor and mutator methods is to capitalize the first letter of variable name and then prepend the variable name with the prefixes get and set respectively. Due to this convention, accessor methods are also known as "getter" and mutator methods as "setter".

Observe that in code listing shown in figure 8.11 we have used "getter" methods getLength() and getWidth() methods. If we change the type of variable 'length', it will be hidden from other users. It affects only the implementation of accessor method 'getLength()'.

If we want to allow other methods to read only the data value, we should use "getter" methods.

If we want to allow other methods to modify the data value, we should use "setter" methods. For example, 'setLength()' method can be defined to set the value of 'length' attribute using passed argument as follows:

```
void setLength(float l) { length = l; }
```

Use of accessor and mutator methods will prevent the variables from getting directly accessed and modified by other users of the class. It may seem a little difficult to get used to this as we need to write get and set method for each and every instance variable as per need. But, this minor inconvenience will reward us with an ease of reusability and maintenance.

Passing object as a parameter in a method

Just like variables of primitive data types and available built-in data types, objects can also be passed as parameters in a method.

For example, we want to determine whether the area of invoking rectangle object is larger than another rectangle or not. For this, we may write a method where another rectangle object is passed as an argument or parameter. Let us add method 'isLarge' in 'Rectangle' class as shown in figure 8.12 and use it in main() method.

```
boolean isLarge (Rectangle rect)

{ if (area() > rect.area() ) return true; else return false; }
```

See the method call in if statement in the main() method as shown in figure 8.12: rect1.isLarge(rect2). Here 'rect1' is a calling or invoking object and 'rect2' is an object passed as parameter. In the isLarge() method, area() refers to area of calling object 'rect1' and rect.area() refers to area of an object passed as argument.

ObjectParameter.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 ObjectParameter.java

```
class Rectangle
{
    private double length, width;

    Rectangle (double x, double y)
        { length = x; width = y; }
    Rectangle ( ) { };

    double area () { return length * width;  }
    void display()
    {
        System.out.println ("Rectangle with length = " + length
                            + " width = " + width );
    }
    double getLength() { return length; }
    double getWidth() { return width; }
    boolean isLarge (Rectangle rect)
    {
        if (area() > rect.area()) return true;
            else return false;
    }
}// end class

class ObjectParameter
{
    public static void main (String[] s)
    {
        Rectangle rect1 = new Rectangle(8, 20);
        Rectangle rect2 = new Rectangle(10, 15);

        rect1.display();
        System.out.println ("Area of rectangle 1 is "+ rect1.area() + "\n");
        rect2.display();
        System.out.println ("Area of rectangle 2 is "+ rect2.area() + "\n");
        if (rect1.isLarge(rect2))
            System.out.println ("Area of rectangle 1 is larger than " +
                                "Area of rectangle 2");
    } //end main()
}// end class ObjectParameter
```

```
>javac ObjectParameter.java
>Exit code: 0
>java -cp . ObjectParameter
Rectangle with length = 8.0 width = 20.0
Area of rectangle 1 is 160.0

Rectangle with length = 10.0 width = 15.0
Area of rectangle 2 is 150.0

Area of rectangle 1 is larger than Area of rectangle 2
>Exit code: 0
```

Figure 8.12 : Passing an object as a parameter

Remember that parameters of primitive types are passed by value. The values of actual parameters are copied to formal parameters and then function is executed. Changes made to formal parameters are not affecting actual parameters.

It is to be noted that object parameters are passed by reference. So, whatever modifications are performed to the object inside the method, the original object is affected as well. Here, an address (and not value) of the actual parameter is copied to formal parameter.

Inheritance

Object-oriented programming languages provide reusability feature using inheritance. Inheritance allows us to build new class with added capabilities by extending existing class.

Inheritance models 'is-a' relationship between two classes. For example, classroom is a room, student is a person. Here, room and person are called parent class; classroom and student are called child classes. Parent class is also referred to as superclass or base class. In the same way; child class is also referred to as subclass, derived class or extended class.

Whenever two classes have 'is-a' relationship, we use inheritance. Common features are kept in superclass. A subclass inherits all instance variables and methods from superclass and it may have its own added variables and methods. Note that constructors are not inherited in subclass. For example, like room, classroom also has variables length, width, height, number of windows. In addition, it has number of benches and capacity of each bench to accommodate students. Similarly, subclass inherits all methods of superclass and it may have additional methods. Here, subclass Classroom has additional methods: show(), display(), getSeats() and its constructor methods.

Figure 8.13 shows a class diagram that has a class named 'Classroom' derived from its parent class named 'Room'. See that the arrow points from subclass towards superclass. In subclass, only additional attributes and methods are to be shown.

Note that a subclass is not a subset of superclass. In fact, subclass usually contains more information and methods than its superclass.

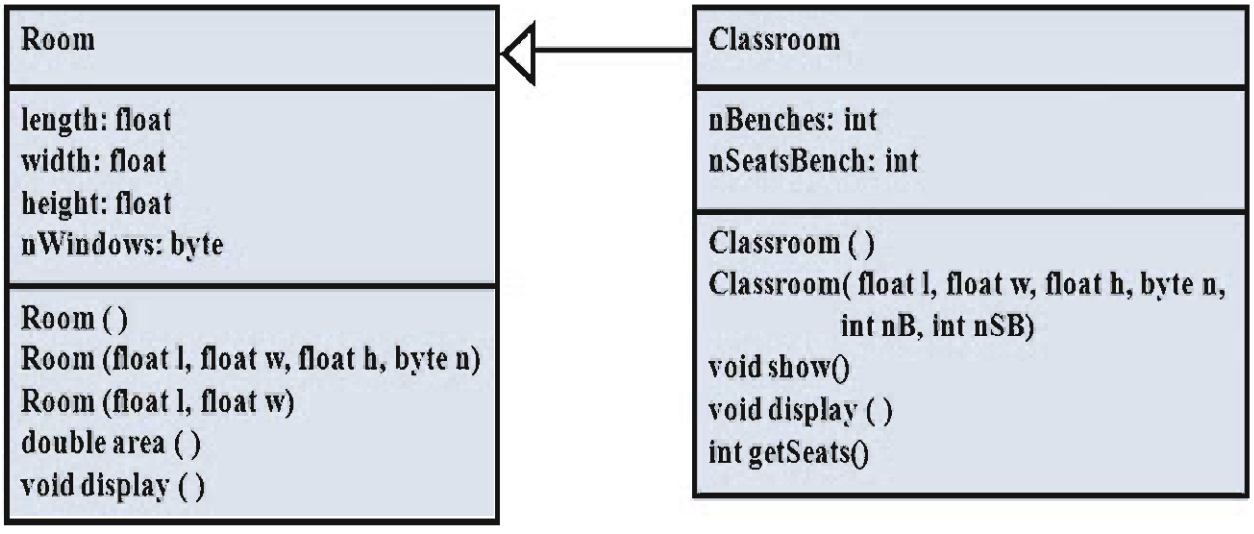


Figure 8.13 : Inheritance Class Diagram

When an object of subclass is instantiated, memory is allocated for all its attributes including inherited ones. Figure 8.14 shows the instances of superclass Room and subclass Classroom.

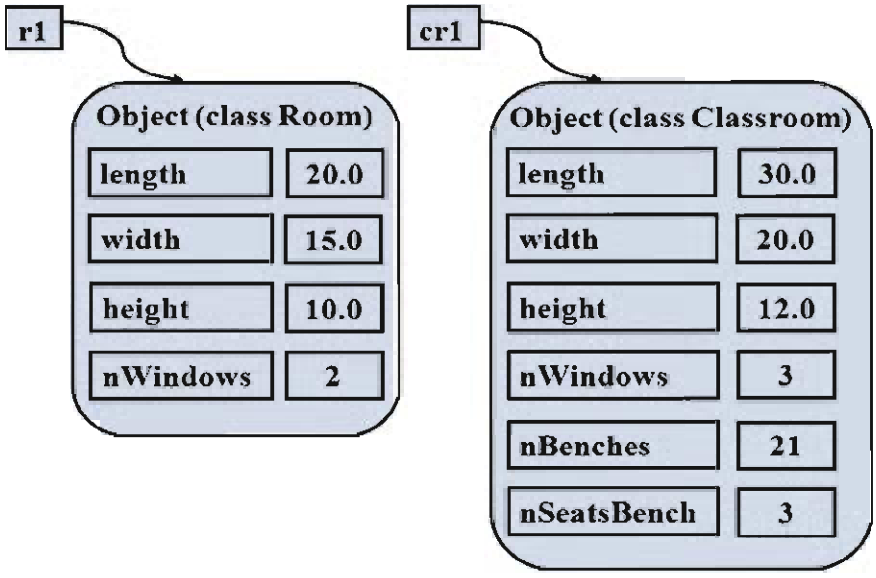


Figure 8.14 : Instances of superclass and subclass

In Java, to create a subclass, keyword 'extends' is used in the class definition. Let us create a subclass 'Classroom' using existing class 'Room' as shown in code listing 8.2.

```
class Room
{
        float length, width, height;
        byte nWindows;
        static int totWindows;  // class variable

        Room ( ) { }; // user-defined no-argument constructor
        Room (float l, float w, float h, byte n)
        {
                length = l; width = w; height = h;
                nWindows = n;   totWindows+=n;
        }
        Room (float l, float w)
        {
                length = l; width = w; height = 10;
                nWindows = 1;       totWindows++;
        }

    double area ( )  // area =  length * width
        {       return (length * width);        } // end area() method

          void display ( )
        {
                System.out.println ("\nLength: " + length + "\nWidth: " + width);
                System.out.println ("Height: " + height);
                System.out.println ( "Windows: " + nWindows);
        } // end display() method
}  // end Room class
```

Code Listing 8.2 : Example of using Inheritance

Now, let us add code to create a subclass 'Classroom' by extending superclass 'Room' as shown in code listing 8.3. Subclass has two additional instance variables nBenches and nSeatsBench. 8Variable nSeatsBench denotes the number of students that can seat on one bench. It has its own additional constructors and three methods: show(), display() and getSeats().

```
class Classroom extends Room
{
        int nBenches, nSeatsBench;
        Classroom( ) {};
        Classroom( float l, float w, float h, byte n, int nB, int nSB)
        {
                super (l,w,h,n);
                nBenches = nB; nSeatsBench = nSB;
        }
        void show()
        {
                super.display();
                System.out.println ("Benches: " + nBenches );
                System.out.println ("Seats per Bench: " + nSeatsBench );
                System.out.println ("Total Seats in a class: " + getSeats() );
        }
        void display()
        {
                System.out.println("\nClassroom with length " + length + " feet, width "
                        + width + " feet\nhas " + nBenches + " Benches, each to accomodate "
                        + nSeatsBench + " Students\nSo, Total seats in a class is  "
                        + getSeats());
        }
        int getSeats() {return nBenches * nSeatsBench; }
}  // end class Classroom
```

Code Listing 8.3 : Code for subclass named Classroom

In sub class 'Classroom', user defined constructor and method show() have reused the code written in superclass 'Room'.

As constructors of a superclass are not inherited in the subclass, keyword 'super' is used to call the constructor of superclass in the constructor of subclass. This call must be the first statement in the constructor. When there is no explicit call to constructor of superclass, no-argument constructor of superclass is implicitly called as the first statement 'super()'.

Method show() is used to display attributes of Classroom object. To display first four attributes inherited from superclass 'Room', we want to use existing code in display() method of superclass.

Now display() method is available in subclass also. In show(), our intention is call display() method of superclass. To do so, we have used super.display().

When superclass and subclass have methods with same signature, a superclass method is said to be overridden in the subclass. Method display() in subclass is used to override display() method of superclass. It means that we want to display the details in a different way without reusing the display() method of superclass. When such method of superclass is to be referred, we need to use keyword 'super' with dot operator and method name. Here we have used super.display() in show() method to invoke display() method of superclass. Method getSeats() is used to compute the seating capacity of a classroom.

Let us use these classes in an application by adding code as given in code listing 8.4. Here, we have created objects of both superclass and subclass.

```
class Inheritance  /* Application using Room, Classroom */
{
        public static void main (String args[])
        {
                Room r1 = new Room(20, 15, 10, (byte)2);
                r1.display();
                Classroom cr1 = new Classroom (30, 20, 12, (byte)3, 21, 3);
                cr1.show();
                System.out.println("Area of classroom1 is " + cr1.area() + " square feet");
                Classroom cr2 = new Classroom (30,30,10, (byte)4, 20, 4);
                cr2.display();
                System.out.println ("\nTotal number of Windows: " + Room.totWindows);
        }  // end main()
}  // end Inheritance
```

Code Listing 8.4 : Application using Room, Classroom

All instance variables and methods are inherited from super class to subclass. Thus method area() of superclass can be invoked using an object of subclass also using cr1.area(). When overridden method is referred in an application using subclass object, it calls the method of subclass as cr2.display(). The output of the code created by combining code 8.2, 8.3 and 8.4 is shown in figure 8.15.

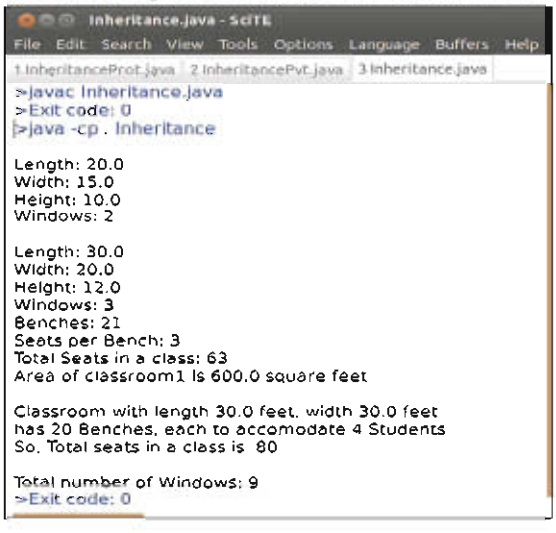


Figure 8.15 : Output of Inheritance using classes Classroom and Room

Private members of superclass

In the previous example, all instances have package visibility. So they are available for direct use in the entire package. If we try to access r1.length or cr1.width directly in main() method, there will not be any error.

If we change the visibility of instance variables of superclass to private, these variables are not directly accessible outside the class. Remember that these attributes belongs to subclass also, but still private instance variables or methods are not visible even in its subclass.

Modify declaration of instance variables in code listing 8.2 to make them private as follows and observe an error in output as shown in figure 8.16.

```
private float length, width, height;

private byte nWindows;
```

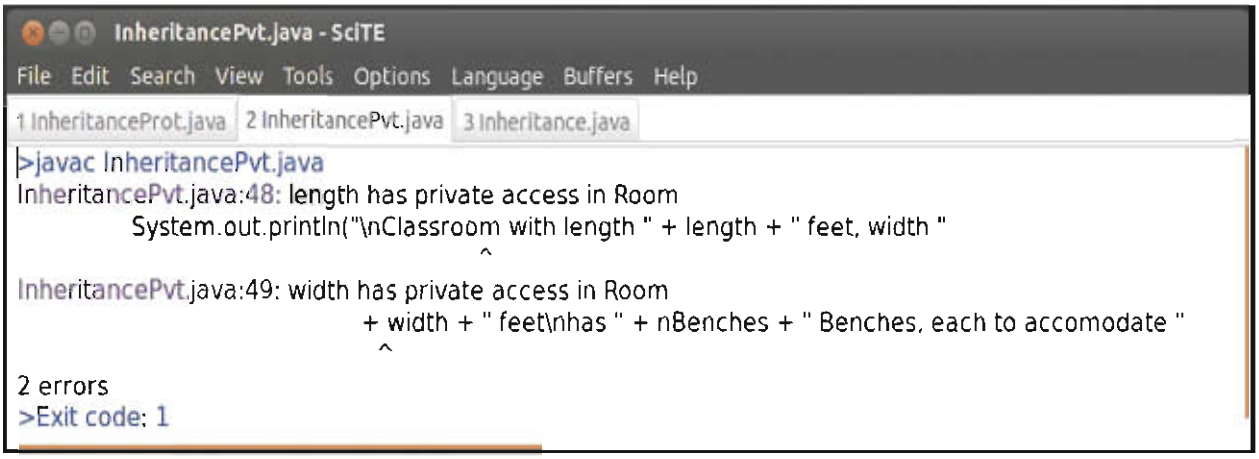


Figure 8.16 : Private members of a class are not accessible outside its class

Remember that private members are directly available only in the class in which they are defined and nowhere else. To make them available elsewhere, use public accessor and mutator methods; 'getter' and 'setter' methods.

Protected members of superclass

As studied before in the topic about 'Visibility Modifiers', 'protected' members are available as 'private' members in the inherited subclass. Modify the instance variables of class 'Room' to protected as follows.

```
protected float length, width, height;

protected byte nWindows;
```

When we execute the modified code we will get the same result as shown in figure 8.15. It is to be noted that Java does not support multiple inheritance. A subclass can be derived from only one superclass.

Composition and Aggregation

Composition and aggregation are the construction of classes that incorporate other objects. They establish a "has-a" relationship between classes.

For example, if we define class 'Library', it has a reading room. Here reading room is an object of the class 'Room'. Thus Library has a Room. When a class includes objects of other class, it is also referred to as container class.

Other examples are :

- Person has a name, address. Here, name is of the class Name with three attributes first name, middle name and last name; address is of class Address with attributes house number, apartment/ society name, area, city, state, country, pin code.
- Car has a steering, wheels and engine. Here steering is of the class Steering, wheel is of the class Wheel and engine is of the class Engine. All the three attributes of class Car are components or parts of a car.

Let us create class 'Library' that contains following attributes :

- nBooks: int, number of books in library
- nMagazines: int, number of magazines subscribed in library
- nNewspapers: int, number of newspapers subscribed in library
- readingRoom: Room

See that readingRoom is not of the basic data types. It is of the type 'Room' class.

Code listing 8.5 shows the code to create class named Room and Library and use them in application named 'Container.java'.

```
/* Using objects as data members in a container class */
class Room
{
        protected float length, width, height;
        protected byte nWindows;
        static int totWindows;  // class variable

        Room ( ) { }; // user-defined no-argument constructor
        Room (float l, float w, float h, byte n)
        {
                length = l; width = w; height = h;
                nWindows = n;   totWindows+=n;
        }
        Room (float l, float w)
        {
                length = l; width = w; height = 10;
                nWindows = 1;       totWindows++;
        }
```

```
        double area ( )  // area =  length * width
        {       return (length * width);        } // end area() method
        void display ( )
        {
                System.out.println ("Length: " + length + "\nWidth: " + width);
                System.out.println ("Height: " + height);
                System.out.println ( "Windows: " + nWindows);
        } // end display() method
}  // end Room class

class Library
{
        int nBooks, nMagazines, nNewspapers;
        Room readingRoom;
        Library( ) {};
        Library( int nB, int nM, int nN, Room r)
        {
                nBooks = nB; nMagazines=nM; nNewspapers=nN;
                readingRoom=r;
        }
        void display()
        {
                System.out.println ("\nLibrary Details:\nNumber of books: " + nBooks );
                System.out.println ("Number of subscribed magazines: " + nMagazines );
                System.out.println ("Number of subscribed newspapers : " + nNewspapers);
                System.out.println ("Reading Room:");
                readingRoom.display();
        }
}  // end class Library

class Container  /* Application using Room, Library */
{
        public static void main (String args[])
        {
                Room r1 = new Room(20, 15, 10, (byte)2);
                r1.display();
                Library lib = new Library (300, 20, 5, r1);
                lib.display();
        }  // end main()
}  // end class Container
```

Code Listing 8.5 : Example of Composition and aggregation

Observe the following points in code listing 8.5,

- In main() method of class 'Container',
  - An object lib of class 'Library' is created using a constructor with four arguments. Last argument is r1 of class 'Room'.
  - Method display() of class 'Library' is invoked using lib object.
- In 'Library' class,
  - An object readingRoom is an attribute of class 'Room'
  - Constructor with four arguments uses last argument 'r' of the class 'Room', assigns the values of Room attributes using an assignment statement 'readingRoom = r;'
  - Defines display() method which invokes display() method of 'Room' class using 'readingRoom.display();'. Note that display() method is not overridden; we have not used inheritance. Readers may use another name for this method, say show() in class 'Library' and use 'lib.show()' in main() method instead of 'lib.display()'.

In this scenario, reader may think: Why not to use inheritance? Why not to inherit 'Library' class from 'Room' class and add three additional attributes?

Note that 'Library' is not of the type of 'Room'. There is no 'is-a' relationship between 'Library' and 'Room'. There is 'has-a' relationship; 'Library' has a 'Room' used as reading room. So, we have used readingRoom of type 'Room' as an attribute of class 'Library'.

Summary

We have seen how to create class and object, how to access instance variables of class and invoke instance methods using objects. Constructors are special methods with the same name as class name, no argument and no return type. Constructors are called implicitly when an object is instantiated using new operator. Instance variables belong to each object. Class variables and methods are defined using 'static' keyword. Static members are allocated memory only once per class and shared by all objects of the class. They belong to class and not to object. We also studied about access modifiers that determine visibility of instance members. Private members are visible only within a class where they are defined, protected members are visible only in inherited subclasses, package members are visible anywhere within a package, public members are visible anywhere. For protection purpose, it is advisable to use private instance variables and supply 'getters' and 'setters' methods as public. In the last, we studied how to inherit new class using existing class; re-use, extend and override the methods. We also see the use of an object as an instance variable in class. This enables creating classes as composition and aggregation of objects.

EXERCISE

1. What do you mean by instantiation of an object ?
2. Give an example for the need of class variable.
3. Write the differences between methods and constructors.

4. Write about accessor and mutator methods.

5. How can a superclass constructor be invoked from the subclass ?

6. How can an overridden method of superclass be invoked from the subclass ?

7. Write a short note on 'Access modifiers'.

8. Explain the use of inheritance and composition or aggregation based on type of relationship between classes.

9. Explain the difference between method overloading and method overriding.

10. Choose the correct option from the following :

(1) Which of the following defines attributes and methods ?

(a) Class (b) Object (c) Instance (d) Variable

(2) Which of the following keyword is used to declare Class variables and class methods ?

(a) static (b) private (c) public (d) package

(3) Which of the following operator creates an object and returns its reference ?

(a) dot (.) (b) new (c) colon (:) (d) assignment (=)

(4) Which of the following method can be called without creating an instance of a class ?

(a) Instance method (b) Class method

(c) Constructor method (d) All of the above

(5) Which of the following refers more than one method having same name but different parameters ?

(a) Overloaded methods (b) Overridden methods

(c) Duplicate methods (d) All of the above

(6) Which method is invoked automatically with creation of an object ?

(a) Instance method (b) Constructor

(c) Class method (d) All of the above

(7) Which of the following is the keyword used to refer a superclass constructor in subclass constructor ?

(a) extends (b) super

(c) name of the superclass (d) new

(8) Which of the following is used to invoke an instance method in Java ?

(a) The name of the object, colon(:) and the name of the method

(b) The name of the object, dot(.) and the name of the method

(c) The name of the class, colon(:) and the name of the method

(d) The name of the class, dot(.) and the name of the method

(9) Which of the following is accessible by instance methods ?

(a) Only instance variables
(b) Only class variables
(c) Both instance variables and class variables
(d) All of the above

(10) When methods in the superclass and subclass have same name and signature, what are they called ?

(a) Overloaded methods
(b) Overridden methods
(c) Inherited methods
(d) All of the above

LABORATORY EXERCISE

Write Java Programs for the following :

1. Create a class named 'FixedDeposit' that contains three attributes (principal amount, annual interest rate and period (years) of deposit) and a method that returns maturity amount using compound interest. Create another class named 'FixedDepositDemo' with main() method. In main() method, create two objects, assign the values to their attributes and display them with maturity amount.

2. Add following constructors in 'FixedDeposit' class and use them to create objects.

   a. No-argument constructor that initializes principal amount with 1000, annual interest rate with 5% and period of deposit as 3 years.

   b. Parameterized constructor with 3 arguments to initialize 3 attributes.

3. Modify the visibility of instance variables in class 'FixedDeposit' as 'private' in lab exercise 1 and try to execute. Will it give an error ? If so, add display() method in the class to display the value of the instance variables.

4. Add accessor and mutator methods to get and set three attributes of the class 'FixedDeposit'. Using these methods, display the value of private instance variables in main() method.

5. Add class variable 'totDeposit' in a class that contains total amount of deposited principal amount. Modify the constructors and setter methods so as to get the total deposit amount. Write a method to display the value of 'totDeposit' variable and show its use.

6. Using 'Rectangle' class, derive a subclass 'Box' having additional attribute 'height' and method 'volume'. (Volume = height x width x length = height x area)

Working with Array and String 9

In chapter 7, we studied basic data types supported in Java. Each variable of such basic data type can store only one value at a time. Such variables are called scalar variable. In this chapter, we will discuss some composite data types that can be used to store a collection of more than one data values. For example, array and string.

Introduction to Array

An array is a variable representing a collection of homogeneous type of elements. Arrays are useful to represent vector, matrix and other multi-dimensional data. Vector is a one dimensional (1-D) data structure that can be used to store list of items like characters, numbers. Matrix is used to represent two dimensional (2-D) data structure like table of rows and columns.

Arrays are useful when same operations are to be performed on various elements of the similar type. All the elements of an array are stored in memory using contiguous storage space. Each element is identified by an index position associated with array variable.

In Java, array is an object used to manage list of items. Creating an array is a two step process:

1. Declare an array object
2. Create an array object

An array object can be created in two ways:

1. Using new operator and specifying the size
2. Directly initializing the content of array

1-D array

Array with single dimension is known as 1-D array. For example, vector which can be seen as a collection of one or more scalar variables. Instead of declaring individual variables like marks1, marks2, marks3, marks4, mark5; we can declare array marks[5] and accesses individual variable elements as marks[0], marks[1], marks[2], marks[3], marks[4].

To declare a 1-D array we use a pair of square brackets [ ] after array name or after data type. The syntax to declare array is as follows:

```
<data type> <array name> [];    or <data type> [] <array name>;
```

For example, to store the marks obtained by a student in five tests in Mathematics subject, we can use an array of five integer elements. Array object named 'marks' with size for 5 elements can be created as follows:

```
int marks[];                    // declare array object
marks = new int [5];            // create array object
```

We can combine above two steps and create array using a single statement also as follows :

int marks[] = new int [5]; or int [] marks = new int [5];

Here name of the array is 'marks'. It refers to memory location where five integer values are stored. Integer of int data type uses 4 bytes storage space. Thus, array 'marks' requires 5 x 4 = 20 bytes in contiguous locations in memory.

To refer an element of an array, we use index or subscript in square bracket [ ] after array variable name as shown in figure 9.1. Index specifies the position of an element in an array; for example, marks[2]. Index value starts from 0.

```
int marks[] = {90, 60, 70, 65, 80};
```

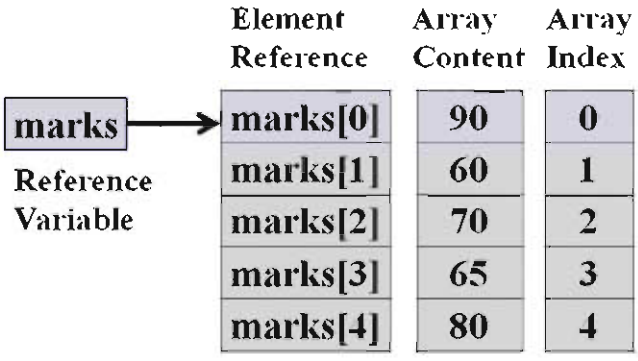


Figure 9.1 : One dimensional array

In figure 9.1 the array variable marks has an index value from 0 to 4. Here, marks[0] refers to the first element and marks[4] refers to the last element.

As explained in chapter 8, an object is a reference variable. Array being an object in Java, array name is also a reference variable. It contains reference to memory location where the array elements are stored. See figure 9.1.

As array is an object, so its elements are initialized with default values. Figure 9.2 shows an example of how to use an array in java programs.

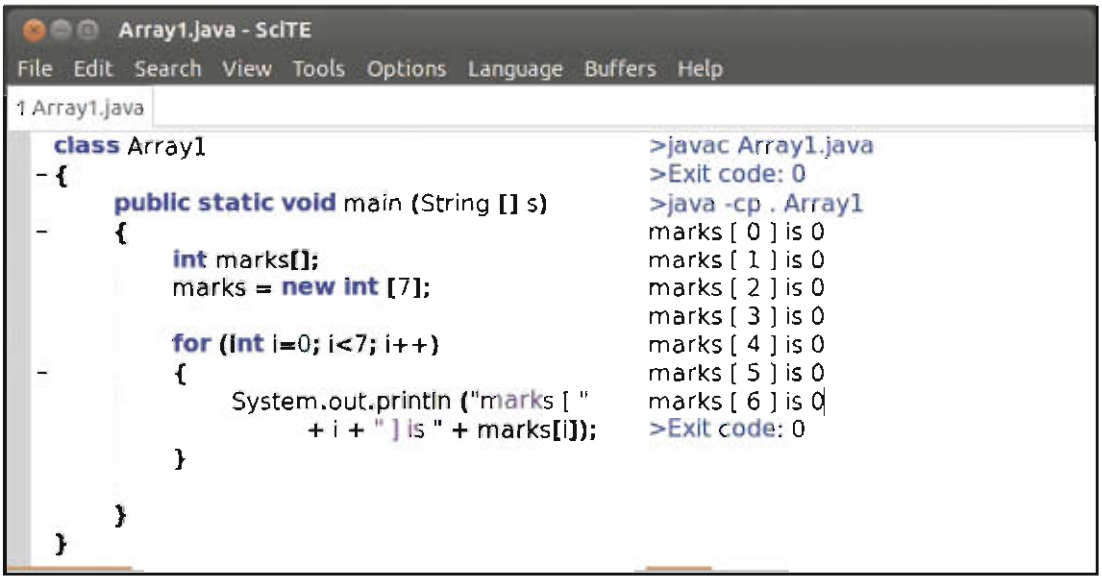


Figure 9.2 : Array object with its elements initialized by default values

One can also specify the initial values of data elements while declaring an array. 1-D array is initialized using comma separated values of data elements in braces { }. For example,

int marks[ ] = {90, 70, 77}; or int [] marks = {90, 70, 77};

Note that when an array is initialized while creating, it does not need the use of new operator. Its size is same as the number of values specified in braces.

The code shown in figure 9.3 uses different ways to create and initialize array. Note the use of class method display() used to display contents of an array. Class methods are declared as static and can be referred without any object in the class.

```
File Edit Search View Tools Options Language Buffers Help
1 Array2.java
/* creating and initializing 1-D array  in different ways */
class Array2
{
    public static void main (String [] s)
    {
        int marks1[];
        marks1 = new int [3];

        int marks2[] = new int[3];
        int [] marks3 = new int [3];
        int marks4[] = {50, 60, 70};
        int [] marks5 = {70, 80, 90};

        System.out.print ("Array marks1:\t");
        display(marks1,3);
        System.out.print ("Array marks2:\t");
        display(marks2,3);
        System.out.print ("Array marks3:\t");
        display(marks3,3);
        System.out.print ("Array marks4:\t");
        display(marks4,3);
        System.out.print ("Array marks5:\t");
        display(marks5,3);
    }

    static void display(int arr[], int size)
    {
        for (int i=0; i<size; i++)
        {
            System.out.print (arr[i] + "\t");
        }
        System.out.println();
    }
}
```

```
>javac Array2.java
>Exit code: 0
>java -cp . Array2
Array marks1:   0    0    0
Array marks2:   0    0    0
Array marks3:   0    0    0
Array marks4:   50   60   70
Array marks5:   70   80   90
>Exit code: 0
```

Figure 9.3 : Different ways to create and initialize array object

In Java, we can not specify both the size of dimensions and initial values of the array elements simultaneously while declaring an array. Figure 9.4 shows the code to illustrate the statement. If we declare an array variable without initialization, only a variable is created to hold the reference as shown in figure 9.1. It does not create an array object, i.e. no memory is allocated for array elements. If we try to access the elements of an array, it will result in an error.

```
File Edit Search View Tools Options Language Buffers Help
1 Array3.java
/* creating and initializing 1-D array
   specifying size with array variable"/

class Array3
{
    public static void main (String [] s)
    {
        int marks4[3] = {50, 60, 70};
        int [5] marks5 = {70, 80, 90};

        System.out.print ("Array marks4:\t");
        display(marks4,3);
        System.out.print ("Array marks5:\t");
        display(marks5,3);
    }

    static void display(int arr[], int size)
    {
        for (int i=0; i<size; i++)
        {
            System.out.print (arr[i] + "\t");
        }
        System.out.println();
    }
}
```

```
>javac Array3.java
Array3.java:8: ']' expected
        int marks4[3] = {50, 60, 70};
                    ^
Array3.java:8: illegal start of expression
        int marks4[3] = {50, 60, 70};
                    ^
Array3.java:8: illegal start of expression
        int marks4[3] = {50, 60, 70};
                    ^
Array3.java:8: not a statement
        int marks4[3] = {50, 60, 70};
                    ^
Array3.java:8: ';' expected
        int marks4[3] = {50, 60, 70};
                    ^
Array3.java:9: ']' expected
        int [5] marks5 = {70, 80, 90};
            ^
Array3.java:9: ';' expected
        int [5] marks5 = {70, 80, 90};
            ^
Array3.java:9: <identifier> expected
        int [5] marks5 = {70, 80, 90};
            ^
Array3.java:11: <identifier> expected
        System.out.print ("Array marks4:\t");
                    ^
Array3.java:11: illegal start of type
        System.out.print ("Array marks4:\t");
```

Figure 9.4 : Illegal to specify size while initializing an array object

Every element of an array is an individual variable that can be referred by using index. Like variables, to change the value of an element, we can use assignment statement referring a variable element on left side. For example, marks[3] = 56; Let us write a program that computes an average of 10 floating point numbers and execute it. Observe the code listing and its output as given in figure 9.5.

ArrayAvg.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 ArrayAvg.java

```
/* compute average of 10 numbers */
class ArrayAvg
{
    public static void main (String [] s)
    {
        double numbers [] = { 10.5, 20.6, 30.8, 15.5,
                              17.3, 25.5, 27.2,
                              20, 30, 18.5};

        byte ctr;
        double sum=0, avg;

        System.out.println ("list of numbers is");
        for (ctr=0; ctr<10; ctr++)
        {
            System.out.println (numbers[ctr]);
            sum = sum + numbers[ctr];
        }
        avg = sum/10;
        System.out.println ("\nAverage of above numbers is "
                            + avg);
    } // main
} // class
```

```
>javac ArrayAvg.java
>Exit code: 0
>java -cp . ArrayAvg
list of numbers is
10.5
20.6
30.8
15.5
17.3
25.5
27.2
20.0
30.0
18.5

Average of above numbers is 21.59
>Exit code: 0
```

Figure 9.5 : Program to compute average of 10 numbers using 1-D array and loop

Here, we need to use the same operation 'add number to sum' repeatedly for different elements of same type. Use of an array helps in declaring 10 variables at a time and using a loop to perform same operation on different variable elements of an array.

We can perform various other operations on array. For example, compare two arrays, copy all the elements of one array to another, search for a specified element in array, sort elements of array and so on. For all such operations, we may write procedures as we did for finding average of elements. Instead of writing such code ourselves, we can use various static methods provided by Java using java.util.Arrays class. In Java, array is treated as an object of this Arrays class. This enables us to make use of methods of Arrays class with array type of data. The code listing and its output in figure 9.6 shows how to use methods sort() and fill() of java.util.Arrays class.

We can use sort() method to sort entire or part of array. When we use only array as an argument, it sorts an entire array. When we invoke this method with 3 arguments: array, start, last; it sorts partial array from element at index start to element at index (last-1). For example, the method call java.util.Arrays.sort (list, 1, 5) sorts elements of array list from list[1] to list[5-1]; See figure 9.6.

Method 'fill' is used to fill the whole or partial array with specified value. When the method is invoked with two arguments: array and value; it assigns the specified value to all array elements. When it is invoked with four arguments: array, start, last, value; it fills partial array from element at start to (last-1) with specified value. For example, fill (list, 7) assigns value 7 to all elements of list array; whereas fill(list, 2, 6, 5) assigns value 5 to elements list[2] to list[6-1]. See figure 9.6.

```
ArraysClassSortFill.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 ArraysClassSortFill.java
class ArraysClassSortFill
{ // sort and fill methods on whole or partial array
    public static void main (String [] s)
    {
        double list[] = {   6.4, 8, 7.8, 9.8, 9.5,
                            6, 7, 8, 8.5, 5.9        };
        int indx;

        System.out.println ("Initial Elements:");
        display(list);
        java.util.Arrays.sort (list, 3, 9);  //sort partial array
        System.out.println ("\nsort partial array: list[3] to list[8]:");
        display(list);
        java.util.Arrays.sort (list);  //sort whole array
        System.out.println ("\nsort whole array:");
        display(list);

        java.util.Arrays.fill (list,7);  //fill whole array
        System.out.println ("\nFill whole array:");
        display(list);
        java.util.Arrays.fill (list, 2, 6, 5);  //fill partial array
        System.out.println ("\nFill partial array: list[2] to list[5]");
        display(list);
    }  // end main

    static void display(double ary[])
    {
        for (int i=0; i<ary.length; i++)
        {
            System.out.print (ary[i] + "\t");
        }
        System.out.println();
    } // end display
} // end class
```

```
>javac ArraysClassSortFill.java
>Exit code: 0
>java -cp . ArraysClassSortFill
Initial Elements:
6.4   8.0   7.8   9.8   9.5   6.0   7.0   8.0   8.5   5.9

sort partial array: list[3] to list[8]:
6.4   8.0   7.8   6.0   7.0   8.0   8.5   9.5   9.8   5.9

sort whole array:
5.9   6.0   6.4   7.0   7.8   8.0   8.0   8.5   9.5   9.8

Fill whole array:
7.0   7.0   7.0   7.0   7.0   7.0   7.0   7.0   7.0   7.0

Fill partial array: list[2] to list[5]
7.0   7.0   5.0   5.0   5.0   5.0   7.0   7.0   7.0   7.0
>Exit code: 0
```

Figure 9.6 : Using sort() and fill() methods of Arrays class

To search an element in an array, Arrays class provides binarySearch() method. We will write our own method to search an element in array using linear search. Linear search method does element by element comparison in a serial fashion. Refer code listing and its output given in figure 9.7. It returns index position when an element is found; otherwise it returns -1.

```
LinearSrch.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 LinearSrch.java
// Search element in array using linear search
class LinearSrch
{
    public static void main (String [] s)
    {
        double list[] = {6, 5, 7, 9, 9.5, 6.5, 7.5, 8 };
        int indx;

        System.out.println ("Given Array elements are:");
        display(list);

        indx=search(list, 8);  // search 8
        if (indx < 0)
            System.out.println("\nelement 8 is not found in array");
        else
            System.out.println("\nelement 8 is found at position " + indx);
        indx=search(list,5.5);  // search 5.5
        if (indx < 0)
            System.out.println("\nelement 5.5 is not found in array");
        else
            System.out.println("\nelement 5.5 is found at position " + indx);
    }  // end main
    static void display(double ary[])
    {
        for (int i=0; i<ary.length; i++)
        {
            System.out.println ( ary[i] );
        }
        System.out.println();
    } // end display
    static int search(double ary[], double x)
    {
        for (int i=0; i<ary.length; i++)
        {
            if (ary[i] == x ) return i;
        }
        return -1;
    } // end search
} // end class
```

```
>javac LinearSrch.java
>Exit code: 0
>java -cp . LinearSrch
Given Array elements are:
6.0
5.0
7.0
9.0
9.5
6.5
7.5
8.0

element 8 is found at position 7

element 5.5 is not found in array
>Exit code: 0
```

Figure 9.7 : Search an element in an array

2-D array

Two dimensional (2-D) arrays are used to store tabular data in the form of rows and columns. For example, to store 5 student's marks in 3 tests, we may use a tabular arrangement of marks in 5 rows and 3 columns as given in figure 9.8. In mathematics, we call it as a matrix of 5 rows and 3 columns where each element contains marks.

Students	Test1	Test2	Test3
1	50	60	70
2	35	30	50
3	70	75	80
4	80	85	90
5	40	50	55

Figure 9.8 : Tabular representation of 2-D array

In Java, 2-D array can be declared using array name and two pairs of square brackets [ ] [ ] to specify the size of two dimensions row and column respectively. For example, following statement declares and creates an array named marks to store 5 x 3 = 15 integer values in contiguous memory locations.

```
int marks [][] = new int [5][3];
```

Here, the logical view of array elements is a table of 5 rows and 3 columns. Physically, they are stored in memory using contiguous memory locations for 15 integers that is 60 bytes.

Java does not support multi-dimensional arrays directly. To create 2-D array, we have to create an array of array. There is no limit on number of dimensions.

In marks [5][3], it creates 1-D array object of 5 elements and each element is 1-D array object of 3 integers as shown in figure 9.9.

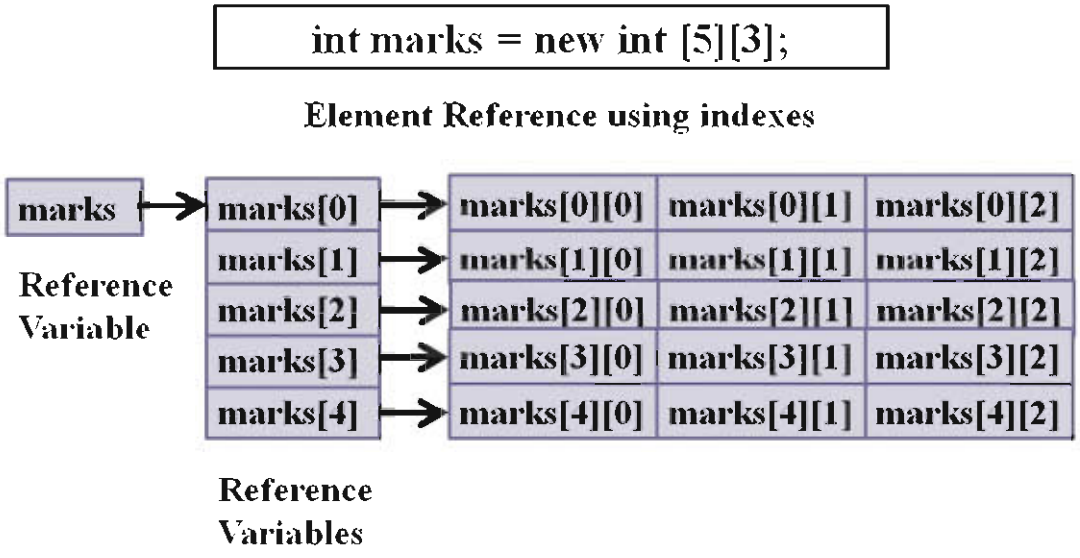


Figure 9.9 : 2-D array as an array of 1-D array

Declaration and initialization of 2-D array is very similar to 1-D array except the number of dimensions. In 2-D array, each row is considered as 1-D array element. Thus, like 1-D array initialization, each row is initialized by enclosing its elements in a pair of braces { } separated by comma. To initialize 2-D array, all these initialized rows are to be enclosed in curly braces {} separating them by comma (,). Like 1-D array, 2-D arrays can be declared in various ways as seen in code listing given in figure 9.10. Figure 9.11 shows output of the code.

```
File Edit Search View Tools Options Language Buffers Help
1 Array2D.java *
/* creating and intializing 2-D array */
class Array2D
{
    public static void main (String [] s)
    {
        int marks1[] [ ];
        marks1 = new int [ 5] [3];
        int marks2[][] = new int [5][3];
        int [][] marks3 = new int [5][3];
        int marks4[] []= { {50, 60, 70}, {35, 30, 50},
                           {70, 75, 80}, {80, 85, 90},
                           {50, 50, 55}  };
        int [] []marks5 = {{50, 60, 70}, {35, 30, 50},
                           {70, 75, 80}, {80, 85, 90},
                           {50, 50, 55}  };

        System.out.print ("2-D Array marks1\n:");
        display(marks1,5,3);
        System.out.print ("2-D Array marks2:\n");
        display(marks2,5,3);
        System.out.print ("2-D Array marks3:\n");
        display(marks3,5,3);
        System.out.print ("2-D Array marks4:\n");
        display(marks4,5,3);
        System.out.print ("2-D Array marks5:\n");
        display(marks5,5,3);
    } // main
    static void display(int arr[][], int rows, int cols)
    {
        for (int i=0; i<rows; i++)
        {
            for (int j=0; j<cols; j++)
            {
                System.out.print (arr[i] [j] + "\t");
            }
            System.out.println();
        }
    } // display
} // class
```

Figure 9.10 : Example of 2-D array

```
Array2D.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 Array2D.java
>javac Array2D.java
>Exit code: 0
>java -cp . Array2D
2-D Array marks1
:0    0    0
0    0    0
0    0    0
0    0    0
0    0    0
2-D Array marks2:
0    0    0
0    0    0
0    0    0
0    0    0
0    0    0
2-D Array marks3:
0    0    0
0    0    0
0    0    0
0    0    0
0    0    0
2-D Array marks4:
50   60   70
35   30   50
70   75   80
80   85   90
50   50   55
2-D Array marks5:
50   60   70
35   30   50
70   75   80
80   85   90
50   50   55
>Exit code: 0
```

Figure 9.11: Output

In Java, 2-D array is considered as an array of 1-D array. So, rows of 2-D array can have variable number of columns. Figure 9.12 shows how to use 2-D array of characters to store five names of computer programming languages.

```
char names [ ][ ] = {   {'J', 'a', 'v', 'a'}, {'C'}, {'C', '+', '+'},
                        {'B', 'a', 's', 'i', 'c'}, {'P', 'a', 's', 'c', 'a', 'l'}   }
```

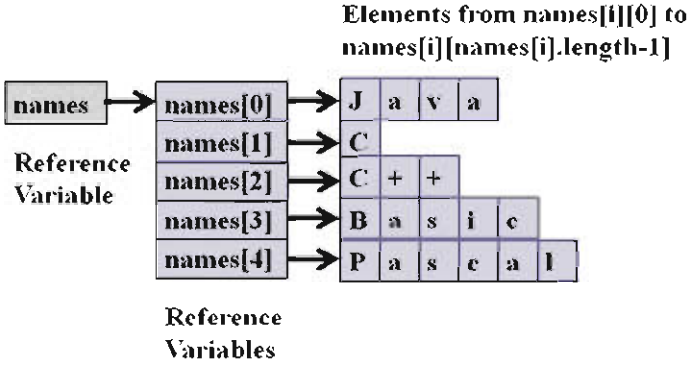


Figure 9.12 : 2-D array with variable number of columns

Each name is stored in different row and all names have different number of characters. Thus, each row is of different size. Size of each row can be known using 'length' property of 1-D array.

Figure 9.13 shows how to use 2-D array with varying column size. Here, we have used 'length' property to get the number of elements in 1-D array. For 2-D array, it returns number of rows. For 1-D array, it returns number of columns in specified row element. Remember that for 2-D array, number of elements is considered as number of rows and each row is 1-D array. In short, length property used with only array name returns the size of its first dimension.

```
Array2Dchar.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 Array2Dchar.java
/* creating and initializing 2-D array */
class Array2Dchar
{
    public static void main (String [] s)
    {
        char names[] []= {
                          {'J','a','v','a'},
                          {'C'},
                          {'C', '+', '+'},
                          {'B', 'a', 's', 'i', 'c'},
                          {'P','a','s','c', 'a', 'l'}
                         };

        System.out.println("Number of elements in 2-D array: "
                           + names.length + "\n");
        System.out.print
                 ("Five names stored in 2-D Array of characters:\n");
        display(names,5);
    } // main
        static void display(char arr[][], int rows)
    {
        for (int i=0; i<rows; i++)
        {
            System.out.print ("Row " + i + " have " + arr[i].length +
                              " character elements: ");
            for (int j=0; j<arr[i].length; j++)
            {
                System.out.print ( arr[i] [j] );
            }
            System.out.println();
        }
    } // display
} // class
```

```
>javac Array2Dchar.java
>Exit code: 0
>java -cp . Array2Dchar
Number of elements in 2-D array: 5

Five names stored in 2-D Array of characters:
Row 0 have 4 character elements: Java
Row 1 have 1 character elements: C
Row 2 have 3 character elements: C++
Row 3 have 5 character elements: Basic
Row 4 have 6 character elements: Pascal
>Exit code: 0
```

Figure 9.13 : Program using 2-D array with variable number of columns

Later we will see that 1-D array of characters can be defined using String class available in Java. In figure 9.14, we have used 2-D array of bytes to store names. Here, it shows that initial chara type data values are converted to byte data type and the characters of names are assigned their corresponding ASCII values.

```
File Edit Search View Tools Options Language Buffers Help
1 Array2Dbyte.java
/* creating and initializing 2-D array */
class Array2Dbyte
{
    public static void main (String [] s)
    {
        byte names[] []= {
                          {'J','a','v','a'},
                          {67},
                          {'C', '+', '+'},
                          {'B', 'a', 's', 'i', 'c'},
                          {'P','a','s','c', 'a', 'l'}
                         };
        System.out.print ("Five names stored in 2-D Array of bytes:\n");
        display(names,5);
    } // main
        static void display( byte arr[][], int rows)
    {
        for (int i=0; i<rows; i++)
        {
            for (int j=0; j<arr[i].length; j++)
            {
                System.out.print ( arr[i] [j] + "\t" );
            }
            System.out.println();
        }
    } // display
} // class
```

```
>java -cp . Array2Dbyte
Five names stored in 2-D Array of bytes:
74   97   118  97
67
67   43   43
66   97   115  105  99
80   97   115  99   97   108
>Exit code: 0
```

Figure 9.14 : 2-D array: Characters stored in bytes using corresponding integer values

Points to be remembered :

- array is a collection of homogeneous type of data
- array elements can be accessed using index for each dimension in [ ]
- index value starts with zero
- multi-dimensional arrays can have variable size of 2nd, 3rd ... dimensions
- attribute 'length' is used to determine the size of the dimension
- array is treated as an object
- declaring an array without initialization does not create an array object

Strings

String is nothing but a sequence of characters. So, 1-D array of characters can be considered as a string. We have already studied and used string literals, where a sequence of characters is enclosed between double quotes.

To use variables that can store strings, Java supports two types of strings that are handled by two classes namely 'String' and 'StringBuffer'. In this chapter, we will see the first one.

Some of the constructors that can be used to create an object are as follows :

- String () without arguments create a String object with no character
- String (char ary[]) creates a String object with its initial value using ary argument
- String (char ary[], int start, int len) creates a String object using 1-D ary argument starting at ary[start] with len number of characters
- String (String strObj) creates a String object which is same as object specified in argument
- String (string literal) creates a String object that refers to the literal specified in argument.

Figure 9.15 shows the use of various types of String class constructors. Do not forget to use new operator while creating object.

String1.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 String1.java

```
/* creating String Objects */
class String1
{
    public static void main (String [] s)
    {
        char name [] = { 'j', 'a', 'v', 'a'};

        String str1 = new String();
        String str2 = new String(name);
        String str3 = new String(name, 1,2);
        String str4 = new String(str3);
        String str5 = new String("Java");

        System.out.println ("str1 is " + str1);
        System.out.println ("str2 is " + str2);
        System.out.println ("str3 is " + str3);
        System.out.println ("str4 is " + str4);
        System.out.println ("str5 is " + str5);
    } // main
} // class
```

```
>javac String1.java
>Exit code: 0
>java -cp . String1
str1 is
str2 is Java
str3 is av
str4 is av
str5 is Java
>Exit code: 0
```

Figure 9.15 : Creating objects of class String using various constructors

In Java, characters are stored using two bytes. To save space, if the characters are ASCII, we should use an array of bytes instead of array of characters. We can use a constructor with array of bytes as an argument. Try the same program by replacing char array with byte array.

Note :

Literals are stored in memory. When two String objects are created using same string literals, memory space is not allocated for second object. Both objects refer to same memory location.

Separate memory is allocated when string objects are created using new operator even if strings are identical. Code listing given in figure 9.16 shows such an example.

```
String2.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 String2.java

/* String Objects */
class String2
{
    public static void main (String [] s)
    {
        String str1 = "I like Java";
        String str2 = "I like Java";
        String str3 = new String("I love India");
        String str4 = new String(str3);

        System.out.println ("str1==str2: "
                            + (str1==str2));
        System.out.println ("str3==str4: "
                            + (str3==str4));

        System.out.println ("str1: " + str1);
        System.out.println ("str2: " + str2);
        System.out.println ("str3: " + str3);
        System.out.println ("str4: " + str4);
    } // main
} // class
```

```
>javac String2.java
>Exit code: 0
>java -cp . String2
str1==str2: true
str3==str4: false
str1: I like Java
str2: I like Java
str3: I love India
str4: I love India
>Exit code: 0
```

Figure 9.16 : Identical String objects created using string literal and using new operator

In code listing given in figure 9.16, note that "str1 == str2" compares the contents of str1 and str2 and not the objects at str1 and str2. String objects str1 and str2 are created using same string literal but without using new operator. Here, both reference variables str1 and str2 refer to the same instance as created for str1. For object str2, separate memory is not allocated as seen in figure 9.17. In the same program, String objects str3 and str4 are created using new operator. The contents of both these objects are same and they refer to different memory locations. Separate memory is allocated for these two objects as seen in figure 9.17.

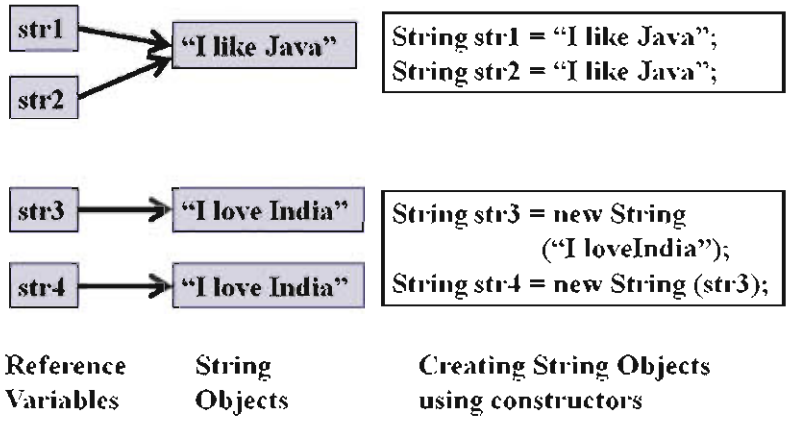


Figure 9.17: Identical strings

String class methods

The String class provides methods to compare strings, find length of string, combining strings, obtaining substrings, converting strings, splitting strings, searching for character or patterns in string etc. We will see the examples of using some of these methods through programs. The program in figure 9.18 shows different ways to compare strings or part of strings.

```
StringComparison.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 StringComparison.java

class StringComparison
{
    public static void main (String [] s)
    {
        String str1 = "India is great";
        String str2 = "INDIA is Great";

        System.out.println ("str1: " + str1);
        System.out.println ("str2: " + str2);
        System.out.println ("str1.equals(str2): "
                    + (str1.equals(str2)));
        System.out.println ("str1.equalsIgnoreCase(str2): "
                    + (str1.equalsIgnoreCase(str2)));
        System.out.println ("str1.compareTo(str2): "
                    + (str1.compareTo(str2)));
        System.out.println ("str1.compareToIgnoreCase(str2): "
                    + (str1.compareToIgnoreCase(str2)));
    } // main
} // class

>javac StringComparison.java
>Exit code: 0
>java -cp . StringComparison
str1: India is great
str2: INDIA is Great
str1.equals(str2): false
str1.equalsIgnoreCase(str2): true
str1.compareTo(str2): 32
str1.compareToIgnoreCase(str2): 0
>Exit code: 0
```

Figure 9.18 : Comparing strings using methods of class String

The syntax and description of various comparison methods is shown in table 9.1.

Method	Description
boolean equals (String str)	Returns true if invoking string is same as str
boolean equalsIgnoreCase (String str)	Returns true if invoking string is same as str after ignoring case (case insensitive)
int compareTo (String str)	Returns 0, >0, <0 integer if invoking string is equal to, greater than or less than str respectively
int compareToIgnoreCase (String str)	Same as CompareTo but case insensitive

Table 9.1 : String class methods for comparing strings

String class provides methods for other tasks as follows. Some of them are given in table 9.2.

- Extracting part of string
- Replacing characters or substrings
- Splitting string into substrings
- Getting number of characters

- Getting character at specified index position
- Converting string into an array of bytes
- Converting string into lowercase or uppercase
- Appending string
- Copy string or part of string

Method	Description
int length()	Returns number of characters in invoking string
char indexAt(int index)	Returns character at index position from the invoking string, index considered from 0
byte [] getBytes()	Returns an array of characters as bytes from invoking string
void getChars (int fromIndx, int toIndx, char target [], int targetIndx)	Copies characters of invoking string from fromIndx to toIndx-1 to target array from targetIndx onwards
String concat(String str)	Returns a string after appending str with the invoking string
String toLowerCase()	Returns a string with all characters of invoking string converted to lowercase
String toUpperCase()	Returns a string with all characters of invoking string converted to uppercase

Table 9.2 : Additional methods of String class

The code listing given in figure 9.19 shows how to use some of these methods.

StringConversion.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 StringConversion.java

```
/* String conversions */
class StringConversion
{
    public static void main (String [] s)
    {
        String str1 = "I like Java";
        String strAry[] = new String[5];
        byte byteAry[] = new byte[20];

        System.out.println ("Given string is \"" + str1 + "\"");
        System.out.println ("String in lowercase: \""
                            + str1.toLowerCase()+ "\"");
        System.out.println ("String in uppercase: \""
                            + str1.toUpperCase()+ "\"");

        System.out.println("\nString converted to array of bytes");
        byteAry = str1.getBytes();
        for (int i=0; i<byteAry.length; i++)
            System.out.println ( byteAry[i] ) ;
    } // main
} // class
```

```
>javac StringConversion.java
>Exit code: 0
>java -cp . StringConversion
Given string is "I like Java"
String in lowercase: "i like java"
String in uppercase: "I LIKE JAVA"

String converted to array of bytes
73
32
108
105
107
101
32
74
97
118
97
>Exit code: 0
```

Figure 9.19 : Using String conversion methods

Note: With array variable, length is an attribute or property of array; whereas with String object, length is a method. So we need to use () while using length method with String object.

String class of Java does not provide any method to reverse a string. So, let us write a program to reverse string. Refer code listing and its output given in figure 9.20.

```
StringReverse.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 StringReverse.java

/* Reverse String */
class StringReverse
{
    public static void main (String [] s)
    {
        String str = new String("I Like Java");
        System.out.println ("Given   String: " + str);
        System.out.println ("Reverse String: " + reverseStr(str));
    } // end main

    static String reverseStr (String str)  // reverse string
    {
        int len = str.length();
        byte byteAry[] = new byte[len];
        byteAry = str.getBytes();        // convert to byte array
        for (int i=0, j=len-1; i<len/2; i++, j--)  // exch half array
            {
                byte tmp = byteAry[i];
                byteAry[i] = byteAry[j] ;
                byteAry[j]=tmp;
            }
        return new String(byteAry);   // used constructor
    }
} // class
```

```
>javac StringReverse.java
>Exit code: 0
>java -cp . StringReverse
Given   String: I Like Java
Reverse String: avaJ ekiL I
>Exit code: 0
```

Figure 9.20 : Reversing a string

Here, we have written a user-defined method reverseStr that reverses the string as follows:

1. Determine the length of string using length() method of String class
2. Convert a string into an array of bytes using method getBytes() of String class
3. Exchange half of the elements on left (starting from first towards right) with half of the elements on right (starting from last towards left) of a byte array
4. Construct a String object from a byte array using constructor and return it

Date class

We have studied about String class and Arrays class provided by Java. Java library also provides Date class in java.util package. Date class encapsulate both date and time and represents the value using milliseconds precision. Figure 9.21 shows the use of Date class with code listing and its output.

```
import java.util.Date;

class Date1
{
    public static void main(String args[])
    {
        Date date1 = new Date();  // current date, time
        // if not used: import java.util.Date; use next statement
        java.util.Date date2 = new java.util.Date();
        System.out.println ("date1: Date & Time: " + date1);
        System.out.println ("date2: Date & Time: " + date2);

        System.out.println (
            "Elapsed time since Jan 1, 1970 is\n\t"
            + date1.getTime() + " milliseconds");
        date1.setTime(date1.getTime() + 10000000);
        System.out.println ("DateTime after 1 crore milliseconds\n\t"
            + date1.toString());
    } // end main()
} // end class
```

```
>javac Date1.java
>Exit code: 0
>java -cp . Date1
date1: Date & Time: Thu Oct 31 08:50:59 IST 2013
date2: Date & Time: Thu Oct 31 08:50:59 IST 2013
Elapsed time since Jan 1, 1970 is
    1383189659935 milliseconds
DateTime after 1 crore milliseconds
    Thu Oct 31 11:37:39 IST 2013
>Exit code: 0
```

Figure 9.21 : Using Date class

Table 9.3 lists some of the methods of Date class.

Method	Description
Date()	Constructs Date object using current system time
Date (long elapsedTime)	Constructs Date object using specified time in milliseconds elapsed since January 1, 1970 GMT
String toString()	Returns a string representing date and time of invoking object
long getTime()	Returns number of milliseconds since January 1, 1970 GMT
void setTime (long elapsedTime)	Sets new date and time of an object using elapsed time

Table 9.3 : Methods of Date class

Calendar class

Like Date class, Calendar class is also provided in java.util package. This class can be used to extract detailed calendar information like year, month, date, hour, minute and second. Here, we will see the use of GregorianCalendar subclass of Calendar class.

Figure 9.22 shows an example program to use Calendar class along with its output. Note the use of user-defined class method display() used to display various components of date and time.

```
Calendar1.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 Calendar1.java

import java.util.*;   // for Date, Calendar, GreGorianCalendar class
public class Calendar1
{
    public static void main(String s[])
    {
        System.out.println ("Current Date & Time: " + new Date()); // current date,time
        //using current date & time
        Calendar calendar1 = new GregorianCalendar();
        display(calendar1);
        //using given year,month,day
        Calendar calendar2 = new GregorianCalendar(2013, 11, 17);
        display(calendar2);
        //using given date,year, month, day and time,hour, minute, second
        Calendar calendar3 = new GregorianCalendar(2013,11,20,210,2);
        display(calendar3);
    } // end main()
    static void display(Calendar calendar)
    {   // using Calendar constants
        System.out.println ("\nYear:\t" + calendar.get(Calendar.YEAR));
        System.out.println ("Month:\t" + calendar.get(Calendar.MONTH));
        System.out.println ("Date:\t" + calendar.get(Calendar.DATE));
        System.out.print ("Hour (12 hour):\t" + calendar.get(Calendar.HOUR));
        if (calendar.get(Calendar.AM_PM) == 1)
            System.out.println (" p.m.");
        else
            System.out.println (" a.m.");
        System.out.println ("Hour (24 hour):\t" + calendar.get(Calendar.HOUR_OF_DAY));
        System.out.println ("Minute:\t" + calendar.get(Calendar.MINUTE));
        System.out.println ("Second:\t" + calendar.get(Calendar.SECOND) );
        System.out.print ("Day of Week: ");
        switch ( calendar.get(Calendar.DAY_OF_WEEK) )
        {
            case 1: System.out.println ("Sunday"); break;
            case 2: System.out.println ("Monday"); break;
            case 3: System.out.println ("Tuesday"); break;
            case 4: System.out.println ("Wednesday"); break;
            case 5: System.out.println ("Thursday"); break;
            case 6: System.out.println ("Friday"); break;
            case 7: System.out.println ("Saturday"); break;
            default: System.out.println("Error...");
        }
    } // end display()
}// end class
```

```
>javac Calendar1.java
>Exit code: 0
>java -cp . Calendar1
Current Date & Time: Thu Oct 31 07:48:28 IST 2013

Year: 2013
Month:    9
Date:     31
Hour (12 hour):     7 a.m.
Hour (24 hour):     7
Minute:   48
Second:   28
Day of Week: Thursday

Year: 2013
Month:    11
Date:     17
Hour (12 hour):     0 a.m.
Hour (24 hour):     0
Minute:   0
Second:   0
Day of Week: Tuesday

Year: 2013
Month:    11
Date:     28
Hour (12 hour):     6 p.m.
Hour (24 hour):     18
Minute:   2
Second:   0
Day of Week: Saturday
>Exit code: 0
```

Figure 9.22 : using Calendar class and its constants

Like get methods, we can use set methods to set the value of the field constants of Calendar class. For example, if calendar is an object of class Calendar, then execution of call 'calendar.set(Calendar.DATE, 20)' will set the date to 20.

Table 9.4 lists the integer constants defined in Calendar class. These constants are given meaningful and self-explanatory names.

Constant	Description
YEAR	Year of calendar
MONTH	Month of calendar (0 for January, 11 for December)
DATE	Day of calendar month
DAY_OF_MONTH	Same as DATE
HOUR	Hour in 12-hour notation
HOUR_OF_DAY	Hour in 24-hour notation
MINUTE	Minute
SECOND	Second
AM_PM	0 for AM, 1 for PM
DAY_OF_WEEK	Day number within a week (1 for Sunday, 7 for Saturday)
WEEK_OF_MONTH	Week number within the month
WEEK_OF_YEAR	Week number within the year
DAY_OF_YEAR	Day number in the year (1 for the first day)

Table 9.4: Constants defined in Calendar class

Summary

This chapter explains how to deal with array, string and date. Array is used to have collection of variables of same data type. Java basically supports only 1-D arrays. Using 1-D array of arrays, we can practically get multi-dimensional arrays. Array is treated as an object of class Arrays, so array name is a reference variable referring to memory location where its elements are stored. All methods of class Arrays can be used with array objects. Examples of such methods are: sort, fill. String class provided in Java enables to work with a sequence of characters. Examples of operations that can be performed on strings are: comparing strings, finding number of characters in a string, converting case of letters of string. Class Date and Calendar are provided to work with date and time. Using Calendar class methods, we can get and set the fields like year, month, date, hour, minute, seconds.

EXERCISE

1. Explain array giving example.
2. Differentiate between 1-D and 2-D arrays.
3. Explain the use of the following classes :

   a. String b. Date c. Calendar

4. Choose the most appropriate from those given below :

   (1) Which of the following refer to the starting index value in arrays ?

   (a) 0 (b) 1 (c) null (d) All of these

   (2) What is the size of second dimension in an array sales[5][12] ?

   (a) 5 (b) 12 (c) 60 (d) 10

   (3) What will expression sales.length return for an array sales[5][12] ?

   (a) 5 (b) 12 (c) 60 (d) 120

   (4) When an array sales [5][12] is declared without specifying initial values, what is the initial value of sales[0][0] ?

   (a) 0 (b) default value (c) compilation error (d) 60

   (5) What does 'length' refer to for an object of String class ?

   (a) attribute (b) method (c) class variable (d) class name

   (6) If 'str' is the object of String class and its content is "Thank GOD", then what is the value of str.length() ?

   (a) 9 (b) 10 (c) 8 (d) 11

(7) What type of value is returned when we use get method of Calendar class with constant DAY_OF_WEEK as an argument ?

(a) int (b) char (c) String (d) boolean

LABORATORY EXERCISE

1. Write a program that uses an array to store 12 values of hourly temperature of a day from 6 am to 6 pm. Use either initialization or assignment statements to assign these values. Print maximum and minimum temperature of the day.

2. Write a program that stores weekly sale of five salespersons. A sales person is given a weekly incentive depending upon his/her average weekly sale. Incentive is: 10 % if average weekly sale is up to Rs. 10000, 15% if average weekly sale is above Rs.10000 but less than or equal to Rs.30000 and 20% if average weekly sale is above Rs. 30000. Compute total daily sale of all salespersons combined together. Also compute weekly incentive for each salesperson.

3. Write a program to see whether a string is palindrome or not. (String is palindrome if its reverse is same as original, example "1771", "madam")

4. Write a program to print current date in the format "DD-MMM-YYYY". For example, 17th November 2013 should be printed as 17-Nov-2013. Also print the day of week in words.

5. Write a program to set the date and time as per your birth date and time. Print the week day on your birthday.

Exception handling in Java 10

Usually we believe that a compiled program is error free and will always execute successfully, but in few cases the program can terminate while it is executing. For example, if we have written a program that connects to a particular website and download the web pages, under normal conditions the program will execute as expected but suppose if the program is executed in a computer where there is no internet connection then the program will produce unexpected output. There can be similar situations when we are dealing with files, for instance the program tries to modify a read-only file or it tries to open a file that does not exist in the system. Such cases are known as "exceptions" in Java. The exception results in an abnormal execution and it may lead to abnormal termination of the program.

An exception is an indication of a problem that occurs during a program's execution, it usually signals an error. Although exceptions occur infrequently, we must be careful to handle such cases while writing the code. Exception handling allows a program to continue executing as if no problem had been encountered or it may notify the user of the problem before terminating in an uncontrolled manner.

In this chapter we will learn techniques to handle exceptions in our programs, few standard exceptions available in Java, a technique to guarantee that a particular block of code will always be executed, even if exceptions are present in our program. Finally we will look at a technique to define and use our own type of exception

Types of Exceptions

In Java, all kinds of error conditions are called exceptions. Errors can be broadly classified into two categories namely Compile-time errors and Run-time errors.

Compile-time errors

We have already learnt in previous chapters how to compile our java programs. A compiler is used to convert source code into object code. If there is a syntax error in the program we will get a compilation error and will not be able to create the ".class" file. Examples of some common syntax errors are missing semicolon, use of undeclared variable, wrong spellings of identifier or keyword and mismatch of bracket. The java program shown in figure 10.1 does not contain a semicolon.

Figure 10.1 : A program that illustrates Compile-time Error

The above code when compiled will generate a compile-time error. This happens because we have missed writing the semicolon ';' in line number 8. The Java compiler when showing the output suggests the type of error, along with the line number where the error has occurred as shown in figure 10.2.

```
ErrorDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
ErrorDemo.java
>javac ErrorDemo.java
ErrorDemo.java:8: ';' expected
        System.out.println("Hello World")  // No Semicolon
                                         ^
1 error
>Exit code: 1
```

Figure 10.2 : Output of the program shown in Figure 10.1

Compile-time errors are usually the mistakes of a programmer and it won't allow the program to compile unless they are solved.

Note : In the field of Computer Science, "Exit code" or "Exit status" indicates whether the command or a program executed successfully or not. Code "0" indicates that the command executed successfully whereas code "1" indicates that some problem occurred while executing the command. In figure 10.2, the last line indicates "Exit Code: 1", it means that the compilation of program - ErrorDemo.java was not successful.

Run-time errors

If there are no syntax errors in the source code then the program will compile successfully and we will get a ".class" file. However this does not guarantee that the program will execute as per our expectations. Let us look at another example shown in figure 10.3 which will compile but will terminate abnormally.

```
RuntimeErrorDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
RuntimeErrorDemo.java
 1 /* A program which generates run-time error. An array of four elements is created but the program tries to
 2    access fifth element of the citylist[] array resulting in run-time error */
 3
 4 class RuntimeErrorDemo
 5 {
 6     public static void main(String args[])
 7     {
 8         /* Create an array of four elements */
 9         String citylist[] = {"Ahmedabad", "Baroda", "Rajkot", "Surat"};
10
11         System.out.println("Statement to be executed before displaying the fifth element");
12
13         /* Statement that generates run-time error  */
14         System.out.println(citylist[5]);
15
16         System.out.println("Statement to be executed after displaying the fifth element");
17     }
18 }
```

Figure 10.3 : A program illustrating run-time error

The program shown in figure 10.3 will compile as there are no syntax errors it. In the program, we have created an array "citylist[]" that contains name of four different cities. In line 14 we are trying to display the contents of element of citylist array that does not exist. This case here is an exception condition. The exception will be generated during runtime; the output of the execution of the program is shown in figure 10.4.

```
RuntimeErrorDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
RuntimeErrorDemo.java
>javac RuntimeErrorDemo.java
>Exit code: 0
>java -cp . RuntimeErrorDemo
Statement to be executed before displaying the fifth element
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 5
        at RuntimeErrorDemo.main(RuntimeErrorDemo.java:14)
>Exit code: 1
```

Figure 10.4 : Output of the program shown in Figure 10.3

From the output shown in figure 10.4, we can notice that the program execution terminated abruptly from line number 14. The output contains a phrase "ArrayIndexOutOfBoundsException" that indicates the type of exception occurred while executing the program.

Let us see few other cases that generate exceptions. For each type of exception, there are corresponding Exception classes in Java.

The java.lang and java.io package contains a hierarchy of classes dealing with various exceptions. Few widely observed exceptions are listed in Table 10.1.

Exception Class	Condition resulting in Exception	Example
ArrayIndexOutOfBoundsException	An attempt to access the array element with an index value that is outside the range of array	int a[] = new int[4]; a[13] = 99;
ArithmeticException	An attempt to divide any number by 0	int a = 50 / 0;
FileNotFoundException	An attempt to access a non-existing file	
NullPointerException	An attempt to use null in a case where an object is required	String s = null; System.out.println(s.length());
NumberFormatException	An attempt to convert string to a number type	String s = "xyz"; int i = Integer.parseInt(s);
PrinterIOException	An I/O error has occurred while printing	

Table 10.1 : List of few widely observed Exceptions

Figure 10.5 shows one more program that generates an exception.

```
RuntimeErrorDemo2.java - SciTE
File Edit Search View Tools Options Language Buffers Help
RuntimeErrorDemo2.java
1   /* A program which generates run-time error.  On dividing any number by zero generates ArithmeticException */
2
3   class RuntimeErrorDemo2
4   {
5       public static void main(String args[])
6       {
7           int numerator = 15;
8           int denominator = 0;
9           int answer;
10
11          System.out.println("Statement to be executed before performing division operation");
12
13          /* Statement that generates run-time error   */
14          answer = numerator / denominator;   // Creates ArithmeticException
15
16          System.out.println("Statement to be executed after performing division operation");
17      }
18  }
```

Figure 10.5 : A program illustrating arithmetic exception

In the program shown in figure 10.5 we try to divide an integer number by zero. A number when divided by zero leads to infinity as a result. Here the program will be compiled successfully but will generate unexpected output due to ArithmeticException. Figure 10.6 shows the output of the execution of program.

```
RuntimeErrorDemo2.java - SciTE
File Edit Search View Tools Options Language Buffers Help
RuntimeErrorDemo2.java
>javac RuntimeErrorDemo2.java
>Exit code: 0
>java -cp . RuntimeErrorDemo2
Statement to be executed before performing division operation
Exception in thread "main" java.lang.ArithmeticException: / by zero
    at RuntimeErrorDemo2.main(RuntimeErrorDemo2.java:14)
>Exit code: 1
```

Figure 10.6 : Output of the code shown in figure 10.5

As we are trying to perform 'division by zero', JVM will terminate the program immediately when it encounters the statement that performs division operation.

Exception Handling

An exception is an error condition. Exception handling is an object-oriented technique for managing errors. While performing exception handling, we try to ensure that the program does not terminate abruptly nor does it generate unexpected output. In this section, we will learn how to handle the exceptions.

Java uses keywords like try, catch and finally to write an exception handler. The keywords try, catch and finally are used in the presence of exceptions, these keywords represent block of statements.

- A try block contains the code that may give rise to one or more exceptions.

- A catch block contains the code that is intended to handle exceptions of a particular type that were created in the associated try block.
- A finally block is always executed before the program ends, regardless of whether any exceptions are generated in the try block or not.

Let us understand these blocks one after another in detail.

The try block

The try statement contains a block of statements within the braces. This is the code that we want to monitor for exceptions. If a problem occurs during its execution, an exception is thrown. Each type of problem (exception) corresponds to an object in Java. A try block may give rise to one or more exceptions. The syntax of the try block is shown below :

```
try
{
   // Set of statements that may generate one or more exceptions
 }
```

Let us see the code that is placed within a try block; it creates a single exception, which we have already seen earlier. The code shown in figure 10.7, when executed will terminate the program as there is no exception handler. The part of program that may lead to runtime error must be written within the try block.

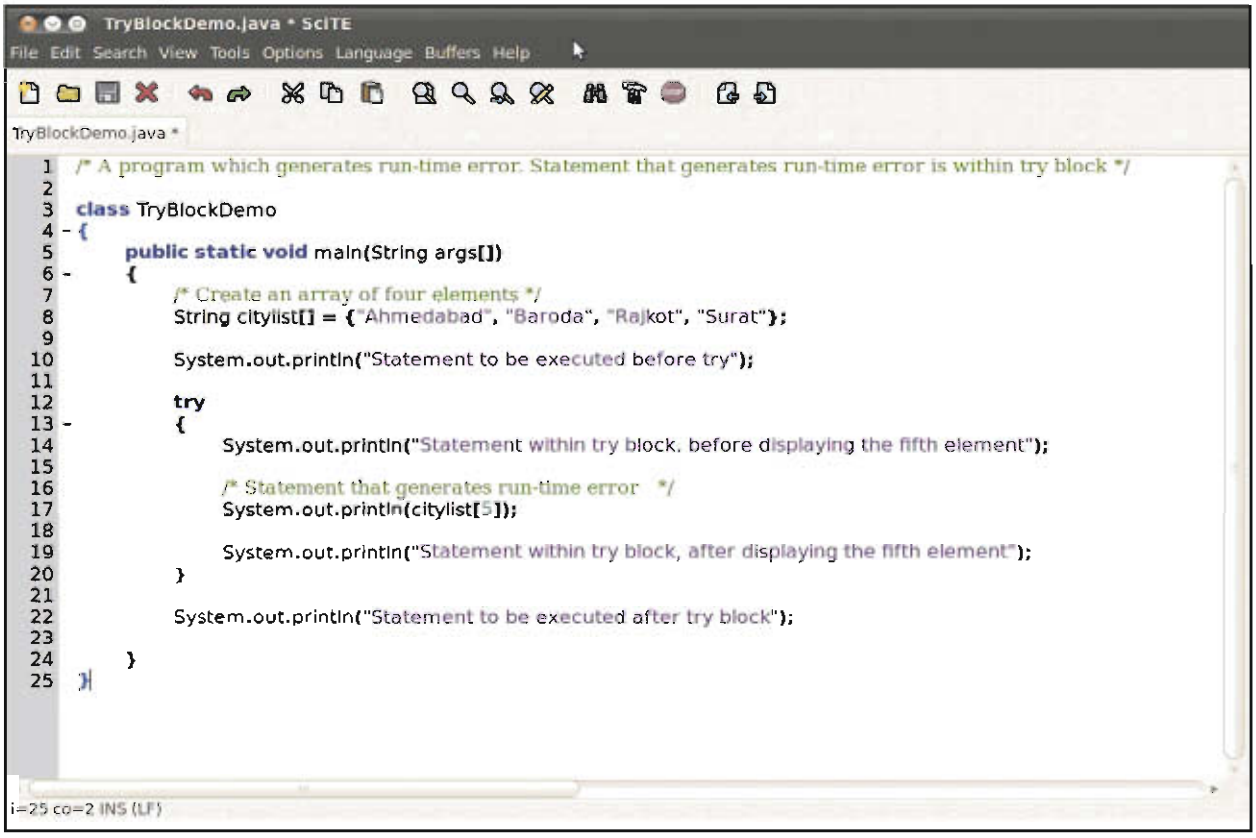


Figure 10.7 : A Program that illustrates try block

On compiling the program shown in figure 10.7, it results in compilation error as there has to be either a catch block or a finally block following the try block. The compilation error is shown in figure 10.8.

Figure 10.8 : Compilation error in program shown in figure 10.7

The catch block

The catch block must immediately follow the try block. It contains the code that is to be executed to handle an exception. The catch block is an exception handler, for a single try block there can be one or more catch blocks. The syntax of the catch block is shown below:

```
try
{
        // Set of statements that may generate one or more exceptions
  }
catch(Exception_Type Exception_object)
  {
        // Code to handle the exception
  }
```

A catch block consists of the keyword catch followed by a single parameter. The code to handle exception has to be written between parentheses. The parameter identifies the type of exception that the block is to deal with. Java supports various types of exceptions, in the later part of this topic we will see how different type of exceptions can be handled by using multiple catch blocks. Figure 10.9 illustrates mechanism of try-catch block.

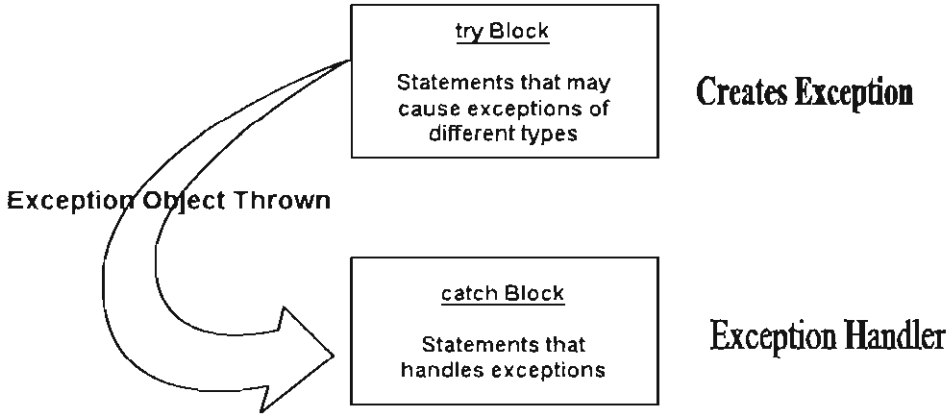


Figure 10.9 : Exception Handling Mechanism

Let us see the example program that uses appropriate exception handler. For the sake of understanding, we will handle ArrayIndexOutOfBounds exception by just catching the object within catch block. In the later sections of the chapter, we will see how the code can be handled to continue the execution of program without termination.

```
TryCatchDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
TryCatchDemo.java
1  /* A program which generates run-time error. Statement that generates run-time error is within
2     try block, there is a corresponding catch block immediately below the try block */
3
4  class TryCatchDemo
5  {
6      public static void main(String args[])
7      {
8          /* Create an array of four elements */
9          String citylist[] = {"Ahmedabad", "Baroda", "Rajkot", "Surat"};
10         System.out.println("Statement to be executed before try");
11         try
12         {
13             System.out.println("Statement within try block, before displaying the fifth element");
14
15             /* Statement that generates run-time error  */
16             System.out.println(citylist[5]);
17
18             System.out.println("Statement within try block, after displaying the fifth element");
19         }
20         catch(ArrayIndexOutOfBoundsException eobj)
21         {
22             System.out.println("Within Catch Block");
23             System.out.println("Caught Exception object of type : " + eobj);
24         }
25         System.out.println("Statement to be executed after try...catch");
26
27     }
28 }
i=28 co=2 INS (LF)
```

Figure 10.10 : A program that illustrates the use of try...catch blocks

The code shown in figure 10.10 will compile successfully and execute. In the program shown in figure 10.10, line 16 contains statement that will generate exception. At line 16, we are trying to access the fifth element of an array citylist[], however the array contains only four members. Our program tries to access array element by specifying index position that is outside the range which leads to an exception. When an exception occurs, an object of type ArrayIndexOutOfBoundsException is created and is thrown; a corresponding catch block handles the exception and does not allow the program to terminate unexpectedly. The catch block contains a reference to object "eobj" which was created and thrown by the try block. Figure 10.11 shows the output of the program.

```
TryCatchDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
TryCatchDemo.java
>javac TryCatchDemo.java
>Exit code: 0
>java -cp . TryCatchDemo
Statement to be executed before try
Statement within try block, before displaying the fifth element
Within Catch Block
Caught Exception object of type : java.lang.ArrayIndexOutOfBoundsException: 5
Statement to be executed after try...catch
>Exit code: 0
```

Figure 10.11 : Output of the program shown in figure 10.10

From figure 10.11, we can observe that the program did not terminate in the presence of exception. The statement displaying "after try…catch" gets executed.

Multiple catch blocks

In a single program multiple exceptions can occur. For instance, if we want to upload particular file to a remote computer, it may lead to two distinct exceptions - an exception may occur if the file is not present in our computer or another exception may occur if the computer is not connected to the network. There is a provision in Java to support multiple exceptions. As discussed before, the code that may generate exception should be written within the try block; apart from this there can be multiple catch blocks to handle each type of exception separately.

If the try block throws several different kinds of exceptions, we can write multiple catch blocks, each handling a specific type of exception. This helps the programmer to write separate logic for each type of exception. Figure 10.12 shows the use of multiple catch blocks.

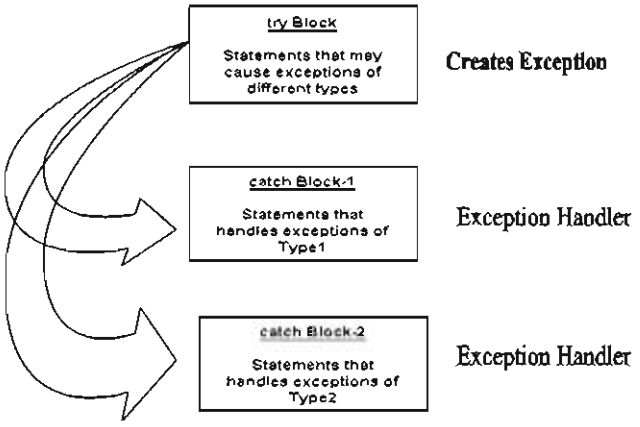
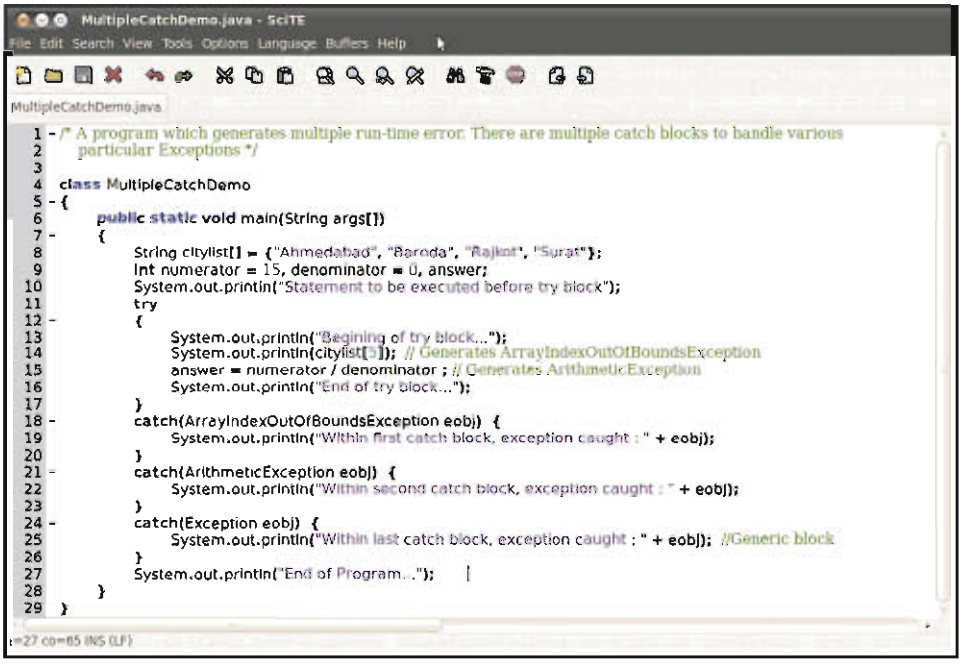


Figure 10.12 : Exception Handling Mechanism with Multiple catch Blocks

Figure 10.13 shows program where multiple catch blocks are used. Here two distinct exceptions are generated within the try block. Exception of type ArrayIndexOutOfBoundsException will be caught by the first catch block and exception of type ArithmeticException will be caught by the second catch block.



```
1  /* A program which generates multiple run-time error. There are multiple catch blocks to handle various
2     particular Exceptions */
3
4  class MultipleCatchDemo
5  {
6      public static void main(String args[])
7      {
8          String citylist[] = {"Ahmedabad", "Baroda", "Rajkot", "Surat"};
9          int numerator = 15, denominator = 0, answer;
10         System.out.println("Statement to be executed before try block");
11         try
12         {
13             System.out.println("Begining of try block...");
14             System.out.println(citylist[5]); // Generates ArrayIndexOutOfBoundsException
15             answer = numerator / denominator ; // Generates ArithmeticException
16             System.out.println("End of try block...");
17         }
18         catch(ArrayIndexOutOfBoundsException eobj) {
19             System.out.println("Within first catch block, exception caught : " + eobj);
20         }
21         catch(ArithmeticException eobj) {
22             System.out.println("Within second catch block, exception caught : " + eobj);
23         }
24         catch(Exception eobj) {
25             System.out.println("Within last catch block, exception caught : " + eobj); //Generic block
26         }
27         System.out.println("End of Program...");
28     }
29 }
```

Figure 10.13 : A program that illustrates use of multiple catch blocks

The last catch block can handle any type of exception. It is a kind of default catch block and must be the last block when there are multiple catch blocks. While writing program, the order of the specific catch blocks does not matter but the default block has to be placed at the end of all catch blocks. For instance, in the program shown in figure 10.13 we can swap the occurrence of catch blocks starting at line number 18 with the one starting at line number 21; however the catch block given at line 24 must be the last block.

Multiple try blocks can be nested together but care must be taken to write a corresponding catch block for each try block.

The finally block

The finally block is generally used to clean up at the end of executing a try block. We use a finally block when we want to be sure that some particular code is to be run, no matter what exceptions are thrown within the associated try block. A finally block is always executed, regardless of whether or not exceptions are thrown during the execution of the associated try block. A finally block is widely used if a file needs to be closed or a critical resource is to be released at the completion of the program. The syntax of finally block is shown below:

```
finally
 {
    // clean-up code to be executed last
 // statements within this block always get executed even though if run-time errors
       terminate the program abruptly
 }
```

Each try block must always be followed by at least one block that is either a catch block or a finally block. Figure 10.14 shows an example of using finally block.

FinallyDemo.java * SciTE

File Edit Search View Tools Options Language Buffers Help

FinallyDemo.java *

```
1  /* A program which generates multiple run-time error. There is a finally block following try block.
2     There are no catch blocks in this program */
3
4  class FinallyDemo
5  {
6      public static void main(String args[])
7      {
8          String citylist[] = {"Ahmedabad", "Baroda", "Rajkot", "Surat"};
9          int numerator = 15, denominator = 0, answer;
10         System.out.println("Statement to be executed before try block");
11         try
12         {
13             System.out.println("Begining of try block...");
14             System.out.println(citylist[5]);  // Generates ArrayIndexOutOfBoundsException
15             answer = numerator / denominator ; // Generates ArithmeticException
16             System.out.println("End of try block...");
17         }
18         finally
19         {
20             System.out.println("This part of code will always get executed");
21         }
22
23         System.out.println("End of Program...");
24     }
25 }
```

Figure 10.14 : Program which illustrates finally block without catch block

In the program shown in figure 10.14, as there is no catch block, the program terminates abruptly due to the exception generated at line 14; statements present in line 15 and line 16 will not be executed. However, in the presence of finally block, the program executes the statements within the finally block before being terminated. The output of program is shown in figure 10.15.

```
FinallyDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
FinallyDemo.java
>javac FinallyDemo.java
>Exit code: 0
>java -cp . FinallyDemo
Statement to be executed before try block
Begining of try block...
This part of code will always get executed
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 5
        at FinallyDemo.main(FinallyDemo.java:14)
>Exit code: 1
```

Figure 10.15 : The output of program shown in figure 10.14

Let us look at an example with multiple catch blocks and a finally block all used together. The program shown in figure 10.16 contains all these blocks. It is a complete program with multiple catch blocks for corresponding exceptions being generated within the try block. Output of program is shown in figure 10.17.

```
AllBlocksDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
AllBlocksDemo.java
 1 /* A program which generates multiple run-time error. There are multiple catch blocks to handle various
 2    particular Exceptions */
 3 class AllBlocksDemo
 4 {
 5     public static void main(String args[]) {
 6         String citylist[] = {"Ahmedabad", "Baroda", "Rajkot", "Surat"};
 7         int numerator = 15, denominator = 0, answer;
 8         System.out.println("Statement to be executed before try block");
 9         try {
10             System.out.println("Begining of try block...");
11             System.out.println(citylist[5]); // Generates ArrayIndexOutOfBoundsException
12             answer = numerator / denominator ; // Generates ArithmeticException
13             System.out.println("End of try block...");
14         }
15         catch(ArrayIndexOutOfBoundsException eobj) {
16             System.out.println("Within first catch block, exception caught : " + eobj);
17         }
18         catch(ArithmeticException eobj) {
19             System.out.println("Within second catch block, exception caught : " + eobj);
20         }
21         catch(Exception eobj) {
22             System.out.println("Within last catch block, exception caught : " + eobj); //Generic block
23         }
24         finally {
25             System.out.println("This part of code will always get executed");
26         }
27         System.out.println("End of Program...");
28     }
29 }
i=19 co=1 INS (LF)
```

Figure 10.16 : Program illustrating try, catch and finally blocks

```
>javac AllBlocksDemo.java
>Exit code: 0
>java -cp . AllBlocksDemo
Statement to be executed before try block
Begining of try block...
Within first catch block, exception caught : java.lang.ArrayIndexOutOfBoundsException: 5
This part of code will always get executed
End of Program...
>Exit code: 0
```

Figure 10.17 : Output of the program shown in figure 10.16

From the output it is clear that the control of program execution switched from line 11 to the first catch block, later it switched to the finally block. The last two catch blocks did not execute. Although the program did not execute completely, it terminated gracefully.

A finally block is associated with a particular try block, and it must be located immediately following any catch blocks for the corresponding try block. If there are no catch blocks, then the finally block can be positioned immediately after the try block. If the finally block or catch blocks are not positioned correctly, then the program will not compile.

The throw statement

The throw keyword is used to explicitly throw an Exception object. In the example programs that we have seen so far, the JVM created an exception object and was throwing it automatically. For example, an object of ArithmeticException was created when we tried to perform a divide by zero operation and it was thrown automatically by the JVM.

Java does provide mechanism to create an Exception object and throw it explicitly. The object that we throw must be of type java.lang.Throwable, (object of Throwable class or any of its sub-classes) otherwise a compile error occurs. The syntax to throw an exception object is as follows:

throw exception_object;

When a throw statement is encountered, a search for matching catch block begins. Any subsequent statements in the try or catch block are not executed. The code in figure 10.18 shows the use of throw statement, its output is given in figure 10.19.

```
1  /* A program which uses throw keyword to throw an exception object explicitly */
2
3  class ThrowDemo
4  {
5      public static void main(String args[])
6      {
7          try
8          {
9              System.out.println("Before throwing an exception object...");
10
11             /* Create an Exception object */
12             Exception myobject = new Exception("Demonstration of throw..");
13
14             throw myobject;  // throw the exception object explicitly
15
16             /* Statements written below throw will generate compile time error */
17
18         }
19         catch(Exception eobj)
20         {
21             System.out.println("Exception caught : " + eobj);
22         }
23     }
24 }
```

Figure 10.18 : Program which illustrates the use of throw keyword

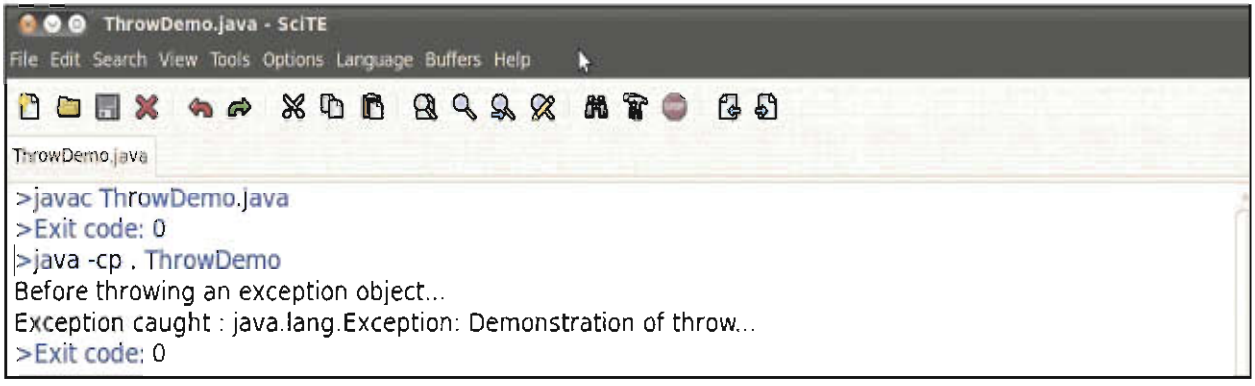


Figure 10.19 : Output of the program in figure 10.18

In the program shown in figure 10.18, we have created an object "obj" of the class Exception, the same object is thrown using throw statement. There has to be catch blocks to handle the exception object thrown explicitly.

The throws Clause

We have explored the try-catch-finally blocks, the programs we discussed so far were simple programs that didn't involve the use of methods. Few questions may arise like what will happen if an exception occurs in a method or a constructor, where will we place the try-catch blocks. There are two alternate approaches to handle exceptions created by a method :

- Write a try-catch block within the method or a constructor that may generate an exception
- Invoking a method (that may generate exception) or constructor within a try block

A throws clause can be used in a method declaration or constructor declaration to inform that the code within the constructor or method may throw an Exception. It also signifies that there is no catch block within the method that can handle the exception. When we write a constructor or a method that can throw exceptions to its caller, it is useful to document that fact. The throws keyword is used with the declaration of method.

A throws clause can be used in a method declaration as follows :

method_Modifiers return_type method_Name(parameters) throws Exception list... {

.....

// body of the method

.....

}

A method can throw multiple exceptions. Each type of exception that a method can throw must be stated in the method header. For example, a method header can be like:

performDivision() throws ArithmeticException, ArrayIndexOutOfBoundsException

{

.....

// body of the method

.....

}

The program in figure 10.20 demonstrates the use of throws clause in the presence of user defined methods. In the following code, it must be noted that if the method throws an Exception object, there must be a matching catch handler. However if we are catching an exception type within the method, there is no need to throw it.

```
ThrowsDemo.java * SciTE
File Edit Search View Tools Options Language Buffers Help
ThrowsDemo.java *
1  /* A program which uses throws keyword to throw an exception from any method */
2
3  class ThrowsDemo
4  {
5      public static void main(String args[])
6      {
7          try
8          {
9              performDivision();   // This method throws exception
10         }
11         catch(ArithmeticException eobj)
12         {
13             System.out.println("Exception caught : " + eobj);
14         }
15     }
16
17     /* Method that throws ArithmeticException object */
18
19     public static void performDivision() throws ArithmeticException
20     {
21         int ans;
22         ans = 15 / 0;
23     }
24 }
```

Figure 10.20 : Program which illustrates the use of throws keyword

In the program of figure 10.20, we have written a method performDivision(), an ArithmeticException object will be generated in this method. The method performDivision() is called in the main() method of our program. It is quite obvious that as there is no exception-handling mechanism within the performDivision() method, the exception will be caught by calling method and there has to be a handler in the calling method. Similarly, there can be exceptions within the Constructors of Java class.

Creating Custom Exceptions

Java allows creating our own exception classes according to application-specific problems. For instance, we are writing a program to generate a mark sheet in which the user is asked to enter marks of different subjects. The marks must be in the range of 0 to 100, suppose if the user enters negative marks or the value is above 100 then the program must generate an Exception. Such kind of exceptions are application specific, Java does not provide built-in exception classes for application specific exceptions.

We can create user-defined exceptions by creating a subclass of Exception class. These exceptions can be thrown explicitly using the throws statement. However it is required to catch this exception and handle it accordingly. Let us see a program code that creates a custom exception to validate the marks.

The program shown in figure 10.21 accepts input from the user. We have used java.util.Scanner class to accept input from the keyboard in line number 21. The "nextInt()" method of the Scanner class helps in reading integer input from the console. We will discuss the functionality of Scanner class in the next chapter that deals with Files and I/O.

Here we have implemented two classes. An additional class is required to create a custom exception. The class "InvalidMarksException" extends Exception class of java.lang package; it contains a single parameter constructor that accepts string to describe the type of error.

```
CustomExceptionDemo2.java - SciTE
File Edit Search View Tools Options Language Buffers Help
CustomExceptionDemo2.java
1  /* A program that creates Custom Exceptions */
2  /* Class InvalidMarksException is a user defined class which is inherited from java.lang.Exception class
3     This program wont proceed untill valid marks are entered*/
4
5  import java.util.Scanner;
6
7  class InvalidMarksException extends java.lang.Exception
8  {
9      public InvalidMarksException(String message)
10     {
11         super(message);
12     }
13 }
14
15
17 class CustomExceptionDemo2
18 {
19     public static void main(String args[])
20     {
21         Scanner kbinput = new Scanner(System.in);
22         int marks;
23         boolean continueLoop = true;
24         do {
25             System.out.println("Enter the marks : ");
26             marks = kbinput.nextInt();
27             System.out.println("You entered " + marks);
28             try
29             {
30                 if(marks < 0 || marks > 100) {
31                     throw new InvalidMarksException("Wrong marks..."); // Throw Customized Exception obj
32                 }
33                 else {
34                     System.out.println("Marks are Valid");
35                 }
36                 continueLoop = false;
37             }
38             catch(InvalidMarksException eobj)
39             {
40                 System.out.println("Exception caught : " + eobj);
41             }
42         }
43         while(continueLoop);
44     }
45 }
```

Figure 10.21 : User defined exception class

In the main method, the business logic is coded (application specific) that ensures whether the marks are in range or not. If the marks are not in range, we create an object of type "InvalidMarksException" and throw it. There has to be a catch block to handle this exception. This program will not proceed unless valid marks are entered. In case if the user enters invalid marks, he/she is asked to keep on re-entering until the input is correct. It must be noted that try-catch blocks are used within a do-while loop. The output of the code is shown in figure 10.22.

```
Terminal
File Edit View Terminal Help

$ javac CustomExceptionDemo2.java
$ java CustomExceptionDemo2
Enter the marks :
145
You entered 145
Exception caught : InvalidMarksException: Wrong marks...
Enter the marks :
-47
You entered -47
Exception caught : InvalidMarksException: Wrong marks...
Enter the marks :
78
You entered 78
Marks are Valid
$
```

Figure 10.22 : Output of the program shown in figure 10.21

Note: SciTE is an editor to type the programs. If the program accepts data from the keyboard, it is advisable to execute the program at command prompt; however, a simple program that does not require user interaction can be executed from within SciTE editor.

Advantages of Exception Handling

Throughout the chapter, we have advocated the use of exception handling in Java programs. By now, it must be clear that a good program must always handle exceptions rather than the program being terminated abruptly. Let us briefly look at the advantages of using exception handling in our Java programs. Few advantages of using exception-handling in Java programs are listed below:

- It allows us to maintain normal flow of program. In the absence of exception handling, the flow of program is disturbed.
- It allows writing separate error handling code from the normal code.
- Error types can be grouped and differentiated within the program.
- Assertions can be used to debug the program before deploying it to the clients.
- It provides an easy mechanism to log various run-time errors while executing the program.

Summary

In chapter we learnt that a program can use exceptions to indicate that an error occurred. A program can catch exceptions by using a combination of the try, catch, and finally blocks.

To throw an exception, we use the throw statement and provide it with an exception object (subclass of java.lang.Throwable class). A method that throws an uncaught, checked exception must include a throws clause in its declaration. The try statement should contain at least one catch block or a finally block and may have multiple catch blocks. Assertions are used to check that something should never happen; they are used by programmers for debugging purpose.

EXERCISE

1. What is the difference between compile-time error and run-time error ?
2. What is an Exception ? Give examples of few Exceptions found in Java.
3. How are the exceptions handled in java ?
4. What is the significance of try, catch and finally block ?
5. What is the use of throw and throws keyword ?
6. How do you create a custom Exception ?
7. Choose the most appropriate option from those given below :

   (1) Which of the following refers to an error condition in object-oriented programming terminology ?

   (a) anomaly (b) abbreviation

   (c) exception (d) deviation

(2) Which of the following is a correct word for all Java Exceptions ?

(a) Errors (b) Runtime Exceptions

(c) Throwables (d) Omissions

(3) Which of the following statements is true ?

(a) Exceptions are more serious than Errors.

(b) Errors are more serious than Exceptions.

(c) Errors and Exceptions are equally serious.

(d) Exceptions and Errors are the same thing.

(4) Which of the following elements is not included in try block ?

(a) the keyword try (b) the keyword catch

(c) the curly braces (d) statements that might cause Exceptions

(5) Which of the following block handles or takes appropriate action when an Exception occurs ?

(a) try (b) catch

(c) throws (d) handles

(6) Which of the following should be within a catch block ?

(a) finally block (b) single statement that handles Exception

(c) any number of statements to handle Exception

(d) throws keyword

(7) What will happen when a try block does not generate an Exception and you have included multiple catch blocks ?

(a) they all execute (b) only the first matching one executes

(c) no catch block executes (d) only the first catch block executes

(8) Which of the following is an advantage of using a try…catch block ?

(a) Exceptional events are eliminated

(b) Exceptional events are reduced

(c) Exceptional events are integrated with regular events

(d) Exceptional events are isolated from regular events

(9) Which of the following methods can throw an Exception ?

(a) methods with throws clause (b) methods with a catch block

(c) methods with a try block (d) methods with finally block

(10) Which of the following is least important to know if you want to be able to use a method to its full potential ?

(a) the method's return type

(b) the type of arguments the method requires

(c) the number of statements within the method

(d) the type of Exceptions the method throws

LABORATORY EXERCISE

1. Write a java program that uses a method - "add()", to add the elements in an array, include a try block within the method, so that the method must handle ArrayIndexOutOfBoundsException.

2. In the above example, remove the try...catch block from the method. By using the throws keyword, the method that invokes "add()" method must handle the exception.

3. Write a java program that throws and catches an ArithmeticException when you attempt to take the square root of a negative value. Prompt the user for an input value and try the Math.sqrt() method on it. The application either displays the square root or catches the thrown Exception and displays an appropriate message. Save the file as SqrtException.java.

4. Write a java program to validate the birth date. Create a custom exception "InvalidBirthDateException". Ask the user to enter any date, if the birth date is after the current date then throw the "InvalidBirthDateException", also write a suitable code to handle such Exceptions.

5. Write a java program to simulate bank transactions. Take two variables, balanceAmount and withdrawAmount. Program must assert if the withdrawAmount is less than balanceAmount.

6. Write a java program to simulate bank transactions. Take two variables, balanceAmount and withdrawAmount. Create a custom exception, "InvalidTransaction", your program must throw the "InvalideTransaction" exception if the withdrawAmount is less than balanceAmount. Write appropriate handlers for this Exception.

7. Write a java program to validate the birth date. Create three different custom exceptions like "InvalidDateException", "InvalidMonthException" and "InvalidYearException". Throw "InvalidDateException" if the date is negative or greater than 30, throw "InvalidMonthException" if the month is negative or greater than 12, throw "InvalidYearException" if the year is below 1950 or after the current year.

File handling 11

In this chapter, we will explore how to access, identify and manipulate files and directories on the hard disk through java programs. Through out the chapter we will focus on input/output processing and file handling. We will use the most common classes available in java.io package and few classes of java.util package. Let us begin the discussion with an overview of files and directories.

Understanding Computer Files

Storage devices of a computer system can be broadly classified into two categories: volatile storage and non-volatile Storage.

Volatile storage is temporary; values stored in variables are lost when a computer is shutdown. A Java program that stores a value in a variable uses Random Access Memory (RAM). Apart from variables, objects and their references are generally stored in RAM, once the program terminates or the computer shuts down, the data is lost.

Non-volatile storage is permanent storage; data is not lost when a computer loses power. When a Java program is saved on a disk, we are using permanent storage. A computer file is a collection of data stored on a non-volatile device. Files exist on permanent storage devices, such as hard disks, USB drives, optical disks and compact discs. Data stored in files is often called persistent data.

Files can be further classified broadly into two categories: text files and binary files.

Text files contain data that can be read in a text editor because the data has been encoded using a scheme such as ASCII or Unicode. Text files can be data files that contain facts, such as a payroll file that contains employee numbers, names, and salaries; or some text files can be program files or application files that store software instructions. Files created through editors like gedit, vi, pico are example of text files. They may have extensions like txt, java or c.

Binary files contain data that has not been encoded as text. Their contents are in binary format, which means the data is accessed in terms of bytes. Some example extensions of binary files are jpeg, mp3 and class.

Java language supports various operations that can be performed on file or on directories. Few operations that can be performed on files using java programs are listed below :

- Determining the path of a file or a directory
- Opening a file
- Writing to a file
- Reading from a file
- Closing a file

- Deleting a file
- Querying the attributes of a file

Java provides built-in classes that contain methods to help us with these tasks. These classes are present in java.io package. Java uses the concepts of streams and it provides two different categories of java classes to perform I/O operations on bytes and characters. A detailed discussion of reading from files and writing to files will be covered in the subsequent sections.

File Class in Java

The java.io.File class encapsulates information about the properties of a file or a directory. File class can be used to access attributes of files and directories. We can create/rename/delete a file or a directory. We can also access the attributes of a file like its permissions, length of a file, or last modification time. The creation of a file object that belongs to File class does not imply that the file or directory exists. A File object encapsulates a pathname or reference to a physical file or a directory on the hard disk.

There are nearly 30 methods of File Class that can be used to perform various operations on a file or a directory. However, File class does not provide any method to read from a file or write into a file, there are several stream classes to perform such operations. So, let us begin our study of few widely used constructors and methods of the File class.

Constructors of File Class

By using the File class, we can create a reference to any file by providing its absolute path in string format or by providing the relative path. The File class provides following constructors to refer a file or a directory.

File(String path)
File(String directory_path, String file_name)
File(File directory, String file_name)

Let us take an example for the above mentioned constructors. In Linux, "passwd" file present in "/etc" directory stores the information of the users existing in the system. Suppose we want to display its attributes, then its java file object can be created using three ways as mentioned below:

- By specifying the path as File fileobj = new File("/etc/passwd");
- By specfying directory and filename as two separate arguments

  File fileobj = new File("/etc", "passwd");

- By using the reference to directory encapsulated in dirobj object

  File dirobj = new File("/etc");

  File fileobj = new File(dirobj, "passwd");

Methods of File Class

The Table 11.1 summarizes few widely used methods of File class.

Method	Description
boolean exists()	Returns true if the file or directory exists, otherwise returns false
boolean isFile()	Returns true if the file exists, otherwise returns false
boolean isDirectory()	Returns true if the directory exists, otherwise returns false
boolean isHidden()	Returns true if the file or directory is hidden
String getAbsolutePath()	Returns the absolute path of the file or directory
String getName()	Returns the name of the file or directory referred by the object.
String getPath()	Returns the path to the file or directory
long length()	Returns the number of bytes in that file
String[] list()	Returns the name of files and directories in a directory
File[] listFiles()	Returns an array of abstract pathnames denoting the files in the directory.

Table 11.1 : Few widely used method of File Class

Let us now try to understand a program that lists the file names in a given directory. After creating an object of file class that refers to a particular directory, we can use the list() method to list all the files present in that directory. The program given in code listing 11.1 demonstrates the listing of files present in "/home/Akash/programs/files" directory. In the program, we have used a variable to count the number of files and directories for a given directory, an array, listOfFiles of the File class is used to store the file objects present in the directory.

```
//List the contents of a Directory
import java.io.File;
public class ListFiles
{
        public static void main(String args[])
        {
                //Provide a Directory Path
                String path = "/home/Akash/programs/files";
                String files;
                int countOfFiles;
                try
                {
```

```
                File folder = new File(path);
                //Store the list of files in an array of File[] objects
                File[] listOfFiles = folder.listFiles();
                //count the number of files in the folder
                countOfFiles = listOfFiles.length;

                System.out.print("List of files in the Directory : ");
                System.out.println(folder.getAbsolutePath());

                //Iterate to display the name of each file
                for(int i=0; i<countOfFiles; i++)
                {
                        if(listOfFiles[i].isFile())
                        {
                                files = listOfFiles[i].getName();
                                System.out.println(files);
                        }
                }
        }
        catch(Exception eobj)
        {
                System.out.println(eobj);
        }
    }
}
```

Code Listing 11.1 : Program to list files in a given directory

The output of code listing 11.1 is shown in figure 11.1

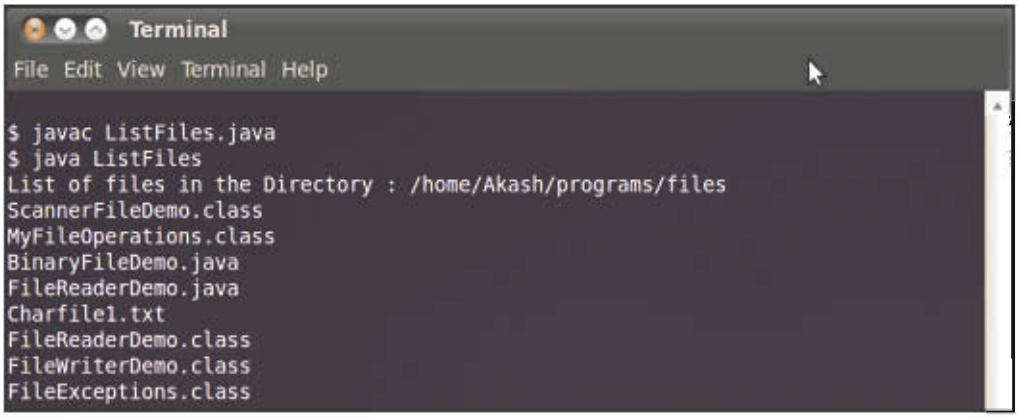


Figure 11.1 : Output of Code Listing 11.1

Note : The above code lists files and directories present in the "/home/Akash/programs/files" directory. You may use appropriate file name and path to see the output.

Introduction to Streams

Till now, we haven't seen how to modify a file or display the contents of a file. To perform such operations we need to understand the concept of streams. Java uses stream classes to carry out read and write operations on files.

We have already studied the types of devices used for input and output. For instance, keyboard is an input device while monitor is an output device. Hard disk can be classified as both input and output device as we can store and read data from the files. There are various devices available in the market from different manufacturers and different capabilities. For example, hard disk are manufactured by various companies and come with different storage capacities like 500GB or 1 TB. Apart from this, hard disk can be connected using different cables like USB or SATA. However, a Java programmer does not need to worry about the technical details like type of hard disk or its capacity while developing a program to perform read/write operations over the files. This is possible because java language provides functionality of streams.

A stream is an abstract representation of an input or output device that is used as a source or destination for data. We can visualize a stream as a sequence of bytes that flows into the program or that flows out of our program. We can write data or read data using streams.

When we write data to stream, the stream is called an output stream. The output stream can transfer data from the program to a file on a hard disk or a monitor or to some other computer over the network. An input stream is used to read data from an external device to the program; it can transfer data from keyboard or from the file on a hard disk to the program.

The main reason for using streams for input or output operations is to make our program independent of the devices involved. Two advantages of streams are:

- Programmer does not need to worry about the technical details of the device
- The program can work for a variety of input/output devices without any changes to the source code.

Stream Classes in Java

To understand byte streams and character streams, let us first try to differentiate between character and byte representation. Let us take an example of number "5"; we can represent the number in two different ways as shown in table 11.2.

Representation	Particulars	Binary Representation
Character 5	ASCII Value : 53	00110101
Binary Number 5	Binary Value : 5	00000101

Table 11.2 : Character and Binary Representation

A character is generally stored using ASCII or Unicode format but when it is used for calculation purpose; its binary value is meaningful. Let us take one more example, the statement int i = 32; declares 'i' as an integer type variable that stores number 32. However 32 can be represented as two seperate characters '3' and '2'. The character representation is advisable when we write sentences like "Human beings have 32 teeth", where we are not performing any kind of operations over the number. But when we apply numerical calculations, we use data types like int, float or double that allows us to store the numbers in binary format.

Java supports two types of streams, byte stream and character stream. Streams that transfer data in the form of bytes to the file or devices are known as byte stream or binary stream. The files that are created using byte stream are known as binary files. Suppose if we wish to store variables like integer, double or boolean into a file, then we must use binary files. Binary files can also be used to store arrays or objects. Similarly text files and program codes are created using character stream. They can be opened in text editors like vi or SciTE.

Java provides two set of classes, character stream class and binary stream class. Character Stream classes present in java.io package deal with the character/text data while byte stream classes present in java.io package deal with binary data. Figure 11.2 shows the classification of stream classes.

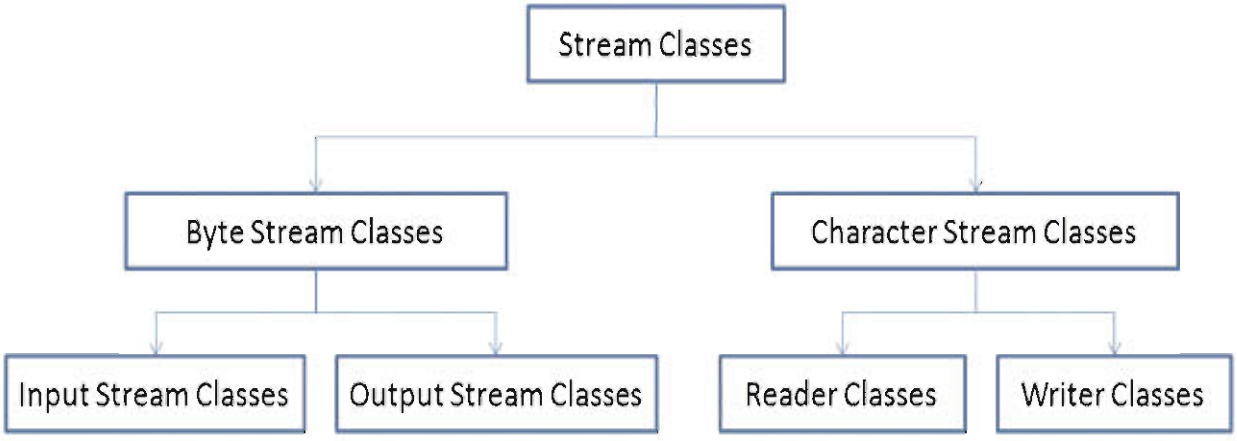


Figure 11.2 : Classification of Stream Classes

Java streams can be classified into two basic types, namely input stream and output stream. An input stream reads data from the source (file, keyboard) while the output stream writes data to the destination (file, output device).

The java.io package contains a collection of stream classes that support reading and writing in a file. To use these classes, a program needs to import the java.io package. Although there are many classes to perform character and byte operations, we will discuss the most widely used classes. Discussion of each class belonging to java.io package is out of the scope for this textbook.

Character Stream Classes

Character stream classes are a group of classes available in java.io package. They can be used to read and write 16-bit Unicode characters. Character stream classes can be further classified into Reader and Writer classes. Reader classes are a group of classes designed to read characters from files. The Writer classes are a group of classes designed to write characters into a file. Figure 11.3 shows the hierarchy of Character Stream Classes.

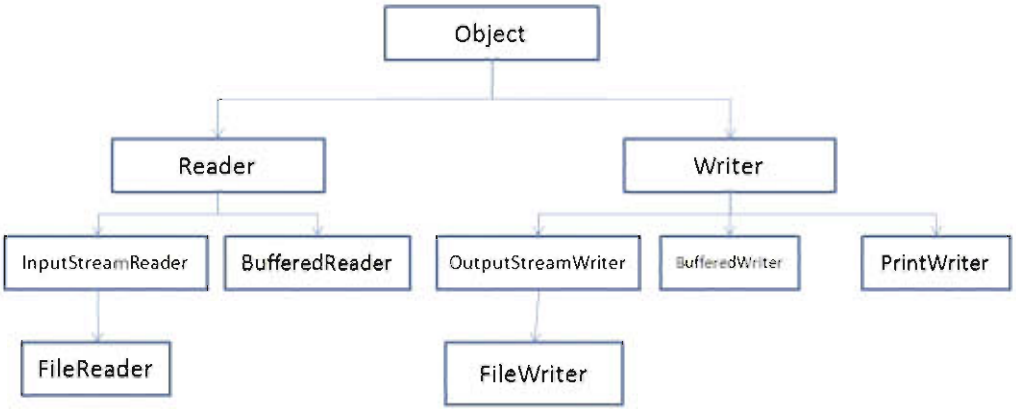


Figure 11.3 : Hierarchy of Character Stream Classes

As seen in the figure 11.3, java.io.Reader class and java.io.Writer class are inherited from the Object class. They are abstract classes (classes that cannot be used to create an object) and come with set of methods to be implemented by its subclasses. InputStreamReader and BufferedReader are the subclass of Reader Class. FileReader class is the subclass of InputStreamReader class. Similarly, OutputStreamWriter, BufferedWriter and PrintWriter are the subclasses of Writer class. FileWriter class is the subclass of OutputStreamWriter class.

Now, let us understand the constructor and methods of some of the above mentioned classes. A detailed description of methods, constructors can be obtained from the online Java documentation at http://docs.oracle.com/javase/6/docs/api/.

Writer Classes

Writer class is the base class for writing a character stream. The abstract Writer class defines the functionality that is available for all character output streams. We will be using the FileWriter class in our program to perform write operations.

The methods of Writer class can throw IOException. An IOException occurs when there is a failed I/O operation. IOException is a checked exception, so we must take care of it otherwise there will be a compiling error. Table 11.3 lists few of the methods of Writer class; these methods are used by its subclasses.

Method	Description
void close()	Closes the stream
void write(int c)	Writes the lower 16 bits of 'c' to the stream
void write(String s)	Writes string 's' to the stream

Table 11.3 : Few methods of FileWriter Class

The OutputStreamWriter class extends Writer class. It converts stream of characters to a stream of bytes. The FileWriter class extends OutputStreamWriter and outputs characters to a file. Some of its constructors are :

FileWriter(String filepath) throws IOException

FileWriter(File fileobj) throws IOException

FileWriter(String filepath, boolean append) throws IOException

In the above constructors, the parameter filepath is the full path name of a file and fileobj is a File class object that describes the file. In the last constructor, if append is true, characters are appended to the end of file, otherwise the existing contents of the file are overwritten. For example we can create an object of FileWriter as shown below:

```
FileWriter  fwobject = new FileWriter("/java/files/Charfile1.txt");
```

Let us take an example that illustrates how to write to a file. We will assume that the files are saved in the working directory. The code listing 11.2 shows how to create a file "Charfile1.txt" that does not exist. Further, we write few lines into that file using the write() method. Methods like write() and close() used in this program are inherited from the Writer class. Output of the code listing is shown in figure 11.4, after execution of the program, we use the cat command to display contents within the newly created file "Charfile1.txt".

```
// Write to a file using character stream
import java.io.*;
class FileWriterDemo
{
        public static void main(String args[])
        {
                FileWriter fwobject = null;
                try   {

                        // Create an object of FileWriter
                        fwobject = new FileWriter("Charfile1.txt");

                        //Write strings to the file
                        fwobject.write("File writing starts...");

                        for(int i = 1; i < 11 ; i++)
                                fwobject.write("Line : " + i + "\n");

                        fwobject.write("File writing ends...");
                }
                catch(Exception eobj)
                {
                        System.out.println(eobj);
```

```
                }
                finally
                {
                        try {
                        // Close the filewriter
                        fwobject.close();
                        }
                        catch(Exception eobj)
                        {
                                System.out.println(eobj);
                        }
                }
        }
}
```

Code Listing 11.2 : Program to illustrate file write operation

It is important to close the stream object after writing to a file is accomplished. Open files consume system resources, and depending on the file mode, other programs may not be able to access them. It's important to close files as soon as the operations are over.

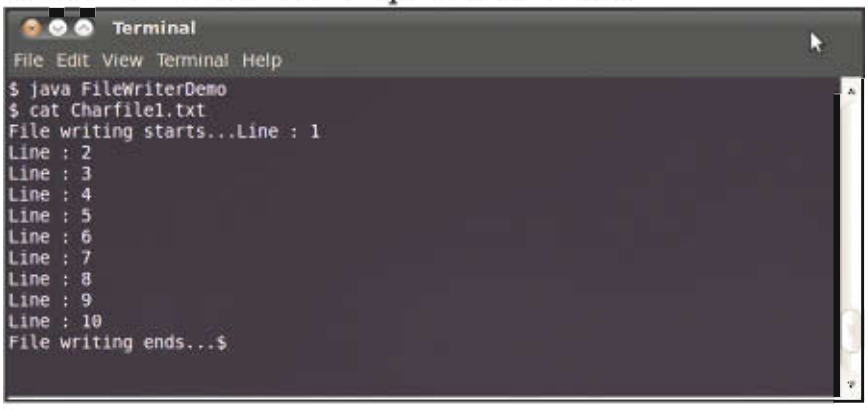


Figure 11.4 : Output of code listing 11.2

Reader Classes

Reader class is the base class for reading a character stream. The abstract Reader class defines the functionality that is available for all character input streams. We will be using the FileReader class in our program to perform read operations. Table 11.4 lists few of the methods of Reader class; these methods are used by its subclasses.

Method	Description
void close()	Closes the stream
int read()	Reads next available character from the stream, it returns "-1" to indicate the end of stream.

Table 11.4 : Few methods of FileReader Class

The InputStreamReader class extends Reader class. It converts a stream of bytes to a stream of characters. The FileReader class extends InputStreamReader class and reads characters from a file. Some of its constructors are:

FileReader(String filepath) throws FileNotFoundException

FileReader(File fileobj) throws FileNotFoundException

We can create an object of FileReader as shown below:

FileReader frobject = new FileReader("/java/files/Charfile1.txt");

In the program shown in code listing 11.2, we attempted to write to a file "Charfile1.txt". Now, let us write a program that reads from the file that we have already created. The program shown in code listing 11.3 reads data from the file using read() method of FileReader class. This read method is inherited from the Reader class.

```
// Reading from a file using character stream
import java.io.*;
class FileReaderDemo
{
        public static void main(String args[])
        {
                FileReader frobject = null;
                try  {

                        // Create an object of FileReader
                        frobject = new FileReader("Charfile1.txt");
                        int i;
                        char ch;
                        while ( ( i = frobject.read()) != -1)
                        {
                                ch = (char) i ;
                                System.out.print(ch);
                        }
                }
                catch(Exception eobj)
                {
                        System.out.println(eobj);
                }
                finally
                {
```

```
                    try {
                    // Close the filewriter
                    frobject.close();
                }
                catch(Exception eobj)
                {
                    System.out.println(eobj);
                }
            }
        }
    }
```

Code Listing 11.3 : Program to illustrate file read operation

Figure 11.5 shows the output of code listing 11.3, here it can be noticed that the execution of program displays the output on the screen. It displays all the contents of the file "Charfile1.txt".

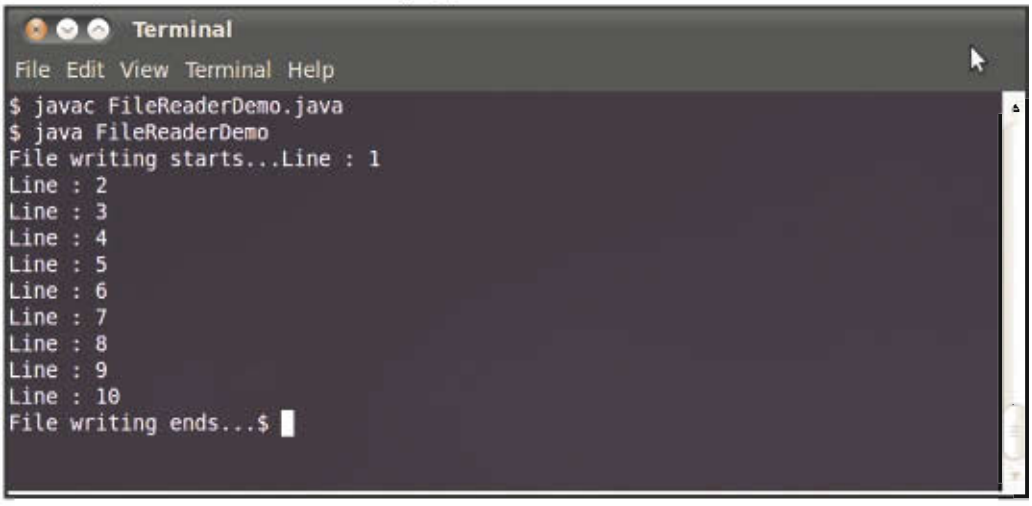


Figure 11.5 : Output of code listing 11.3

There is a special symbol to identify End of File (EOF). While reading from a file, program must identify that the file has ended. However the program reads from input stream so java read() method returns "-1" to identify the end of data in the stream.

Byte Stream Classes

Till now, we have seen how to process characters using java.io package. Most of the real life applications require numeric calculations like storing the details of inventory in a file and calculating the costs involved or like storing the employee details and their salaries in a file. For such kind of processing, java provides set of binary streams and their associated classes.

The FileInputStream and FileOutputStream classes in the java.io package give us the ability to read and write bytes from and into any files in the disk. These classes are the sub-classes of InputStream and OutputSream classes.

FileOutputStream

The FileOutputStream is a subclass of OutputStream and is used to write bytes to the file or some output stream. To use this class and its methods, first we need to create a file object, then we could use the write method derived from the abstract class OutputStream to write byte data into a file. Few widely used methods of FileOutputStream classes are shown in table 11.5:

Method	Description
void close()	Closes this file output stream and releases any system resources associated with the stream.
void write(int b)	Writes the specified byte to this file output stream.
void write(byte[] b)	Writes b.length bytes from the specified byte array to this file output stream.

Table 11.5 : Few methods of FileOutputStream Class

The constructors of FileOutputStream can accept either a string containing the path to the file location or an object of the File class. Let us see few constructors that are normally used :

FileOutputStream(String name) throws FileNotFoundException

Or

FileOutputStream(File file) throws FileNotFoundException

Examples of creating instances of FileOutputStream are as shown below:

```
FileOutputStream fosobject = new FileOutputStream("/home/Akash/myfile.txt");
```

Alternatively we can also use

```
File fobj = new File("/home/Akash/myfile.txt");
FileOutputStream fosobject = new FileOutputStream(fobj);
```

The above listed constructors can throw FileNotFoundException, if the file name refers to directory rather than a regular file, file does not exist, or file cannot be opened for some reason.

FileInputStream

FileInputStream is a subclass of InputStream and is generally used to read byte data from the files. It provides set of methods to perform write operations over the files as shown in table 11.6.

Method	Description
void close()	Closes this file input stream and releases any system resources associated with the stream.
int read()	Reads a byte of data from this input stream.
int read(byte[] b)	Reads up to b.length bytes of data from this input stream into an array of bytes.

Table 11.6 : Few methods of FileInputStream Class

Let us now see a program that uses the above described classes to perform write and read operations over a file as shown in code listing 11.4.

```
//Program to read and write bytes to binary file
import java.io.*;
class BinaryFileDemo {
        public static void main(String args[])
        {
                FileOutputStream outobject = null;
                FileInputStream inobject = null;
                String cities = " Rajkot \n Ahmedabad \n Vadodara \n Vapi \n";

                //Convert cities into byte array
                byte citiesarray[] = cities.getBytes();

                try {
                        // Create object of Binary output stream
                        outobject = new FileOutputStream("Binaryfile.dat");
                        //Write the array of bytes into file
                        outobject.write(citiesarray);
                        outobject.close();

                        //Create object of Binary input stream
                        inobject = new FileInputStream("Binaryfile.dat");
                        //Variable to read each byte
                        int i;
                        //Read each byte from the file and display
                        while((i = inobject.read())!=-1)
                        {
                                System.out.print((char)i);
                        }
                        inobject.close();

                }
                catch(Exception eobj)
                {
                        System.out.println(eobj);
                }
        }
}
```

Code Listing 11.4 : Program to read and write bytes to binary file

The program shown in code listing 11.4 writes a string to a file "Binaryfile.dat", later it creates an object of FileInputStream to read the data from the same file and display it on the screen. We must note that in the above program, typecasting is applied to convert integer to character after read operation is performed.

Note: to observe the difference between text file and binary file, open the files created using the programs given in code listing 11.2 and 11.4.

Processing Input from Keyboard

In this section, let us explore the different ways to input the data to java program through the keyboard. As we know, a program can get input of data from live interaction through keyboard/GUI or it may take input as command line arguments or from files. Although there are many classes that facilitate the input from keyboard, here we will discuss two of the most widely used technique, using Scanner class of java.util package, and using the Console class of java.io package.

Scanner Class

Scanner class belongs to the java.util package; it provides various methods to read input from the keyboard or from the file. A special feature of this class is that it breaks the input string into tokens (words) using a delimiter. (White space is the default delimiter). Each token can be of different type, for example a string like "India-1947" can be read as "String-int" values. Let us explore the constructors and few methods of the Scanner class.

Scanner(String str)

Scanner(InputStream isobject)

Scanner(File fobject) throws FileNotFoundException

A Scanner object can be created from a string, file object or InputStream object. For instance we may use the constructor in the following way to read from a file and keyboard respectively:

```
Scanner fileinput = new Scanner(new File("Students.dat"));

Scanner kbinput = new Scanner(System.in);
```

Some of the important methods available with the Scanner class are listed in table 11.7.

Method	Descrption
void close()	Closes the Scanner
String next()	Returns the next token
boolean hasNext()	Returns true if there is a token in input
int nextInt()	Scans the next token of the input as Int.
float nextFloat()	Scans the next token of the input as Float.
String nextLine()	Scans the next token of the input as Line

Table 11.7 : Methods of Scanner Class

Let us now see a program that reads two numbers interactively from the user and displays the addition of those two numbers. Code listing 11.5 shows the program.

```
//Accepts input at command prompt
import java.io.*;
import java.util.*;
class ScannerInputDemo
{
        public static void main(String args[])
        {
                Scanner kbinput = null;
                int number1;
                int number2;
                int sum=0;
                try
                {
                        // Create an object of Scanner class
                        // that reads from Standard Input
                        kbinput = new Scanner(System.in);
                        System.out.println("Enter the first number : ");
                        //Read the integer number from console
                        number1 = kbinput.nextInt();
                        System.out.println("Enter the second number : ");
                        //Read the integer number from console
                        number2 = kbinput.nextInt();
                        sum = number1 + number2;
                        System.out.println("Sum is : " + sum);
                }
                catch(Exception eobj)
                {
                        System.out.println(eobj);
                }
                finally {
                        try {
                                kbinput.close();
                        }
                        catch(Exception eobj)
                        {
                                System.out.println(eobj);
                        }
                }

        }
}
```

Code listing 11.5 : Program to add two numbers

In the code listing 11.5, we have created an object of Scanner class, the constructor of Scanner class accepts "System.in" as an argument so that it reads from the standard input (keyboard). Both the numbers are scanned as integer numbers using the nextInt() method of the Scanner class. Figure 11.6 shows the output of the program.

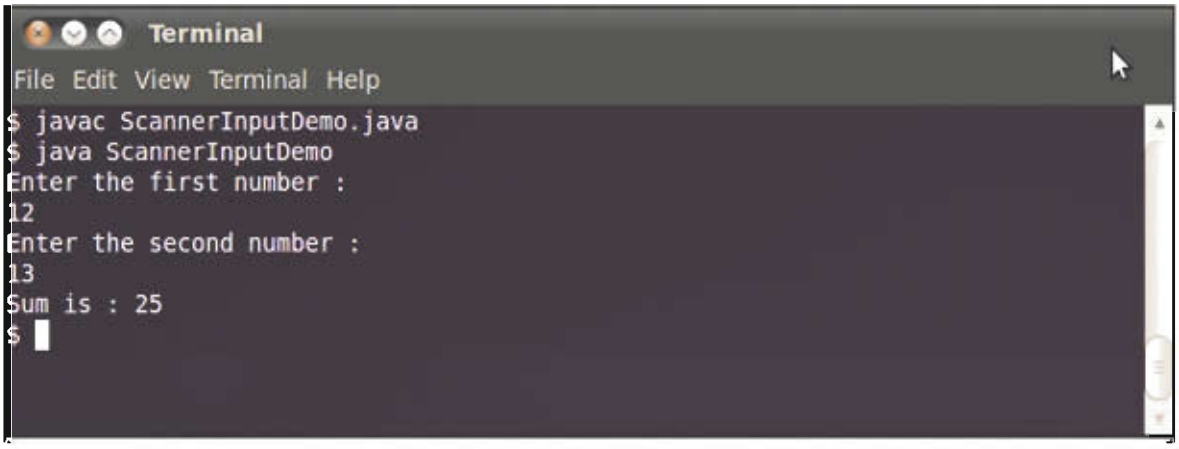


Figure 11.6 : Output of Code Listing 11.5

Scanner class can also be used to read from a file. Let us see an example where we use the Scanner class to read the data from a file that contains information about few students. It is assumed that the file "Students.dat" already exists. It contains five fields, student_no, student_name, marks of three subjects. The format and data of the file Students.dat is as shown below:

1 Akash 45 65 55

2 Badal 10 20 30

3 Zakir 45 40 60

4 David 65 50 75

We will write a program as shown in code listing 11.6 to read the data of each student and perform operations like calculating the total marks and displaying them on the output.

```
//Accepts input from a file "Students.dat" and calculate the total marks of each student
import java.io.*;
import java.util.*;
class ScannerFileDemo
{
        public static void main(String args[])
        {
                Scanner fileinput = null;
                int rollno, mark1, mark2, mark3, totalmarks;
                String name = null;
                File fobject;
        try
        {
```

```
                // Specify the file from where data is to be read
                fobject = new File("Students.dat");
                // Create an object of Scanner class that reads from File
                fileinput = new Scanner(fobject);
                //Display the default Delimiter to separate fields within a file
                System.out.println("Default delimeter is : " + fileinput.delimiter() + "\n");
                // Iterate to read the values of each record
                while(fileinput.hasNext()) {
                rollno = fileinput.nextInt();
                name = fileinput.next();
                mark1=fileinput.nextInt();
                mark2=fileinput.nextInt();
                mark3=fileinput.nextInt();
                totalmarks = mark1 + mark2 + mark3;
                System.out.println("Total marks of Rollno " + rollno + " ," +name+ " are
                                                                    : " + totalmarks);
                }
                fileinput.close();
        }
        catch(Exception eobj)
        {
                System.out.println(eobj);
        }
    }
}
```

Code Listing 11.6 : Program to calculate the total marks of each student

The output of the program is displayed on the monitor as shown in figure 11.7.

Figure 11.7 : Output of code Listing 11.6

Console Class

In the previous example, we used the Scanner class to read user input. Apart from the Scanner class, there is another class java.io.Console which can be used to get the input from the keyboard. We use this class especially when the input is to be typed in hidden form (characters must not be echoed on screen).

The Console class provides a method for reading password. When reading the password the user input will be hidden or not shown in the console screen. And it will return an array of character as the return type. The Console class belongs to java.io package. Few widely used methods of Console class are listed in table 11.8:

Methods	Description
String readLine()	This method reads a single line of text from the console.
char[] readPassword()	This method reads a password or passphrase from the console with echoing disabled.
Console printf(String format, Object args)	This method is used to write a formatted string to this console's output stream using the specified format string and arguments.

Table 11.8 : Few Methods of Console Class

The program shown in code listing 11.7 demonstrates the use of Console class to read username and password, further, it validates whether the username and password are correct or not.

```
// Program to reading passwords
import java.io.Console;
import java.util.Arrays;
public class ConsoleDemo {

    public static void main(String[] args) {
     Console console = System.console();
    String username = console.readLine("Username: ");
     char[] password = console.readPassword("Password: ");

   if (username.equals("admin") &&  String.valueOf(password).equals("secret")) {
      console.printf("Welcome to Java Application \n");
     } else {
     console.printf("Invalid username or password.\n");
      }
   }
}
```

Code Listing 11.7 : Program to read username and password

The output of the program is shown in figure 11.8. We have used two attempts to input the username and password. In the first attempt the username is false and that is why we use invalid username message. In the second attempt, we entered the correct username and password.

```
Terminal
File Edit View Terminal Help
$ javac ConsoleDemo.java
$ java ConsoleDemo
Username: Akash
Password:
Invalid username or password.
$ java ConsoleDemo
Username: admin
Password:
Welcome to Java Application
$
```

Figure 11.8 : Output of Code Listing 11.7

The program can be extended to read list of users and match their passwords from a file in the presence of multi-user environment.

Apart from various classes discussed in this chapter, java provides classes to store and retrieve objects using files. It also provides classes and methods to access a file randomly. Till now, we have seen the methods that perform sequential operations; however, there are classes which allow us to directly jump to nth record instead of accessing it sequentially. Discussion of these classes being out of scope for this text, we leave it to the students for exploring various other interesting features of java.io package.

Summary

In this chapter we learnt about file handling operations. We saw how to use the java.io.File class to perform file operations. We learnt the concept of stream and saw how input and output streams of different types can be used. We also learnt how to use the Scanner class to access data from a keyboard or a file. Finally we learnt how to use the Console class to input data from keyboard.

EXERCISE

1. Why is a file important in java programming? Under what circumstances will you store the data in files ?
2. State various operations that can be performed on a file and directory.

3. Why is the concept of Streams introduced in java, give the advantages of using streams.

4. Choose the most appropriate option from the following :

(1) Which of the following statements is true ?

(a) Volatile storage lasts only a few seconds.

(b) Volatile storage is lost when a computer is shutdown.

(c) Computer disks are volatile storage devices.

(d) All of the above are true.

(2) Which of the following refers to a collection of data stored on a nonvolatile device in a computer system ?

(a) file (b) application (c) volatile data (d) hard-disk

(3) The data hierarchy occurs in which of the following order from the smallest to largest piece of data ?

(a) file:character:field:record (b) file:character:record:field

(c) character:field:file:record (d) character:field:record:file

(4) Which of the following is true about streams ?

(a) Streams always flow in two directions

(b) Streams are channels through which the data flow

(c) Only one stream can be open in a program at a time

(d) All of the above are true

(5) Which of the following is used as a separator between fields of a record ?

(a) path (b) delimiter (c) variable (d) space

(6) Scanner class can be used to for performing which of the following operations ?

(a) accept input from the keyboard (b) read from the file

(c) parse a string separated by delimiters (d) All of the above

LABORATORY EXERCISES

1. Write a java program to list all the files in a given directory with ".txt" extension.

2. Write a java program to enter a filename, your program must check whether the file exists or not, if it exists then display the properties of that file.

3. Write a java program to create a file "friends.dat"; write the name of your friends in that file using Writer classes.

4. Write a java program in which a user enters the filename, further copy the file into another file.

5. Write a java program to copy the file /etc/passwd into your directory with the name mypasswd.dat.

6. Write a java program to count the number of characters in a given file.

7. Write a java program to display the size of file in terms of bytes.

8. Write a java program to calculate the simple interest, the user must enter principal amount, rate of interest and period in years.

9. Write an interactive program to convert inches into centimeter and reverse by entering the data at console.

Publishing documents using LaTeX 12

We have learned how to use OpenOffice.org Writer to create documents. In this chapter we shall discuss the TeX and LaTeX typesetting software. We shall discuss the advantages of using LaTeX. Then we will learn how to use LaTeX using TeX Live, a software package of the TeX/LaTeX software combination along with the SciTE text editor.

Using LaTeX

To use LaTeX, we need any LaTeX distribution (software). Most include TeX and some additional software in them. TeX Live is a very popular LaTeX distribution available in the standard Ubuntu repositories. We also need a plain text editor and software to view the output file. LaTeX can produce output in different file formats, and depending on the output file format, corresponding viewer software is needed.

LaTeX documents are typically created using any plain text editor (like gedit or SciTE). The different parts of the text are marked using LaTeX commands that associate a meaning to them. For example, **\title** is used to define the document's title, **\author** to specify the author(s) of the document and **\date** to indicate the date the document was created. Similarly, **\chapter, \section, \subsection, \paragraph** can be used to explicitly specify the logical structure of the document.

LaTeX comes with built-in ways to format these document elements in a pleasant-looking professional style. However, when typing the document, we will only see it as plain unformatted text with these commands written as part of the text. The document is then compiled (processed) using the LaTeX system and an output file is produced. LaTeX may also produce some additional files. In most cases, these additional files can be deleted safely without losing any information.

When we view the output file using appropriate software or print it on a printer, we see the formatted document. If we are not satisfied with the looks of the document, we may customize the built-in styles or define our own styles. Every time we make some modification in the source text, we need to compile it again for seeing the effect of our change on the output document.

Both TeX and LaTeX use the file extension .tex; LaTeX now has a command called pdflatex that produces the popular PDF (Portable Document Format) format files. PDF files can be viewed on screen as well as printed to printers and the printout looks exactly same as the monitor display. PDF files are very popular for sharing printable documents on the Web. PDF documents can be viewed in Ubuntu's default document viewer, evince. Hence the edit-compile-view cycle becomes

- Edit the document using any plain text editor like gedit
- Compile the document by issuing the command ***pdflatex filename*** at the command prompt (in the directory where the tex file was saved)
- View the generated PDF file by either opening it from the GUI or by issuing the command ***evince pdffilename*** at the command prompt (the terminal will display the next prompt only when you close the PDF file)

We may also use the SciTE editor for editing LaTeX files. While both gedit and SciTE have syntax highlighting (displaying different language elements in different colors for easy identification and readability), SciTE has one advantage over gedit - one can compile and view the document from within the SciTE program itself. As you are already familiar with SciTE, we shall use SciTE as the editor for learning LaTeX.

To use SciTE with pdflatex, we need to make changes to its configuration file as shown at the end of this chapter. The user needs to perform the operation only once, after installing the required software.

The LaTeX Language

LaTeX is essentially a markup language. The LaTeX source consists of plain text, with some parts of the text marked up using markers known as commands. Some commands are independent commands - they do not mark any specific part of the text. These commands can perform a variety of tasks when the document is processed by the LaTeX system - they may provide information about the text or the document, they may indicate the role of the marked text in the overall structure of the document (and hence cause LaTeX to format the text in certain way), they may directly specify formatting, or they may instruct LaTeX to process the document in a certain way (for example, use a certain page size, start a new chapter only on an odd-numbered page, etc.).

LaTeX commands start with a \ (backslash) character followed by the command name. The command name may be a string of alphabetic letters only, or may be a single non-letter. LaTeX commands are case-sensitive (capital and small letters are treated as being different). Some commands may accept additional information (for example, the \textcolor command expects the color in which the text is to be displayed). This additional information is called arguments.

There are two types of arguments. Optional arguments, as the name suggests, are not mandatory. We may or may not provide them. If we want to provide one or more optional arguments, we write them after the command name, enclosed in [ ] (square brackets) and separated by comma. These are followed by mandatory arguments (if there are any) in { } (curly braces) with each mandatory argument written in its own set of curly braces. For example, if we issue a command \documentclass[12pt]{article}; then documentclass is the name of the command, 12pt is an optional argument while article is a compulsory argument.

LaTeX treats all whitespace characters (the space, tab and newline characters) as the same. It converts all occurrences of multiple consecutive whitespace into a single space character. The white spaces at the beginning of a line are generally ignored and one or more consecutive blank lines are considered to mark the beginning of a new paragraph. This means that even if you type your text as several lines, it will appear in the output as a continuous flow unless there is a blank line in it. To insert a break in lines use \\ (the line break command) at the end of each line except the last line in the paragraph. Figure 12.1 shows an example of continuous text as well as how to insert explicit line breaks (We shall discuss the \textsf command later). Figure 12.2 shows the output of the LaTeX file.

```
\documentclass[12pt]{article}
\title{Line handling in \LaTeX}
\date{May 2013}
\begin{document}
 \maketitle
 \section{Continuous Text}\textsf{
 We have no wings,  we cannot fly
 But we have legs to sail and climb
 By slow degrees and by and by
 The cloudy summits of our time}
 \section{Text with Seperate Lines}\textsf{
 Heights by great men reached and kept \\
 Were not attained by a sudden flight \\
 But they, while their companions slept, \\
 Were toiling upwards in the night}
\end{document}
```

Figure 12.1 : Line Handling Example Source File

1 Continuous Text

We have no wings, we cannot fly But we have legs to sail and climb By slow degrees and by and by The cloudy summits of our time

2 Text with Seperate Lines

Heights by great men reached and kept
Were not attained by a sudden flight
But they, while their companions slept,
Were toiling upwards in the night

Figure 12.2 : Line Handling Example Output

The following characters are reserved characters in LaTeX;

$ % & _ (underscore) { } ^ ~ \

They have a special meaning in LaTeX. These characters cannot be used directly in our text in LaTeX. If we want to use them in our text, we must use the following forms.

\# \$ \% \& _ \{ \} \^{} \~{} \textbackslash{}

Note the special cases of the last three characters. The symbols < and > print very differently by default (except in math mode). Hence they must be written as **\textless** and **\textgreater.** The ` (grave accent or backquote) and ' (apostrophe or straight quote) are used around text to put it in single quotes, like `Book Code'. Double quotes are produced by repeating them twice, like ``Book Code'' (these are two straight quotes, not a single double quote). These may look odd in the source file, but are typeset properly in the output file.

LaTeX uses groups to mark portions of text. A group is enclosed between curly braces { and }. Any command in a group applies only to text following the command within that group. Also, some commands are followed by a group and apply to the whole group. Groups are useful for applying few commands to a small amount of text such as a part of a line, few lines or a paragraph.

For the cases where a multitude of commands must be applied (for example, for formatting a table or mathematical equation properly) or where some command(s) have to be applied to large portions of the text (like several paragraphs, whole sections), LaTeX provides a facility called environments; an environment begins with a \begin{environment-name} command and ends with a \end{environment-name} command.

All the formatting characteristics of the environment are applied to the entire text inside the environment. Environments can be nested; we can have one environment inside another. There are several standard environments meant for specific types of content, like equation, quotation, table and list that come with nice ready-made formatting commands for these specific content types.

While LaTeX documents are meant to be displayed and printed, LaTeX has several advanced features, including programming and automatically generating parts of the documents or multiple documents (mail merge). Often a complicated LaTeX template or package developed by one person or team is used by many others. Sometimes those people also need to tweak the LaTeX code.

In such cases, providing explanation for the complicated parts makes it easy for others to understand the code. Such explanation is provided in the form of comment. In LaTeX, the % character marks the beginning of a comment and everything from the % character up to the end of the line is treated as a comment. Comments are meant for the humans who read the LaTeX source code in a text editor for understanding and possibly modifying it. They are completely ignored by the compilation process and hence never make it into the output.

The Structure of a LaTeX Document

A LaTeX document has two parts namely preamble and content. The preamble contains metadata (data about data). In this case, the metadata is information about the document (for example, what kind of document it is, who is the author, when was it created.) and instructions on how LaTeX should process the document. The actual content is always inside the environment document, written between \begin{document} and \end{document}.

The Preamble

As LaTeX supports creation of a wide variety of documents, each with different characteristics and format, LaTeX needs to know what type of a document used as a source file is. The very first element in the preamble must be **\documentclass**(document-class-name} specifying the type of the document. Some common document classes are as shown in Table 12.1. Many document classes have options. Table 12.2 lists some common options.

Document Class	Purpose
article	For writing individual articles.
book	For writing entire books.
slides	For creating presentation slides. It automatically sets larger font size.
letter	For writing letters.
beamer	For generating presentations similar to office suites using the beamer package.

Table 12.1: Some Common Document Classes

Options	Function
10pt, 11pt, 12pt	Sets the size of the main font in the document to 10 points (the default), 11 points and 12 points respectively.
a4paper, letterpaper, legalpaper	Defines the paper size. These are several international standard paper sizes. The most common paper sizes in regular office use are A4, letter and legal.
fleqn	Displayed formulas and equations are flushed left (left-aligned) rather than centered (the default).
landscape	Changes the layout of the document to print in landscape mode.

Table 12.2 : Some Common Options of Some Document Classes

The document class declaration is followed by optional package declaration. While the LaTeX system itself provides for many common typesetting requirements, it also recognizes that it cannot provide everything that users may need. Hence LaTeX allows users to write packages that provide additional functionality. There is a large community of LaTeX users who develop new LaTeX packages or enhance existing ones to cater to their own needs and then share them with others over the Comprehensive TeX Archive Network (CTAN).

LaTeX distributions themselves usually come with a large number of such packages preinstalled. To use one or more packages in our document, we need to declare them in the preamble as \usepackage{package-name}. Some packages may also have options to customize their behaviour. If we are not going to use options for any of the packages, multiple package names can be declared in a single \usepackage command, separated by commas. Table 12.3 shows only a few of the commonly used packages.

Package	Description
amsmath	It contains the advanced math extensions for LaTeX originally developed for the American Mathematical Society.
color	It adds support for colored text.
easylist	Adds support for multilevel lists.
geometry	For page layout tasks like setting paper size, orientation, margins, etc.
listings	Has special features for including programming code within the document
setspace	Lets you change line spacing

Table 12.3 : Some Commonly Used Packages

Three more pieces of information are typically supplied. They are as mentioned :

\title{the-title-of-the-document}

\author{author(s) of the document}

\date{date of creation / last update of the document, in any format}

It is necessary to provide the title and the author if you want LaTeX to create an automatic title for you. Providing date is optional; if it is omitted then the date of compilation is used in the title. These pieces of information can be supplied in the preamble or as the first thing in the document environment also.

The Document Environment

The document environments for articles and slides will only have a title followed by the main content of the document. A title is automatically generated by LaTeX when it sees the \maketitle command. Of course, the \title, \author and \date commands must precede the \maketitle command because their information is used in creating the title.

A book can have a far more elaborate structure, though most elements are optional. The document environment of a book is divided into three main parts - the front matter, the main matter and the back matter, denoted by the commands \frontmatter, \mainmatter and \backmatter respectively. This structure has things like the title, the table of contents and the preface in the front matter, the bibliography, index and references in the back matter and the primary content in the form of chapters, sections and subsections in the main matter.

The main contents of a book has a hierarchical structure, where a book is divided into parts, parts are divided into chapters, chapters are divided into sections which are divided into subsections which are divided into sub subsections which are divided into paragraphs which are divided into subparagraphs. These are marked with the commands \part, \chapter, \section, \subsection, \subsubsection, \paragraph and \subparagraph respectively.

Each accepts one compulsory argument, the title; and one optional argument, the title to be displayed in the table of contents. They start a new part, chapter or section respectively. The title to be displayed in the main text may be longer and may have internal formatting (like boldface or italics applied to a part of the title), while the title to be displayed in the table of contents is expected to be short and devoid of any special formatting to maintain consistency in looks. These 7 elements nest inside one another and have an integer level assigned to them, with part having a level of 1, chapter a level of 0, section a level of 1 and so on.

The parts, chapters, sections are numbered automatically by LaTeX. Hence the author does not have to worry about it. The author can also move chapters, sections and subsections without having to bother about renumbering. The parts are numbered in Roman numerals (I, II, III, and so on.), while the chapters, sections, subsections, etc. are numbered in Arabic numerals (1, 2, 3, and so on.).

All chapters after the special \appendix command (to be used only once) are treated as appendices and get uppercase alphabetic numbering (A, B, C, and so on.). Pages in the front matter are numbered

using Roman numerals, while the pages in the main matter and back matter are numbered in Arabic numerals with the numbering restarting from 1. While both the front matter and back matter can have chapters (like preface, acknowledgments, bibliography), those chapters usually do not have sections or other sub-elements. The commands defining different elements of the document also have their starred equivalents that are not numbered. For example, \section* can be used to create a section that is not automatically numbered.

By default elements are assigned number up to level 2 that is up to subsections. Sub subsections and further divisions are not assigned numbers. This can be changed by modifying one of the built-in counters of LaTeX in the preamble. For example, the command

\setcounter{secnumdepth}{3} ensures that elements up to level 3 (subsubsection) are assigned numbers. Elements are assigned a number formed by appending a period (.) and the element number to the number of the parent element. Chapters are an exception as their assigned number does not have the part number and period in front of them. Chapters are assigned simple Arabic numeral numbers. If part II of a book has chapter 5 in it, which, in turn, has section 4 in it that has subsection 1 in it, the subsection would be numbered 5.4.1.

A well-formatted table of contents (TOC) is generated automatically from the element titles by LaTeX when it encounters the command \tableofcontents. Again, by default a TOC has entries up to level 2 (subsection), but this can be changed by altering the value of another built-in counter tocdepth.

There is one important point to be noted. LaTeX processes the source file sequentially from beginning to end in a single pass. It produces the output file also sequentially. It cannot move back and forth in either of the file. This poses a problem. The TOC comes early in the document and must output the entries for chapters or sections along with their page numbers. However, at that point LaTeX has no knowledge about chapters or sections that would follow. In this kind of situation, LaTeX must be run multiple times.

In the first run LaTeX collects information about the document structure and stores it in supplementary files. The TOC in the output files would be empty at this point or may have information from old supplementary files. In the second run, it will pick up the correct information from the supplementary files in the beginning to produce the correct TOC.

Just like the TOC, LaTeX can also maintain list of figures, list of tables, cross references, bibliography, glossary or index automatically. This offloads a great burden from the author's back and is a major reason for the popularity of LaTeX.

Example

To have an idea of how LaTeX typesets books, let us create one source file and see its output.

We will use the SciTE editor with LaTeX to work on the example.

- Create a new file in SciTE using the **File → New** menu option.
- Type the content shown in Listing 12.1 in a SciTE editor.

```
\documentclass[12pt]{book}
\usepackage{amsmath}
\title{\huge Mathematics \\[3\baselineskip]
\Large Standard 12}
\author{Gujarat State Board of School Textbooks}
\date{2013}
\setcounter{secnumdepth}{2}
\setcounter{tocdepth}{1}
 \begin{document}
 \frontmatter
  \maketitle
  \chapter{\MakeUppercase{Fundamental Duties}}
  \tableofcontents
  \chapter{\MakeUppercase{About This Textbook...}}
\mainmatter
 \part{Semester I}
 \chapter{Set Operations}
 \section{Introduction}
 \section*{Exercise 1.1}
 \section{Properties of the Union Operation}
 \subsection{Union is a Binary Operation}
 \section{Properties of the Intersection Operation}
 \subsection{Intersection is a Binary Operation}
 \subsection{Associative Law}
 \chapter{Number Systems}
 \section{Introduction}
 \section*{Exercise 2.1}
 \section{Irrational Numbers}
 \chapter{Polynomials}
 \chapter{Coordinate Geometry}
 \chapter{Some Primary Concepts in Geometry : 1}
 \chapter*{Answers}
 \markboth{\MakeUppercase{Answers}}{}
 \addcontentsline{toc}{chapter}{Answers}
 \part{Semester II}
 \chapter{Quadrilaterals}
```

```
\section{Introduction}
\section{Plane Quadrilateral}
\chapter{Areas of Parallelograms and Triangles}
\section{Introduction}
\section{Interior of a Triangle}
\chapter{Circle}
\chapter{Surface Area and Volume}
\chapter*{Answers}
\markboth{\MakeUppercase{Answers}}{}
\addcontentsline{toc}{chapter}{Answers}
\appendix
\chapter{Terminology}
\backmatter
\end{document}
```

Listing 12.1 : A Sample Book in LaTeX

- Save the file using the **File → Save** menu option. Note that the extension of the file should be .tex.
- Now select the **Tools → Build** menu option (shortcut key: F7) to compile your LaTeX file. The output window will show several messages. If the last line (in blue color) reads Exit code: 0 then compilation was successful. Otherwise the error messages may point to the error(s), but often they are difficult to interpret.
- If the compilation was successful, you may select the **Tools → Go** menu option (shortcut key: F5) to view the file in the default document viewer. Don't forget to close the document viewer before returning to SciTE.

Note that we have done some customization of the default book style in listing 12.1. Some of the features used are not discussed in this text. LaTeX typesetting is heavily customizable.

Text Formatting

In LaTeX documents, we type a paragraph in a continuous flow without pressing the ENTER key. LaTeX then adjusts the text automatically. It decides how much text should go in the first line, how much text should go in the second line, and so on according to the page width, font size, alignment option used. LaTeX usually avoids breaking a word in two parts. Where it must do so, it has a hyphenation algorithm to decide the most appropriate way of breaking the word with a hyphen (-). For example, if the word formatting cannot be accommodated on a single line, the first line may end with formatt- and the next line may begin with ing.

On the other hand, there are some situations where some text, even though technically forming multiple words, should not be split into multiple lines. For example, up to are two words, and yet, it is not desirable to have one line ending in up and the next line beginning with to because up is an independent word with different meaning; so the reader is surprised for a moment on seeing to on the next line. This is an obstacle

to smooth reading experience. To avoid it, both up and to should be on the same line, either the first line or the second. In LaTeX, this can be taken care of by inserting a non-breaking space between up and to. LaTeX uses the ~ (tilde) character to denote a non-breaking space.

LaTeX divides font families into three categorie; Roman (also called serif) fonts have a tiny line or curve called serif at the end of the strokes (lines), sans serif fonts do not have serifs while monospace fonts use equal width for all characters. Monospace fonts are typically used for computer code listings. The default fonts are Roman. These three types of fonts can be used for any text by employing the commands \textrm{text}, \textsf{text} and \texttt{text} respectively. You can see the difference between serif and sans serif fonts in figure 12.2 where the section titles are in the default serif fonts while the body text is in sans serif.

The font size can be changed using the commands \tiny, \scriptsize, \footnotesize, \small, \normalsize, \large, \LARGE, \huge and \Huge. Notice that the commands are case-sensitive. The commands \textbf, \textit and \emph can be used to add the bold, italic and emphasis (generally same as italics) effect to text. \textsc provides small capital letters. Superscripts and subscripts can be created in text mode using the commands \textsuperscript and \textsubscript respectively from the package fixltx2e.

Paragraph Formatting

In LaTeX, the setspace package provides the singlespace, onehalfspace, doublespace and spacing{amount-of-spacing} environments for setting the line spacing. By default, body text is fully justified in LaTeX. To achieve left alignment, right alignment or center alignment, one may use the environments flushleft, flushright and center respectively. The first line of a paragraph is indented, except for the paragraphs immediately following a heading.

The \indent and \noindent commands can be used immediately before a paragraph to explicitly make the first line indented and unindented respectively. The verbatim environment outputs everything inside it (including special characters, spaces, newlines and LaTeX commands) as it is without any processing. The moreverb package provides a listing environment with one mandatory argument line-number-of-first-line for program code listing with line numbers.

Page Layout

In LaTeX, the geometry package can be used for page layout. The paper size and margins can be passed as optional arguments with the \usepackage command itself. E.g. the command

\usepackage[a4paper, top=1in, bottom=2in, left=1.5in, right=1in]{geometry}

sets the page size to A4, top margin to 1", bottom margin to 2", left margin to 1.5" and right margin to 1". Page sizes have been standardized internationally. The page sizes commonly used with regular printers are A4, letter and legal. These can be specified by a4paper, letterpaper and legalpaper respectively. The page orientation can also be specified using the portrait (default) and landscape options.

Documents can be either one-sided or two-sided. Articles are by default one-sided while books are two-sided. Two-sided documents differentiate between the left (even) and right (odd) pages and can have different margins for both, to take care of amount of page space used up by the binding. There can also be rules like all chapters must start on an odd page.

Typesetting Mathematical Content in LaTeX

Ability of automatically laying out complex mathematical content is a major strength of LaTeX. The most common way of laying out mathematical content in LaTeX is using the packages amsmath, amssymb and amsfonts created by the American Mathematical Society.

The amsmath package defines several environments for mathematical content. There are two ways of typesetting formulas and equations. We may have them printed as part of the running text (inline) or they may be printed independently on their own lines (called display in LaTeX parlance). The former form can be obtained using the math environment, while the latter can be obtained using the displaymath environment.

The equation environment is a display environment that automatically numbers equations. A convenient way of embedding math environment in running text is to enclose the mathematical content between $...$. In the mathematical environments, each letter is treated as a mathematical variable. Hence they are different from the text environments. To better understand these environments, create a tex file by using code in listing 12.2.

- Create a new file in SciTE using the **File → New** menu option.
- Type the content shown in Listing 12.2 in a SciTE editor.

```
\documentclass[12pt]{article}
\usepackage{amsmath}
\title{Introduction to \LaTeX}
\date{May 2013}
\begin{document}
\section*{math environment}
The quadratic equation, in its general form, is
\begin{math}
ax^2 + bx + c = 0
\end{math}.
You learnt about them in class X.

The quadratic equation, in its general form, is $ax^2 + bx + c = 0$. You learnt about
them in class X.

\section*{displaymath environment}
The quadratic equation, in its general form, is
\begin{displaymath}
ax^2 + bx + c = 0
\end{displaymath}. You learnt about them in class X.

\end{document}
```

Listing 12.2 : Demonstration of the math environments

- Save the file using the **File → Save** menu option.
- Now select the **Tools → Build** menu option (shortcut key: F7) to compile your LaTeX file.
- If the compilation was successful, you may select the **Tools → Go** menu option (shortcut key: F5) to view the file in the default document viewer. Figure 12.3 shows the output of the file when viewed in document viewer.

math environment

The quadratic equation, in its general form, is $ax^2 + bx + c = 0$. You learnt about them in class X.

The quadratic equation, in its general form, is $ax^2 + bx + c = 0$. You learnt about them in class X.

displaymath environment

The quadratic equation, in its general form, is

$$ax^2 + bx + c = 0$$

. You learnt about them in class X.

Figure 12.3 : Part output of the TeX code in Listing 12.2

Using Mathematical Symbols

Mathematics uses a large number of symbols. We shall discuss how some of those symbols can be used in LaTeX. Letters of the Greek alphabet have their corresponding commands, like \alpha, \beta, \gamma, \pi that produce the lower case letters. The same commands, when used with the first letter in the uppercase, for example \Alpha produce capital Greek letters.

There are commands for other mathematical symbols as well. Figure 12.4 and Figure 12.5 show some LaTeX commands and the mathematical symbols produced by them using the AMS packages. The reason for providing commands for the trigonometric functions sin, cos and others are that they need to be recognized and typeset as functions rather than as individual variables.

$<$	$<$	$>$	$>$
$=$	$=$	\leq	$\leq$
\geq	$\geq$	\neq	$\neq$
\times	$\times$	\div	$\div$
\pm	$\pm$	\mp	$\mp$
\in	$\in$	\notin	$\notin$
\supset	$\supset$	\subset	$\subset$
\supseteq	$\supseteq$	\subseteq	$\subseteq$
\cup	$\cup$	\cap	$\cap$

Figure 12.4 : Some Mathematical Symbols in LaTeX

\cong	$\cong$	\propto	$\propto$
\rightarrow	$\rightarrow$	\parallel	$\parallel$
\leftrightarrow	$\leftrightarrow$	\leftarrow	$\leftarrow$
\angle	$\angle$	\bigodot	$\bigodot$
\triangle	$\triangle$	\overleftrightarrow{AB}	$\overleftrightarrow{AB}$
\stackrel\frown{AB}	$\stackrel{\frown}{AB}$	\overrightarrow{AB}	$\overrightarrow{AB}$
\overline{AB}	$\overline{AB}$	\perp	$\perp$
45^{\circ}	45°	\implies	$\implies$
\iff	$\iff$	\therefore	$\therefore$
\because	$\because$	\sin	$\sin$
\cos	$\cos$	\tan	$\tan$
\sec	$\sec$	\csc	$\csc$
\cot	$\cot$	\theta	θ

Figure 12.5 : Some More Mathematical Symbols in LaTeX

Using Mathematical Operators

LaTeX supports a large number of mathematical operators. The power operator (x2) and the index operator (x1) are not provided separately, but are implemented using the generic superscript operator ^ (the caret character) and the generic subscript operator _ (the underscore character) respectively. The superscript operator raises the text that follows it, while the subscript operator lowers it. Both reduce the size of the text as well. Absolute values can be denoted by enclosing the expression between two | (vertical bar) symbols.

Fractions are created using the command \frac{numerator}{denominator}, while square root of a number x is denoted using the command \sqrt{x}. These operators can be nested one inside another. That means we may have a fraction in the denominator of another fraction, whose square root is in the numerator of yet another fraction, and so on. There is no practical limit on this, and LaTeX takes care of sizing and placing the elements appropriately. Round brackets work normally. An example usage of mathematical operators is given in listing 12.3.

- Create a new file in SciTE using the **File → New** menu option.
- Type the content shown in Listing 12.3 in a SciTE editor.

```
\documentclass[12pt]{article}
\usepackage{amsmath}
\setlength{\parindent}{0pt}
\title{Introduction to \LaTeX}
\date{May 2013}
\begin{document}
\begin{math}
x^2 \\[6pt]
x_1 \\[6pt]
x_i \\[6pt]
x_i^2 \\[6pt]
x^y \\[6pt]
a^bc \\[6pt]
{a^b}^c \\[6pt]
a^{b^c} \\[6pt]
\frac{x}{y} \\[6pt]
\sqrt{b} \\[6pt]
\frac{\frac{1}{3}+\frac{3}{2}}{\frac{2}{3}+\frac{1}{2}} \\[6pt]
\frac{-b\pm\sqrt{b^2-4ac}}{2a} \\[6pt]
\frac{-b\pm\sqrt{\Delta}}{2a} \\[6pt]
\sqrt{1+\frac{1}{\sqrt{1+\frac{1}{\sqrt{1+\frac{1}{2}}}}}} \\[6pt]
\sqrt{(x_1-x_2)^2+(y_1-y_2)^2} \\[6pt]
|x-y| \\[6pt]
(\frac{x_1+x_2}{2},\frac{y_1+y_2}{2})
\end{math}
\end{document}
```

Listing 12.3 : Example of Mathematical Operators

- Save the file using the **File → Save** menu option.
- Now select the **Tools → Build** menu option (shortcut key: F7) to compile your LaTeX file.
- If the compilation was successful, you may select the **Tools → Go** menu option (shortcut key : F5) to view the file in the default document viewer. Figure 12.6 shows the output of the file when viewed in document viewer.

$$x^2 \qquad \sqrt{b}$$

$$x_1 \qquad \frac{\frac{1}{3}+\frac{3}{2}}{\frac{2}{3}+\frac{1}{2}}$$

$$x_i \qquad \frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

$$x_i^2 \qquad \frac{-b\pm\sqrt{\Delta}}{2a}$$

$$x^y \qquad \sqrt{1+\frac{1}{\sqrt{1+\frac{1}{\sqrt{1+\frac{1}{2}}}}}}$$

$$a^b c \qquad \sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$

$$a^{b^c} \qquad |x-y|$$

$$a^{b^c} \qquad \left(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2}\right)$$

$$\frac{x}{y}$$

Figure 12.6 : Output of the TeX code in Listing 12.3 broken in two parts

Using Equations

LaTeX provides a special equation environment for typesetting equations. Each equation is to be enclosed in the equation environment, which cannot be embedded in a math environment. Equations are numbered automatically and center-aligned. Listing 12.4 shows an example of using the equation environment, while Figure 12.7 shows a part of the output.

```
\documentclass[12pt]{article}
\setlength{\parindent}{0pt}
\usepackage{amsmath}
\title{Introduction to \LaTeX}
\date{May 2013}
\begin{document}
\begin{equation}
\sin^2\theta + \cos^2\theta=1
\end{equation}
\begin{equation}
```

```
\sec^2\theta - \tan^2\theta=1
\end{equation}
\begin{equation}
\csc^2\theta - \cot^2\theta=1
\end{equation}
\end{document}
```

Listing 12.4 : Using the Equation environment

$$\sin^2\theta + \cos^2\theta = 1 \qquad (1)$$

$$\sec^2\theta - \tan^2\theta = 1 \qquad (2)$$

$$\csc^2\theta - \cot^2\theta = 1 \qquad (3)$$

Figure 12.7 : Output of the TeX code in Listing 12.4

Advantages of LaTeX

LaTeX has several benefits. Just as TeX was extended to LaTeX, LaTeX itself also can be extended. Anyone can create additional packages to enhance the features provided in LaTeX, to add new features or to provide alternate implementations. Thousands of such packages have been created by LaTeX users around the world to cater to different needs. Most of them are free. They are hosted on the CTAN (The Comprehensive TeX Archive Network) website at www.ctan.org.

LaTeX is extremely good at laying out complex mathematical formulae in nice looking and appropriate way. Because of this, it is quite popular among authors and publishers in mathematics, engineering, computer science and other technical areas. Because it is open source, highly systematic and is developed in the academic spirit of knowledge sharing and collaboration, it is popular among academicians and scholars. People in these fields use LaTeX, share their views and experiences, help one another, develop and share new packages and drive the further development of LaTeX.

LaTeX has built-in facilities to automatically update numbering and references and to automatically create table of contents, indexes, and other such needs, taking a major burden off the mind of the author.

Summary

In this chapter, we learnt how to use LaTeX for typesetting technical and mathematical documents. We studied the essentials of using LaTeX. We discussed its syntax and the structure of a LaTeX document. Then we discussed basic document creation and formatting in LaTeX. Finally, we saw how LaTeX can be used to typeset mathematical content, one of its major strength. We also discussed how it differs from word processors. We discussed the advantages of using LaTeX.

Configuring SciTE and LaTeX (Only for teachers)

- Close SciTE if it is already open.
- Open the terminal.
- Run the command

  sudo gedit /usr/share/scite/tex.properties&

  This will open the file in gedit as administrator. Edit it as follows :
- Delete the lines

  file.patterns.tex=*.tex;*.sty

  file.patterns.context=*.tex;*.tui;*.tuo;*.sty

  Note make sure that you do not delete the line

  file.patterns.latex=*.tex;*.sty;*.aux;*.toc;*.idx
- Change the line

  command.go.$(file.patterns.latex)=gv $(FileName).pdf

  to

  command.go.$(file.patterns.latex)=evince $(FileName).pdf
- Save the file and close gedit.
- Open SciTE; open a proper LaTeX file and press F7 followed by F5. The LaTeX file should compile and the PDF file should open. Close the PDF file before returning to SciTE.

EXERCISE

1. Compare LaTeX with word processors. List the strengths and weaknesses of both.
2. List key reasons for the popularity of LaTeX.
3. List the reserved characters and characters that cannot be used directly in LaTeX.
4. Explain the structure of a LaTeX document.
5. Explain the \frac command and its nesting inside another \frac command with an example.
6. Choose the most appropriate option from those given below :

   (1) Modern word processing software operates in which of the following mode ?

   (a) WIGIWIS (b) WISYWIG (c) WYSIWYG (d) WISYWYG

(2) Which of the following is not a reserved character in LaTeX ?

(a) @ (b) % (c) $ (d) ^

(3) Which of the following begins with \begin{name} and ends with \end{name}.

(a) group (b) section (c) environment (d) preamble

(4) Which of the following character is used to mark a comment in LaTeX ?

(a) $ (b) % (c) # (d) &

(5) Which of the following part of the LaTeX document contains the metadata ?

(a) preface (b) TOC (c) preamble (d) environment

(6) Which of the following web site hosts the LaTeX packages ?

(a) CTAN (b) CLAN (c) CTEN (d) CLEN

(7) Which of the following will not be automatically numbered ?

(a) \section (b) \subsection (c) \chapter* (d) \part

(8) Which of these environments displays mathematical content inline with the text ?

(a) displaymath (b) math (c) equation (d) text

(9) Which of the following commands generate the set union symbol ?

(a) \cup (b) \setunion (c) \cap (d) \union

(10) Which of the following operator is used to denote a subscript or an index ?

(a) _ (underscore) (b) ^ (caret) (c) - (minus) (d) < (less than)

(11) Which of the following refer to the very first line in the preamble ?

(a) \usepackage (b) \title (c) \maketitle (d) \documentclass

LABORATORY EXERCISE

1. Create a skeletal structure of this book in LaTeX with preface, table of contents, chapters, sections, subsections, etc. Type only a few words in each.
2. Create a LaTeX document to illustrate text formatting.
3. Create a LaTeX document to illustrate paragraph formatting.
4. Create a LaTeX document and experiment with page layout.
5. Create LaTeX documents with the given contents.

(a) If A = {x | x ∈ N, x ≤ 4}, B = {-1, 0, 1, 2, 3} and C = {0, 1, 2} then (A ∪ B) ∩ (A ∪ C) = {0, 1, 2, 3, 4}.

(b) If A = {x | x ∈ N, x ≤ 7} and B = {2, 4, 6} then B ⊂ C.

(c) $\frac{(81)^{\frac{1}{4}}}{(625)^{\frac{1}{4}}}+\frac{(216)^{\frac{1}{3}}}{(8)^{\frac{1}{3}}}-(729)^{\frac{1}{6}}$

(d) X ⊂ Y and Y ⊂ X ⇒ X = Y

(e) $\overrightarrow{PQ}\cap\overrightarrow{PR}=P$

(f) $BE\cap EG=\{E\}$

(g) $\sqrt{s(s-a)(s-b)(s-c)}$

(h) $Areaof\odot(O,r)=\pi r^2$

(i) $\sqrt{a+2\sqrt{b}}=\sqrt{x}+\sqrt{y}$

(j) $x=\frac{a+\sqrt{a^2-4b}}{2}, y=\frac{a-\sqrt{a^2-4b}}{2}$

(k) $\frac{h(\tan\alpha-\tan\beta)}{\tan\alpha\tan\beta}$

Other useful free tools and services 13

In this chapter we shall discuss some useful free tools and services. We shall have a quick introduction to data compression techniques. These techniques allow us to reduce the amount of memory and disk space required to store the information. We shall discuss the Archive Manager tool that lets us apply these techniques to files and entire directories. We shall briefly discuss VLC media player, a versatile multimedia tool. We shall also discuss the Google Maps service, which is an online service providing, maps of different areas. We will see two small but useful applications - Character Map, which allows us to insert various symbols and characters from different languages in our text. We shall then have an overview of the R environment for statistical computing. We shall learn some basic statistical operations like finding mean and median in R and also learn to draw bar graphs and histograms in R. Lastly we will look at two tools RationaPlan and Skype. While some of these applications come pre-installed with Ubuntu 10.04 LTS, others need to be installed from the Ubuntu repositories.

Data Compression

A modern computer system has lakhs of files on them. Often one needs to transfer a large number of these files or an entire directory structure to another computer or storage device. When we want to transfer computer files (containing data or programs or both) to another computer or storage device, the amount of data to be transferred or stored becomes a concern. Both computer networks and external storage devices are usually not as fast as internal components of a computer. Thus, transferring larger amount of data may take more time. If the Internet is used for such transfers, more time may be taken in the transfer due to slow Internet speed. Also, such transfers put load on the usually clogged Internet connection. Unless the user or organization has unlimited Internet plan, higher amount of data may result in higher cost as well.

Similarly, if the files and directories are transferred to a storage device, the amount of the data to be transferred again becomes an issue because of the finite capacity of storage devices and the multiple uses that they are put through. Considering these issues, there is a need to reduce the amount of storage space occupied by computer files (and entire directory structures), wherever possible. From a convenience point of view, in many cases it is also desirable to have a single file to handle rather than a large bunch of files or a complex directory structure.

Computer scientists have developed techniques to place a whole directory structure into a single file for convenience. Such a file is called an "archive". They have also developed a number of techniques for reducing the storage requirements of computer files and directory structures. These techniques are called data compression.

Data compression generally works by identifying repetition in the data and encoding the data in a way that reduces or eliminates such repetition. Some techniques also identify and eliminate less important information to conserve space. As an example, consider the children's rhyme shown in

figure 13.1. It certainly has a lot of repetition of words. We may take advantage of this and represent each word by a single digit or letter. At the beginning of our encoded file we shall have a table that tells us which digit/letter represents which word (this information is needed to convert the file back in its original uncompressed form). Afterwards, we may use a single digit/letter to represent a word. Thus, lengthy words appear only one time and subsequently they are represented by a single character. We leave punctuation as it is. We may mark the beginning of the table with a ^ (caret) symbol and end of the table with a $ symbol. The encoded file is also shown in the second half of figure 13.1.

```
One little, two little, three little Indians
Four little, five little, six little Indians
Seven little, eight little, nine little Indians
Ten little Indian boys.

Ten little, nine little, eight little Indians
Seven little, six little, five little Indians
Four little, three little, two little Indians
One little Indian boy.
```

Actual Contents

```
^0:One 1:little 2:two 3:three 4:Indians 5:Four
6:five 7:six 8:Seven 9:eight A:nine B:Ten
C:Indian D:boys E:boy$0 1, 2 1, 3 1 4
5 1, 6 1, 7 1 4
8 1, 9 1, A 1 4
B 1 C D.

B 1, A 1, 9 1 4
8 1, 7 1, 6 1 4
5 1, 3 1, 2 1 4
0 1 C E.
```

Encoded Contents

Figure 13.1 : Example of data compression

While the size of the original file was 324 bytes, the size of the encoded file is just 226 bytes. This is because each word is used in its full form only once; subsequently it is represented by a single character. The encoded file can be transformed into the original file at any time using the reverse process.

The scheme described here is fairly simple. Actual data compression programs use far more sophisticated algorithms based on information theory to achieve even greater reduction in size. Fortunately, Linux provides ready-made free and open source software for managing archive files. It is called Archive Manager.

Archive Manager

Archive Manager as shown in figure 13.2 permits us to combine many files or an entire directory tree into a single file known as an archive. On Linux systems, tar (tape archiver) is the most common archive format. It also provides the facility of compressing the archive to reduce its file size. It supports multiple compression algorithms and compressed file formats. The most common compressed file formats are the zip file format and the tar.gz file format. While the zip file format supports storing multiple files in a single zip file, on Unix/Linux systems the common practice is to combine the files into a single uncompressed file in the tar format and then compress the file in the zip format using gzip (GNU zip, an open source archiving program). Such a file commonly has the extension tar.gz and is called a "tar ball". The tar ball is a very common format for distributing software or bundles of files on the Linux platform. The zip format is also used by different applications under different names. For example, the JAR files in Java and the OpenOffice.org office suite file formats use zip compression. Archive Manager uses the extension part of the filename to identify the file format when opening an archive and to select the file format when creating a new archive.

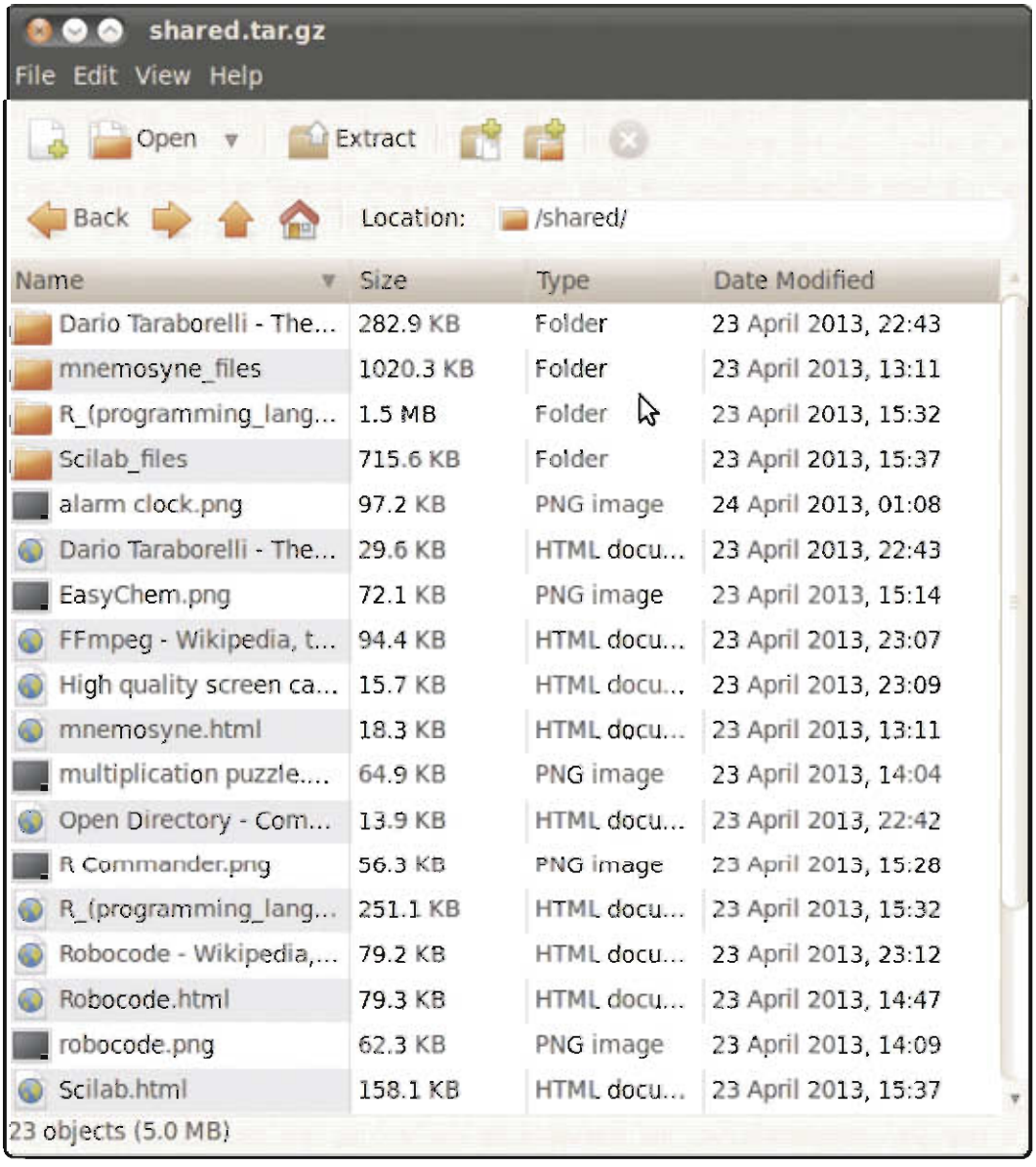


Figure 13.2 : Archive Manager

Archive Manager can also be used to explore the contents of archives, to extract files from the archives, to add new files to the archive, to delete files from the archive, and perform other such operations. There is no menu item to start the archive manager by default. One may double-click on an archive file in the Nautilus file browser to start the archive manager. A new archive can be created by right-clicking a file or directory and selecting the Compress... option from the context menu. This will create a new archive with that file or directory as its contents. This option is not available for files that are themselves archives. If we wish, we may add Archive Manager to the Accessories sub menu of the Application menu. For this, right-click the Ubuntu icon in the upper-left corner, select Edit Menu from the list, click on Accessories in the dialog box that opens and select the check box for Archive Manager and close the dialog box.

When an archive is open in the Archive Manager, the view is similar to a directory open in the Nautilus file browser as can be seen in figure 13.2. The operations and keyboard shortcuts are also similar. Double-clicking a directory opens that directory and shows its contents. The Up and Back buttons in the toolbar can be used to go to the parent directory (if it exists in the archive) and the previous directory (if any) respectively. They are enabled only when applicable.

Thus the Archive Manager helps us in creating and using archive files for backup, for transfer via an external storage device, for transfer via network, for saving disk space, etc.

The VLC Media Player

We have discussed earlier that Ubuntu comes with built-in tools to play multimedia content like audio, video, etc. However, another great open source tool called VLC (originally it was short form of VideoLAN Client) media player is shown in figure 13.3. It is also extremely popular because if its versatility and features.

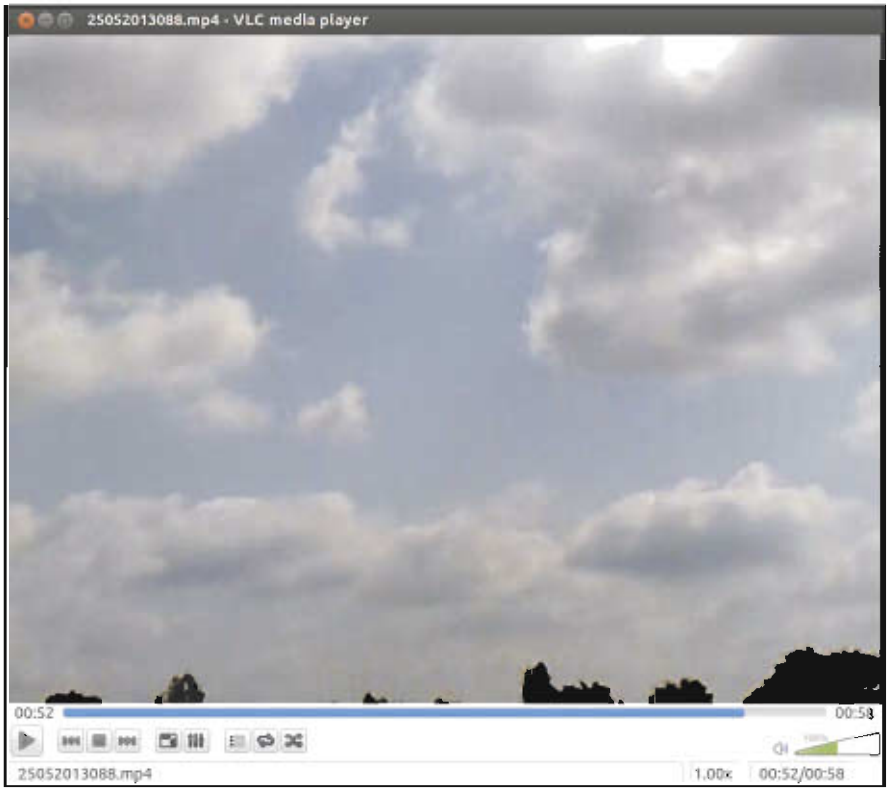


Figure 13.3 : The VLC Media Player

Started as an academic project by the music-loving students of a university in Paris, it is now a community project available under multiple operating systems and has been downloaded over a billion times. One of the issues faced with multimedia content is that there are a variety of ways of converting multimedia information coming from hardware devices (audio/video streams) into computer data and then converting the computer data back into audio/video streams for playing on the hardware devices. This conversion (coding) and reverse conversion (decoding) is performed by a software component called codec (coder decoder). Each multimedia data format requires its own codec. The advantage of VLC is that it supports all the popular codecs and hence all the popular formats. It also supports all the major types of devices including web cameras, HD monitors, speakers of all types, microphones, headphones. VLC provides several options for playback of audio and video. It can convert multimedia files from one format to another. It can also stream (send) audio/video to and receive audio/video from another computer(s) over the network.

After starting VLC from the **Applications → Sound & Video** menu, we may open one or more files using the **Media → Open File...** menu item. We may open an entire directory using the **Media → Open Directory...** option. If we open a directory, VLC will actually open all media files in that directory playable by it. When one or more files are opened, they are added to the playlist. The playlist is a list of media files (tracks) to be played. We may play the media in the playlist in sequence, or in random order. We may go to the previous or next track. We may also save the playlist for playing the same tracks again. The playlist can be opened using the **View → Playlist** option and saved using the **Media → Save Playlist to File...** option. VLC supports multiple file formats for playlist, but M3U is more common.

At the bottom of the window, VLC displays a progress bar showing how much of the current track has been played and what is its total duration. By sliding the little slider on the bar, we may move back and forth in the current track. Below that, we have the play button (marked with a little right-pointing triangle). When the track is playing, this button turns into a pause button (with two parallel bars on it). If we pause the playback, the button reverts back to play mode. The middle button in the next three buttons is used to stop the playback completely, while the buttons with double arrows on the left and right allow us to jump to the previous and next track respectively. The next button acts as a toggle between watching a video in a window against watching it in full screen mode. That means, every time we click the button, it switches from one of these states to the other. In full screen mode, even the controls like play/pause, stop is hidden. They can be displayed in a temporary floating window by moving the mouse cursor over the video. The next button is for displaying the playlist. These options are available in the menu also.

While VLC has a lot of other functionalities as well, here we shall discuss how VLC can be used to convert multimedia files from one format to another. For this, select the **Media → Convert / Save** option from the menu. Using the Add button, select and add the file to be converted. Then from the list besides the Convert / Save button, select the Convert option. This will open another dialog box. Here, we need to provide the destination file (in which the converted track will be saved). We need to select appropriate file format (audio or video), from the drop down provided. Clicking on the Start button will now start the process and the progress will be shown. Sometimes, we may get error in the conversion process also.

Google Maps

Google Maps shown in figure 13.4 is a free Internet-based service provided by Google, Inc. Google has, over a period of years, collected extensive map data for the whole earth through various means like satellite imagery, cars with cameras mounted on them, data purchased from other organizations and data provided by millions of individual users around the world. Google Maps allows anybody to edit the maps and identify landmarks, buildings, etc. It also allows users to upload photographs of the place and post reviews of the place. The service is also heavily used on mobile devices like smartphones and tablets.

Figure 13.4 : The Google Maps Service

The service can be accessed from the web browser by opening the web site http://www.google.co.in and selecting the Maps option from the top menu, or directly entering the URL http://maps.google.co.in. On mobile phones, the service may be accessed from the mobile phone's web browser as well as from the Google Maps application. When accessed, the service first tries to know our current location. On the PC, a rough idea of the user's location can be obtained from the user's Internet connection. On a mobile phone, the user's location can be known with very high accuracy if the GPS (Global Positioning System) facility is used. This system finds a user's location with an estimated error of only few meters using satellite signals. Google maps also allows the user to correct the location, if it is inaccurate. Then it shows a map of the current location. We may pan (scroll) the map by dragging it in different directions or by clicking on the arrows in a circle on the left hand side. A vertical slider bar on the left allows us to zoom in and out on the map.

Google maps can show the information of a place in different forms. The default view is map view, shown in figure 13.4. By clicking on the Satellite button, we may switch to the satellite image view, shown in figure 13.5. In this view, the images taken by satellites in the sky are shown. The images are clear enough to identify most familiar buildings and roads. The service also provides the facility of getting directions on how to reach from place A to place B. The two places may be two different cities or places within the same city. The service also provides a choice of routes where available. Figure 13.6 shows an example. The service even provides turn-by-turn guidance while traveling if we use a mobile device equipped with GPS receiver. On a highway or in a crowded city area, walking or driving, the service decides the route based on the current location and destination and guides us with on-screen as well as voice (spoken) instructions.

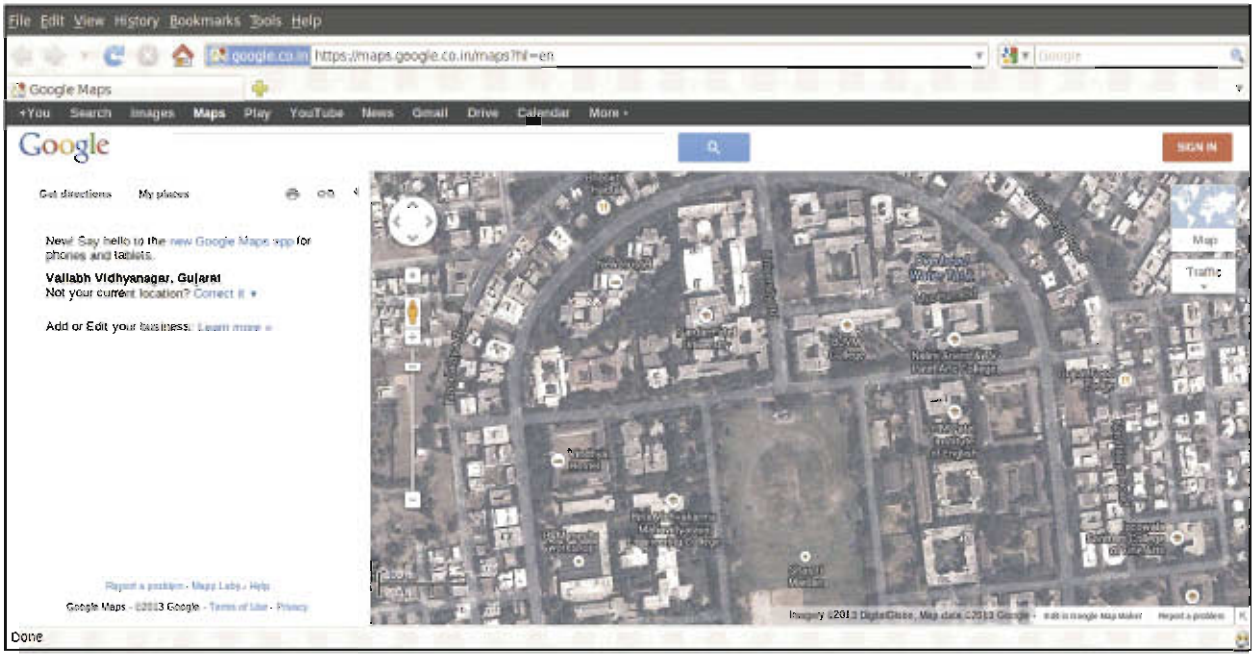


Figure 13.5 : The Google Maps Satellite Image View

There are several uses of the service. It can be used to search for a place and find its exact location. It can also be used to find directions to places unknown to us or in cities we are not familiar with. Government organizations and businesses use the service to provide people information on how to reach their offices. The service also has a facility using which anyone can display a map of any place on one's web site. Many organizations display such maps on their web sites. Tourist information web sites and Tourism departments of governments also make use of the service. It can also be used to display bus routes, current location of running trains, etc. The service also provides the facility of finding the nearest ATM centre, bank, restaurant, bus stop or any other such essential place.

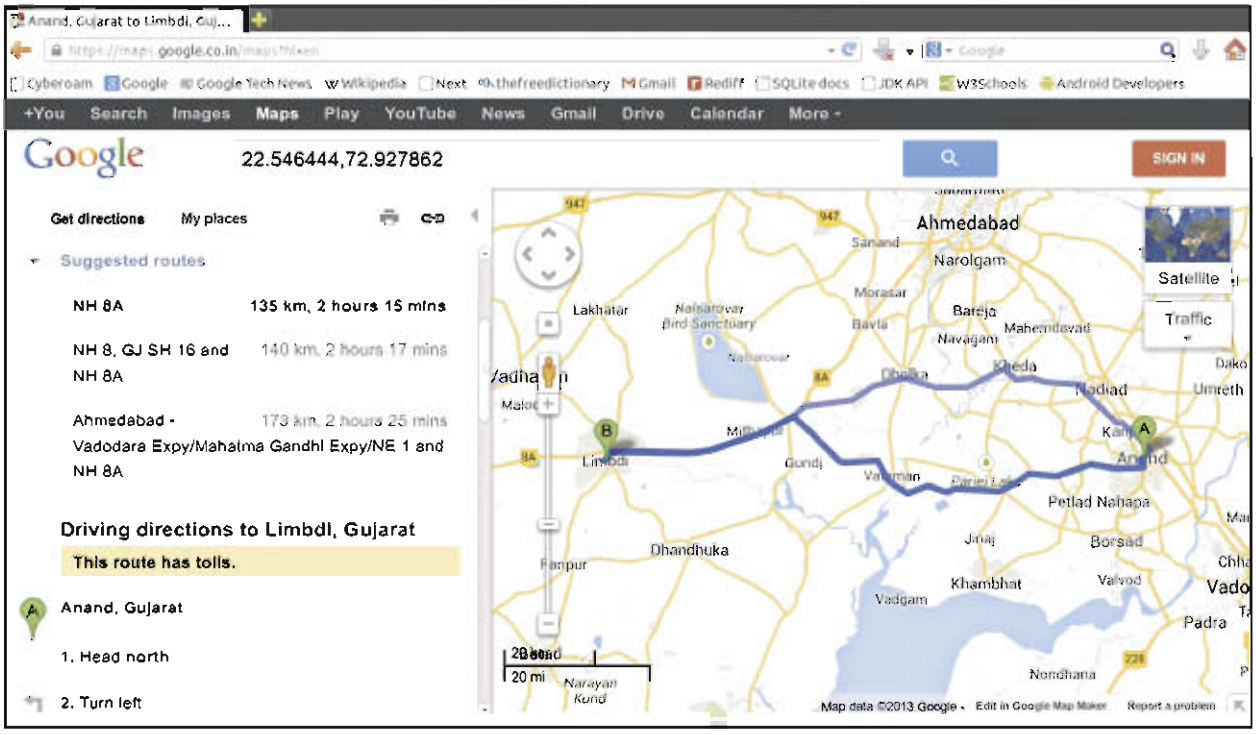


Figure 13.6 : Finding Directions using Google Maps

Character Map

The Character Map program shown in figure 13.7 can be used to enter Unicode characters into any application. We first select the script from the left pane and then double click characters to insert them into the Text to copy area below. Spaces and punctuation may be entered from the keyboard directly into the area after clicking in it. Combining characters, like the long "II" (??????) in Gujarati can also be entered and combine with the corresponding consonant appropriately. Brief details of the character currently selected with single-click or entered with double-click are shown in the status line at the bottom, while more details on the character is available in the Character Details tab. After we get a substantial amount of content, we may copy the content and paste into an application. If you only have to type a few characters in another script occasionally, then Character Map is a good solution for that.

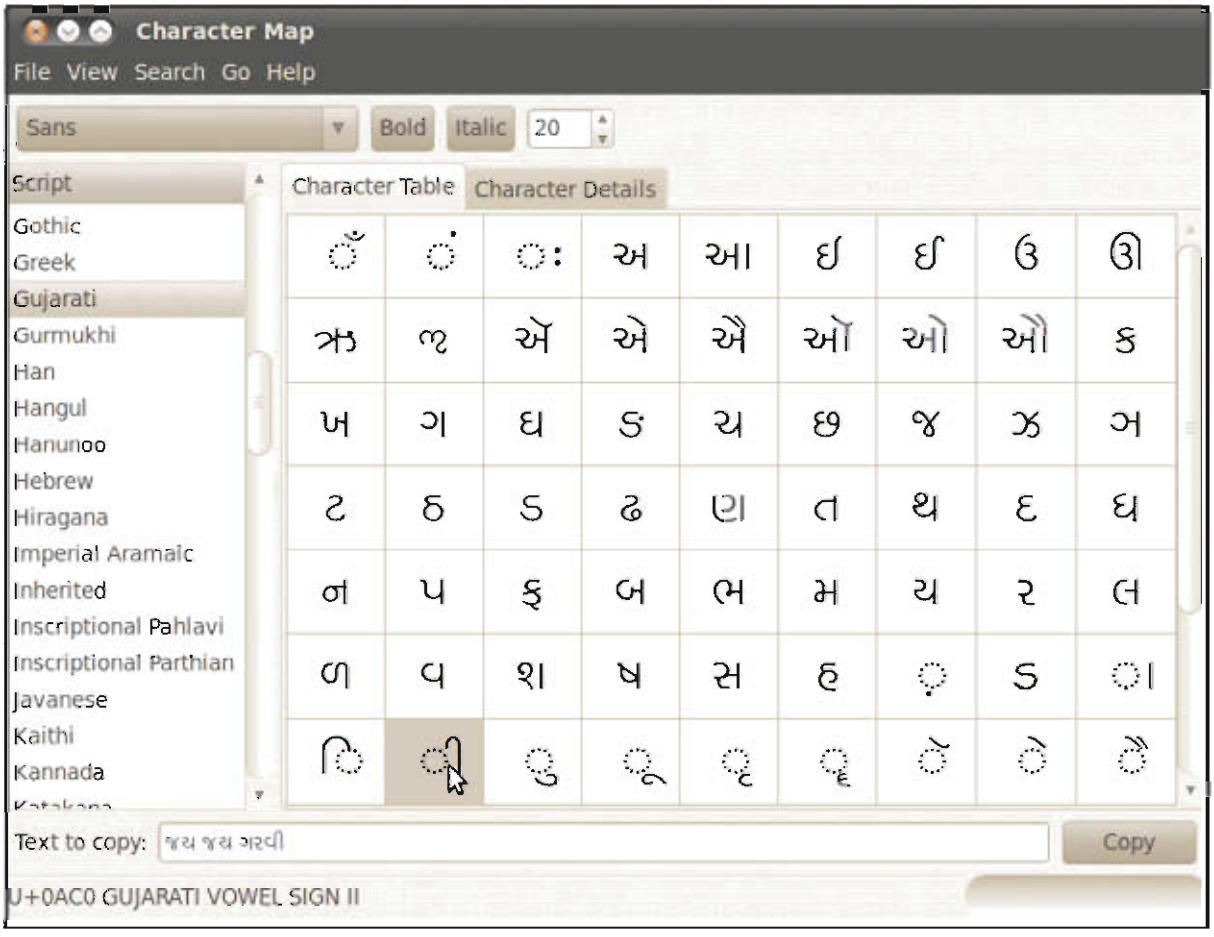


Figure 13.7 : The Character Map program

The R Software

R is a free software environment for statistical computing. It is a GNU project. It is widely used for statistical analysis. It has its own scripting language. It is a case-sensitive language. R has two work environments namely Command Line and Graphical. To use it in a GUI environment we need to install graphic editors like R Commander or RStudio from Ubuntu Software Center. The command to install R is given at the end of this chapter. To invoke the R scripts from terminal window, open terminal and type R on the command prompt. You will get a welcome message along with R prompt as shown in figure 13.8

```
harshal@HarshalAtUbuntu: ~
harshal@HarshalAtUbuntu:~$ R

R version 2.14.1 (2011-12-22)
Copyright (C) 2011 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: i686-pc-linux-gnu (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

Figure 13.8 : R Command prompt

R uses # as the comment marker, just like the Linux shell. Any text following #, up to the end of the line, is treated as a comment. Number and string are the basic data types. Strings may be enclosed in single quotes or double quotes. Common operators and functions are available, just like other programming languages. Ordered lists of items (arrays or lists) are commonly used and are also referred to as vectors. Lists are created using the c function to combine several numbers into a single list. Most operators like +, -, *, /, and others work equally well on single number as well as on lists. However, when both operands of binary operators are lists, their lengths (number of elements in them) should ordinarily be same. Entering the name of a variable and pressing ENTER displays its value. An example of this concept is shown in figure 13.9.

```
harshal@HarshalAtUbuntu: ~
harshal@HarshalAtUbuntu:~$ R

> a <- 10
> a
[1] 10
> b <- 20
> a*b
[1] 200
>
```

Figure 13.9 : Working with variables in R

Here the symbol > shows the prompt of R. The statement > a ← 10 defines a variable 'a' and assigns it a value 10. The statement > a followed by Enter key press displays the value of a. Similarly the statement > b ← 20 defines a variable 'b' and assigns it a value 20. The statement > a*b followed by Enter key press multiplies the value of variable 'a' and 'b' and display the result on the prompt.

Some common R commands include q() for quitting R, help() for accessing the on line help, demo() for viewing some demonstrations and help.start() to open the on line help in a browser. To get help on a particular function we need to use the syntax help(function name). When we quit R, we are asked whether we want to save the data (values of variables) from the current session. If we respond "yes" or "y", the data will be saved in a file in the current directory and will be available in the next session of R we start from the same directory. The function ls() displays a list of all the variables we have defined as shown in figure 13.10.

```
harshal@HarshalAtUbuntu: ~
harshal@HarshalAtUbuntu:~$ R

> a <- 10
> a
[1] 10
> b <- 20
> a*b
[1] 200
> ls()
[1] "a" "b"
>
```

Figure 13.10 : Use of ls() in R

A series of consecutive numbers (range) may be generated using the syntax start:end. For example, 1:5 is same as c(1, 2, 3, 4, 5).

R also provides basic statistical operations quite straightforward forms. For example we may used the functions min(list), max(list), mean(list) and median(list) to find the minimum value, maximum value, mean and median for the list as shown in figure 13.11.

```
harshal@HarshalAtUbuntu: ~
harshal@HarshalAtUbuntu:~$ R

> ll <- c(3,6,8,3,4,6,9,4,7,3,9)
> min(ll)
[1] 3
> max(ll)
[1] 9
> mean(ll)
[1] 5.636364
> median(ll)
[1] 6
>
```

Figure 13.11 : Statistical functions in R

Plotting in R

Assume that we want to plot a bar chart that represents faculty wise students in a university. We may define the number of students as a list and the names of faculties as another list. Then we may use the function barplot with various arguments for drawing the plot as shown in figure 13.12.

```
harshal@HarshalAtUbuntu: ~
harshal@HarshalAtUbuntu:~$ R
> no_students <- c(140,210,700,950,810,320)
> faculty_names <- c("Medical","Engineering","Science","Commerce","Arts","Law")
> barplot(no_students,main="Faculty wise students",xlab="faculty",
+ ylab="Number of students",names.arg=faculty_names,
+ ylim=c(0,1000),border="blue")
>
```

Figure 13.12 : Plotting graph in R

The first argument specifies the data list, while all other arguments have the form name=value. We use the arguments main, xlab, ylab, names.arg, ylim and border to specify the main title, X-axis label, Y-axis label, the values to be displayed for the bars, the range of values to be plotted on the Y-axis and border color respectively. Observe that for long commands, when we press ENTER before the command is complete, R issues the + prompt until we complete the command. By default the graph does not have grid lines as can be seen in figure 13.13.

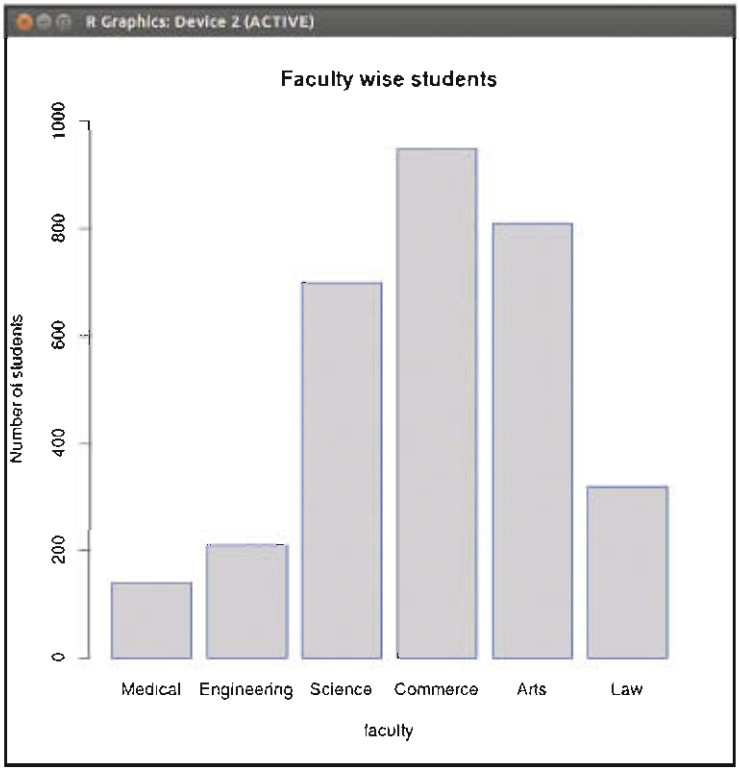


Figure 13.13 : A Bar Chart Produced in R

Producing histograms is also not difficult. Suppose we have marks out of 100 in a particular subject for several students. The histogram can be drawn using the code shown in figure 13.14 and will look like the plot shown in the same figure.

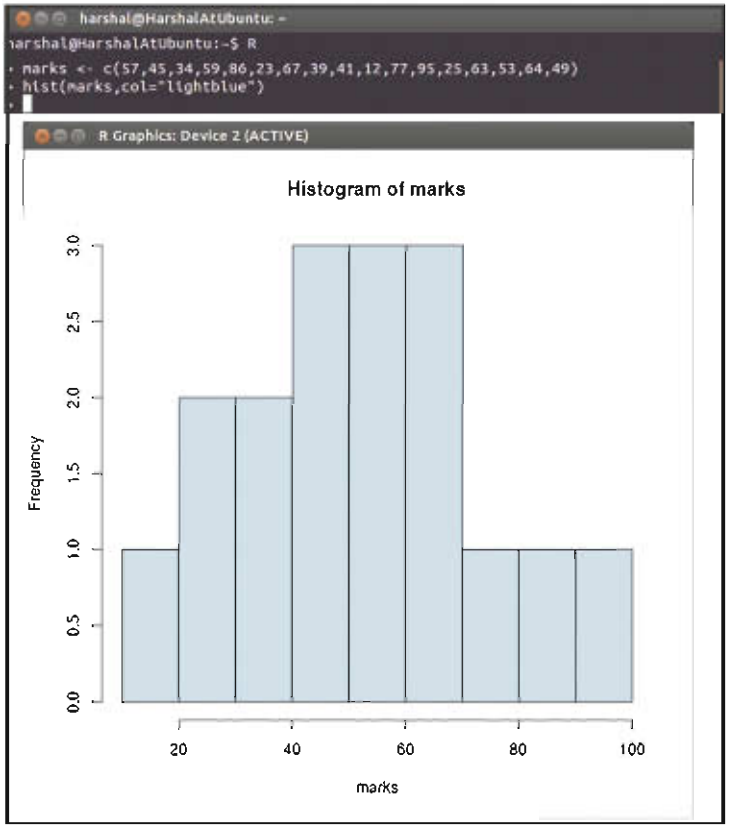


Figure 13.14 : A Histogram Produced in R

RationalPlan

People working in different areas like construction, engineering, services and consulting, business or software development usually work on a project that is having constraints of time, resource and budget. Any delay in completion of the project increases the cost of the project. At some point of time to manage this kind of project we need to prepare, maintain and follow a well defined plan. Such a plan helps us to complete the project as scheduled, on time and within budget.

RationalPlan is one such open source software designed to assist project managers in preparing, maintaining and following a well defined project plan. It assists the project managers throughout the life cycle of their projects. It comprises three different desktop products namely RationalPlan Single, Multi and Viewer.

RationalPlan Single allows us to manage independent projects that do not have common resources and has no interrelations between different projects. It allows the manager to associate general project information like name, notes, links, assumptions, constraints or risks. It allows creating, editing and deleting calendars, building schedules. Once the resources are created we can assign tasks to them. It provides project tracking tools such as critical path, mark completion value for tasks, work and cost time phased information. We can generate a printable report as well as import data from other project management tool or export our data into other formats.

RationalPlan Multi allows us to manage projects that share company resources across projects. It also manages interdependencies between projects. It includes all the features of RationalPlan Single Project. Further it calculates the resources data (work, cost, over allocation) considering their assignments in all projects. It allows links between tasks pertaining to different projects, analyzes all our projects data and creates a project Portfolio view.

RationalPlan Viewer is an additional tool developed to share the project in its original file format (.xrp), but it can be used as well to open Microsoft Project files. It is useful to anybody who needs to check and overview the project's evolution without making changes to the schedule, and for resources to see their assignments.

Though as of today the support of RationPlan on Unbuntu 10.04 has been removed it is available for Ubuntu 12.04 and later versions.

Skype

You must have used various instant messaging services like Yahoo Messenger, Google Talk or Rediff Bol. We use them for real time chatting with text, audio and video facilities. Skype is one such software that allows us to make calls over the Internet using our computer.

The Skype service allows users to communicate with peers using voice, video as well as text. We can use Skype to place phone call to recipients on the traditional telephone network also. The calls to landline telephones or mobile phones are charged via a debit based user account system while calls to other users within the Skype service are free of charge. It also provides additional features like file transfer and videoconferencing.

The Skype software can be downloaded free of cost, though its source code is proprietary and not available for modification. To use Skype we need a working sound input and output configuration. Most

modern computers today support both these features. In laptops they are inbuilt while on desktop we may use extensions like a headset, speaker and a web cam. We can install Skype from Software Center.

To start Skype, choose **Applications → Internet → Skype.** If you are opening it for the first time it will open an End User License Agreement window. Read the contents and click on "I agree" button. We will now see a window as shown in figure 13.15.

Figure 13.15 : Skype initial window

To use the Skype service you will need a Skype user account. In case you don't have one, then click on the link with contents "Don't have a Skype Name yet?", follow procedure to create a Skype name. Once you have a Skype name and password you can enter the details in the textbox of the window shown in figure 13.15 and click on the "Sign in" button. If everything goes well you can now start interacting with other Skype users or make phone calls using Skype.

Summary

In this chapter we discussed some useful free tools and services. We learnt about data compression techniques that allow us to reduce the amount of memory and disk space required to store the information. We discussed the Archive Manager tool that lets us apply these techniques to files and entire directories. We also introduced VLC media player, a feature-rich multimedia tool. We discussed the Google Maps service, which is an online service providing, maps of different areas. We learnt how to use Character Map to insert various symbols and characters from different languages in our text. We had an overview of the R environment for statistical computing. We learnt some basic statistical operations like calculating mean and median in R and learnt to draw bar graphs and histograms in R. Finally we saw the use of RationalPlan and Skype.

Instruction for Teachers

The tools covered in this chapter are only for demonstration purpose. The teachers are requested to show demo of these tools to the students. The examples given in laboratory exercise section can be used by teachers for demonstration. Teacher needs to install R software on the machines. To install R use the command given below :

sudo apt-get install r-base r-base-dev

EXERCISE

1. Write a short note on data compression.
2. Why do we need data compression tools ?
3. List main features of Archive Manager.
4. Write the main features of the VLC media player.
5. List the applications of Google Maps.
6. Explain the use of the Character Map program.
7. Choose the most appropriate option from those given below :

(1) Which of the following refers to a file that has an entire directory structure inside it ?

(a) apache (b) archie (c) archi (d) archive

(2) What is the full form of tar ?

(a) tape archiver (b) tech archiver (c) test archiver (d) tight archiver

(3) For which types of archives is the password protection option available ?

(a) zip (b) tar (c) tar.gz (d) both zip and tar.gz

(4) Which of the following is a feature-rich media player ?

(a) VAC (b) VEC (c) VLC (d) VNC

(5) What is the full form of VLC ?

(a) VideoLAN Client (b) Video Line Coder

(c) Video Length Coder (d) Video List Creater

(6) Which technology gives our location with accuracy ?

(a) GRS (b) GPRS (c) GRPS (d) GPS

(7) Which program is used to enter Unicode characters into any application ?

(a) Character Display (b) Character Insert

(c) Character Map (d) Character Select

(8) Which command is used to quit from R ?

(a) quit() (b) q() (c) exit() (d) close()

(9) Which function is used to create a bar graph in R ?

(a) bar() (b) plot() (c) bargraph() (d) barplot()

(10) Which of the following are different variants of RationalPlan ?

(a) Single, Multi, Viewer (b) Singular, Multiple

(c) View, Preview (d) Server, Client

LABORATORY EXERCISE

1. Create a zip file and store an entire directory in it.
2. Create a tar.gz file and store an entire directory in it.
3. Extract all files from the zip archive above into a subdirectory files1.
4. Extract all files from the tar.gz archive above into a subdirectory files2.
5. Locate your school on Google Maps. Can you identify the building in the satellite image?
6. Locate your home in Google Maps. Obtain directions for reaching your school from your home. Is it the same path you take?
7. Type सत्यमेवजयते in gedit using Character Map.
8. Type $\sin(\alpha - \beta) = \sin \alpha \cos \beta - \cos \alpha \sin \beta$ in gedit with help from Character Map.
9. Plot a bar graph of the number of days in each month of the (non-leap) year.
10. Plot a bar graph of the number of pages in your textbooks.
11. Plot a histogram of the frequency of letters in the names of days of the week (sunday, monday, …saturday). Only include the letters that occur at least once.
12. Create your own Skype user account and try to explore the facilities provided by Skype service.

ગુજરાત રાજ્યના શિક્ષણવિભાગના પત્ર-ક્રમાંક
મશબ/1214/15/છ, તા. 29-1-2014થી–મંજૂર

COMPUTER STUDIES

Standard 12

PLEDGE

India is my country.
All Indians are my brothers and sisters.
I love my country and I am proud of its rich and varied heritage.
I shall always strive to be worthy of it.
I shall respect my parents, teachers and all my elders and treat everyone with courtesy.
I pledge my devotion to my country and its people.
My happiness lies in their well-being and prosperity.

Price : ₹ 105.00

Gujarat State Board of School Textbooks
'Vidyayan', Sector 10-A, Gandhinagar-382010

© Gujarat State Board of School Textbooks, Gandhinagar
Copyright of this book is reserved by Gujarat State Board of School Textbooks.
No reproduction of this book in whole or in part, in any form is permitted without written permission of the Director, Gujarat State Board of School Textbooks.

Subject Adviser

Prof. R. P. Soni

Authors

Dr Harshal Arolkar (Convener)
Shri Jigneshbhai Smart
Shri Jyotikaben Doshi
Shri Triptiben Dodiya
Shri Hardikbhai Joshi

Reviewers

Shri Girish S. Brahmbhatt
Shri Saket A. Dave
Shri Bimal K. Raval
Shri Rajanikant A. Pandya
Shri Ramaniklal L. Gilatar
Shri Ankitaben S. Dwivedi
Shri Pankaj R. Shukla
Shri Mayuriben I. Shah
Shri Rajshreeben N. Padia
Shri Nishitaben N. Gandhi
Shri Sejalben D. Trivedi

Co-ordinator

Shri Ashish H. Borisagar
(Subject Co-ordinator : Mathematics)

Preparation and Planning

Shri C. D. Pandya
(Dy. Director : Academic)

Lay-out and Planning

Shri Haresh S. Limbachiya
(Dy. Director : Production)

PREFACE

The Gujarat State Secondary and Higher Secondary Education Board has prepared new syllabi based on the open source operating system and compitible open source software tools for various topics of Computer Studies. These syllabi are sanctioned by the Government of Gujarat.

It is a matter of pleasure for the Gujarat State Board of School Textbooks to place this textbook of **Computer Studies** prepared according to the new syllabus before the students of **Standard 12**.

Before publishing the textbook, its manuscript has been fully reviewed by experts and teachers teaching at this level. Carrying out suggestions given by teachers and experts, we have made necessary changes in the manuscript and then have published the textbook.

The board has taken special care to ensure that this textbook is interesting, useful and free from errors. However, we welcome suggestions to enhance the quality of the textbook.

Dr Bharat Pandit	**Dr Nitin Pethani**
Director	Executive President
Date : 1-2-2014	Gandhinagar

First Edition : 2014
Published by : Bharat Pandit, Director, on behalf of Gujarat State Board of School Textbooks, 'Vidyayan', Sector 10-A, Gandhinagar
Printed by :

FUNDAMENTAL DUTIES

It shall be the duty of every citizen of India :

(a) to abide by the Constitution and respect its ideals and institutions, the National Flag and the National Anthem;

(b) to cherish and follow the noble ideals which inspired our national struggle for freedom;

(c) to uphold and protect the sovereignty, unity and integrity of India;

(d) to defend the country and render national service when called upon to do so;

(e) to promote harmony and the spirit of common brotherhood amongst all the people of India transcending religious, linguistic and regional or sectional diversities; to renounce practices derogatory to the dignity of women;

(f) to value and preserve the rich heritage or our composite culture;

(g) to protect and improve the natural environment including forests, lakes, rivers and wild life, and to have compassion for living creatures;

(h) to develop the scientific temper, humanism and the spirit of inquiry and reform;

(i) to safeguard public property and to abjure violence;

(j) to strive towards excellence in all spheres of individual and collective activity so that the nation constantly rises to higher levels of endeavour and achievement.

(k) who is a parent or guardian to provide opportunities for education to his child or, as the case may be, ward between the age of six and fourteen years.

CONTENTS

1. Creating HTML forms using KompoZer 1
2. Cascading Style Sheets and Java script 25
3. Designing simple website using KompoZer 52
4. Introduction to E-Commerce 72
5. Introduction to M-Commerce 90
6. Object-Oriented concepts 116
7. Java Basics 127
8. Classes and objects in Java 158
9. Working with Array and String 186
10. Exception handling in Java 203
11. File handling 220
12. Publishing documents using LaTeX 241
13. Other useful free tools and services 260

About This Textbook...

Dear Teachers,

With a mission to spread computer literacy on a fast track, the Gujarat Government has provided latest computer equipments to more than 6000 aided schools under the ICT@School program. As a new policy initiative, all the schools are given the Ubuntu (a variant of Linux) Operating System and other Open Source software packages so that schools can freely use and exchange the software without bothering about the licensing issues. Since earlier text books were largely based on proprietary software, there was a need to rewrite the textbooks based on new syllabus. This was also necessary in view of the fact that the 8th standard has been transferred to primary section. Therefore, new content has been provided for 9 to 12 standard in a phased manner based on the open source Operating System and compatible open source software tools for various topics of computer studies.

Students are now quite familiar with computer software fundamentals such as Operating System, Open Office components, HTML, C Programming, Internet and Web surfing etc. This textbook for 12th standard is the fourth in series for the subject of 'Computer Studies'. It aims at introducing creation of HTML forms, Writing Javascripts, Using Kompozer for designing simple websites, and E-Commerce as well as M-Commerce. Moreover, after introducing new paradigm of Object Oriented Concepts, it describes basic JAVA language which is gaining popularity since it is the language extensively used in web applications. At the end, some useful and popular open source packages like LaTeX, and other miscellaneous tools are introduced for the knowledge of students.

We hope, this coverage will be useful to the students of 12th standard and you will enjoy teaching and conducting practicals on these topics using open source Ubuntu operating system environment and related tools.

Dear Students,

So far in standards 9 to 11, you have learnt Operating System Ubuntu, Open Office components such as Writer, Calc, Impress; and database management tool Base as well as web page presentation in HTML. You also learnt algorithm development and basic C Programming which is considered as procedure oriented programming language. In addition you were introduced to multimedia tools for animation.

This text of 12th standard presents Web Page designing using Kompozer, E-Commerce, fundamentals of Object Oriented Concepts, JAVA programming and other useful tools.

Chapters 1 to 3 lead you to try your hand at creating simple yet working websites. For this, HTML forms, Stylesheets, JavaScripts and Kompozer tool for creation of websites are discussed in these chapters. E-Commerce, M-Commerce and L-Commerce are introduced in chapters 4 to 5. Nowadays most of the software development is done using Object Oriented Methodology. Therefore, you will learn Object Oriented Concepts in chapter 6. The prominent and current web application development language Java which is also object oriented language is introduced in chapter 7. The features of Java such as Classes and Objects, Arrays and Strings, Exception handling and File handling are covered in chapters 8 to 11. There is another open source word processing package very popular amongst book authors, writers, publishers and scientists known as LaTeX; and it is introduced in chapter 12. In the last chapter, some useful open source tools i.e., Archive Manager, VLC Media Player, Google Maps, statistical tool R, Skype and Rational Plan are briefly discussed just to apprise you about their usefulness in normal working and doing communication with computers.

It is expected that if you carefully study the text and practice the laboratory exercises, you will develop reasonable confidence in creating simple websites and writing simple Java programs.

Creating HTML forms using KompoZer 1

With increase in the use of Internet many activities have become online. We may use a web page to fill information about ourselves or a product. HTML forms are used to help the visitors of the website to input data. It allows for more interactivity and control in data entry. For instance, if you want to open a mail account or register on a website, you need to enter your personal details in a form. This information is used to setup the account for the user. In order to obtain such information on Internet, HTML forms are used. This data is further stored by the application and used to retrieve the details about the users registered on the website.

A form in HTML is a container used to collect different kinds of inputs from the user. HTML forms contains elements like label, checkbox, text input field, radio button, submit button, reset button and many more. These elements are used to enter the data as well as validate the data within the forms. We will create a simple form to enter the data using HTML tags, but before that let us discuss about the elements used to create HTML forms. The elements used are described in the section below :

- Form
- Input
- Textarea
- Select and Option

Form elements

The form element is used to create an HTML form. It acts as a container for all the elements used in the form. The tag <form>… </form> is used to implement this element. The example shows the use of the form element :

```
<form action="register.html" method="post">
.
.
input elements
.
.
</form>
```

Observe that the form element uses two attributes namely action and method. The action attribute is used to specify where to send the form data when the form is submitted. It takes a filename as value. This file is opened when the user clicks on the submit button after filling the data in the form.

The method attribute specifies the HTTP method to be used when sending the data. It can take

two values; GET and POST. The GET method retrieves the data from the form and sends it to the server by attaching it at the end of the URL. This method allows only a limited amount of information to be sent at a time. In the POST method, the data is sent as a block through the HTTP transaction. The data is included in the body of the request. This method does not have any restrictions on data length. The default value for method attribute is GET.

Input element

The input elements are used to insert various fields like radio button, text box and checkbox in the form. The tag <input> ….</input> or <input> is used to implement this element. The input tag has different attributes like type, name and value.

The type attribute of the input element specifies the field that is to be created in the form. The name attribute specifies the name to be used for the field in the form. The value attribute specifies the default value of the field in the form. Table 1.1 shows different values of attribute and its usage.

Type	Description	Example
Radio	Creates radio buttons in the form. Any one radio button can be selected at a time from a group of radio buttons. Generally used to select a single item from a given group of items.	<INPUT TYPE = "radio" NAME = "var" VALUE = "txt">
Checkbox	Creates checkboxes in the form. Multiple check boxes can be selected at a time. Generally used to select a multiple items from a given group of items.	<INPUT TYPE="checkbox" NAME = "var" VALUE ="txt" >
Text	Creates a text field to enter text in the form. A user can enter any data of his choice in the text field.	<INPUT TYPE = "text" NAME = "var" VALUE = "txt" >
Password	Creates a password field in the form. Similar to the text field but the characters are not displayed to the user. Instead the character typed is converted into non readable format.	<INPUT TYPE = "password" NAME = "var" >
Submit	Creates a submit button in the form. On clicking the submit button, the values of data entered in the form is submitted to the file specified in the action attribute of the form element.	<INPUT TYPE = "submit" VALUE = "label" >
Reset	Creates a reset button in the form. On clicking the reset button, the values of data entered in the form are cleared and set back to default values.	<INPUT TYPE = "reset" VALUE = "label" >

Table 1.1: Values of Type attribute used with input tag

Textarea element

The Textarea element allows multi-line text input. The tag <textarea>…</textarea> is used to implement this element. It allows entering unlimited number of characters. It can be used to enter comment, report or a long description of product. The size of a textarea element can be specified using rows and cols attributes. The rows attribute is used to set the number of rows of text that will be visible without scrolling up or down. The cols attribute is used to set the number of columns of text that will be visible without scrolling right or left. The following example shows how to insert a textarea in the form.

```
<form method="post" action="comment.html">
Input your comments: <br /> <textarea name="comments" rows="4" cols="20">
...Your comments here...
</textarea>
</form>
```

Select and Option element

The select element is used to create a drop down list or menu in a form. The option element is used to specify the values that are to be displayed in the menu. The tag <select>….</select> is used to create a drop down menu. The tag <option>…</option> is used to create the elements within the menu. Following example shows the use of select and option element.

```
<select>
<option value="Ahmedabad" >Ahmedabad</option>
<option value="Rajkot" >Rajkot</option>
<option value="Surat" >Surat </option>
</select>
```

Let us now create a sample registration form using the elements learned so far. Code listing 1.1 shows the HTML source code used to generate the form. The output of the code is shown in figure 1.1.

```
<HEAD>
        <TITLE>Registration Form</TITLE>
</HEAD>
<BODY bgcolor="lightblue">
<h1> <center>Registration Form</center></h1>
<FORM name="frmRegistration" action="form.html">
        <center>
        <TABLE BORDER="0">
<TR>
        <TD width="12%">First Name</TD>
        <TD width="1%"> </TD>
```

```
        <TD> <INPUT type="textbox" name="txtFirstName"></TD>
</TR>
<TR>
        <TD>Middle Name</TD>
        <TD> </TD>
        <TD><INPUT type="text box" name="txtMiddleName"></TD>
</TR>
<TR>
        <TD>Last Name</TD>
        <TD> </TD>
        <TD> <INPUT type="text box" name="txtLastName"></TD>
</TR>
<TR>
        <TD>Gender</TD>
        <TD> </TD>
        <TD>
        <INPUT type="radio" name="Gender" value="male" CHECKED>Male
        <INPUT type="radio" name="Gender" value="female" >Female
        </TD>
</TR>
<TR>
        <TD>Hobby</TD>
        <TD> </TD>
        <TD>
        <INPUT type="checkbox" name="chkSinging" value="Sing" CHECKED>Singing
        <INPUT type="checkbox" name="chkDancing" value="Dance">Dancing
        <INPUT type="checkbox" name="chkReading" value="Read">Reading
        </TD>
</TR>
<TR>
        <TD>Address</TD>
        <TD> </TD>
        <TD>
        <Textarea name="txtAddress" rows="5" cols="70">Insert Address Here</Textarea>
        </TD>
</TR>
<TR>
        <TD>City</TD>
        <TD> </TD>
        <TD>
```

```
            <Select Name="cmbCity">
            <Option >Ahmedabad</Option>
            <Option >Baroda</Option>
            <Option selected>Rajkot</Option>
            <Option >Surat</Option>
            </Select>
            </TD>
      </TR>
      <TR>
            <TD> </TD>
            <TD> </TD>
            <TD> <INPUT type="submit" name="cmdSubmit" value="Submit">
            <INPUT type="reset" name="cmdReset" value="Reset">
            </TD>
      </TR>
      </TABLE>
      </center>
</FORM>
</BODY>
</HTML>
```

Code listing 1.1: HTML code to generate sample registration form

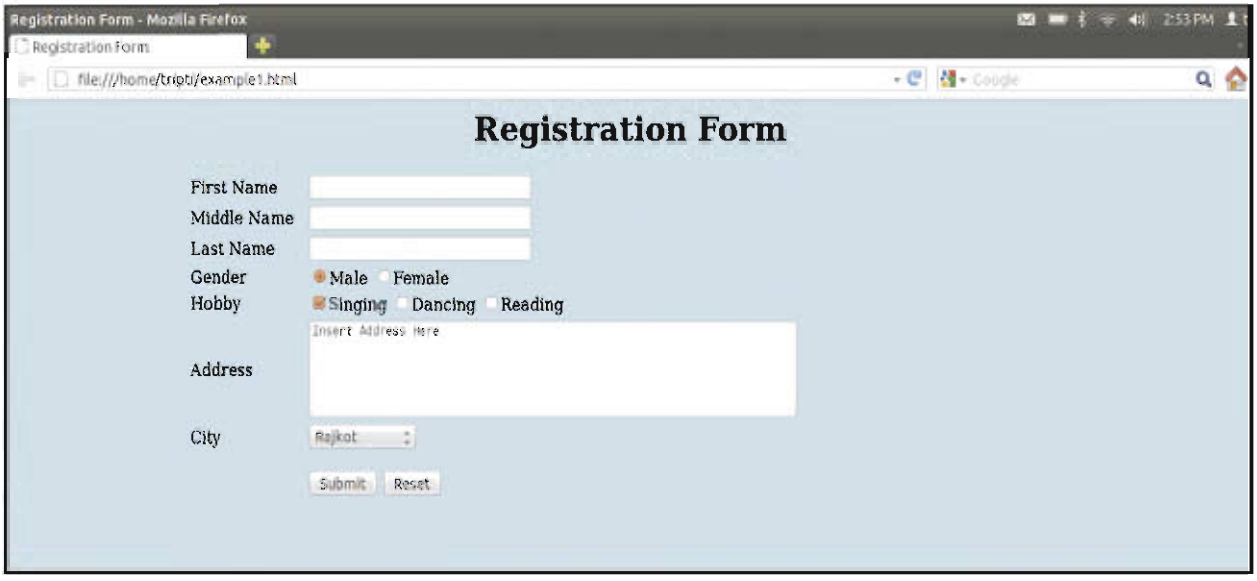


Figure 1.1: Registration Form as displayed in the web browser

Thus, you can see that creating a form using HTML tags is a tedious process. A simpler method is to use an IDE (Integrated Development Environment). An IDE is a software application that provides complete facilities to programmer to develop software. It provides a GUI (Graphical User Interface), text or code editor, a compiler and/or interpreter and a debugger. KompoZer, Eclipse, JBuilder and Netbeans are all examples of some open source IDEs. Let us discuss how to use the KompoZer to create web pages.

Introduction to KompoZer

KompoZer is a free open source web development IDE. It can be downloaded from http://www.KompoZer.net. It provides a web page editor which has a simple graphical interface known as WYSIWYG "what you see is what you get". It is a complete web authoring system which integrates web page development and web file management. Creating new web pages using KompoZer is quick and easy. The users can also edit the web pages by using the source code and making changes. KompoZer incorporates a Site Manager which gives rapid access to the files on both local machines and remote servers. Web pages and associated files can be uploaded to a remote server from within KompoZer. It also supports the use of "Styles" through Cascading Style Sheet (CSS). We will learn about CSS in the next chapter.

Before we learn how to create forms using KompoZer, let us first learn the KompoZer interface. Open KompoZer by locating its icon. To view the different toolbars and status bar (if not visible) click on **View → Show/Hide.** Select all the options listed: Composition Toolbar, Format Toolbar1, Format Toolbar2, Edit Mode Toolbar and Status bar. Site Manager and Rulers option should also be checked. Figure 1.2 shows the window that will be displayed after selecting the toolbars.

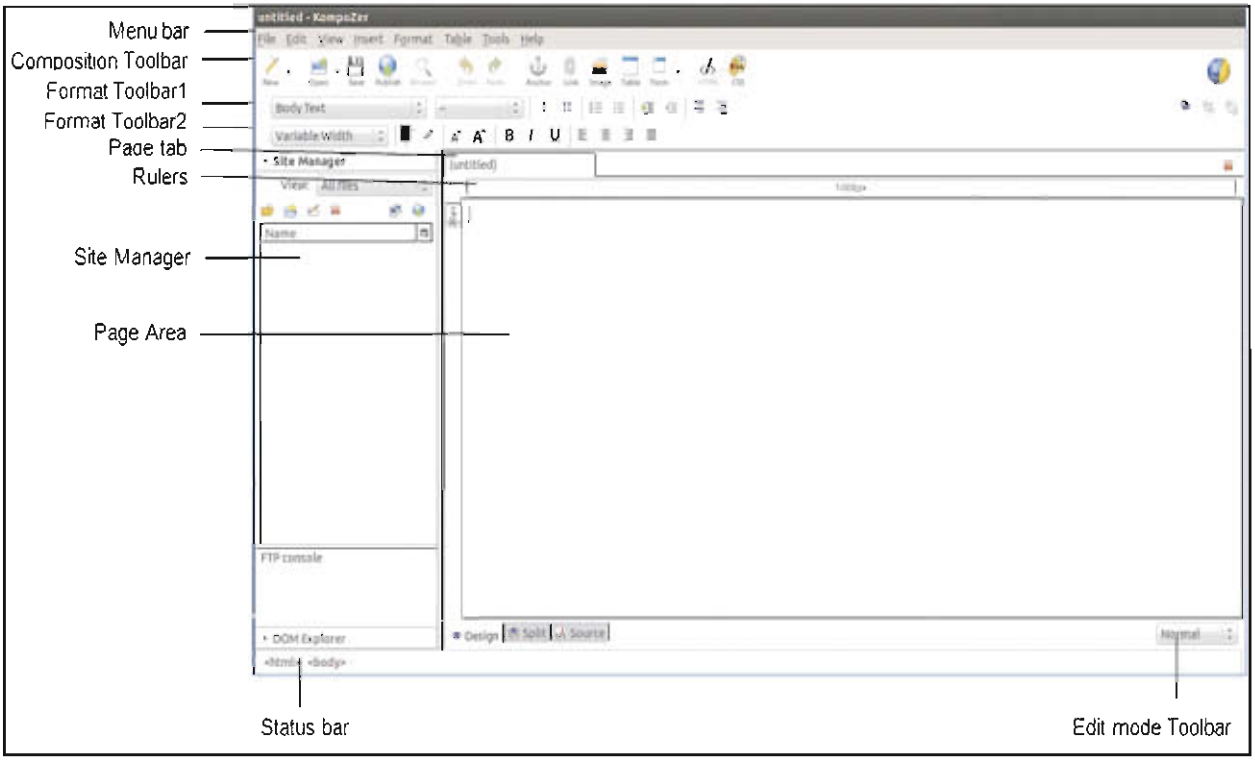


Figure 1.2: KompoZer Interface

Observe that in figure 1.2 you can see the menu bar on the top with the options like File, Edit, View, Insert, Format, Table, Tools and Help. Below the menu bar there are three Toolbars: Composition, Format Toolbar1 and Format Toolbar2.

The composition toolbar is used to create new file, open a file, save a file or publish a web page. The Format toolbar1 and Format toolbar2 are used to format the text, add bullets, numbering and perform similar formatting operations.

In the centre of the window, you can see two panes: Site Manager and blank web page. Site manager is a powerful tool used to navigate within the site or between the sites. You can close the site manager pane by clicking on close button or press F9. The page pane shows a blank untitled web page. The bottom right side of the window shows Edit mode toolbar with three viewing modes: Normal, HTML Tags and Preview. All the three viewing modes provide editing facilities.

The Preview mode offers the page view as seen in a browser. The difference is that in the preview mode the scripts do not run and therefore their effects will not be seen. The links also does not operate in preview mode.

The Normal view is very similar to preview mode. In this mode the table outlines are visible.

The HTML Tags view helps those who are familiar with HTML. A yellow marker is used to indicate the start tag for all elements. Clicking on a marker selects and highlights whole of the element.

The left side of the page pane shows Design, Split and Source tabs. The Design tab is used to design the web page. The Split tab displays the HTML source of the current element. Source tab shows all details of the HTML code. It helps in editing the source code.

On the bottom of the window you can see the status bar. When we click on any item in page, its structure appears in the status bar. If you want to customize any toolbar, right click on the respective toolbar and click on customize toolbar option. You can then customize the toolbar as per your choice.

Create a New File

To create a new file, open KompoZer. In the menu bar click **File → New**. A dialog box as shown in figure 1.3 appears with title "Create a new document or template". Select "Blank document" option from the options that are visible in the dialog box. At the bottom of the dialog box you can see a label "Create in". Select New Tab option from the drop down menu next to it. This allows us to create the page in a new tab. Click on the Create button.

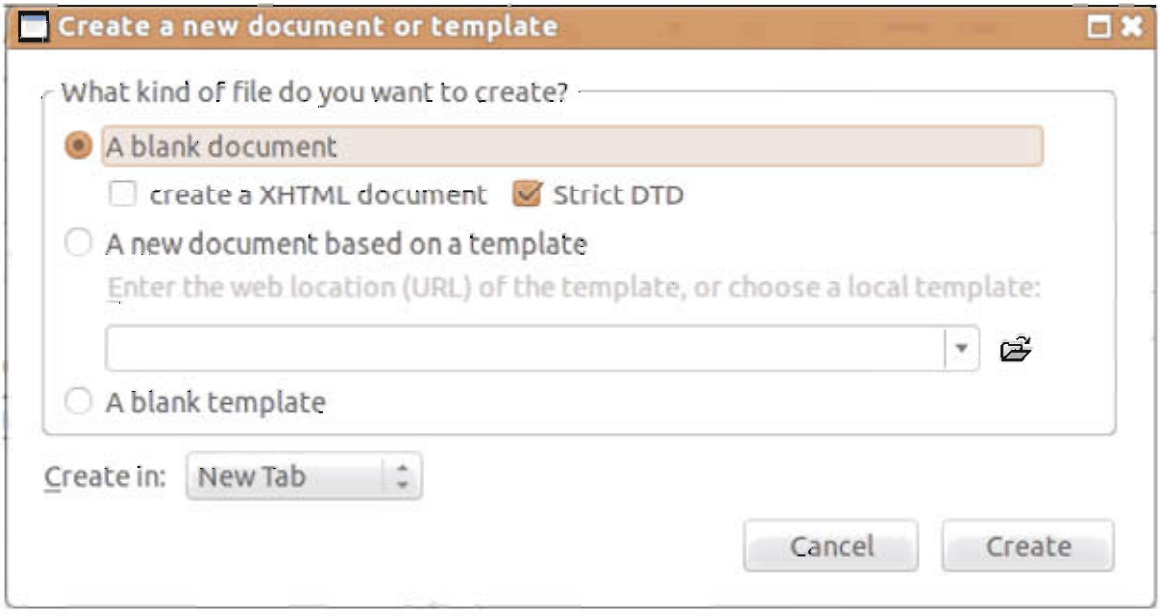


Figure 1.3 : Create a new file

Open an existing File

To open an existing file, click on

icon present on the composition toolbar. Alternatively, you can also click on **File → Open.** If the file has been opened recently, then you can also open the file from **File → Recent Pages.**

Now let us learn to create a forms using KompoZer. We will create a simple form with two input fields: Name and E-mail address and a submit button. Follow the steps given to create the form.

- Open KompoZer. Create a new file.
- From the menu bar, select **Insert → Form → Define Form.** Alternatively, you can also click on

in the composition toolbar. This will open a Form Properties dialog box as shown in figure 1.4. Clicking on More Properties shows added options for the form.

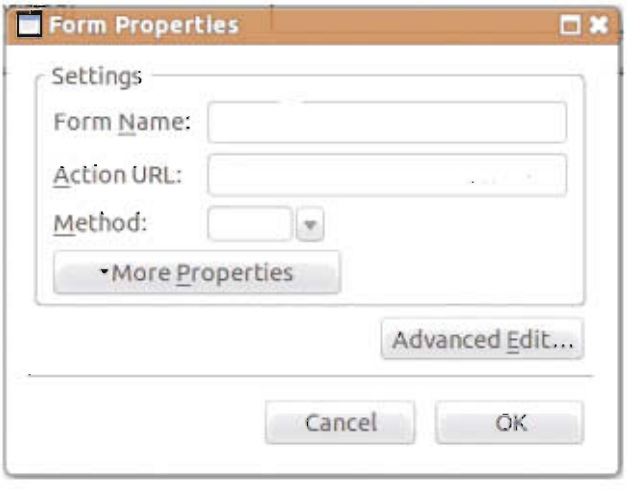


Figure 1.4 : Form properties dialog box

- Enter appropriate name for the form. In the Action URL, enter the file name where you want the form data to be submitted. Select the method POST from the method drop down menu, and click on the OK button. Figure 1.5 shows the details added in the form property dialog box.

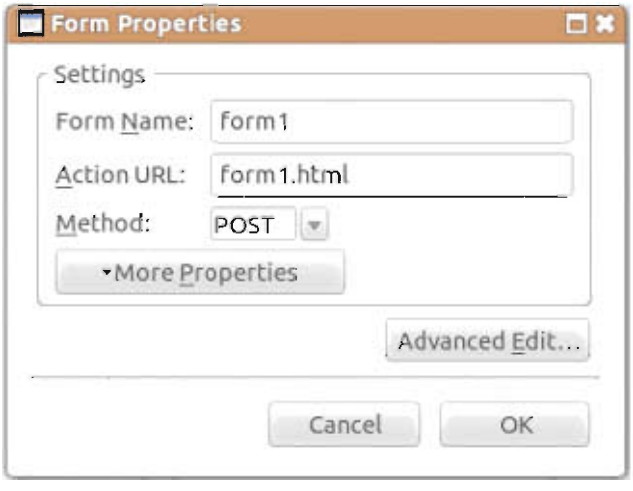


Figure 1.5 : Details entered in Form properties dialog box

- You can see the form inserted with light blue colored outline in the untitled page as shown in figure 1.6. In normal view, the forms are shown surrounded by a dotted blue box. All the form elements like text box, radio button, check box and drop down box will be placed within this box. Press the Enter key a few times to give some space to work on the form.

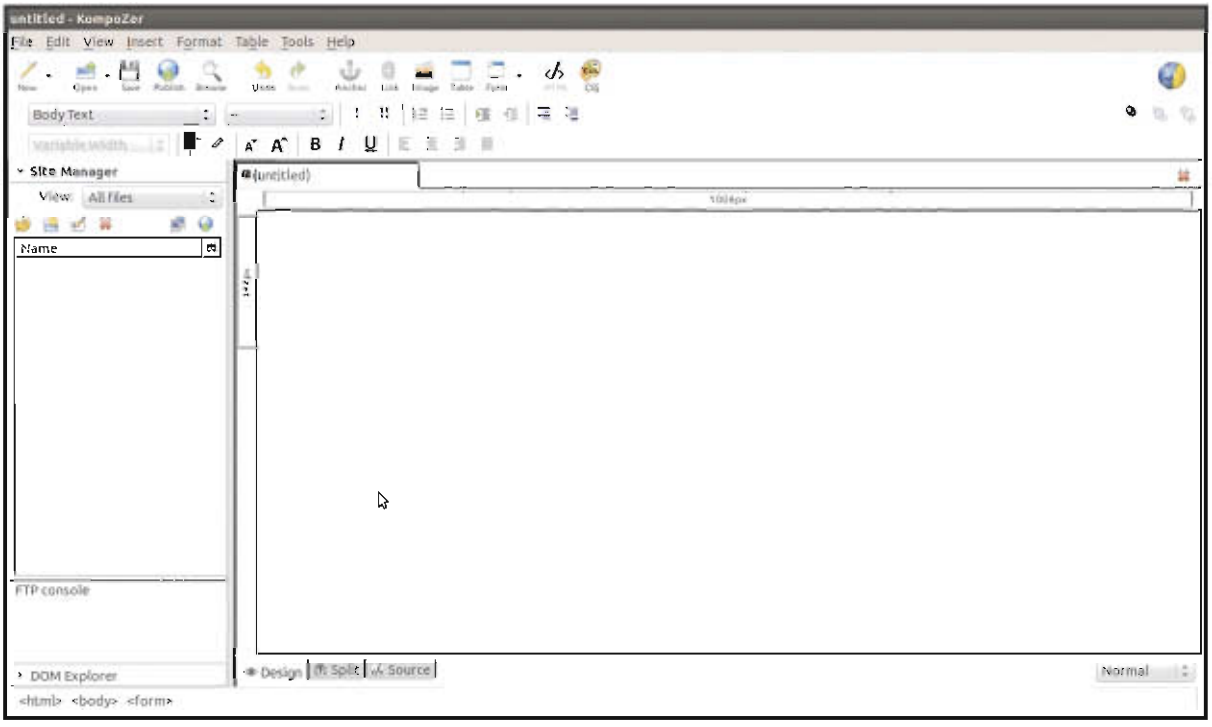


Figure 1.6 : Light blue colored form outline.

- First, we will insert a label for the name field. Click on **Insert → Form → Define label.** Place the cursor in the form where you want your label to appear. Type the text "Name" in the label as shown in figure 1.7. To come out of the label field, click outside the field.

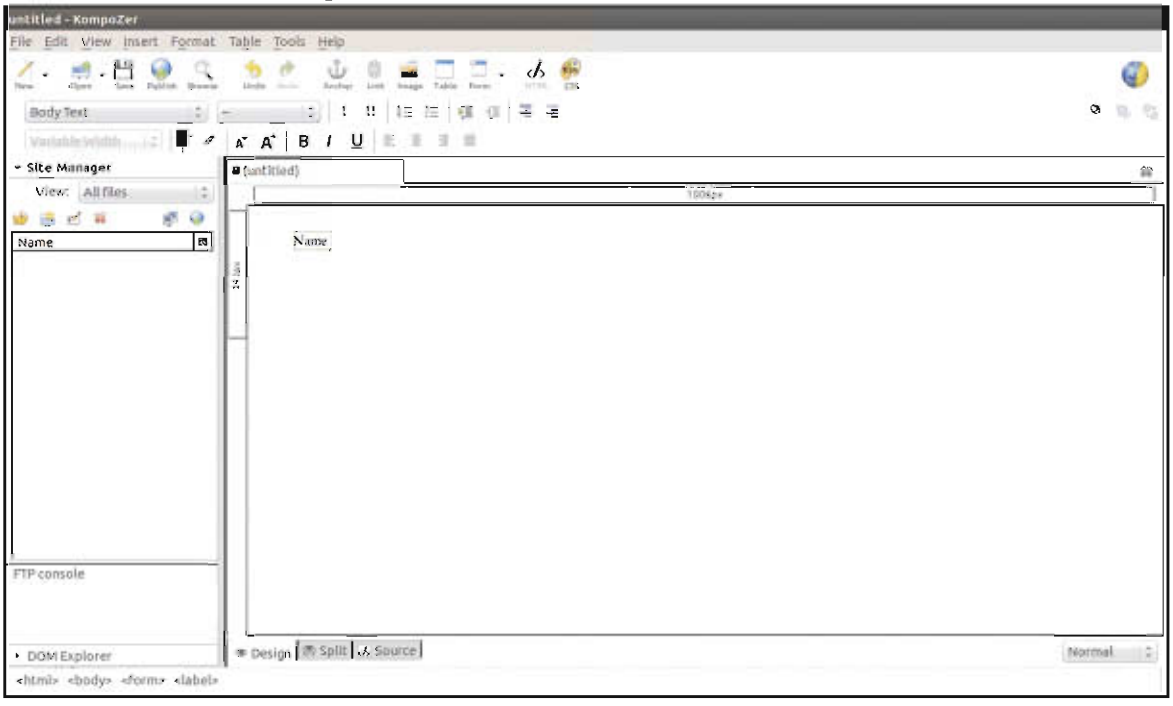


Figure 1.7 : Label field inserted in the form

- To insert an input text field in the form, click **Insert → Form → Form Field.** Figure 1.8 shows the form field properties dialog box. The drop down menu shows various input field type options which we had discussed earlier. Click on more properties, it shows advanced properties related to the field like field size, maximum length.

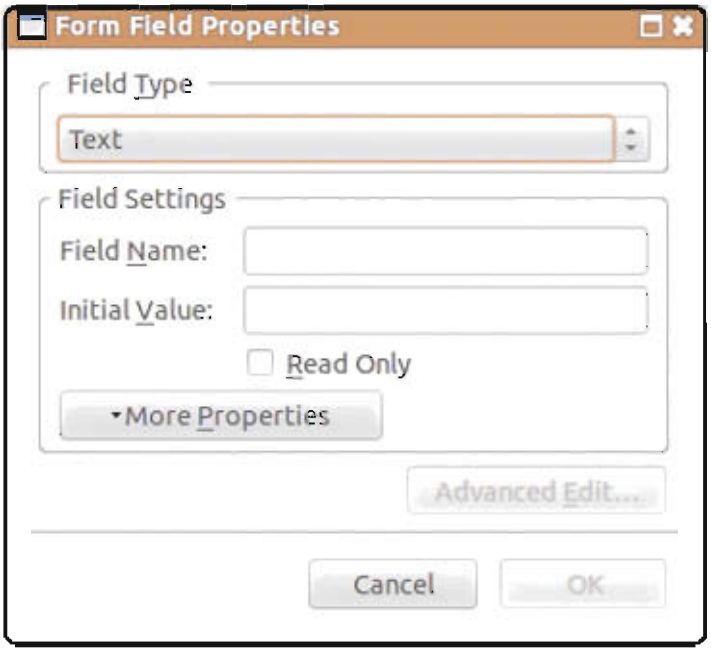


Figure 1.8 : Add form field

- From the drop down menu select Text. Under the Field Settings heading, enter a name in the Field Name text box. In our case we have used name as the field name. Enter some text in the Initial Value field, if you want to show some text before the user actually enters data. Here we have left this field empty. Click on the OK button. Figure 1.9 shows the form after text input field is added.

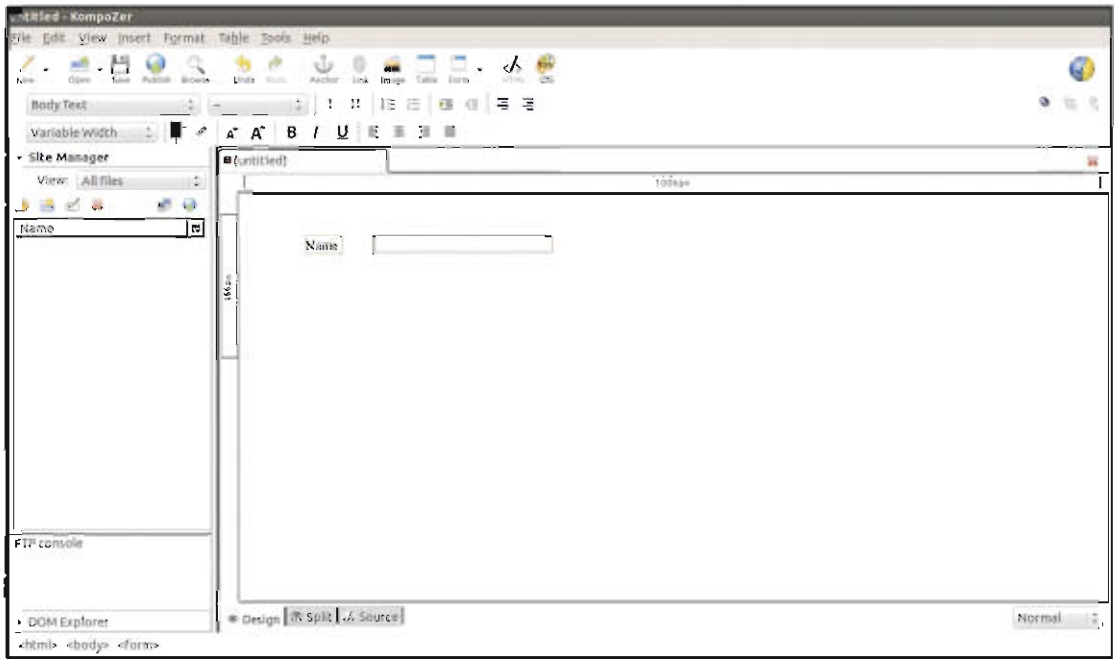


Figure 1.9 : Label and Text field added

- Similarly, we will add another label "E-mail" below the name label field. Add an input text field for E-mail, just like we did for the name field. Here enter text abc@xyz.com in the Initial Value textbox. This will help the user understand the format of E-mail address. Figure 1.10 shows the form with both the text field added.

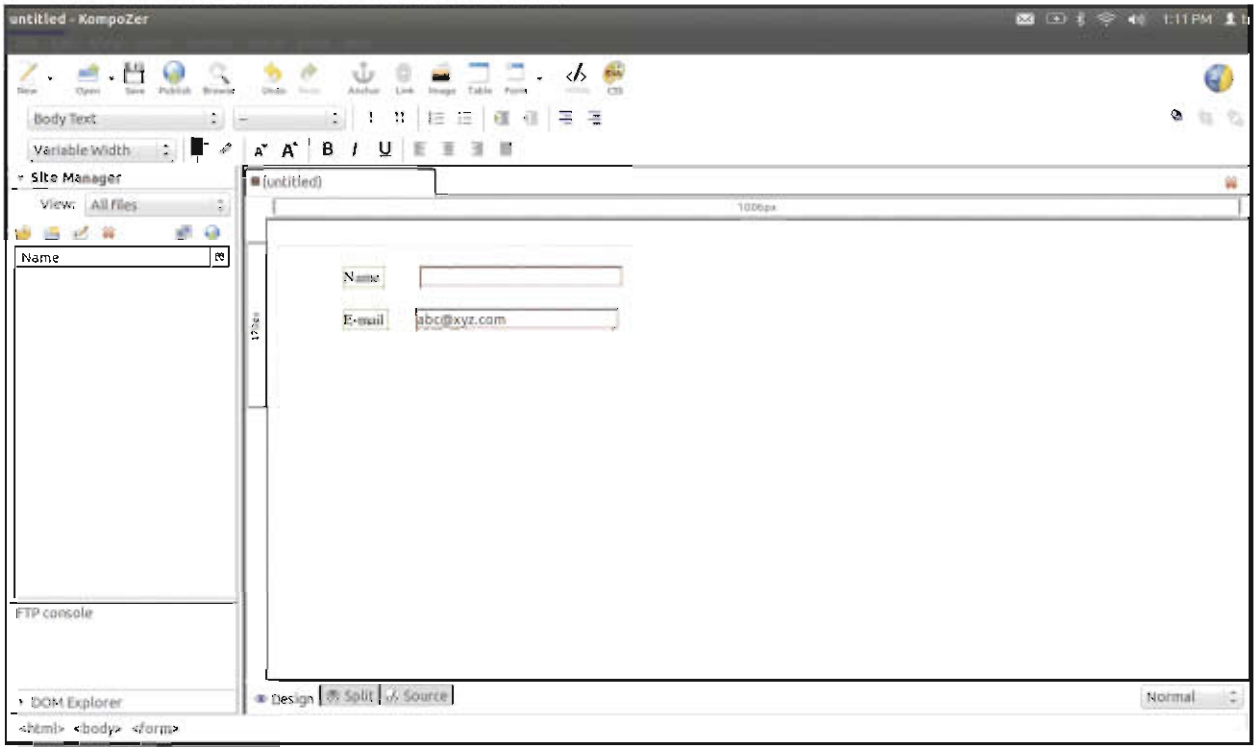


Figure 1.10 : Form with both text fields added

- Finally we will add a submit button in the form. Click **Insert → Form → Form Field.** From the drop-down menu select Submit Button. Type Submit in the both Field Name and Field Value text boxes, and click on the OK button. Figure 1.11 shows the look of form properties dialog box for submit button.

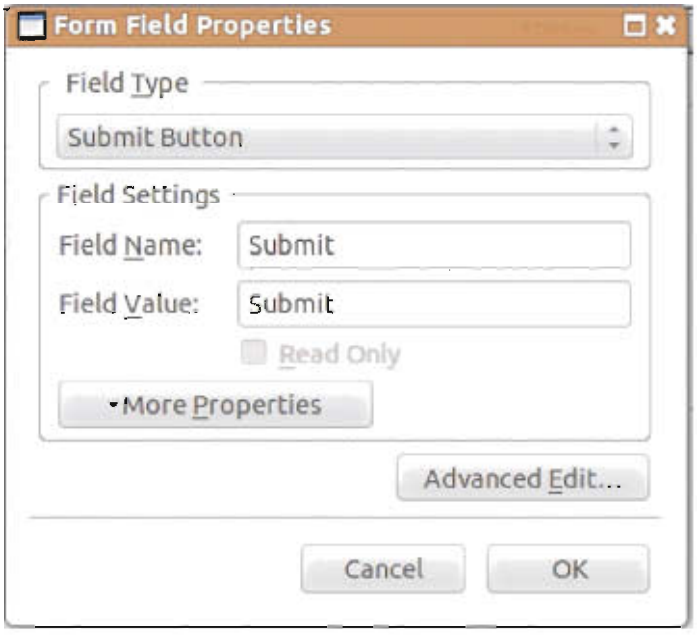


Figure 1.11 : Input field submit button

- The form at present is in the normal view. To have a preview of the form click on Edit mode toolbar and select Preview. Figure 1.12 shows the form in the preview mode.

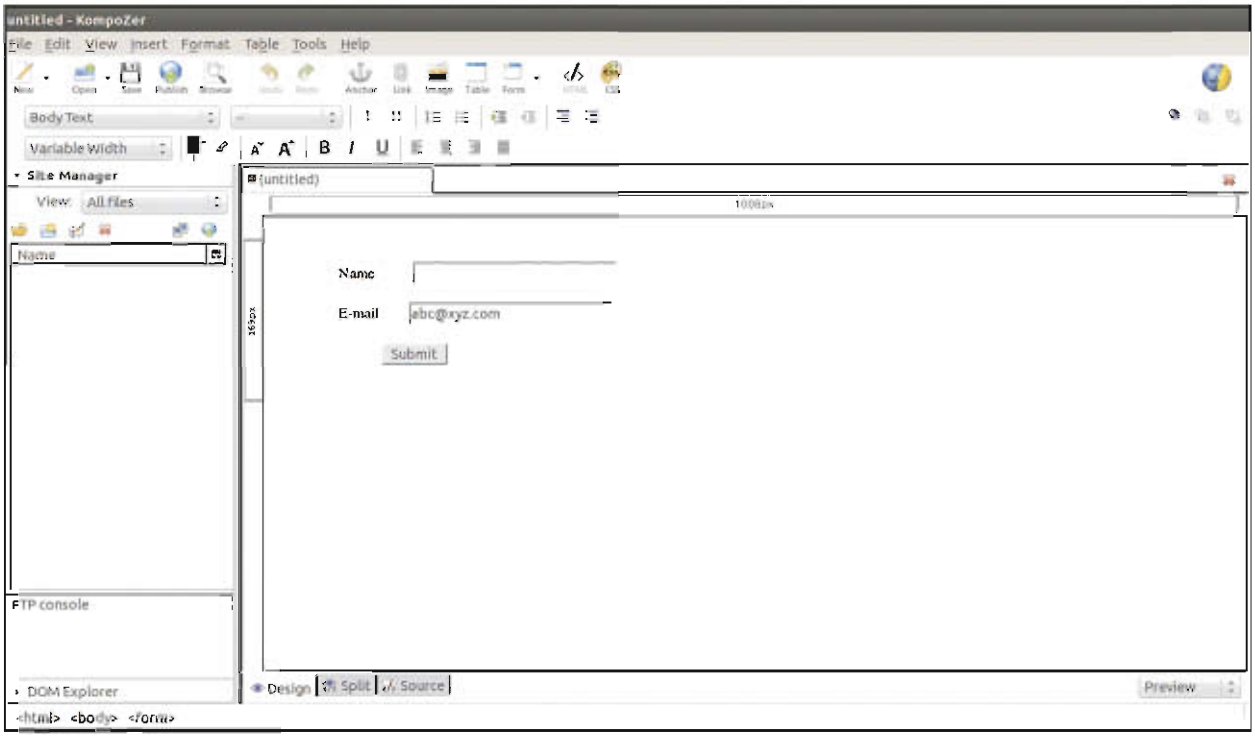


Figure 1.12 : Form as seen in preview mode

Thus, we have created our first form using KompoZer. You can see how KompoZer helps you creating forms within a short span and also relieves you from the tedious job of writing the source code which takes a long time. You can also see the source code of the form just created by clicking on the Source tab. See figure 1.13.

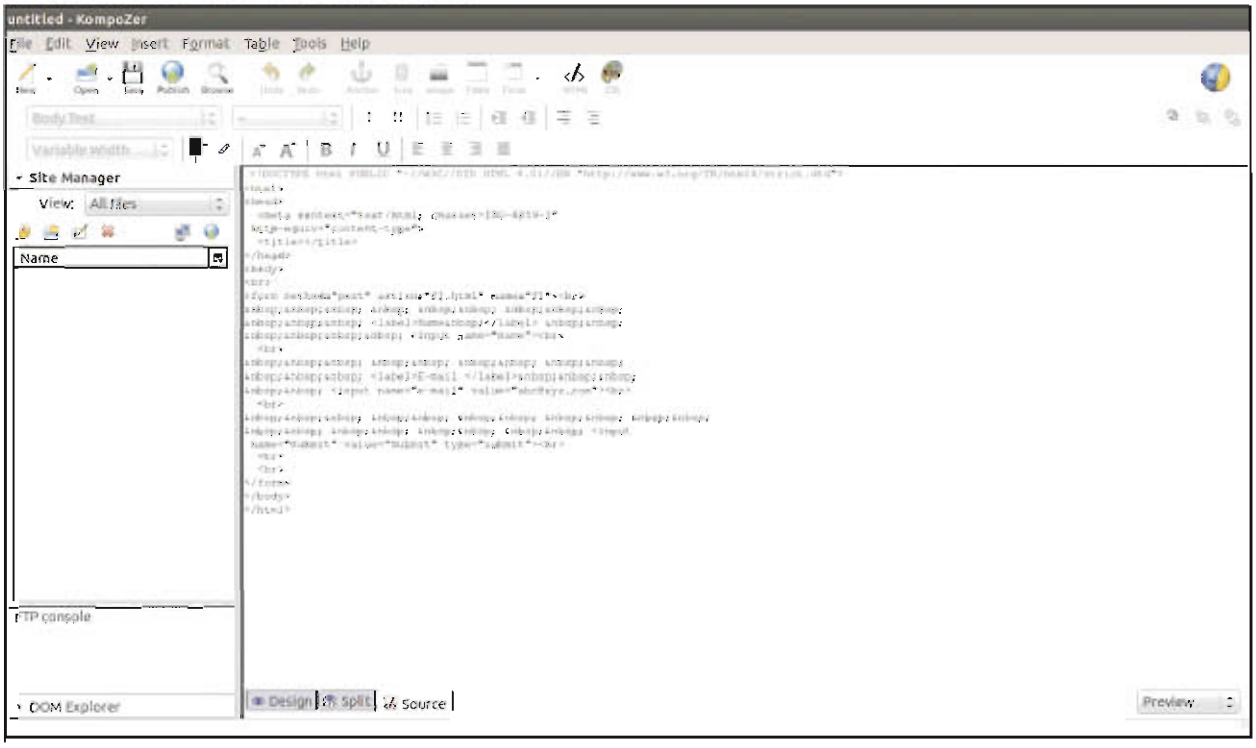


Figure 1.13 : Source code view of form

- Let us save the file which we have created. Click **File → Save.** Alternatively you can also click on the save button in the composition toolbar. This opens a Page Title dialog box as shown in figure 1.14. We can give a suitable title to the web page here. We have given the name "example1" to the title page. Now click on OK button.

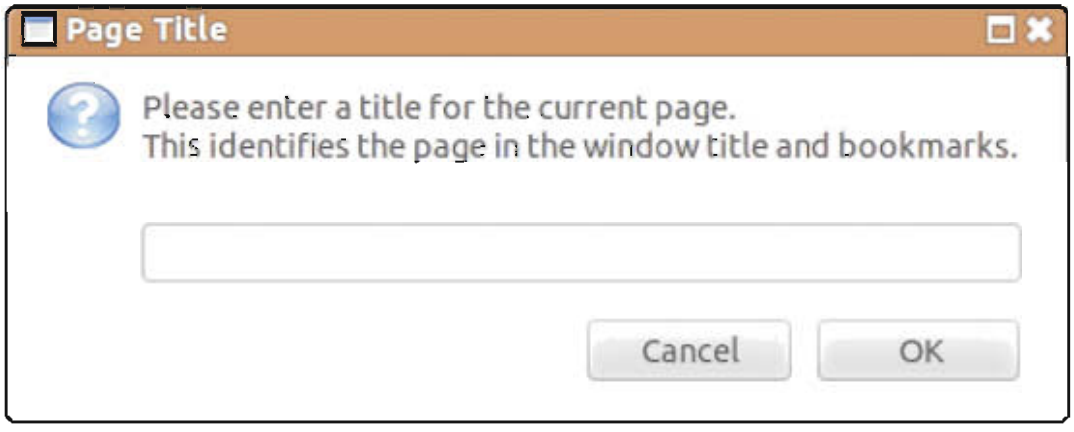


Figure 1.14 : Page title dialog box

The page title will be displayed in the browser windows title bar when viewed in the browser. In case we have created multiple web pages we should give the name of the website as the title page. In this example, since we created a single web page with a form, the title page is named as example1.

- After clicking the OK button, a new dialog box "Save Page As" is opened which prompts to enter a filename and specify the location where you want to save the file as shown in figure 1.15.

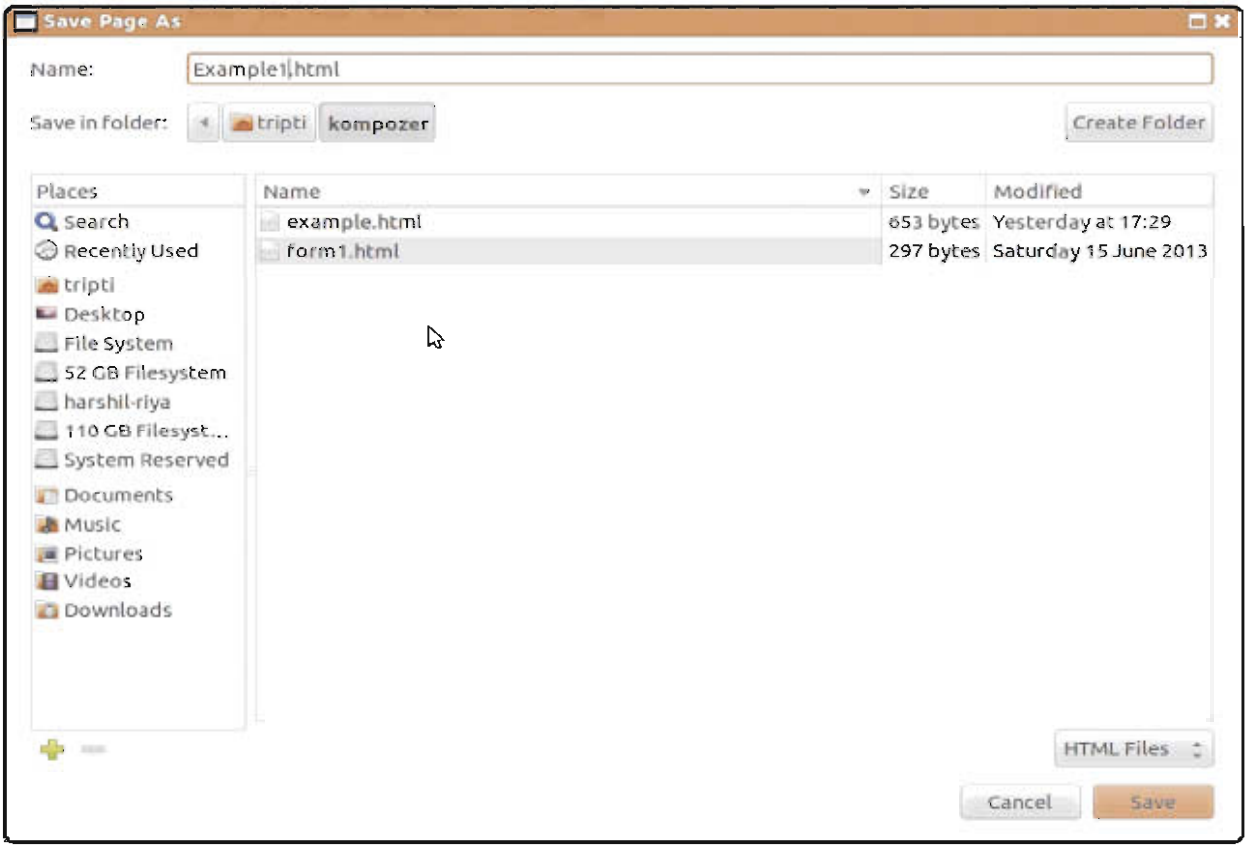


Figure 1.15 : Save Page As Dialog box

Remember to save the file with ***html*** or ***htm*** as an extension. Click on Save button. This will take us back to the main window.

Note : If you are creating a website and this page is the home page that will open when you type the website's URL, then save the page with the name index.html.

Having learnt how to create, open, save and create a simple form in KompoZer, let us now create the registration form, which we had created earlier using the HTML tags. Follow the steps given to create the registration form.

- Create a new file.
- From the menu bar, select **Insert → Form → Define Form.** In the Form Properties dialog box enter the details as shown in figure 1.16.

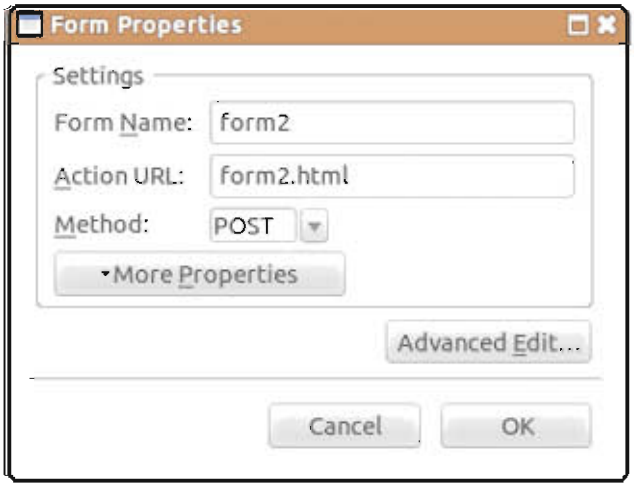


Figure 1.16 : Form Properties dialog box

- Press OK. A form will be displayed showing the light blue colored outline. Press the Enter key to create space in the form.
- To give the heading to the form, select Heading1 from Format Toolbar1. In Format Toolbar2, select centre align icon. Enter the text "Registration Form".
- To insert the label, Click **Form → Define Label.** Type "First Name" in the Field name. Next, to insert input field for the label "First name", click **Form → Form Field.** Select Text in the Field Type menu. Figure 1.17 shows the Form Field Properties dialog box. Observe that we have used "firstname" as the field name. Press OK button.

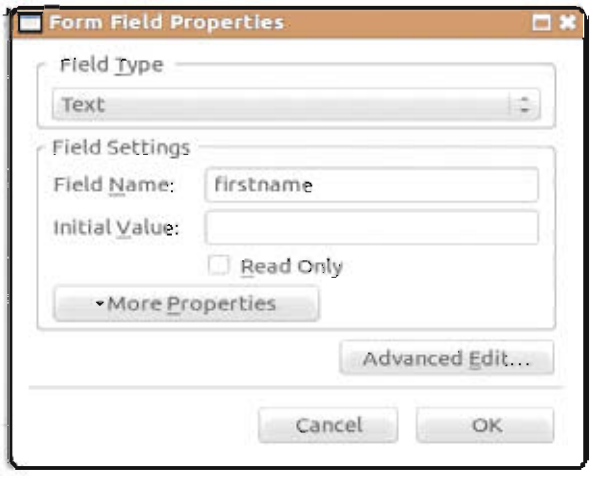


Figure 1.17: Form Field Properties dialog box for Firstname

- Similarly, insert the label "Middle Name" and "Last Name" in the form. The form after adding the fields will look similar to the one shown in figure 1.18.

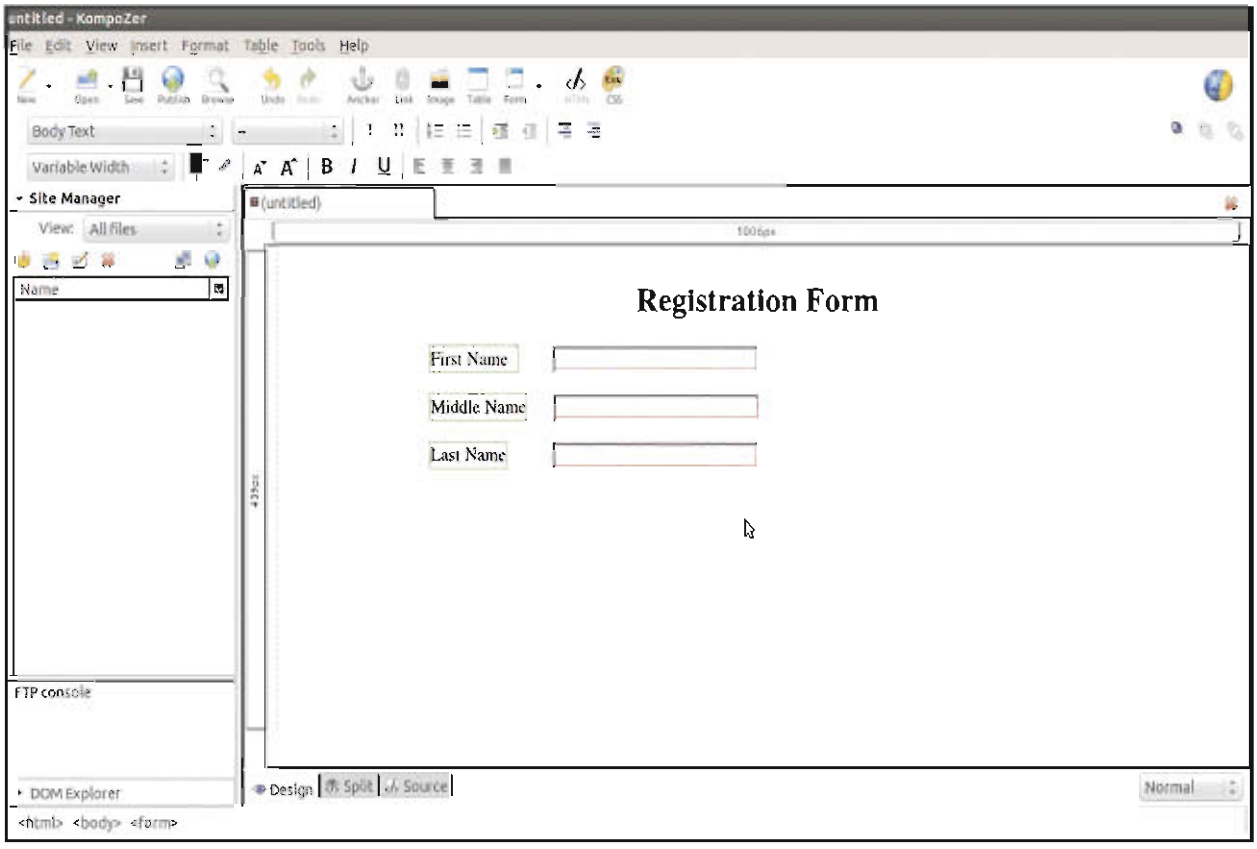


Figure 1.18 : Form displayed after adding the fields

- Now we need to create radio buttons for the Gender field. First, create a label named "Gender".
- For creating a radio button, click **Form → Form Field** and select field type as Radio Button from the drop down menu as shown in figure 1.19.

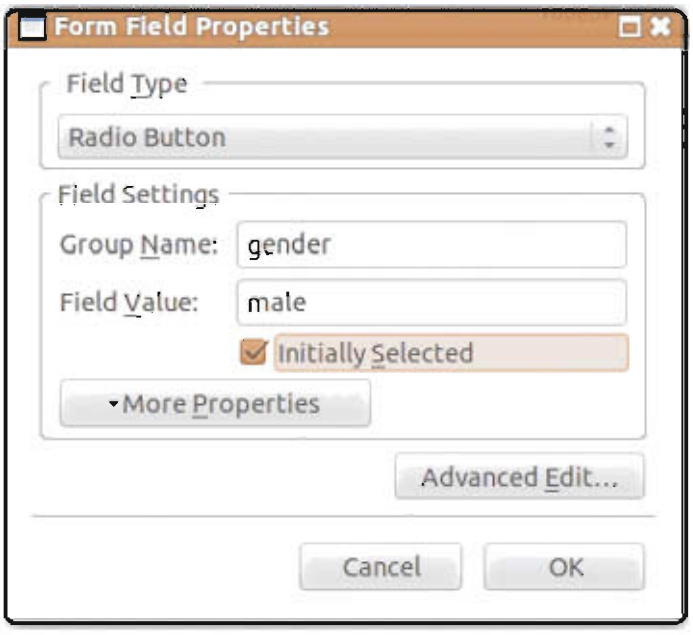


Figure 1.19 : Form field property for Radio Button

Type a name in the Group name box (note that the name should not contain spaces). Here we have entered Group Name as "gender". Similarly in the Field Value text box type "male". If we want the male option of the radio button selected when the form is loaded then, check the box in front of the text "Initially Selected" and click OK button.

- Insert a label with title "Male" near the radio button created.
- Similarly create another radio button named "Female". Remember, when we create the radio buttons within a group, the group name must be the same for all the possible answers. Hence enter the Group Name as "gender". In the Field Value text box type "female". Click OK button.
- Insert a label "Female" near the radio button created. The form after inserting the radio buttons and their labels will look as shown in figure 1.20.

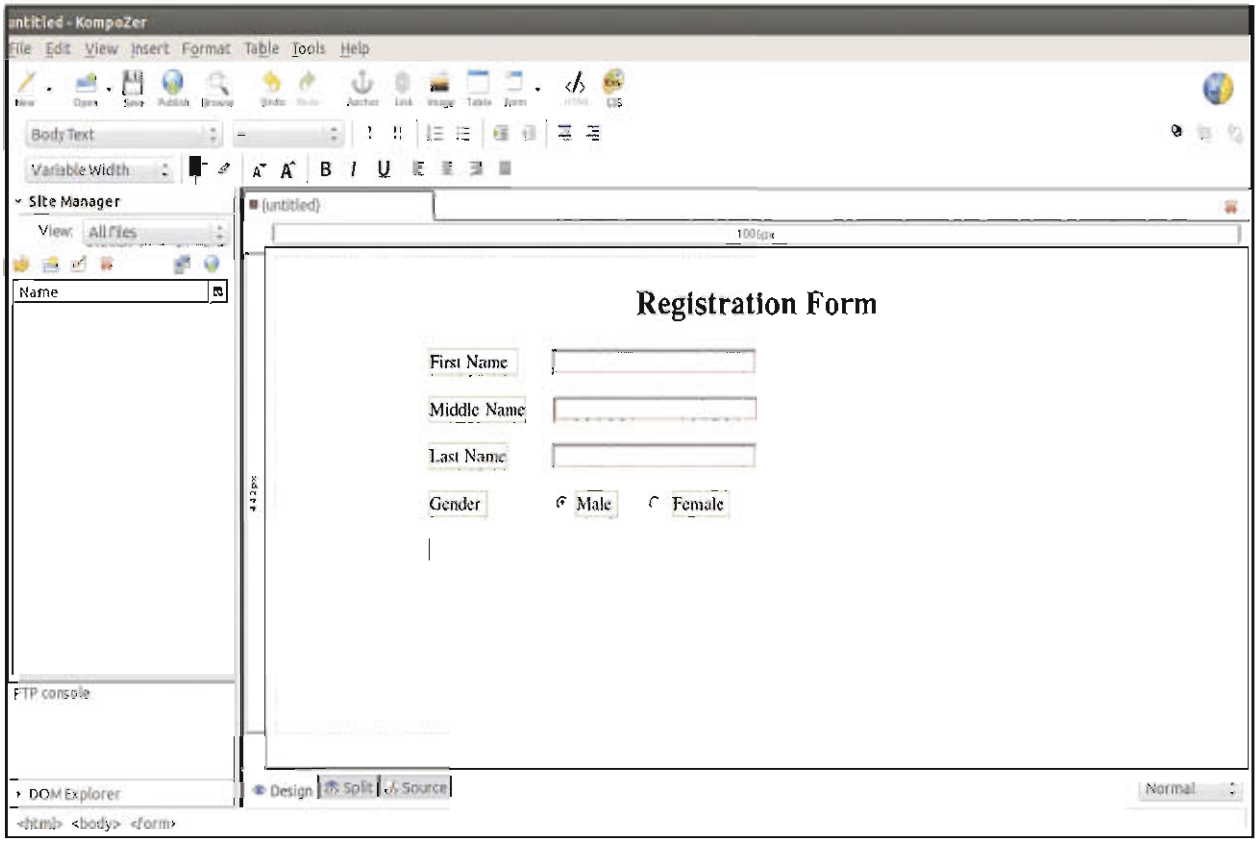


Figure 1.20 : Form displayed after radio buttons are added

- Now we need to add "Hobby" field. As a person can have more than one hobby, hence multiple selections of hobbies are possible. Thus we need to create check box for the hobby field. Create a label for hobby.
- Now, click **Form → Form Field** and from the drop down menu select Check Box field type. Enter name in the Field Name box and a value in the Field Value box as shown in figure 1.21. Check the box in front of "Initially Selected" so as to keep the option checked when the form loads. Click OK button.

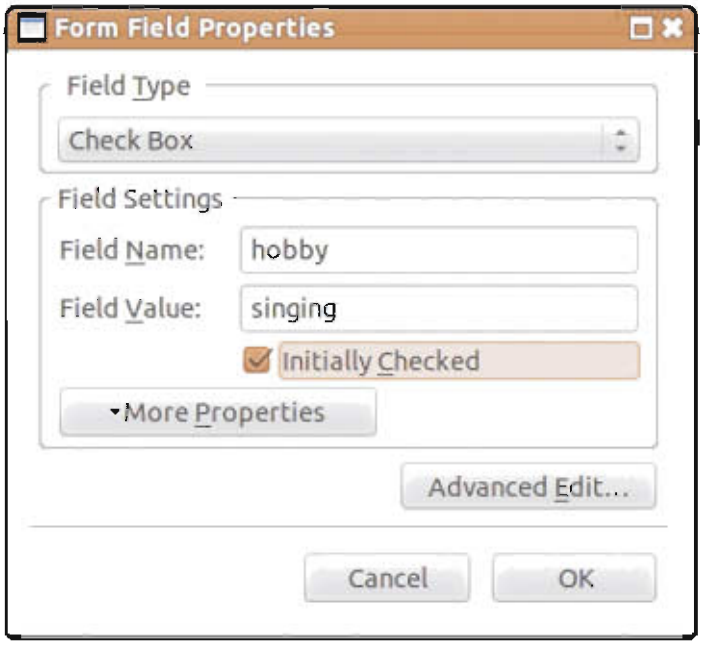


Figure 1.21 : Checkbox field type settings

- Insert a label "Singing" near the checkbox created.
- Similarly create two checkboxes with Field Values as "dancing" and "reading" respectively. Remember, to keep the Field Name same for all the options of checkbox.
- Insert the labels "Dancing" and "Reading" near the checkboxes created. The form after inserting checkboxes and their labels will look as shown in figure 1.22.

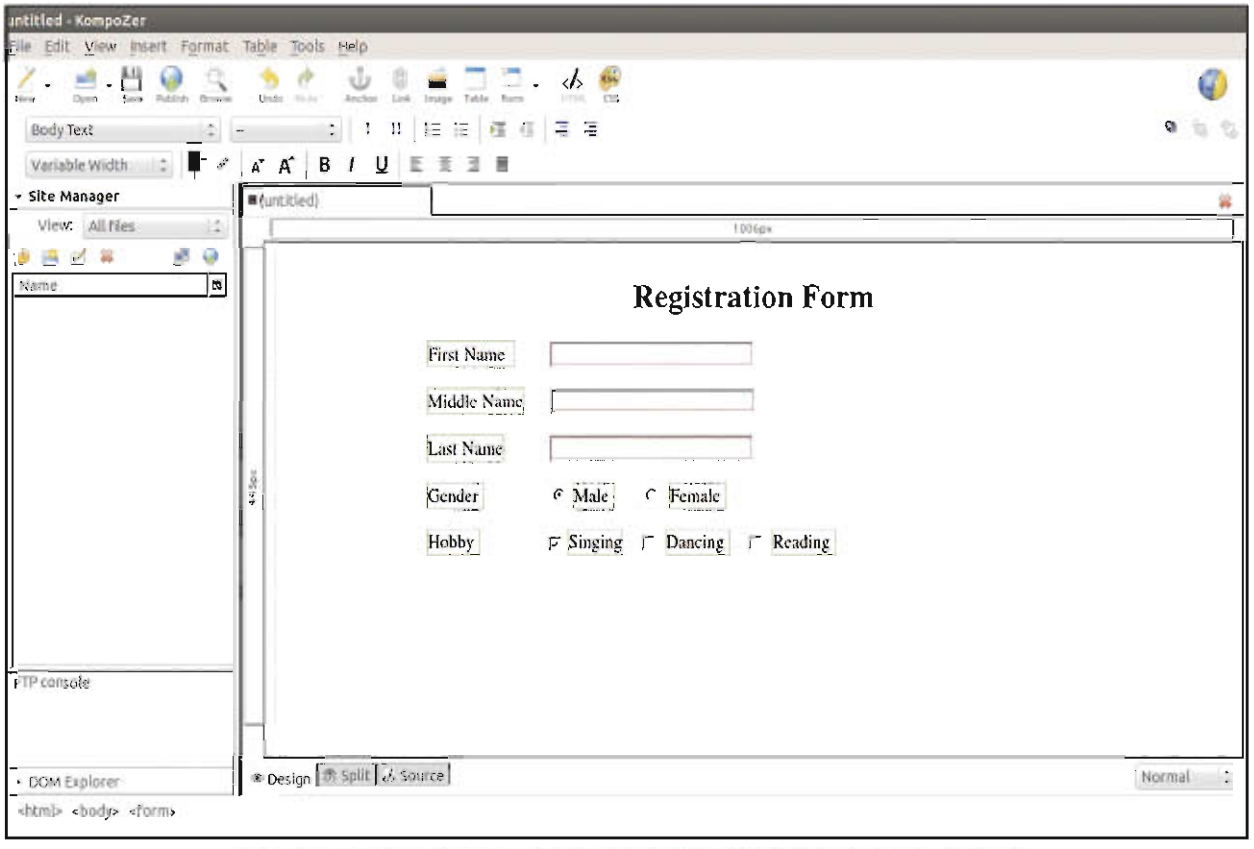


Figure 1.22 : Form display after checkboxes are added

- Next, we need to the "Address" field. As the user enters a large text in the address field, we will keep the field type as textarea. First, create a label named "Address". Now, click **Form → Text Area.** This will open a Text Area Properties dialog box as shown in figure 1.23.

Figure 1.23 : Text Area Properties dialog box

Enter the Field Name. Select the rows and the columns required for the textarea. Here we have kept the rows as 5 and columns as 70. In the Initial Text field enter a suitable text which will be displayed when the form loads. Click OK button.

- Next, we will insert the "City" field. Add a label for the city field. The user will be asked to select the city from the drop down menu. So we need to create the city field using selection list option. Click **Form → Selection List.** This will open a Selection List Properties dialog box as shown in figure 1.24.

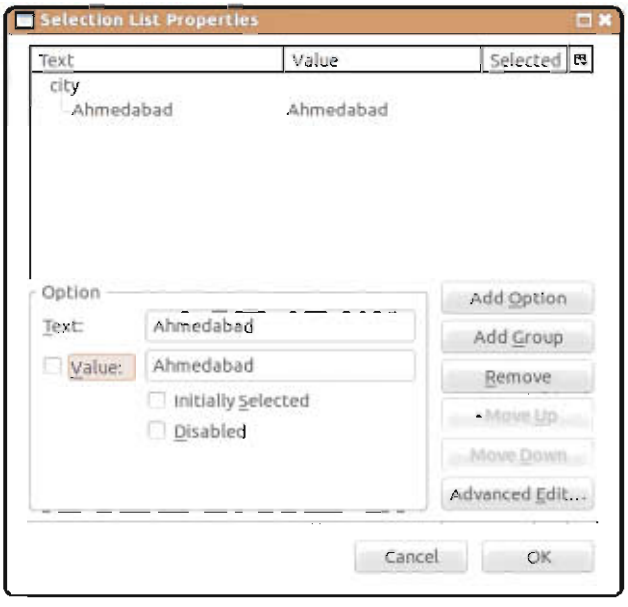


Figure 1.24: Selection List Properties dialog box

Type the name "city" in the List Name box and press Add Option button. Next, type "Ahmedabad" in the Text field. Again press Add option to add the city "Baroda". Likewise, add the city "Rajkot" and "Surat". Remember, to select the option "Initially Selected" when adding the Rajkot city. Click OK button. Figure 1.25 shows the Selection List Properties dialog for the added cities.

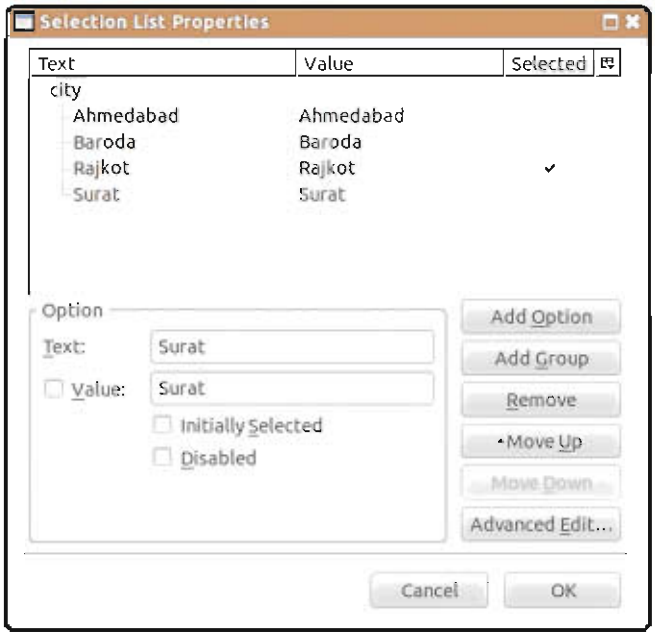


Figure 1.25 : Selection List Properties dialog box for various cities

- Next, we will add the "Submit" button. Click **Form → Form Field.** From the drop down menu select Submit Button. Enter text Submit in Field Name and Field Value text box respectively. Click OK button.
- Similarly we will add the "Reset" button. From the drop down menu select Reset Button. Enter text Reset in Field Name and Field Value text box respectively. Click OK button. The final form in the normal will be as shown in figure 1.26.

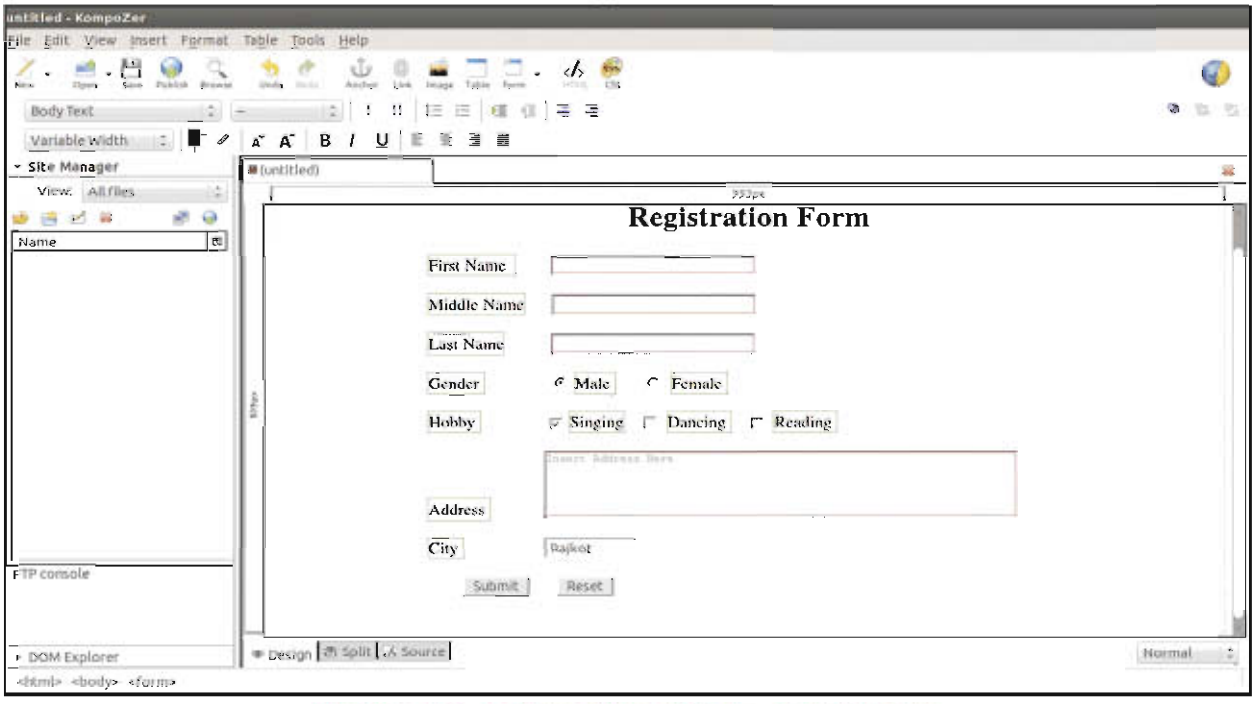


Figure 1.26 : Form displayed in normal view

Save the file with name "example2". Figure 1.27 shows the preview mode of the form.

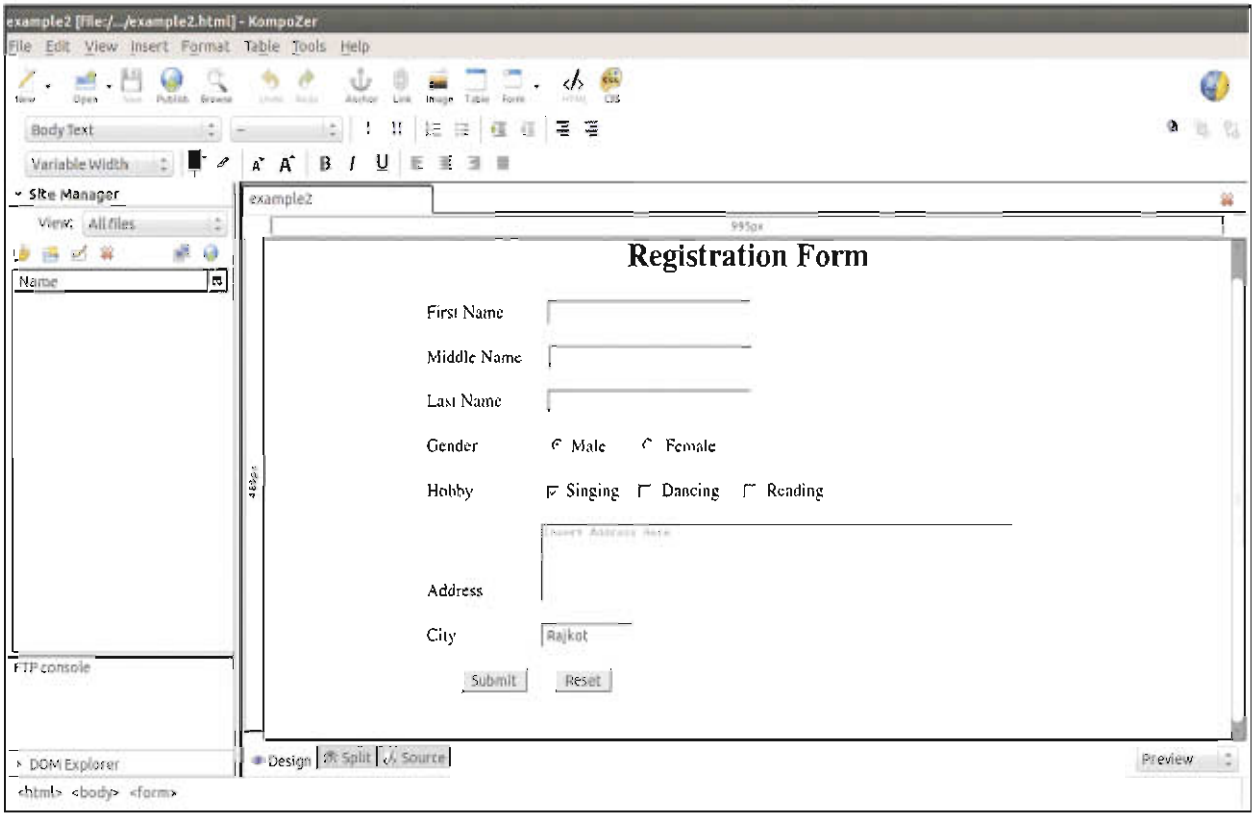


Figure 1.27 : Form displayed in preview mode

Figure 1.28 Shows the registration form as seen in the browser.

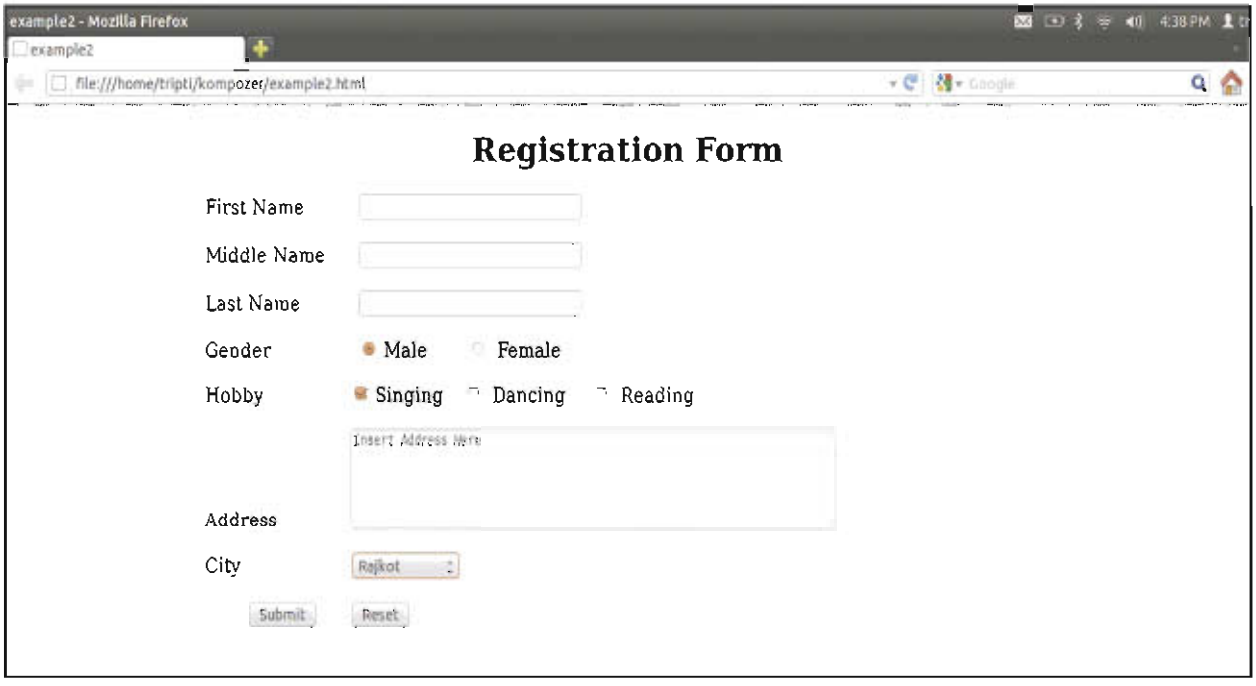


Figure 1.28 : Form displayed in Browser

Observe that the form background is white colored. If you want to give a background color to the form, go to **Format → Page Colors and Background.** This will open a dialog box as shown in figure 1.29.

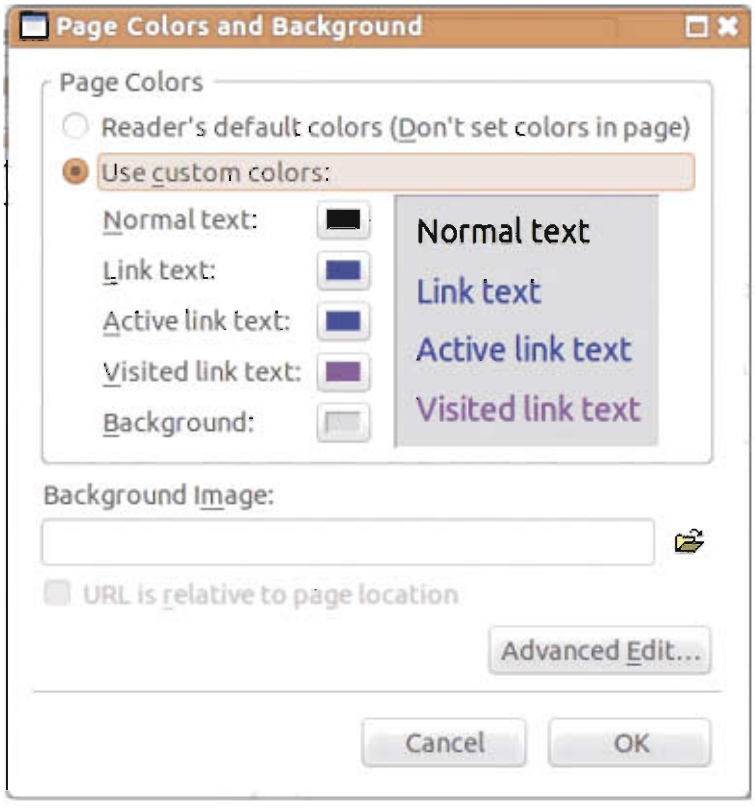


Figure 1.29 : Page Colors and Background dialog box

Select "Use custom colors" option. Click the background option and select the color of your choice from the Block Background Color dialog box as seen in figure 1.30. Click OK button. This will lead you back to the dialog box shown in figure 1.29. Again click OK button.

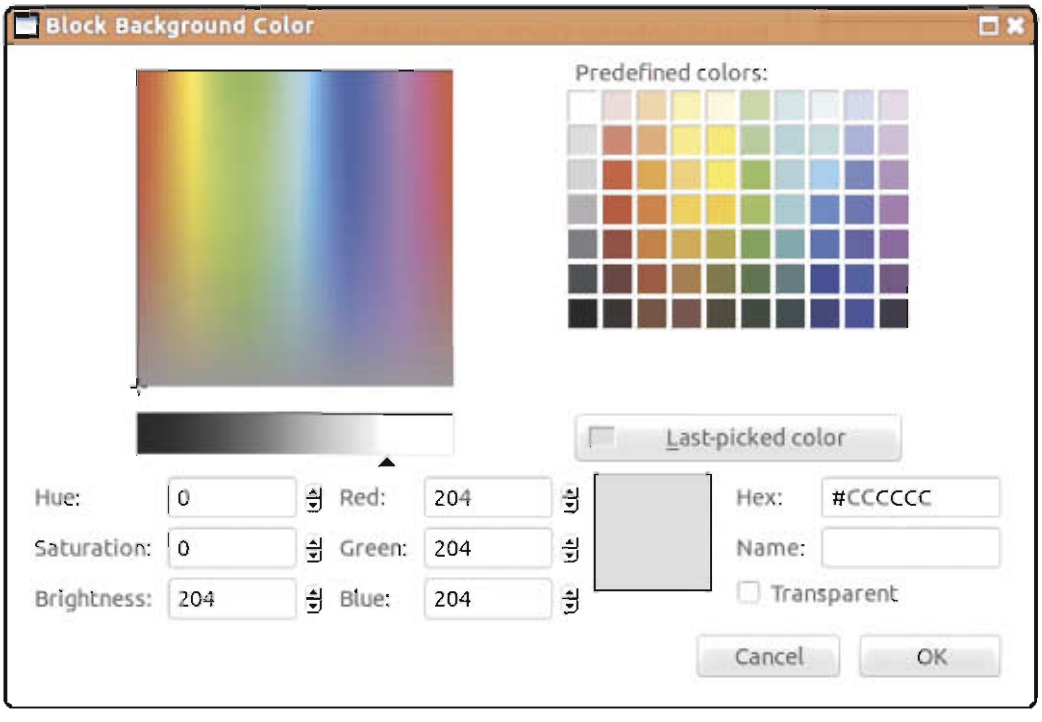


Figure 1.30 : Background color selection

After selecting the color the form will look as shown in figure 1.31.

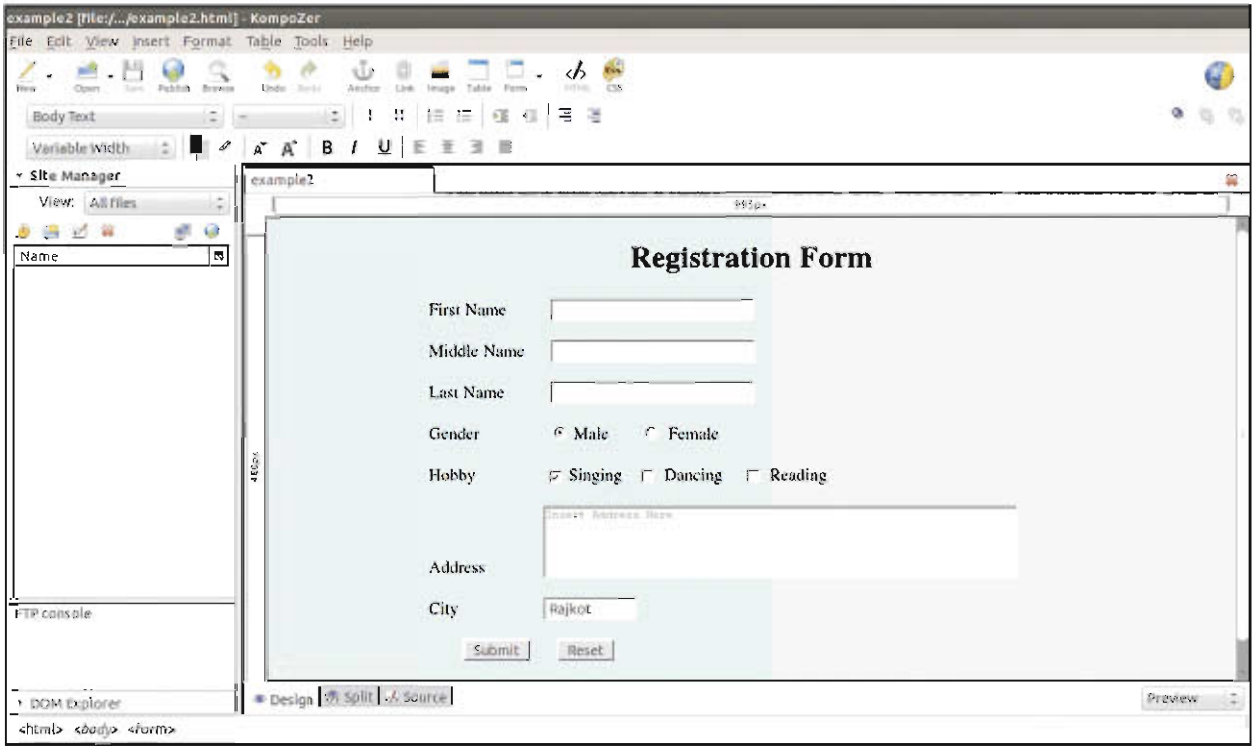


Figure 1.31 : Form displayed in preview mode after adding background color

View the form using the browser and observe the change in the background color.

Summary

Forms are used to accept the data over the web. A form in HTML is a container used to collect different kind of inputs from the user. The users enter the information in the form which can be their personal information, a feedback about a product, a survey or shipping and credit card details. KompoZer is a free open source web development IDE used to create websites. It provides a web page editor which has a simple graphical interface known as WYSIWYG "what you see is what you get". Creating forms using Kompozer is simple and fast.

EXERCISE

1. What is a Form ? List the elements used to create forms in HTML.
2. State the use of Input element in HTML forms. Write about the different attributes of input tag.
3. What is the purpose of textarea element in HTML forms ?
4. Write about select and option element.
5. List the various toolbars seen in the Kompozer interface.
6. Which are the two attributes of form? Explain.

7. Choose the most appropriate option from those given below :

(1) Which of the following is a container used to collect different kinds of inputs from the user.

(a) Form (b) Webpage (c) Text (d) Input

(2) Which of the following element is used to create an HTML form ?

(a) Textarea (b) Form (c) Select and Option (d) Input

(3) Which of the following is the tag used to implement form element ?

(a) <form>… </form> (b) <form>… <form>

(c) </form>… </form> (d) <frm>… </frm>

(4) Which of the following attribute of form is used to specify where to send the form data when the form is submitted ?

(a) method (b) action (c) submit (d) input

(5) Which of the following attribute of form specifies the HTTP method to be used when sending the data ?

(a) submit (b) action (c) method (d) input

(6) Which of the following values are used by method attribute ?

(a) GET and POST (b) GET and SET

(c) GET and PUT (d) SET and POST

(7) Which of the following method allows only a limited amount of information to be sent at a time?

(a) GET (b) POST (c) SET (d) PUT

(8) Which of the following method sends the data as a block through the HTTP transaction ?

(a) GET (b) SET (c) PUT (d) POST

(9) Which of the following attribute of the input element specifies the field that is to be created in the form ?

(a) Input (b) Type (c) Name (d) Value

(10) Which of the following element allows multi-line text input ?

(a) Textarea (b) Input (c) Select and Option (d) Form

(11) Which of the following element is used to create a drop down list or menu in a form ?

(a) Input (b) Textarea (c) Select (d) Form

(13) Which of the following is a free open source web development IDE ?

(a) HTML (b) Kompozer (c) Scite (d) Base

(14) Which of the following stands for "WYSIWYG" ?

(a) When You See Is When You Get (b) What You See Is When You Get

(c) What You See Is What You Get (d) When You See Is What You Get

LABORATORY EXERCISE

1. Create a form for student's personal details.
2. Create a feedback form for the guests who visit your school.
3. You had gone for a vacation with your parents; the tour operator has asked you to give reviews of your trip. Create a form for the same.

Cascading Style Sheets and Java script 2

Cascading Style Sheets or CSS allows you to specify styles for the visual elements of the website. It helps us to keep the information content of a document separate from the details of how to display it. This detail of how to display the document is known as style. Styles specify the appearance of particular page elements on the screen. Keeping the style separate from the content helps us to :

- Avoid duplication in coding
- Use the same content with different styles for different purposes
- Easy maintenance of code

In HTML, for example, if we want contents of some paragraph tag <p> to appear in bold, then we have to insert the bold tag <b> every time the <p> tag appears in the source code. If the website is large, repeating the tag every time becomes tedious and time-consuming. But using CSS, we can set all element types to appear in a style as per our choice. So even if we have more number of <p> tags in the website, using CSS we can set the style for all the <p> tags at once. Thus we can say that, HTML is used to describe the information content of the document and not the style, while CSS describes the style of the elements in the document and not its contents. Using CSS, we can control the font types, font and element colors, pad spaces, margins, and element positions in our website.

Syntax of CSS

The syntax of CSS consists of special symbols known as rules. A CSS rule has two main parts: a selector, and one or more declarations. The selector is the HTML element on which you want to apply the style. Declaration consists of a property associated with the HTML element used in selector and its corresponding value. The general syntax of CSS is defined as:

selector {property : value}

Figure 2.1 gives an example of one such syntax :

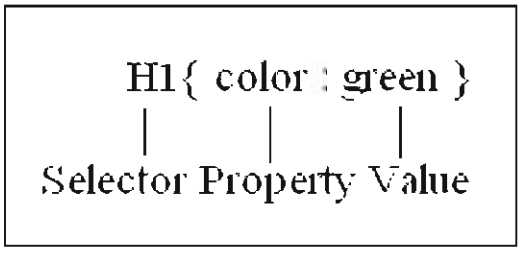


Figure 2.1 : Syntax of CSS

We can easily add CSS using KompoZer. Let us learn how to add CSS to a web page in KompoZer. To make sure that KompoZer uses CSS by default, click **Edit → Preferences.** This will open Options dialog box as shown in figure 2.2. Select the General category on the left side of the window. Select the checkbox "Use CSS styles instead of HTML elements and attributes" if it is not selected. KompoZer will now use the CSS styling instead of HTML to format the text.

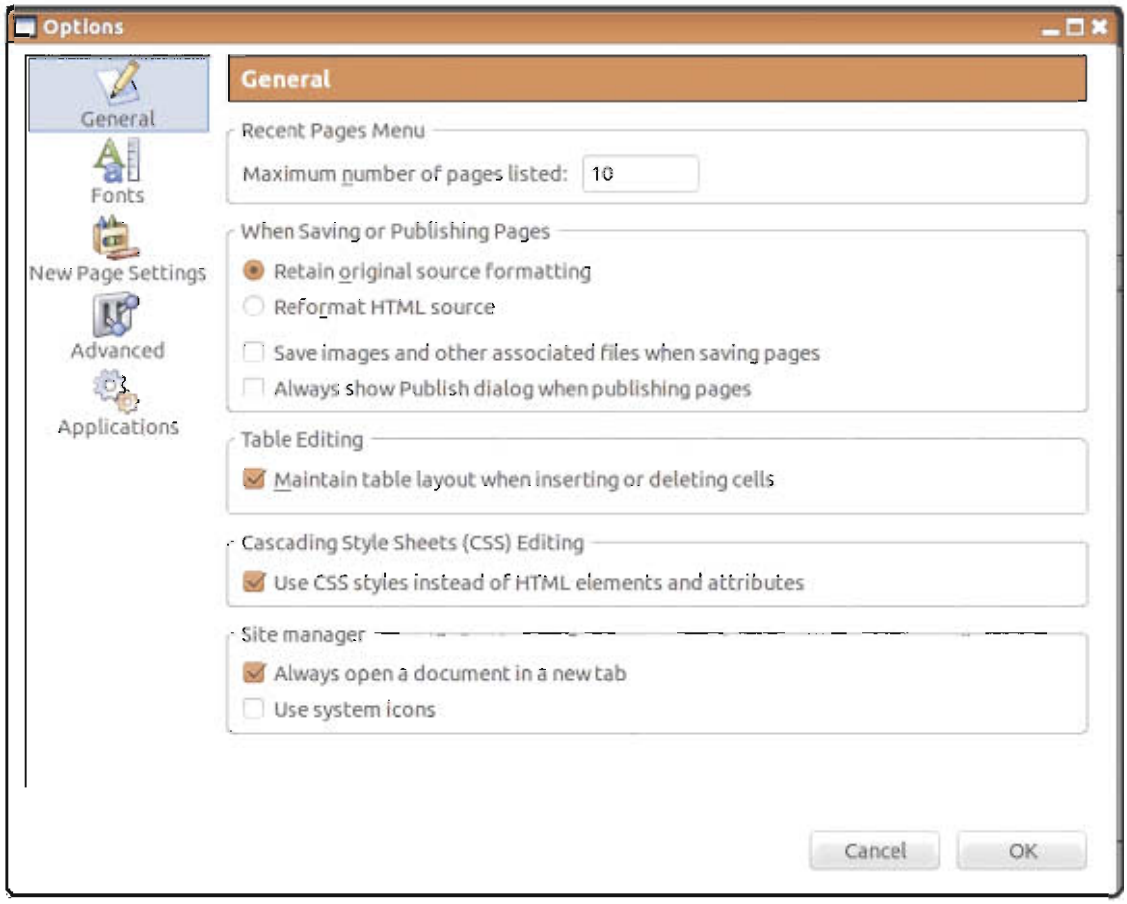


Figure 2.2 : Options dialog box

Assume that we want to design a website and the heading on all the web pages should follow a particular style. For example, all the Headings1 (h1) inserted in the web pages should follow the style as given :

- Font : Times New Roman
- Case : Uppercase
- Alignment : Centre aligned
- Background color : Light Blue
- Border : Dotted Border

Follow the steps given to create the CSS for the above heading.

- Open a new file. Give the Title name and save the file.
- In the composition toolbar, select the Cascade button . (Note: If the file is not saved, the Page title dialog box will be opened. After giving the page title, save the file). This will open the dialog box as shown in figure 2.3. Using this dialog box we will define the styles for each element. Click on the first radio button "style applied to all the elements of type".

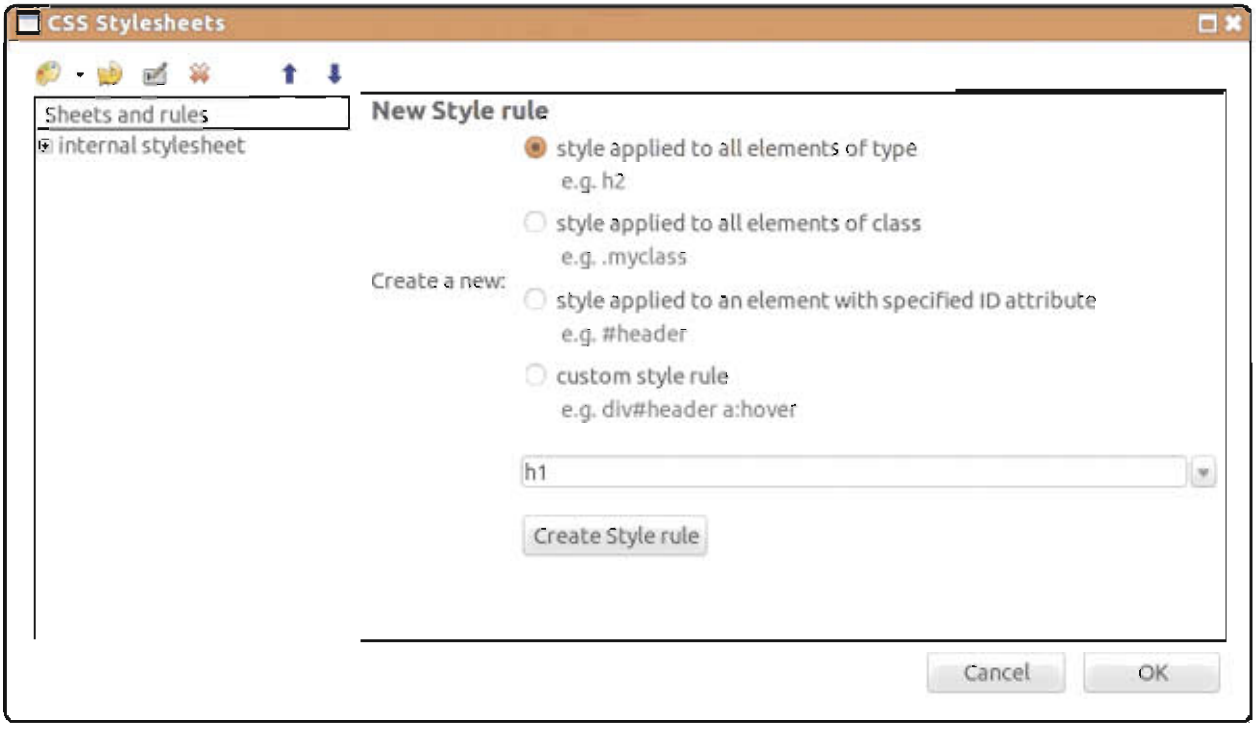


Figure 2.3: CSS Stylesheets dialog box

- From the drop down menu choose an element to create the style rule. As we want to create a style for Heading1, we have selected h1 (Heading 1) from the drop down menu as shown in figure 2.3.
- Click the Create Style rule button. You will see that the CSS Stylesheets dialog box remains open but the options will change.
- Figure 2.4 shows the h1 element just below the heading "internal stylesheet" on the left pane of the window. For each element we add, a style rule will appear in this list below the heading "internal stylesheet". On the right side of the window, you will see various tabs like General, Text, Background, Borders, Box, Lists and Aural. Each of these tabs can be used to give a specific style to the element. But all the tabs may not be applicable to the selector for which the style rule is created. Now, let us create the style rule for h1 selector.

Figure 2.4: CSS stylesheets dialog box with options for h1

- Select h1 in the left pane of the window. Click the Text tab as shown in figure 2.5. This will show various style options like Font family, Font size, Line height, Color, Case, Alignment and many more.

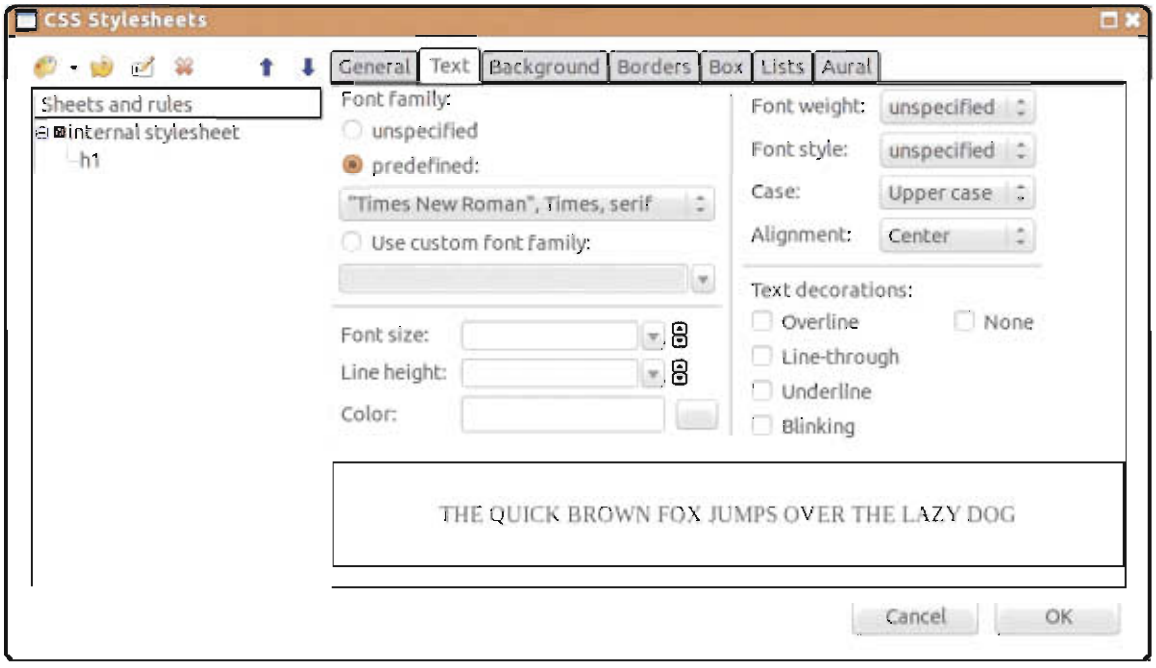


Figure 2.5 : Text tab

- We will now select the options as per the style required for h1. In the Font family section, select the "predefined" radio button and choose "Times New Roman" from the drop down menu. Choose "Uppercase" from the Case menu and "Center" from the Alignment menu. Figure 2.5 shows the options selected. As you change the options, you can see the default text in the window changes automatically according to the styles.
- To set the background color, select the Background tab. This will open the options for the background as shown in figure 2.6.

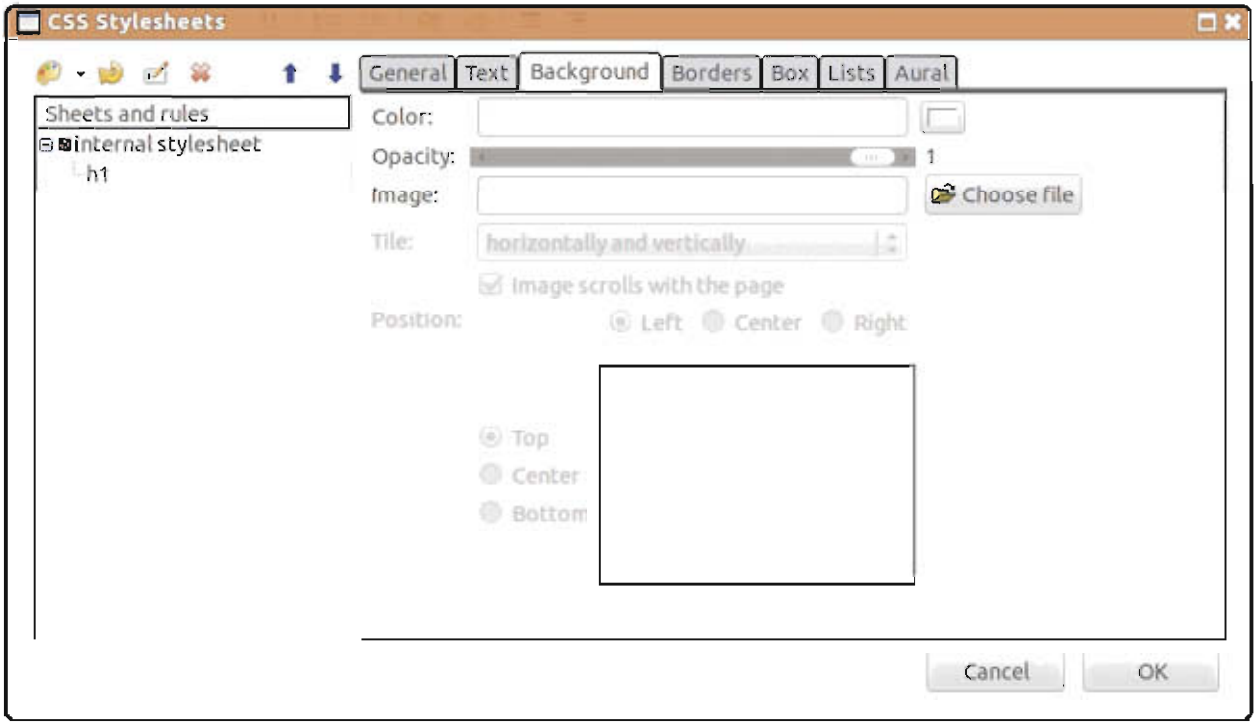


Figure 2.6 : Background tab

- In the Color option, click the color palette button. This opens Block Background Color dialog box as shown in figure 2.7. Select the color of your choice and press OK button.

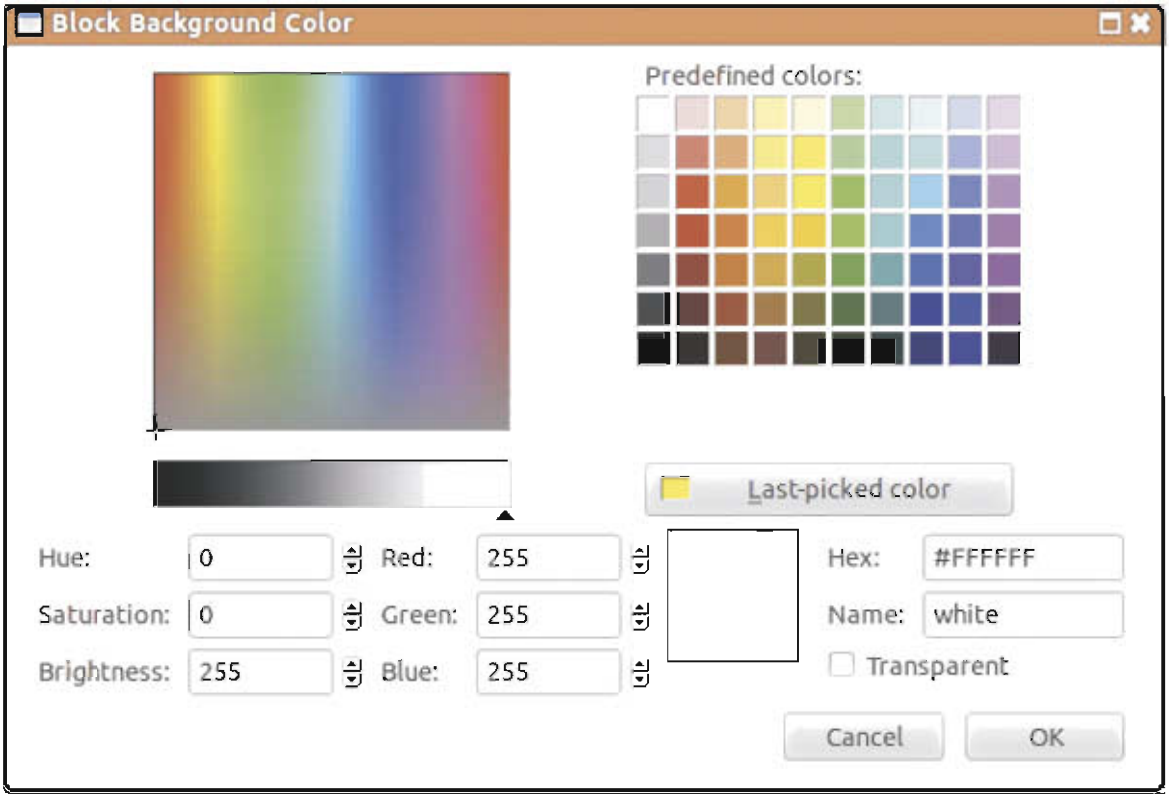


Figure 2.7 : Block Background Color dialog box

- Figure 2.8 shows the background tab after selecting the color.
  Note : If you want to keep an image in the background then in the Image option, click on "Choose file" and select the file for the background.

Figure 2.8 : Options selected in Background tab for h1

- To set the border, select the Border tab. This will open the options for the border as shown in figure 2.9. If you want to apply the same border on all the four sides, tick the checkbox in front of the text "All four sides use same border style". You will see that only one option "Top" will be enabled and all others are disabled. This is because; the effect made on a single side will be applied on all the four sides. From the drop down menu of Style, select dotted. You can also specify the width and the color of the border as per your choice. We can also see the preview of the border style used by us. Press OK button.

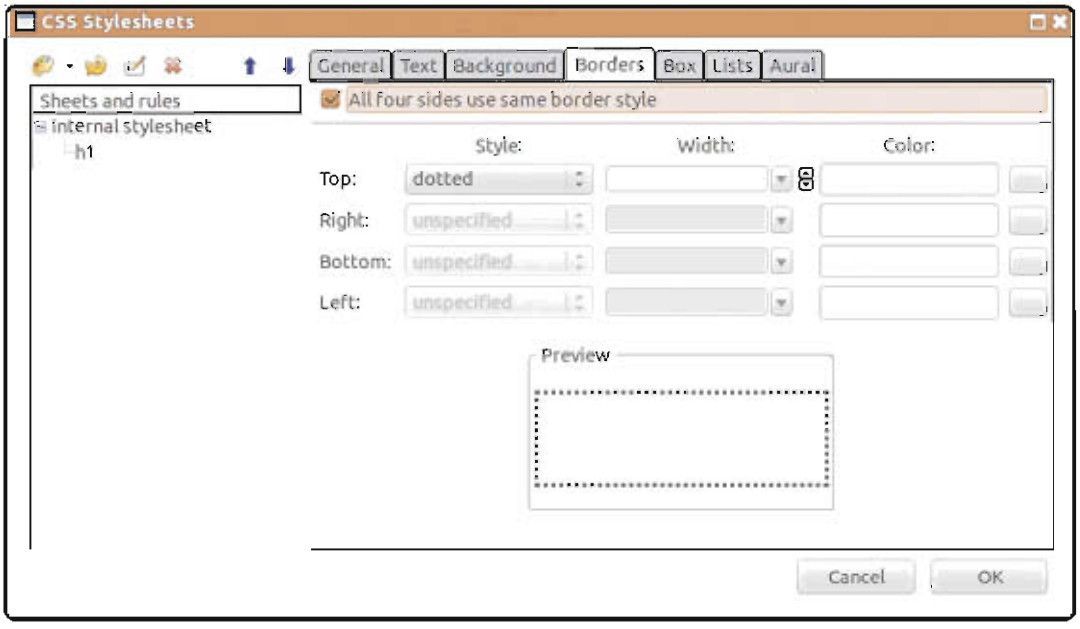


Figure 2.9 : Options selected in Border tab for h1

There are more options available if you select the other tabs in the window. But in our example, we require only the selected options for applying style to h1.

- Click on the General tab. This will show the CSS code which is generated as shown in figure 2.10. If you know HTML coding you can also edit the code.

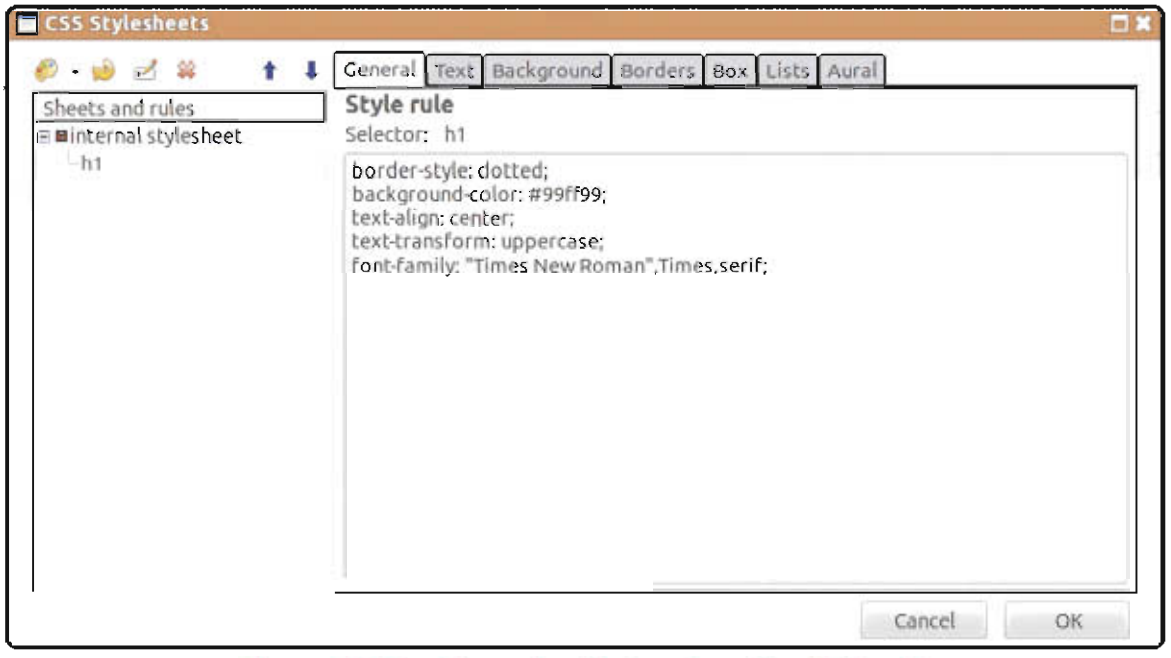


Figure 2.10 : General tab showing the CSS code

Thus, we have created the style to be applied to h1 tag. We can use the same procedure to create a style for any other element in the web page. To add other elements, click on the arrow of CSS button in the upper left corner of the CSS Stylesheets dialog box. Choose "style rule" from the drop down menu. After adding the style rule for all the elements click OK button.

Let us now use the CSS created for h1 tag in a web page. In Format toolbar1, select the Heading1 option to insert a heading in the page. When you select the Heading1 option, your cursor will automatically be placed in the center of the web page (as we had applied center alignment in CSS). Type some text. The text will appear in uppercase with font style "Times new roman". Figure 2.11 shows a web page that uses Heading1 text with CSS applied to it. So, now wherever you insert Heading1 in any of the web pages, the style will remain the same for all.

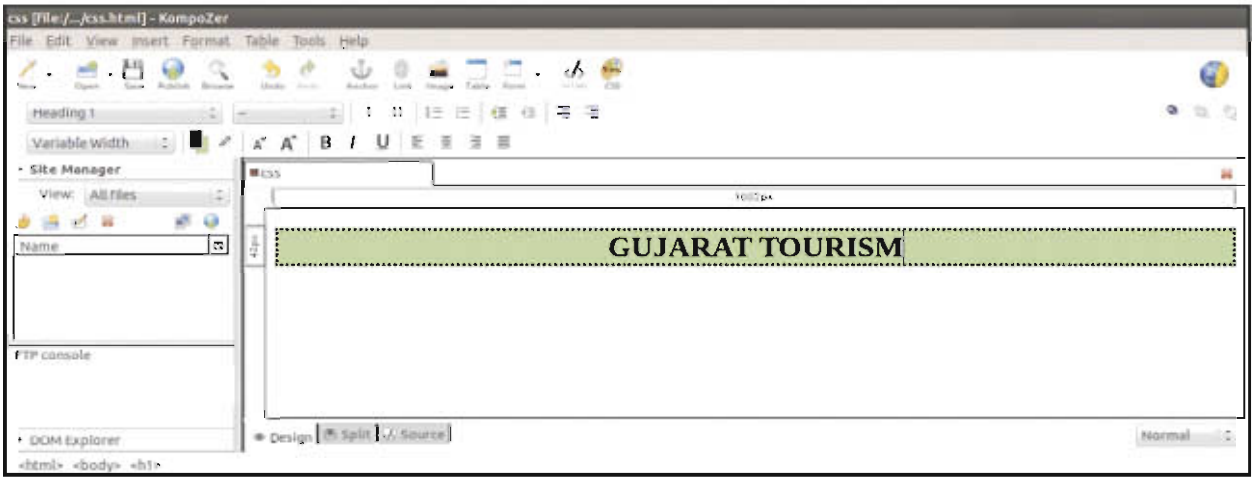


Figure 2.11 : CSS applied to the text in web page

Select the source tab at the bottom of the window to view the CSS stylesheet code. Figure 2.12 shows the CSS code in the head section of the page source code.

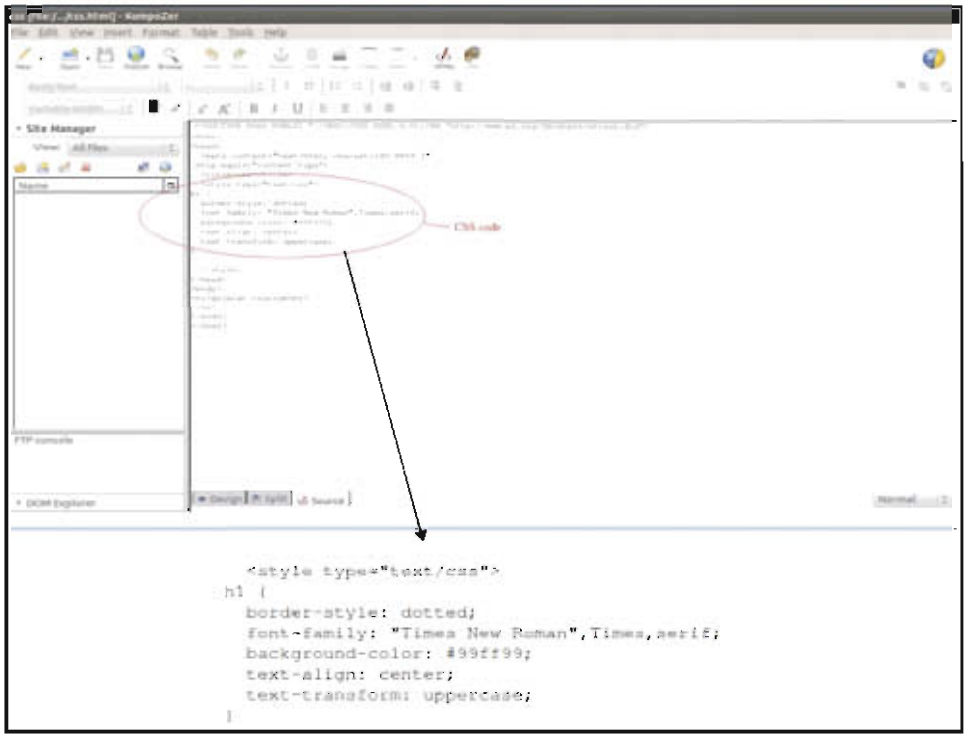


Figure 2.12 : CSS code in the Source view

Advantages of CSS

From the above discussion we can say that CSS allows us to set the format of the website without changing the underlying structure. By separating the layout and format properties of the website from its underlying logical structure, we can make changes to the website without the fear of accidently changing the data.

The web designer's job is made easier as to change the style of an element he only needs to make modifications in the CSS file. Since we have to set the style for each element only once, the CSS has less code compared to that in HTML. Thus the web pages will load faster. Using CSS makes website designing quick and efficient.

Disadvantages of CSS

The CSS compatibility varies with different browsers. This means that some of the style sheet features are not supported by the browser and the style cannot be displayed as per the users design. But now-a-days, the latest browser versions are more standard-compliant and the compatibility issue has reduced.

JavaScript

HTML was originally used to control the appearance of web pages. The web pages designed using HTML, were static and could not be changed after the browser rendered them. However, with the growth of Internet, people demanded the websites to have greater interactivity and better visual design. But, HTML could only provide static web pages. Thus, the demand of more interactivity created the need for a new web programming language. Hence, Netscape developed JavaScript. JavaScript is a scripting language that allows you to add programming aspects to your web pages.

A scripting language is a simple, lightweight programming language that does not contain the advanced programming functionalities of languages like C and Java. JavaScript is used in web pages to improve the design and validate the forms. It adds interactivity to HTML pages and is inserted directly into the HTML code. Today, mostly all the web browsers like Mozilla Firefox, Chrome, Safari and Internet Explorer support JavaScript. Using JavaScript, a web page no longer remains static, but can include coding that allows interactivity with the user, control the browser, and dynamically create HTML content.

We learnt how to create forms using KompoZer. When the user enters data in the form, he might leave some of the important fields empty or may enter the data in a wrong format inside the field. In such cases there must be some type of validation provided. This validation will restrict users from making mistakes. When the user enters wrong data or leaves a field empty, an error message should be generated and the form should not be submitted.

The most common form of JavaScript application today is client side script which runs inside a web browser. It is used to validate the data entered in the HTML forms on the client side before sending it to the server. Using JavaScript, the form is generally checked for the following things:

- Has the user left any required field empty ?
- Has the user entered a valid E-mail address ?

- Check whether contents of two fields are same or not ?
- Has the user entered a valid date ?
- Has the user entered text in the numeric data field ? Say for example; in the quantity field instead of numeric data, user entered text.

Before starting to learn about how to validate a form, let us first learn how to write a JavaScript code.

JavaScript code runs from within the HTML web page. Thus, the JavaScript code can be put directly inside the web page code as a separate section. The JavaScript code is inserted into an HTML page, using the <script>… </script> tag. The lines between the <script> and </script> tag contains the JavaScript code. Remember, JavaScript is a case sensitive language.

JavaScript can be placed inside the <body> or <head> section of an HTML page. But generally it is preferred that the code be placed in the <head> tag. The <script> tag tells the browser to interpret all the text between the <script> tag as a script. Let us write a simple JavaScript code using the source view in KompoZer. Figure 2.13 shows the source view with the JavaScript code. Save the file and open it in the browser.

```
 1. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
 2. <html>
 3. <head>
 4.   <meta content="text/html; charset=ISO-8859-1"
 5.  http-equiv="content-type">
 6.   <title>js1</title>
 7.   <script>
 8. document.write("HELLO STUDENTS");
 9.   </script>
10. </head>
11. <body>
12. <br>
13. </body>
14. </html>
```

Figure 2.13 : JavaScript in source view

You can see that inside the <script> tag we have written a single statement "document.write("HELLO STUDENTS")". Here write is a method which is a part of the document object. The string to be displayed is passed as variable to the method document.write().

The semicolon seen in figure 2.13 at the end of the statement is optional in JavaScript. The semicolon is used as a mark of separation rather than termination. Thus, if you place each statement on a new line then you need not put a semicolon. In our example, even if we remove the semicolon after the document.write statement, the result would be same. However, putting a semicolon is a good programming practice. The output of the code as seen in the browser is shown in figure 2.14.

Figure 2.14 : Output in Browser

Note : JavaScript statements can be grouped together in a block. A block starts and ends with the curly bracket.

Let us now learn how to write a simple JavaScript for form validation. First, we will create a simple form with two fields: a name field and submit button. For form validation, if the user leaves the name field empty and clicks the submit button then a message should be displayed that asks the user to enter the name. Follow the steps given to perform the operation.

- Create a new file and save it. In the example we have saved the file with name js1.html.
- Create a new form using **Form → Define Form.** Give a name to the form. In our example, we have named the form as js1. As shown in figure 2.15, insert a label and input text field for the displaying name, also insert a Submit button.

  Note : The names given to the form, input field or any other fields created in the form should be kept simple and easy to remember, as they will be referenced in the script frequently.

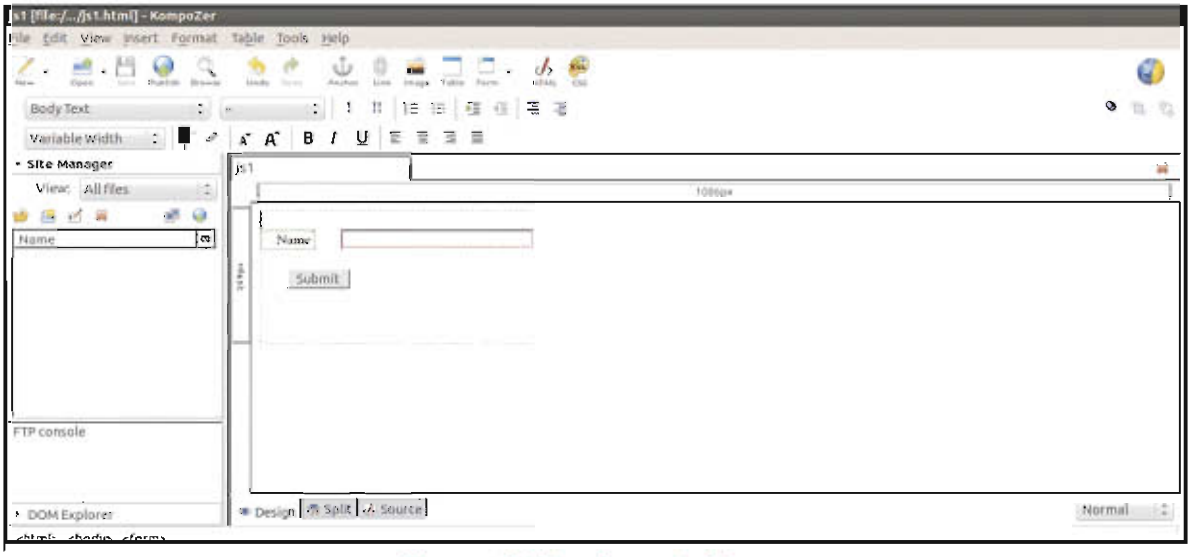


Figure 2.15 : Sample Form

- Go to **Form → Define Form.** This will open the Form Properties dialog box as shown in figure 2.16.

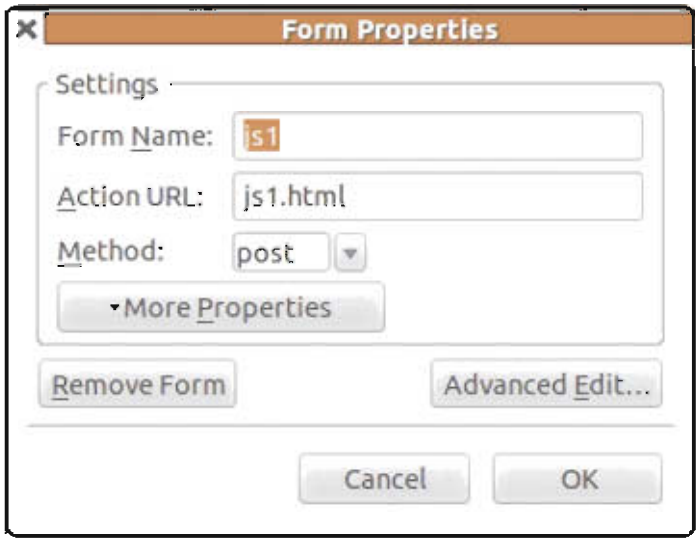


Figure 2.16 : Form Properties dialog box

- Click on Advanced Edit button. This will open Advanced Property Editor dialog box as shown in figure 2.17. Select the JavaScript Events tab. In the Attribute option select "onsubmit" and in the Value write "return validateForm()" as shown in figure 2.17. Press OK button.

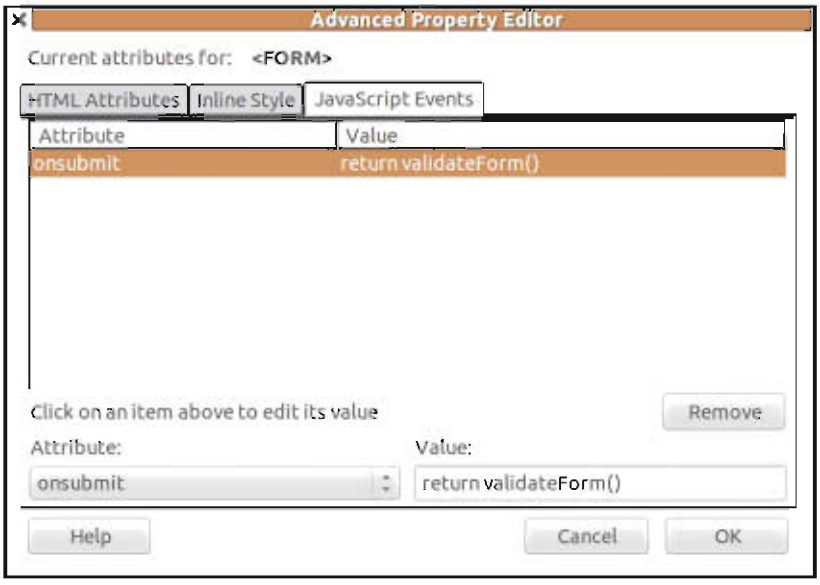


Figure 2.17 : Advanced Property Editor dialog box

- Now, open the source view. Add the script inside the <head> tag as shown in figure 2.18.

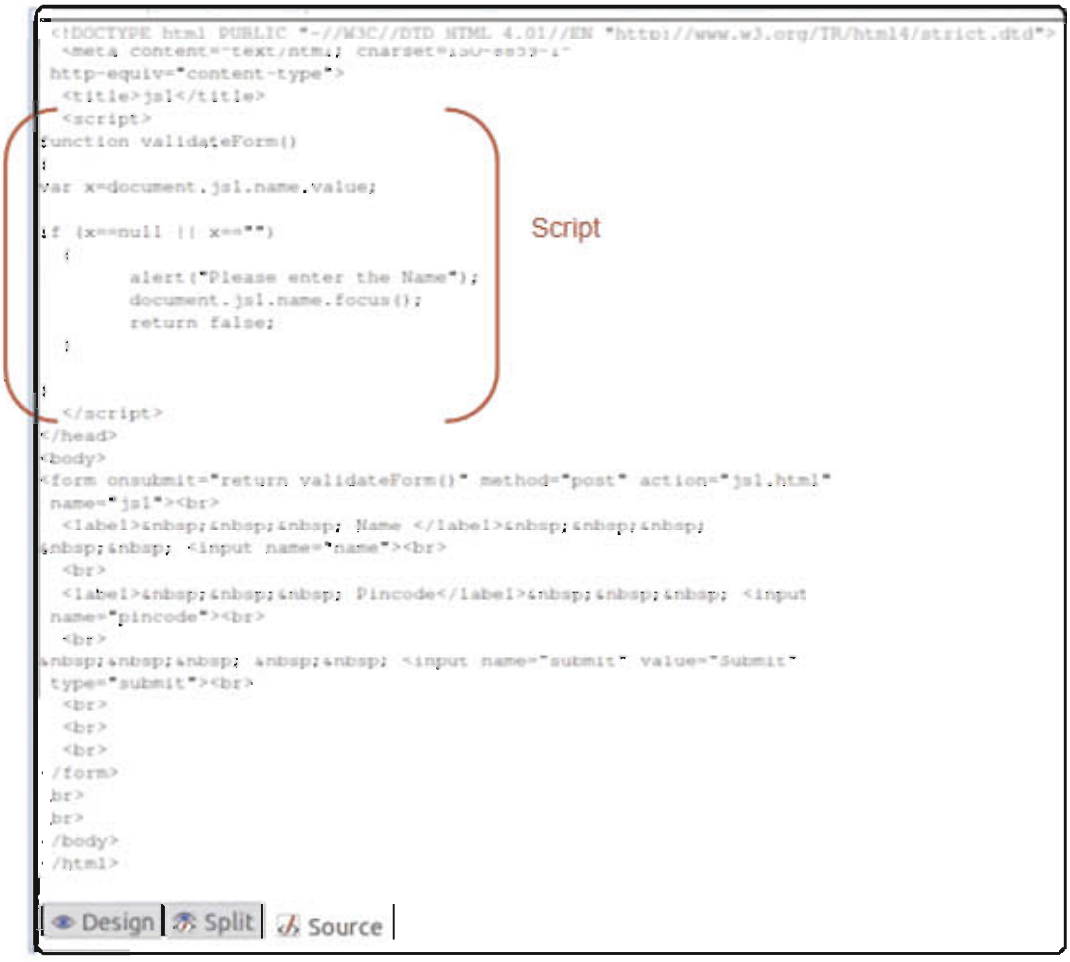


Figure 2.18 : JavaScript in source view

As seen in the script tag of figure 2.18, we have defined :

- a function validateForm()
- a variable x
- a conditional statement if
- built-in function alert()
- an object named document
- a method named focus

Let us discuss each of them in brief.

Function validateForm()

A function (also called routine) is a reusable block of code that performs a particular task. A function is defined by the keyword function and the block of code is written inside the curly braces "{ }". Here, validateForm() is a function.

JavaScript provides some inbuilt functions such as alert(). The alert() function takes the text and displays it in an alert box.

For example,

```
function hello()
{
 alert("Hello Students");
}
```

Here the function "hello()" when called, will display an alert box with the message "Hello Students".

The function will be executed by an event or when it is called inside a source code. A function can be called anywhere from within the source code. We can define the function in the head or the body section of the HTML code. A function is called by its name. It can also take a value inside the parenthesis. This value is passed as a parameter to the function. Sometimes you want the function to return a value. This is possible by using the return statement. The use of return statement, will stop the function from executing, and return the specified value.

In our example, the function validateForm() is called on the click event of the submit button. Thus, when the user clicks the submit button, which is known as an event, the function is loaded and the statements inside the function will be executed. The function returns a false value.

Events

JavaScript is useful in creating interactive web pages which responds to the action performed by the user. The interaction between the user and the web page causes the browser to generate an event. In other words, when the user does something an event takes place. Table 2.1 gives a list of JavaScript events.

Event	Description
abort	Loading of image is cancelled
blur	Element such as a radio button becomes inactive
click	User clicks on a form element
change	Value of a form field is changed by the user
error	Error occurs during loading of a document or image
focus	Element such as button becomes active
load	Document or image is loaded
mouseout	Mouse moves off the element
mouseover	Mouse moves over the element
reset	Form fields are reset to default values
select	User selects a form field
submit	User submits a form
unload	User leaves a page

Table 2.1 : JavaScript Events

When an event occurs, a specific JavaScript code is executed in response to a given situation. This JavaScript code is known as event handler. Event handler names are kept the same as the name of the event. For example, the click event will have an event handler as onclick(). Likewise, submit event will have an event handler as onsubmit().

In our example, we want the validation to be done when the user clicks the submit button i.e. on the "submit" event. So, we need to use the onsubmit() event handler. Therefore, in figure 2.18 we had added an event handler onsubmit(). This event handler will call the function validateForm(). The function will return a value either True or False depending on whether the name field has a value or null (no value).

Variable

In our example, inside the function validateForm() we have a statement

```
var x=document.js1.name.value
```

We have declared a variable named x. A variable is a container for storing data. Remember, that the variables in JavaScript are also case sensitive. So, a variable "x" is not the same as variable "X". The variables can store numbers, strings or text. In JavaScript, we can declare a variable using var keyword. For example,

```
var x=3;
var y="Hello";
var z="Hello Students";
```

If statement

You must be familiar with conditional statement if. The conditional statement if is used to change the flow of the program. The if statement evaluates an expression to validate specific condition. If the condition is true, then the program enters if block and executes the statements inside it. In our example, we have used the if statement to check whether the variable x is having null value or not. If the variable x is having null value then the statements inside the if block will be executed.

Document Object Model (document.js1.name.value)

Sometimes, you may want to use JavaScript to control the web browser. For example, you may want to change the web page or control the elements of the web page. To control the web browsers window or the web page we use the Browser Object Model (BOM).

All browsers are split into different parts or objects that can be accessed using JavaScript. These parts are known as the Browser Object Model, or BOM. On the top of this browser hierarchy is the Window object as shown in figure 2.19.

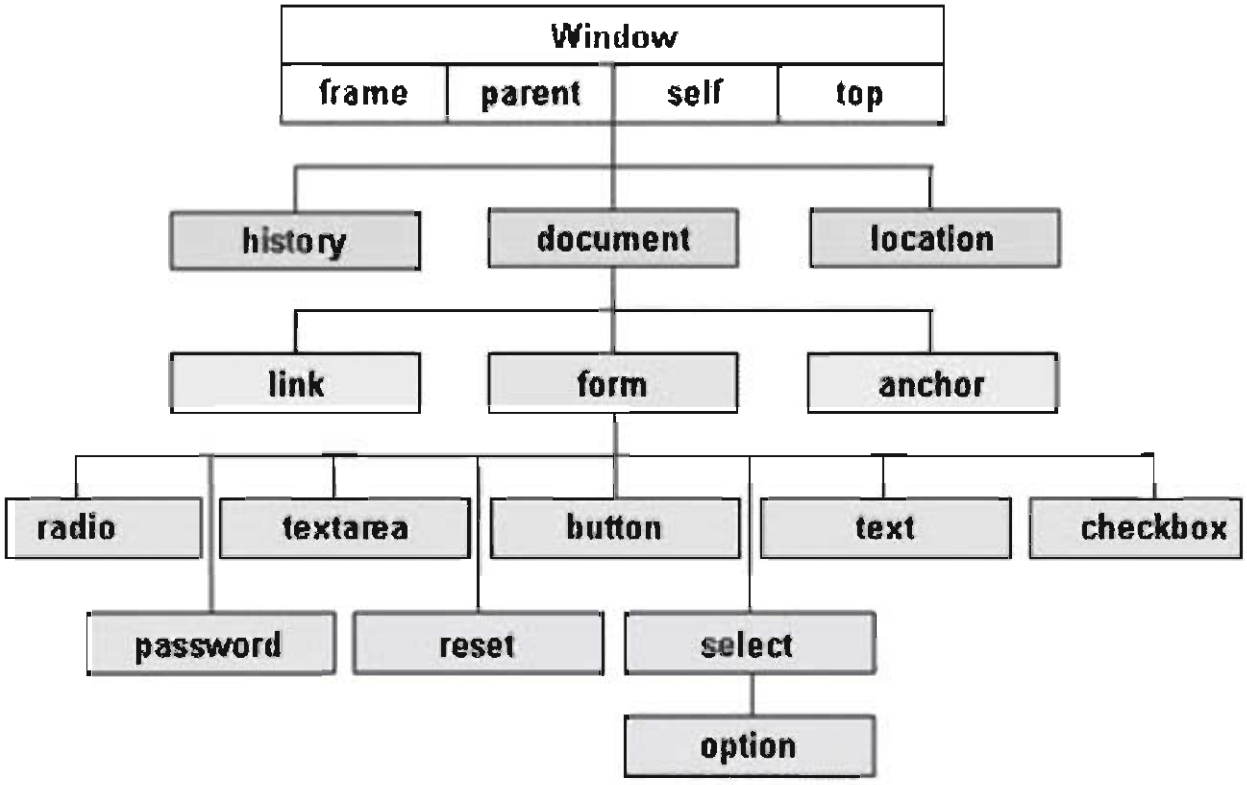


Figure 2.19 : Hierarchy of browser object

It represents the hierarchy of entire browser along with the toolbar, menu, status bar, web page and many more. We can use the properties and the methods of objects in the browser object model to change the window or elements displayed in the browser. We do not have to create an object in a browser object model; they are created automatically when the browser opens the web page.

The top level object in the browser object model is Window object. Window object represents the browser window or individual frame within the window. It is created automatically by the browser. The window object is the global object as it contains all the other objects of the browser object model within it. For example, window object contains the document object. We use the methods and properties of the window object to control the web browser window, while the methods and properties of document object are used to control the web page.

The document object is the most important object in the browser object model. It is used to represent the web page displayed in the browser. All the elements of the web page like forms, images, links and others are contained within the document object. For example, the form object which is used by JavaScript to represent forms created using the <form> element is within the document object. Just as the document object contains a form object, similarly the form object contains element object. The element object is used to reference each element in a form. The element objects can be radio, text, checkbox or any other object.

The focus Method

In the script, we have also used a method named focus. It is used to apply focus on a particular element of the form. Thus, after the user gets an alert message, calling the focus method, the cursor will be placed inside the element and the particular element will be highlighted. Here the name element will be focussed.

Let us now open the file in the browser. The output as shown in figure 2.20 will be visible to us.

Figure 2.20 : Output in Browser

Keep the Name field empty and press the Submit button. You will see an alert message as shown in figure 2.21 appear on the screen.

Figure 2.21 : Alert message for the empty name field

The alert message is the result of the script that we had written in the source view. This is the

validation that we have provided in the form using JavaScript. Thus, a form having more fields can be validated to check if the user has kept any of the fields empty.

You must have observed while filling forms on Internet, some fields are marked with a red color asterisk mark "*". This is an indication to the user that it is a compulsory field which should not be kept empty. Using JavaScript, we can check the entry in such fields and provide the alert message to the user.

Let us now add another field in the form named "Pincode". The validations which we will apply on this field are :

- User should not keep the field empty.
- Only numbers are allowed (characters are not allowed).
- The Pincode should be 6 digits long.

If any of the validation is violated, the user will be displaced an alert message. Follow the steps given to create such a validation.

- In the same form insert a label and a input field for Pincode.
- In the source view, add the script as shown in figure 2.22.

```
<script>
function validateForm()
{
 var x=document.js1.name.value;
 var y=document.js1.pincode.value;
 if (x==null || x==“”)
 {
  alert(“Please enter the Name”);
  document.js1.name.focus();
  return false;
 }

 if (y==“” || isNaN(y) || y.length>6 || y.length<6)
 {
  alert(“Please enter the Pincode properly”);
  document.js1.pincode.focus();
  return false;
 }
}
</script>
```

Figure 2.22 : JavaScript for validating name and Pincode

As seen in the script, we have declared another variable named y. The variable y will hold the value of the Pincode field. In the if condition, we have used a function named isNaN(). Let us understand this function.

isNaN()

To work with numeric values, JavaScript uses built-in functions. One of most commonly used numeric function is isNaN(). NaN stands for "Not a Number". The function isNaN(value) determines whether the value is NaN (Not-a-Number). It returns true if the value is not-a-number and false if the value is a number. For example,

isNaN(123) returns false as the value "123" is a number

isNaN("hello") returns true as the value "hello" is not a number

Thus, using if statement we check if y is; empty, not a number or not equal to six digits. If any of the condition is violated then the alert message will be displayed to the user.

Open the file in the browser. The output will be as shown in figure 2.23. Enter the name in the name field (Do not keep the name field empty, as this will again generate the alert message for it).

Figure 2.23 : Output in the Browser

Keep the Pincode field empty and press the Submit button. This will show an alert message as shown in figure 2.24.

Figure 2.24 : Alert message for empty field

Now, enter some characters or numeric value less than 6 digits length in the Pincode field. This will generate the same alert message as shown in figure 2.24.

Here, we have generated the same alert message for all the three validations in the Pincode field. If you want that the alert message should be displayed appropriately as per the validation, then we need to modify the script. In the example, we used a single if condition for all the three validations and gave the alert message. Now, to display the alert message as per the validation, we need to add three if conditions. Figure 2.25 shows the script with three if conditions.

```
<script>
function validateForm()
{
 var x=document.js1.name.value;
 var y=document.js1.pincode.value;
 if (x==null || x==="")
 {
  alert("Please enter the Name");
  document.js1.name.focus();
  return false;
 }
if (y==="")
 {
  alert("please enter the pincode properly");
  document.js1.pincode.focus();
  return false;
 }
if(isNaN(y))
{
 alert("please enter a number");
 document.js1.pincode.focus();
 return false;
}
if(y.length>6 || y.length<6)
{
 alert("please enter a six digits");
 document.js1.pincode.focus();
 return false;
}
}
</script>
```

Figure 2.25 : JavaScript to display appropriate message

Open the file in the browser. Enter some characters in the Pincode field and press the submit button. This will show an alert message as shown in figure 2.26.

Figure 2.26 : Alert message to check numeric entry

Enter some numeric value less than 6 digits length in the Pincode field and press the submit button. This will show an alert message as shown in figure 2.27.

Figure 2.27 : Alert message to check length

Thus, you learnt how using JavaScript we have added interactivity to our web pages. Now, let us take the "Registration Form" that we had created in the Chapter 1 and add JavaScript to it.

To apply validations, we will make some simple changes in the form that are listed below :

- Gender and Hobby field will not be kept initially selected.
- Address field has no initial text.
- City field has an option "choose the city" whose value is -1. All the other options in the city field like Ahmedabad, Rajkot, Surat have values 1, 2, 3 respectively.

To make the above changes, Open the form and select the respective field in the form. Go to **Form → Form Field.** This will show the properties of the respective field and make the changes. The form after making the changes will look as shown in figure 2.28.

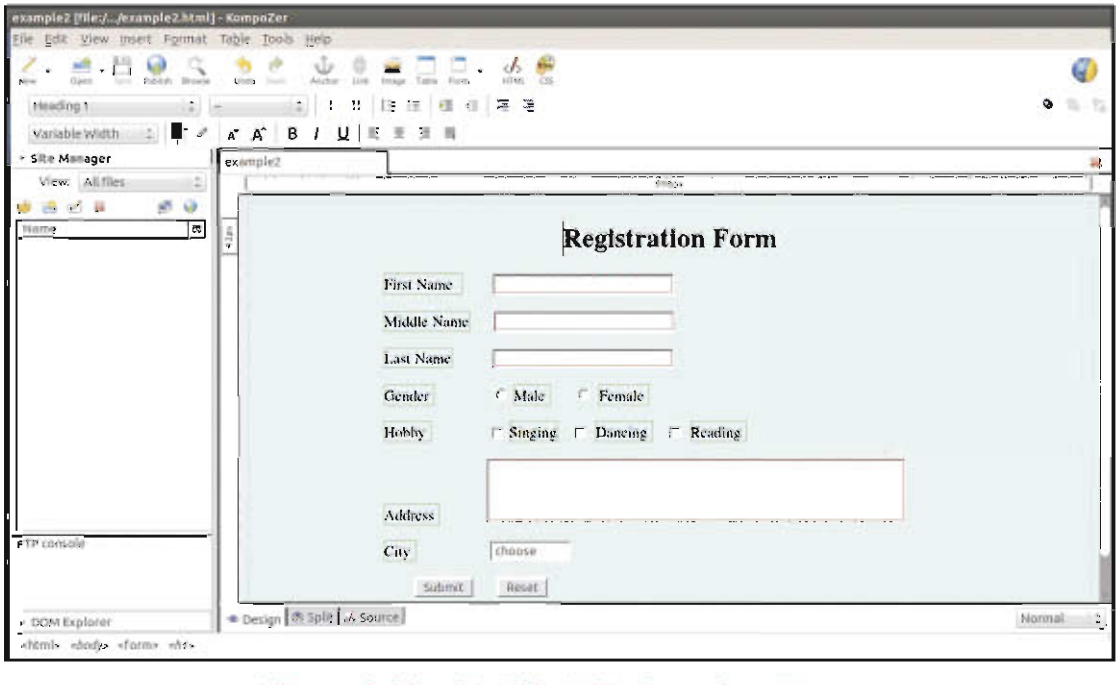


Figure 2.28 : Modified Registration Form

- Go to **Form → Define Form.** Alternatively, you can right click in the form and select properties option. This will open the Form Properties dialog box as shown in figure 2.29.

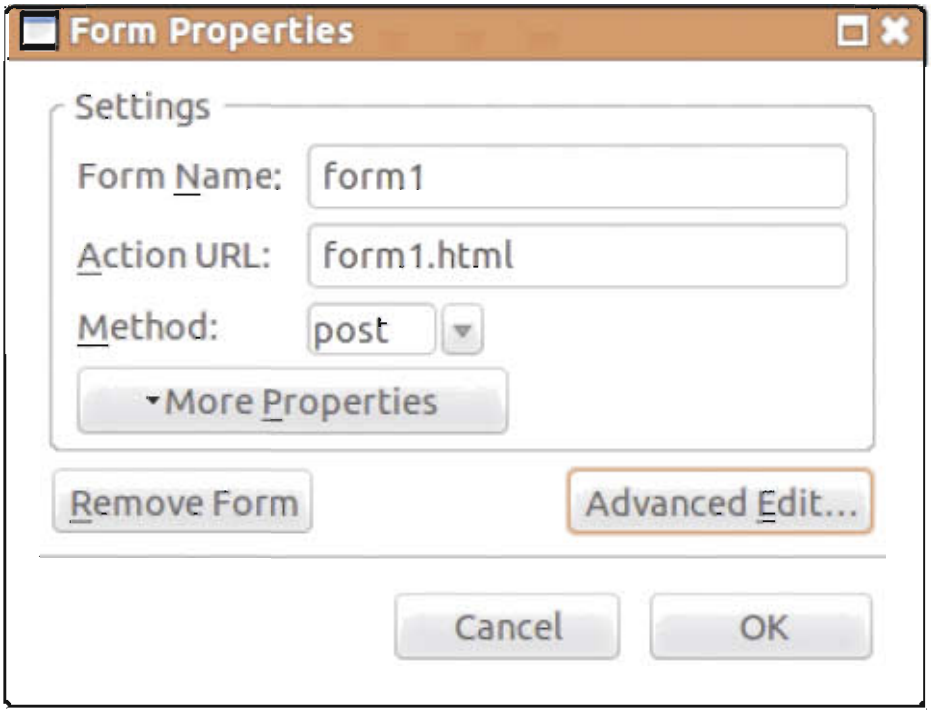


Figure 2.29 : Form Properties dialog box

- Click on Advanced Edit button. This will open Advanced Property Editor dialog box as shown in figure 2.30. Select the JavaScript Events tab. In the Attribute option select "onsubmit" and under the Value option write "return validateForm(). Press OK button.

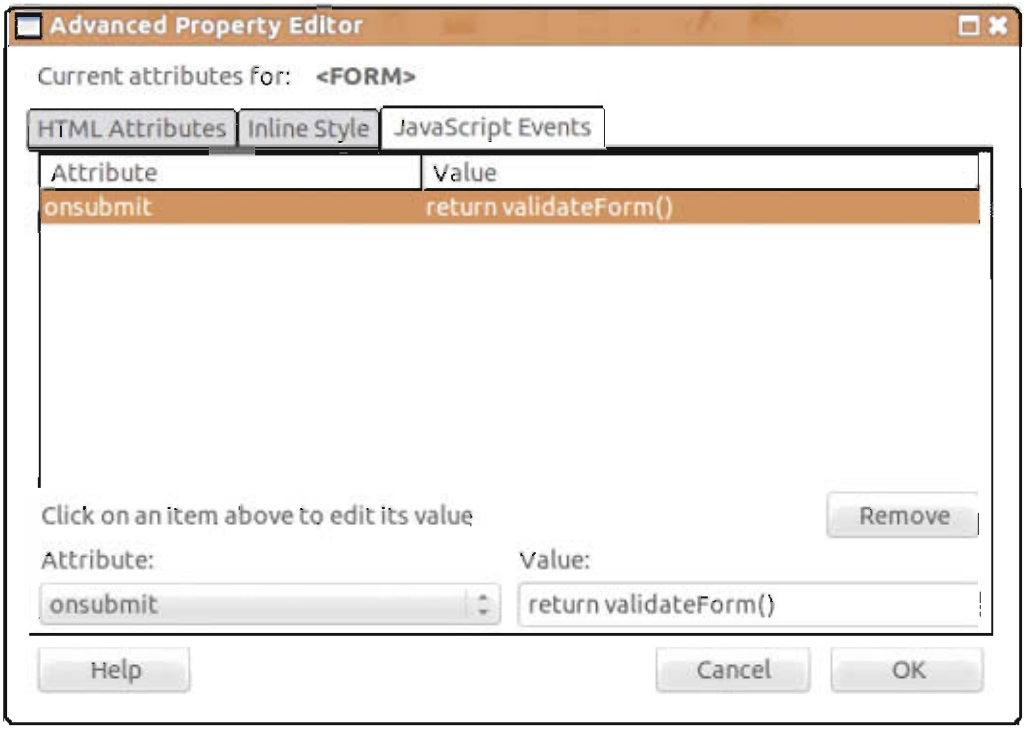


Figure 2.30 : Advanced Property Editor

- Add the script shown in figure 2.31 inside the <head> section as of the HTML.

```
<script>
function validateForm()
{
var x=document.form1.firstname.value;
var y=document.form1.middlename.value;
var z=document.form1.lname.value;
var r=document.form1.address.value;
if (x==null || x==="")
{
alert("Please enter the first name properly");
document.form1.firstname.focus();
return false;
}

if (y==""||y==null)
{
alert("Please enter the middle name properly");
document.form1.middlename.focus();
return false;
}
if(z==""||z==null)
{
alert("Please enter a last name properly");
document.form1.lname.focus();
return false;
}
if(r==="")
{
alert("Please enter address");
document.form1.address.focus();
return false;
}
if (( document.form1.gender[0].checked == false ) && ( document.form1.gender[1].checked ==
false ) )
{
alert ( "Please choose your Gender: Male or Female" );
document.form1.gender[0].focus();
return false;
}
if ((document.form1.hobby[0].checked == false) && (document.form1.hobby[1].checked ==
false ) && (document.form1.hobby[2].checked == false))
{
alert ( "Please choose a hobby" );
document.form1.hobby[0].focus();
return false;
}
if( document.form1.city.value == "-1" )
{
alert( "Please provide your city!" );
document.form1.city.focus();
return false;
}
}
</script>
```

Figure 2.31 : Validation JavaScript

As seen in the script, we have initialised four variables x, y, z and r.

In the statement var x=document.form1.firstname.value. The term form1 refers to the form name; term firstname refers to the element name (the name given to the input field "first name"). Thus,

- variable x stores the value of first name
- variable y stores the value of middle name
- variable z stores the value of last name
- variable r stores the value of address

In the script, using the if condition,

- variable x checks, if the first name is empty or not.
- variable y checks, if the middle name is empty or not.
- variable z checks, if the last name is empty or not.
- variable r checks, if the address is empty or not.

The radio buttons for gender field are grouped by the name "gender". Thus, gender is an array with two elements. The first element of the gender array is checked using the if condition with the statement document.form1.gender[0].checked and the second element of the array is checked using the statement document.form1.gender[1].checked. If both of them are false then no option is selected by the user. Thus, alert message is displayed and the focus is placed on radio button.

Similarly, the check boxes for the hobby field are grouped by the name "hobby". Thus, hobby is an array with three elements. Just as we checked the gender using the if condition, we check the hobby and if all the three elements are false then no option is selected and the user is displayed the alert message to select a hobby and the focus is placed on the check box.

For the city field, we check if the value is equal to -1 (Note: -1 is kept as the value for option "choose the city"), in which case no option is selected and the user is given an alert message. Figure 2.32 to 2.35 shows some of the alert messages displayed after the fields are left empty.

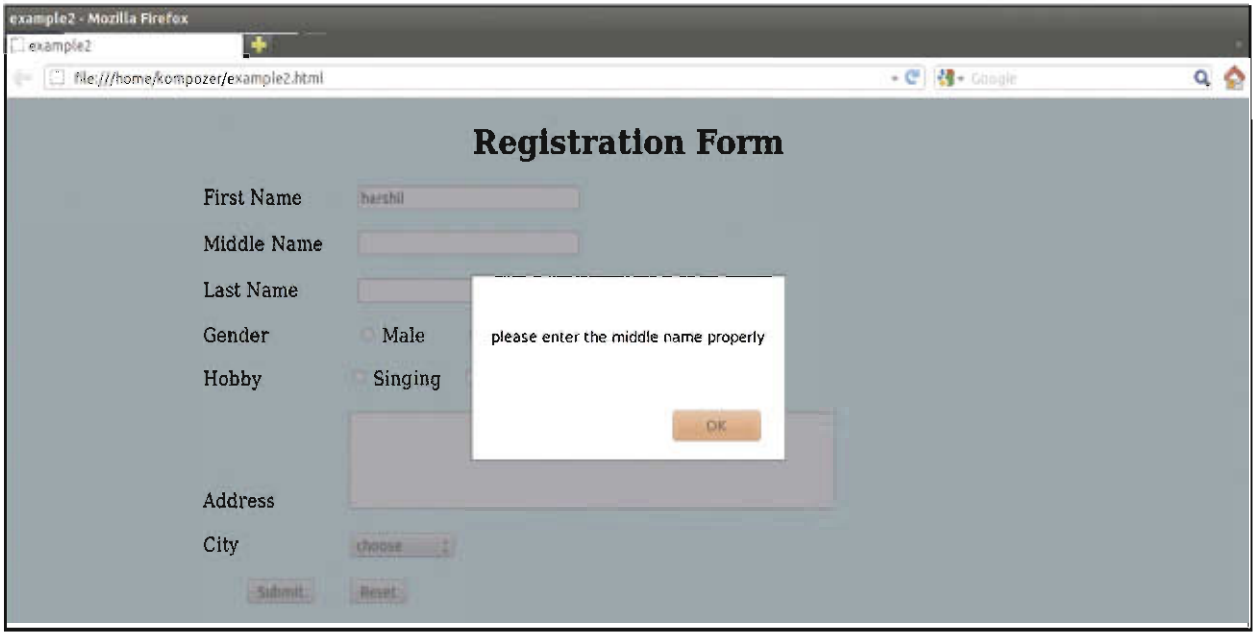


Figure 2.32 : Alert message when middle name left empty

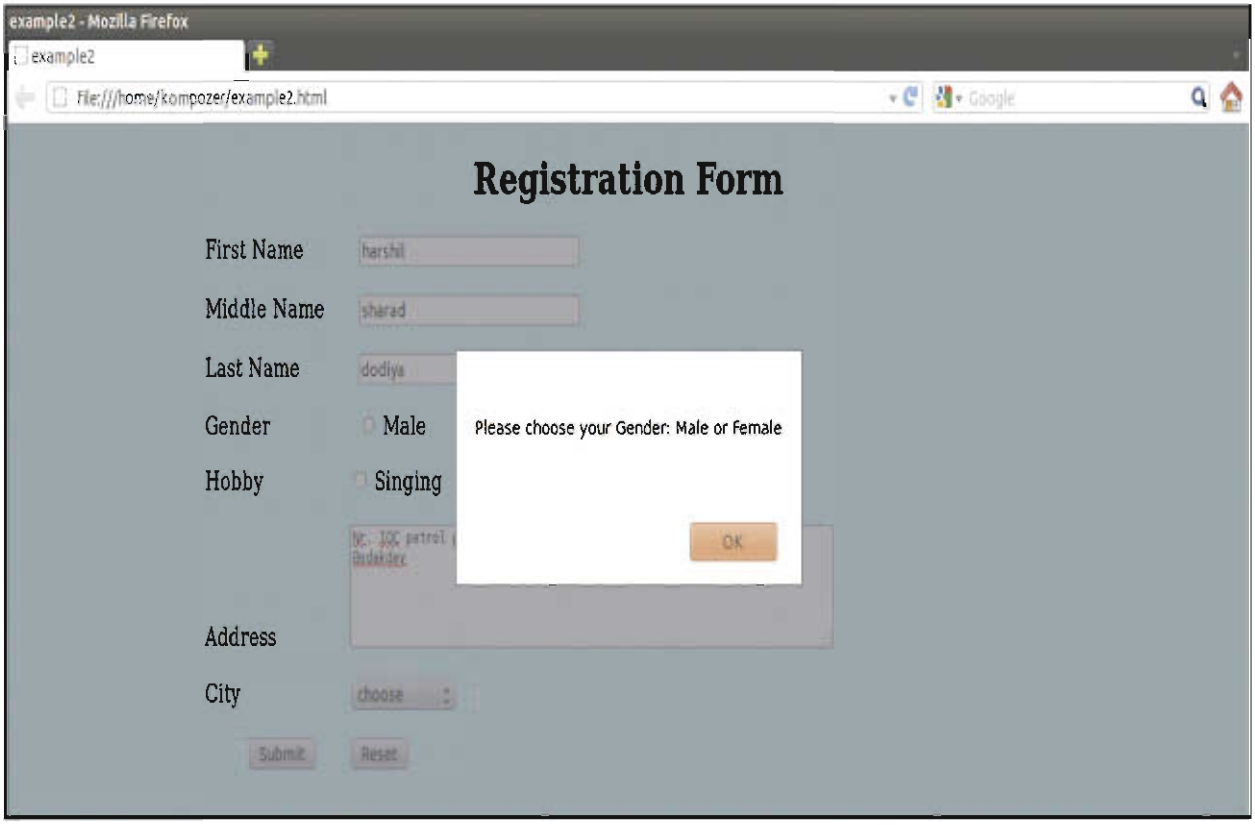


Figure 2.33 : Alert message when gender left empty

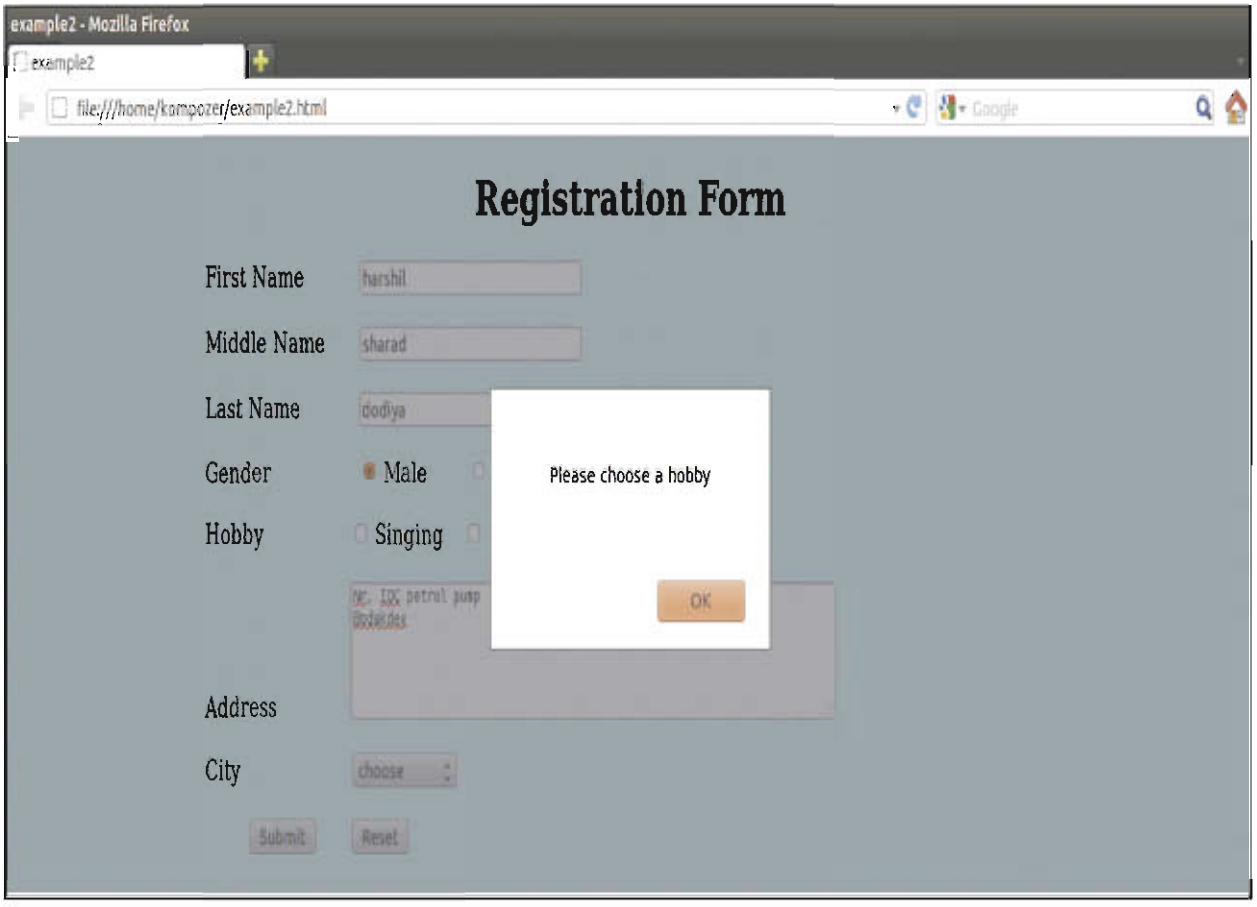


Figure 2.34 : Alert message when hobby left empty

Figure 2.35 shows the final form with all the fields filled as seen in the browser.

Figure 2.35 : Final form

Thus, we learnt how to use JavaScript for form validation and make the web pages more interactive.

Summary

In this chapter we learnt about Cascading Style Sheets and JavaScript. CSS allows you to specify styles for the visual elements of the website. It helps us to keep the information content of a document separate from the details of how to display it. We discussed about the advantages and disadvantages of CSS. JavaScript is a scripting language that allows us to add logical aspects to our web pages. A scripting language is a simple, lightweight programming language that does not contain the advanced programming functionalities. It is used in web pages to improve the design and validate forms. It adds interactivity to HTML pages and is inserted directly into the HTML code. The JavaScript code is inserted into an HTML page, using the <script>… </script> tag.

EXERCISE

1. State the purpose of Cascading style sheets.
2. Why should we keep the style and the content of a webpage separate ?

3. Explain the syntax of CSS.

4. State the advantages of CSS.

5. List the disadvantages of CSS.

6. Why do we use JavaScript in HTML pages ?

7. How does an HTML page identify the JavaScript code ? Give an example.

8. Choose the most appropriate option from those given below :

(1) Which of the following allows specifying styles for the visual elements of the website ?

(a) Cascading Style Sheets (b) Webpage

(c) Form (d) Animation

(2) Which of the following is known as special symbol in the syntax of CSS ?

(a) Rules (b) Selector (c) Declaration (d) Input

(3) Which of the following are two main parts of CSS rule ?

(a) Selector, declaration (b) Select, declaration

(c) Selector, declare (d) Selection, declaration

(4) Which of the following is an HTML element on which style can be applied ?

(a) declaration (b) selector (c) select (d) declare

(5) Which of the following is the syntax of CSS ?

(a) select {property : value} (b) selector {value : property}

(c) selector {property : value} (d) selection {property : value}

(6) Which of the following has developed JavaScript ?

(a) Yahoo (b) Google (c) Wikipedia (d) Netscape

(7) Which of the following is a scripting language that allows adding programming to web pages ?

(a) Action script (b) JavaScript (c) HTML (d) CSS

(8) Which of the following is a scripting language that is simple, lightweight programming language that does not contain advanced programming functionalities ?

(a) JavaScript (b) HTML (c) C (d) Java

(9) Which of the following tag is used to insert JavaScript code into an HTML page ?

(a) <script>... <script> (b) <script>... </script>

(c) <script>... <//script> (d) </script>... </script>

(10) Which of the following symbol signifies the start and end of a JavaScript block ?

(a) semicolon (b) square bracket

(c) curly bracket (d) round bracket

(11) Which of the following is a reusable block of code that performs a particular task ?

(a) Array (b) Code (c) Program (d) Function

(12) Which of the following statement is used to return a value in a function ?

(a) return (b) function (c) select (d) send

(13) Which of the following is generated by the browser due to interaction between the user and the web page ?

(a) Function (b) Response (c) Event (d) Value

(14) Which of the following is not an event ?

(a) Abort (b) Mouseover (c) Set (d) Load

(15) Which of the following is a container for storing data ?

(a) Variable (b) Integer (c) Event (d) Event handler

(16) Which of the following stands for BOM ?

(a) Browser Of Model (b) Browser Object Model

(c) Browser Object Modelling (d) Browse Object Model

(17) Which of the following is the top level object in the browser object model ?

(a) Window (b) Document (c) Page (d) Location

(18) Which of the following stands for NaN ?

(a) Not a Numeric (b) Not a Number

(c) Not a Noun (d) Not an Numeric

LABORATORY EXERCISE

1. Create a CSS for H1 as per the given rules and apply it on the text.

- Font: Times New Roman
- Color: red

- Case: Lowercase
- Font style: Italic
- Alignment: Centre
- Text decoration: Underline
- Background color : Light Grey
- Border: Dotted

2. Create a form with fields: name, email address, phone number and submit button. Give validation as given below :
   - Fields should not be empty.
   - Only numbers are allowed in the phone field.
   - The Phone number should be 10 digits long.
3. Give validation to the student's personal details form created in chapter 1.
4. Give validation to the feedback form created in chapter 1.

Designing simple website using KompoZer 3

A website plays a very important role in a business now-a-days. It helps in presenting the business to the world. It helps in promoting the business, selling the products and attracting a large number of customers. Therefore it is important that the website should have good design techniques which will enable the company to get the maximum amount of traffic, therefore making the maximum amount of profit.

A website is a collection of interlinked web pages for a specific purpose. The challenge however is in creating a website that is interactive, user friendly and providing accurate and useful information. The website should be designed in such a way that the users find the website informative and visit it repeatedly. It should increase customer base if the site is commercial. This objective can be achieved if appropriate care is taken at the time of website designing. This chapter discusses the general points to be considered while planning for a website. We will create a website using KompoZer and publish it on Internet.

Planning for the website

Developing a good website is a project similar to any other project and requires detailed planning. The better the planning, the chances of success in terms of usefulness of the website is higher. In other words we can say that, the desired goals for developing the website can be achieved. Following are the important points which should be considered for developing a good website as part of planning process.

Purpose

The purpose of the website should be clearly identified. Before creating a website, we should be clear with the definition and goal of the website. A properly designed website will be helpful to the organization. The purpose for creating the website may be to provide information to a group of people, or it may be to attract the new customers or it may be to sell the products online. Having decided the purpose, the content and layout of the website can be properly developed.

Audience

Before we start with the designing part, we should know the expected users of the website. The website should contain both general and detailed information. It should also contain specific and non-specific data. The proportion of the different type and extent of detailed and specific information depends on the nature of the expected audience. The user expectations from the website are important to know, as it helps to develop the appropriate content and layout of the site. This also ensures that the website is useful and successful.

Another consideration is the speed of connection that the expected users will have. In case, large graphical files are kept on the website, they will take a long time to download. The users will become impatient while waiting for large files and leave the site. Thus, it is desirable that the website should contain minimal required graphic and multimedia files.

Content

The website must contain complete and relevant information. Excessive use of irrelevant information can cause frustration to the users as they are unable to seek the specific data. Also if the information provided is incomplete, the users can leave the website. Placing appropriate and relevant content will satisfy users and sustain their interest in the website.

The website content should be classified into general and detailed categories.

The general content provides an overview of the site, organization, products and services, and other items. Brief descriptive information helps user in determine what he is searching. It directs to the appropriate specific data and informs about other available items at the site.

The detailed content provides users with the detailed information on the site like products and services description. The layout of the site must ensure easy access to the detailed information, and specific information they seek.

Text paragraphs should always be kept at reasonable lengths. If there is a lot of content to be displayed on a webpage, then try to split it into small text blocks, this way visitors will be easily able to pick out the content they need to know. The navigation in the website should also be made as easy as possible. Moving from one page to another should be easy so that the user knows where he is and how to get back to home page.

Ensuring that appropriate information (both general and detailed) is available and easily accessible ensures that site can serve its intended purpose. A web site is a valuable tool for businesses and other organizations when composed of the right material (content). As with other tools, the web site can perform an outstanding job.

Medium

More and more people are using Internet through smart phones and tablets. The website design should scale for all the devices like computers, smart phones, and tablets. Also the website should be designed keeping in mind that it is well displayed in all the popular web browsers like Mozilla Firefox, Chrome, Opera or Internet Explorer.

Creating a simple website using KompoZer

In earlier chapters we discussed how to create web pages using KompoZer and use JavaScript for interactivity. We have also seen the general considerations while designing a website. Now we will see how to design and develop a small E-commerce website in KompoZer.

Let us create a simple Ecommerce website named "School Plaza", which caters to the needs of the school going students. On our website, all the school related items like books, stationary, school bags, water bottles, lunch box, files, folders and other such related items are available and can be easily ordered. We will also create a shopping cart which will show us all the products and the total amount of the items that we have purchased. Figure 3.1 and 3.2 shows the look of the home page and shopping cart of the website that we plan to design.

Figure 3.1 : Homepage of website

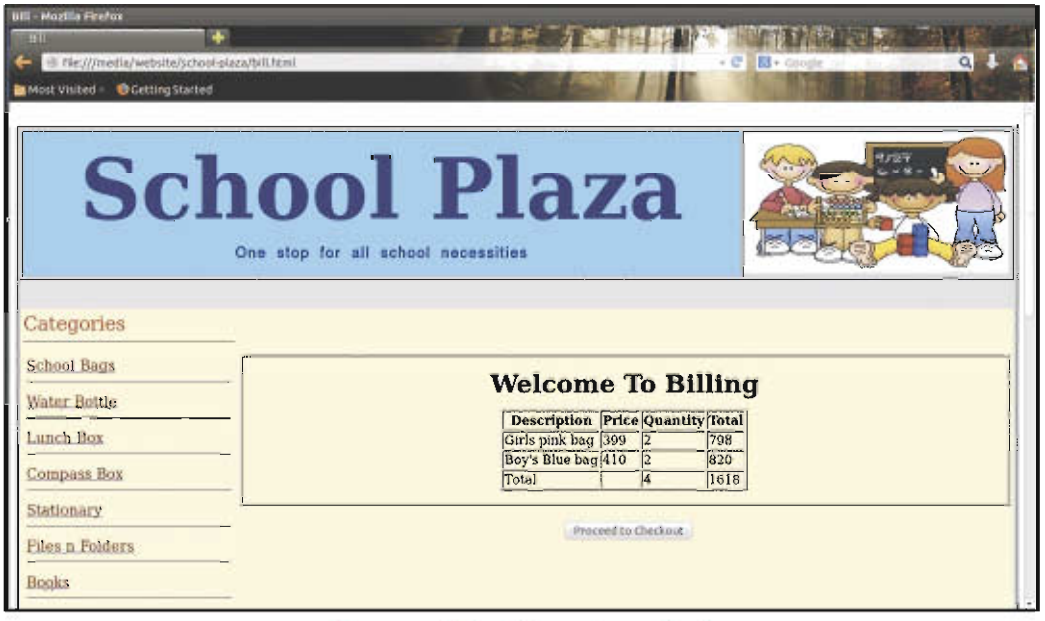


Figure 3.2 : Shopping Cart

Let us learn to create our website step by step. To start with, we should first design the home page of our website. The home page of the website is the first page that opens when the user enters the URL address in the address bar of the browser. The home page of our website should list all the categories of the items that are available. Clicking on any one of the categories will take us to another webpage showing the items of that particular category. To create the website, various designs which are known as templates are freely available on Internet. We can easily download the templates and can customize them as per our needs.

To start with the designing of home page, open KompoZer and create a new webpage. To organize all the contents and the images related to the theme we will insert a table with single cell as shown in the figure 3.3. The table will contain all the text and the images related to the website. To insert the table go to **Insert → Table.** Select single cell table and press OK button.

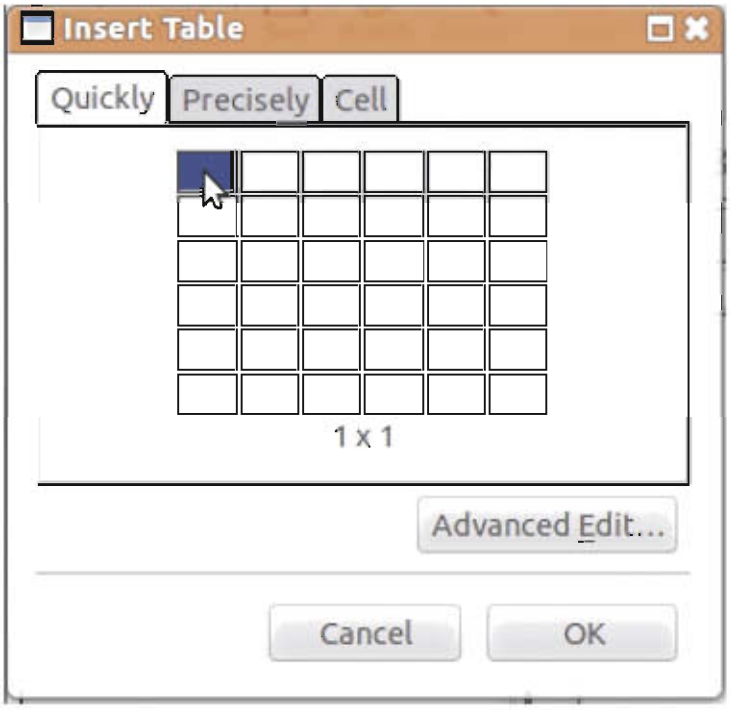


Figure 3.3: Insert 1×1 Table

Save the webpage with a suitable title. The title will be displayed on the title bar of the web browser. Here, we have given the title as "School Plaza". Save the file with the name "index.html". Generally, whenever we create the home page of the website it is named as index.html. Every website is built inside directories on a web server and each webpage is a separate file on that web server. Whenever the user types the URL of the website in the browser, say for example, www.schoolplaza.com, he/she does not specify the file to be opened. But the web server needs a default file name in order to display some content. This default file is the page named index.html. Thus, when we type a URL without the filename, the server looks for the default file and displays it automatically.

Now, we will give a background color to the inserted table. Select the table and double click it. Alternatively, we can also right-click the table and select 'Table Cell properties'. This will open Table Properties dialog box as shown in figure 3.4.

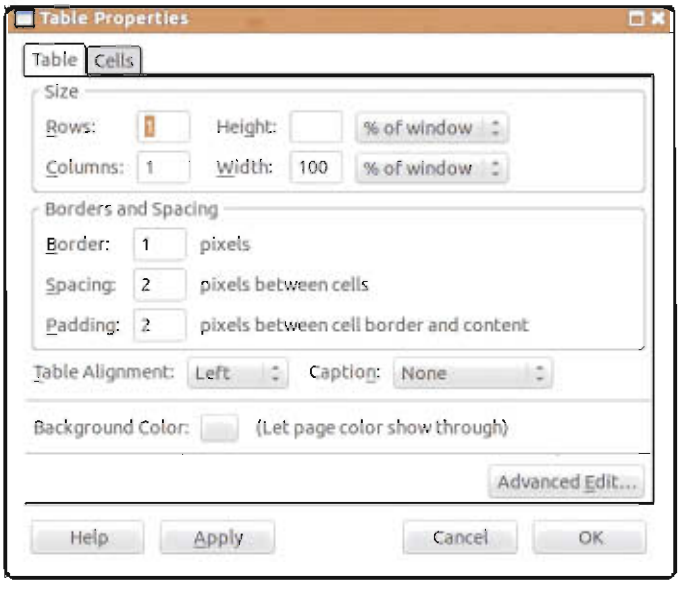


Figure 3.4 : Table Properties dialog box

You can see two tabs 'Table' and 'Cells' which allows user to control several aspects of either the table or individual cells. You can see the cells related to a particular table. The options seen in the table tab are :

1. Size : it shows the number of rows, columns and height and width of the table. Height and width can be specified in pixels or % of the window.
2. Borders and Spacing: it shows the border option in case you want to give a border to the table. Spacing is used to specify gap between cells. Padding gives a gap between the edge of the cell and the text within it.
3. Table Alignment : to align the table as left, centre or right.
4. Caption : Used to give caption to table if require.
5. Background Color : to give a background color to the table.

Now select a background color of your choice as shown in figure 3.5.

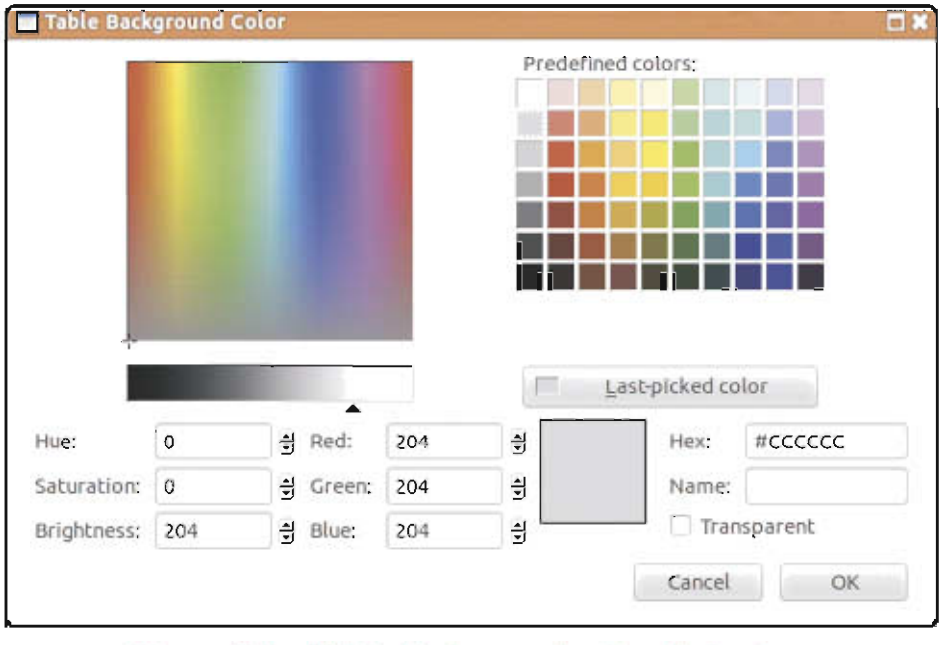


Figure 3.5 : Table Background color dialog box

Now, to display the text "School Plaza" and the related image, we will insert another table with one row and two columns. One of the columns will display the text and the other displays the image. Go to **Insert → Table.** Select 1 × 2 table as shown in figure 3.6 and press OK button.

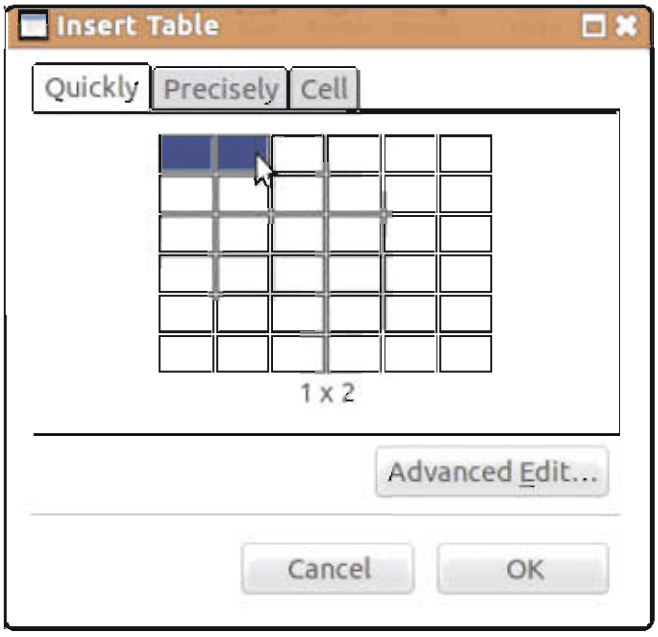


Figure 3.6 : Insert 1×2 table

Give a background color to the table. Now in the left cell we will insert the text and in the right cell the image. To insert image, select **Insert → Image.** Alternatively, you can click the image icon on the toolbar. This will open the dialog box as shown in figure 3.7. Specify the location of the image. Specify the alternate text in the input box otherwise select the "Don't use alternate text" radio button in case you do not want to give alternate text. Giving alternate text will display the text in case the image is not displayed in the browser.

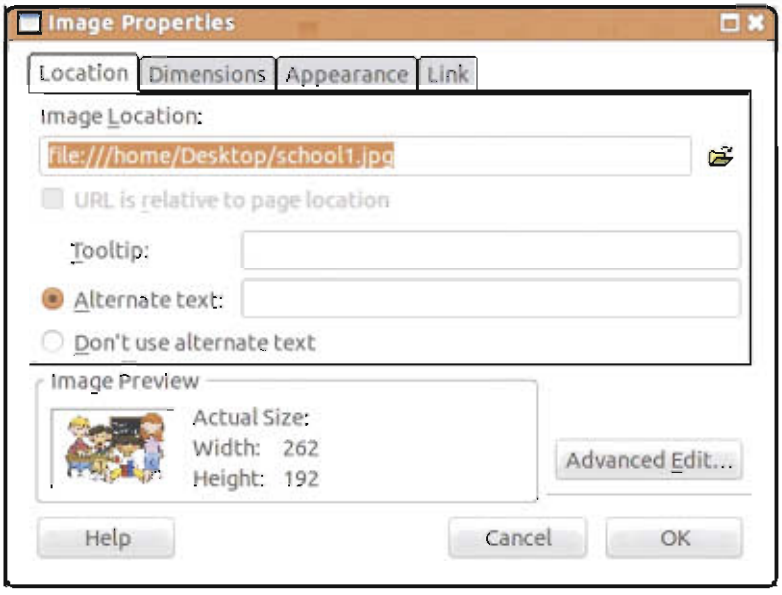


Figure 3.7 : Image Properties dialog box

Note : We have used an image named school1.jpg. Students may design their own image or use any other alternative image of their choice.

Now, select the Dimensions tab and select the Custom Size option. As we want to display the image in the whole cell we will specify the height and the width as 100% of the cell as shown in figure 3.8. Insert the text as shown in figure 3.9 in another cell.

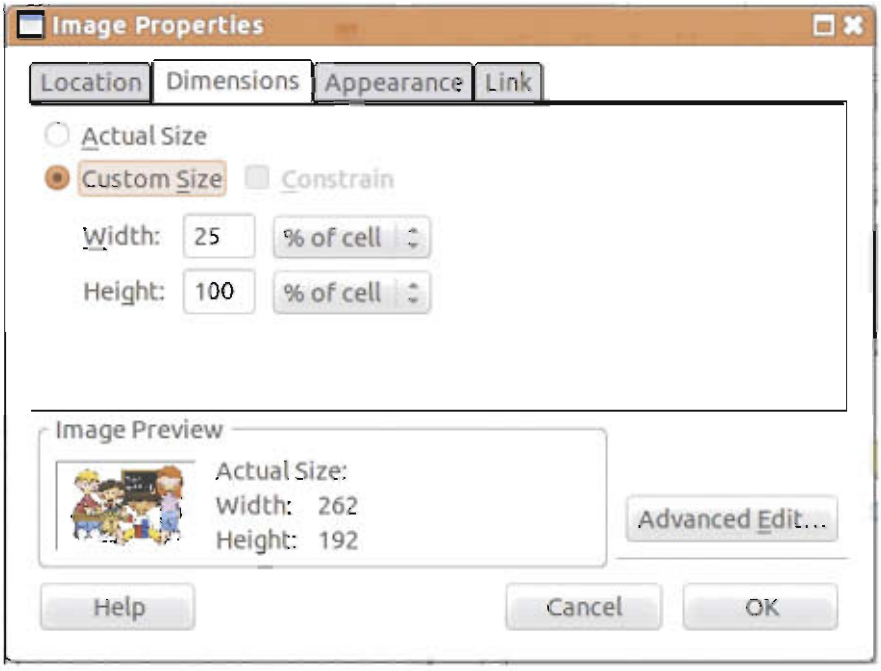


Figure 3.8 : Dimensions tab of Image Properties dialog box

Figure 3.9 shows the inserted text and the image.

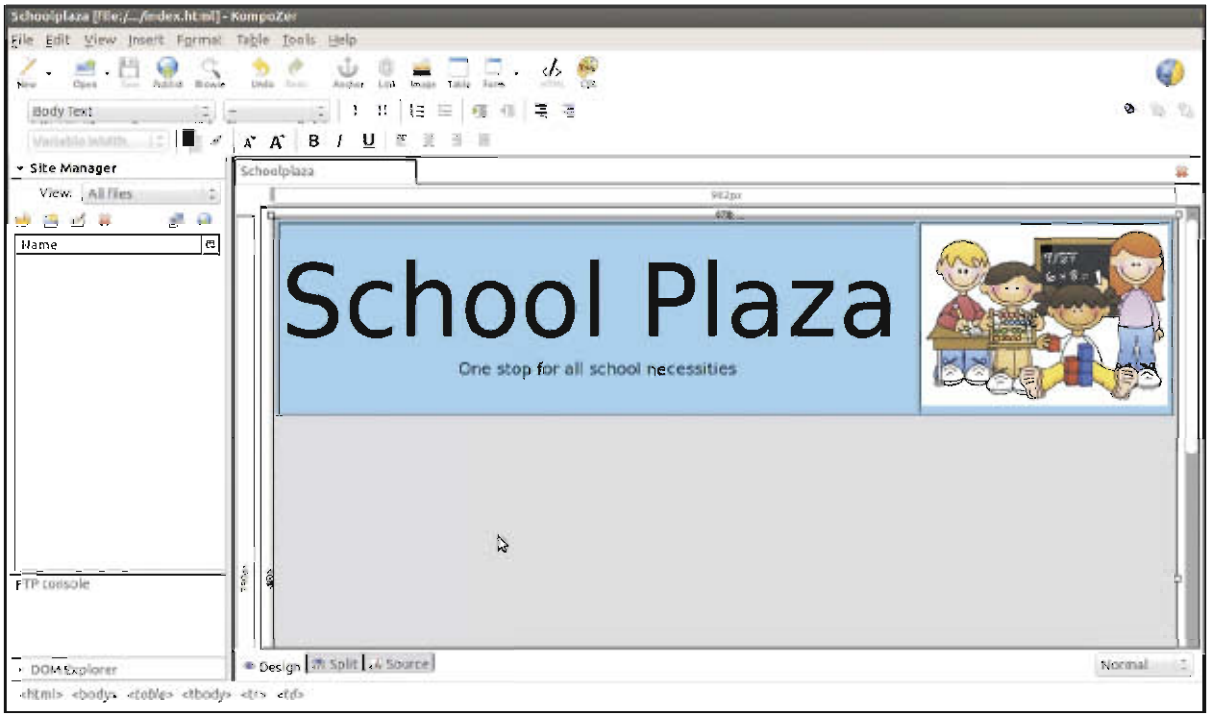


Figure 3.9 : Table with image and text

Now, we will insert another table with 2 rows and 2 columns to display the categories and New Products. In the first row cells, type the text "Categories" and "New Products". Give a suitable color to the text by going to **Format → Text color.** Figure 3.10 shows the row with the text.

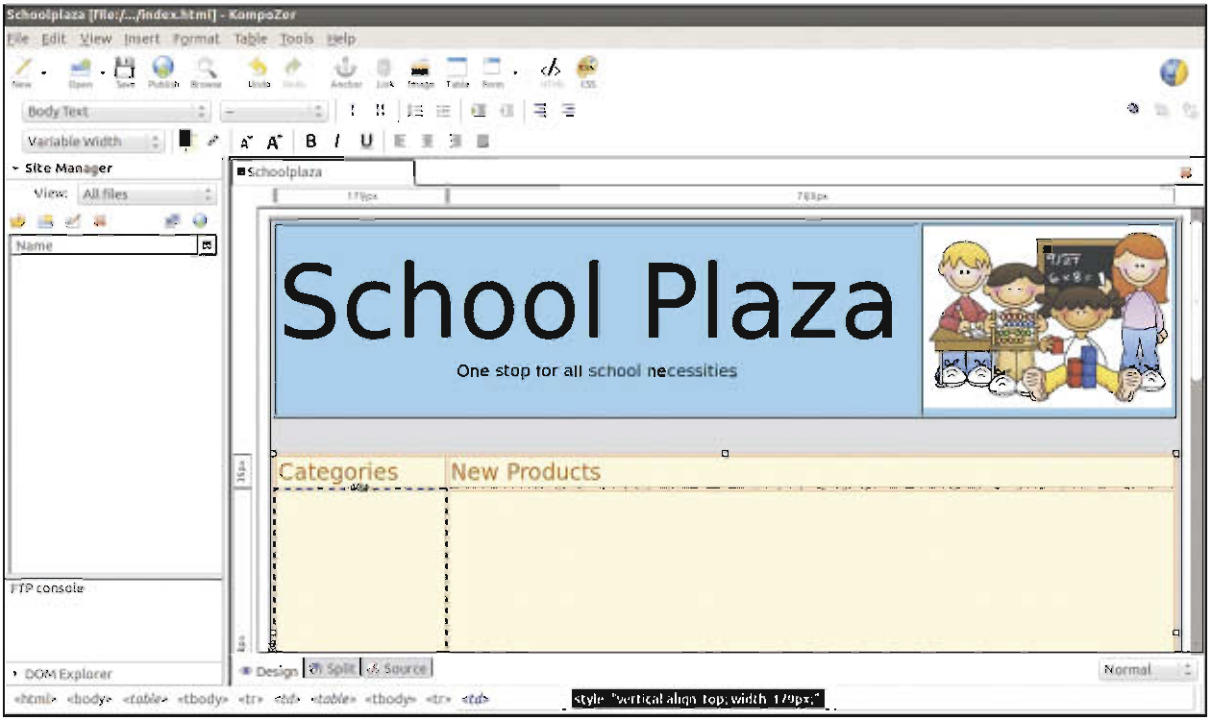


Figure 3.10 : Text inserted in the table row

Next, in the second row just below the categories we will insert another table with single column and multiple rows. In this table, we will insert different categories in each row like school bag, water bottle and others. Give a suitable color to the text. Here, we have inserted a horizontal line to differentiate the categories. To insert a horizontal line, go to **Insert → Horizontal line.** Also, the categories will be a link which takes us to the related page of the selected category. We will make the link on the categories later on. Let us now create table to add the images of the items and the related details in the cell below the new products. Insert the table depending on the items we want to display on the page. Here we have inserted a table with 3 rows and 3 columns. Figure 3.11 shows the added tables in the web page.

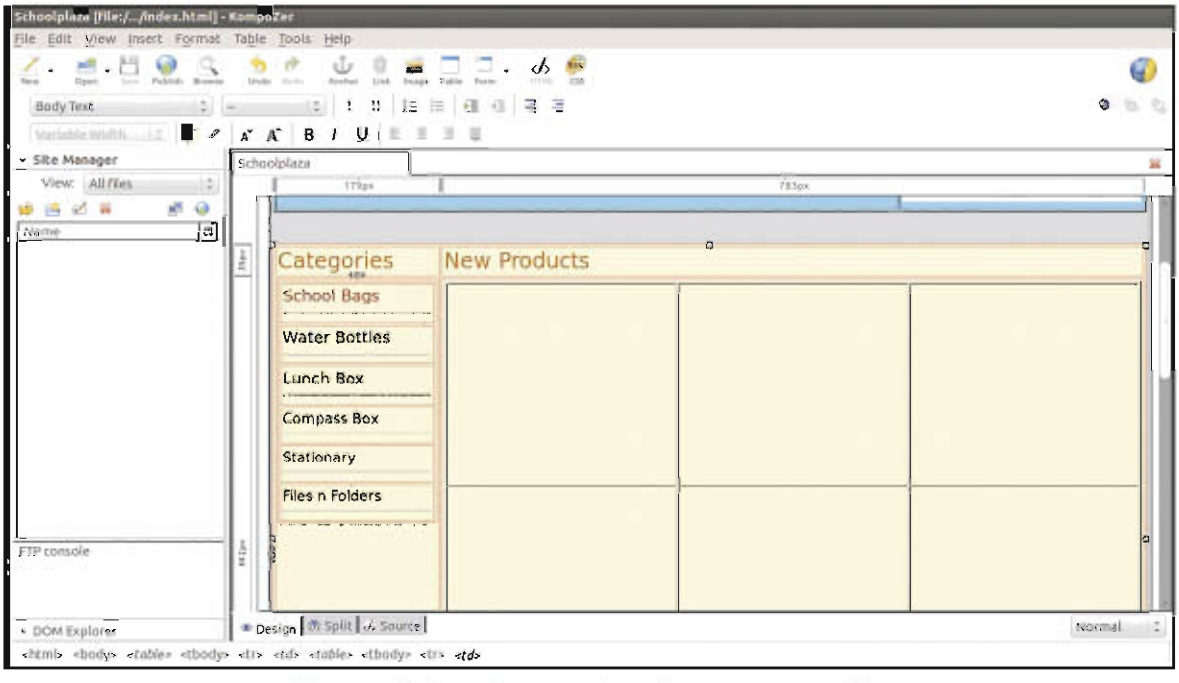


Figure 3.11 : Screen showing category list

Next, we will insert another table at the bottom of the page for the details like "About school plaza", "Feedback", "Contact us" and "Site map". Create the table and insert the text. This text will also be a link.

We have designed most of the home page. Whenever we click on any of the categories, the design of the page remains the same, but the images related to the category are changed. Thus we need not create other web pages from scratch but make multiple copies of our home page which can then be customized as per our need.

Go to Save As option and save the file with different names. We have saved the file as waterbottles.html, school-bags.html and lucnchbox.html. Now when the link of a particular category is clicked the corresponding web page will be opened. You must have observed that the web pages thus created have the same title as that of the home page. Open the web pages one by one and change the page title by going to **Format → Page Title and Properties.** Now, open the respective web pages and put the images as per the category. We will add the images on the home page and the same way they can be added to other web pages also.

Before inserting images of the items and their details on the home page, let us create link on the different categories. Clicking on a category will open the related web page.

Open the index.html. To create link, select the text (School Bags), go to **Insert → Link.** This will open a dialog box as shown in figure 3.12. Under the Link Location option, select the file to be opened (school-bags.html). Press OK button.

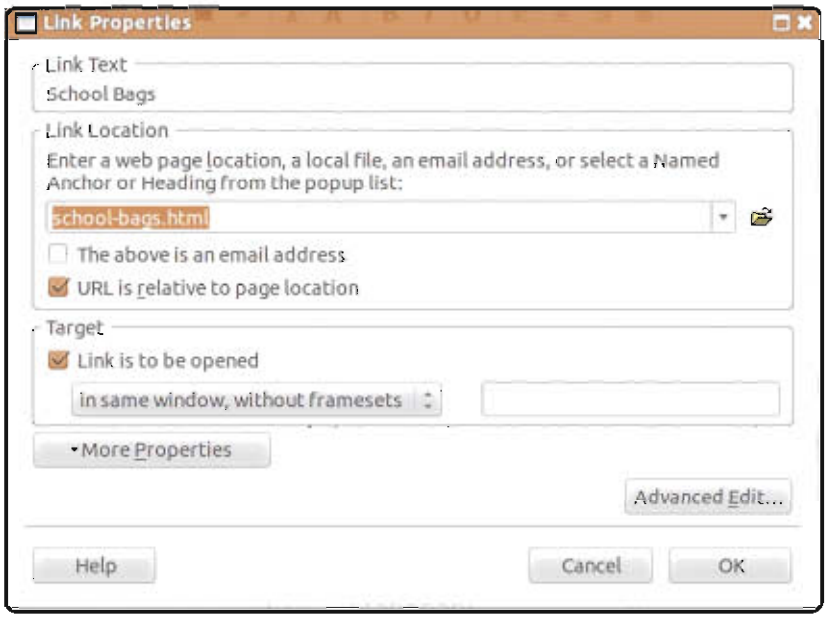


Figure 3.12 : Link Properties dialog box

Now, let us insert the details of the new products in the home page. In each cell of the table, we will insert an image of the item, name of the item, price and add to cart checkbox and quantity textbox. Insert the image as discussed earlier. Add the product name and price. Now, insert a checkbox by going to form field. We will add an HTML attribute "id" named b1 to the checkbox which will be used in JavaScript later on. Add the label "Add to cart". Then, insert textbox with initial value 1 and an id "q1". Add the label "Quantity". When the user selects add to cart checkbox, the item and its quantity are saved in the shopping cart. Figure 3.13 shows the details of the products added.

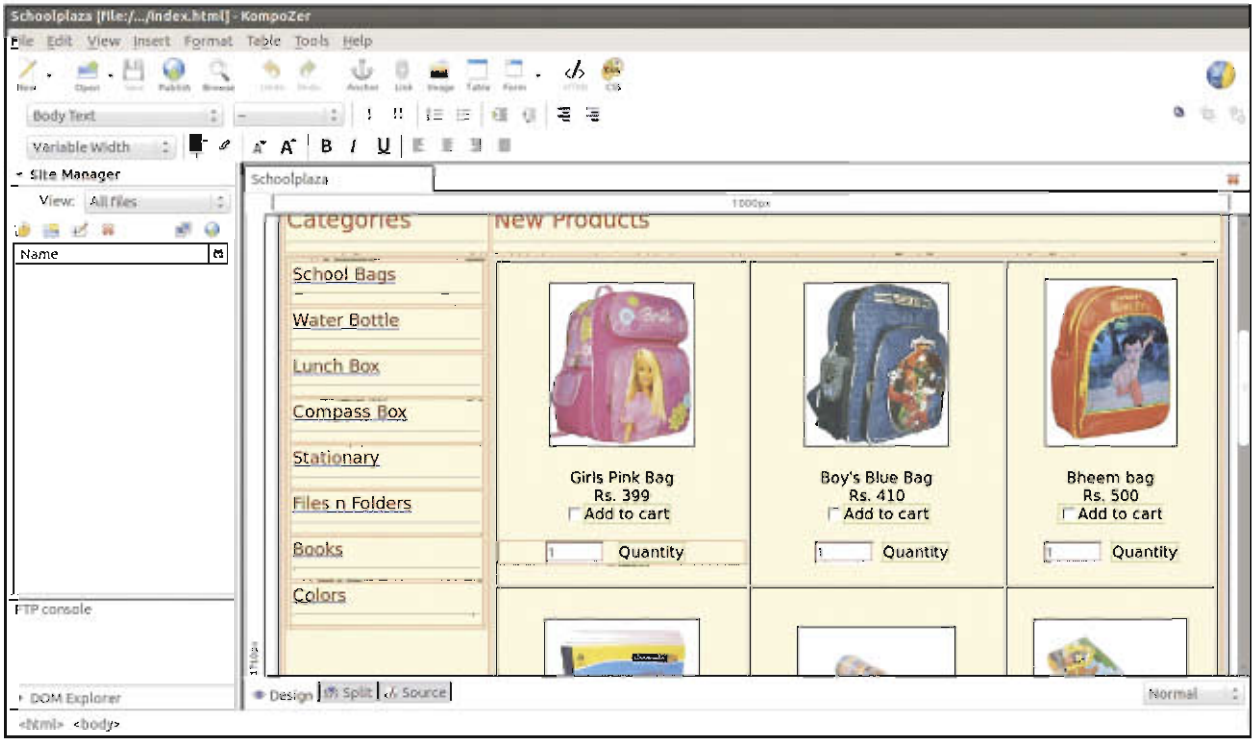


Figure 3.13 : Product Details

Note: Add the images related to the category on the web pages that we have created. Rest of the features remains the same. We have not shown the other web pages here.

We will add a button "Purchase" at the bottom of the page which will take us to the bill details web page. This page displays the total items purchased and the total amount. On the buttons onclick event we will add a Javascript shown in code listing 3.1. It calls the function oncart(), the function oncart() will then call the bill web page.

```
<script>
function oncart()
{
var check=false;
var cookievalue="";

if(document.getElementById("b1").checked)
{
check=true;
var quantity=document.getElementById("q1").value;
var price = 399;
var total=(price*quantity);
cookievalue+=":Girls pink bag,"+ price + "," + quantity + "," + total;
}

if(document.getElementById("b2").checked)
{
```

```
check=true;
var quantity=document.getElementById("q2").value;
var price = 410;
var total=(price*quantity);
cookievalue+=":Boy's Blue bag,"+ price + "," + quantity + "," + total;
}

if(document.getElementById("b3").checked)
{
check=true;
var quantity=document.getElementById("q3").value;
var price=500;
var total=(500*quantity);
cookievalue+=":Bheem bag,"+ price + "," + quantity + "," + total;
}

if(document.getElementById("nb1").checked)
{
check=true;
var quantity=document.getElementById("q4").value;
var total=(50*quantity);
cookievalue+=":NoteBook (set of 3),"+total;
}

if(document.getElementById("cp1").checked)
{
check=true;
var quantity=document.getElementById("q5").value;
var total=(45*quantity);
cookievalue+=":Color pencils,"+total;
}

if(document.getElementById("p1").checked)
{
check=true;
var quantity=document.getElementById("q6").value;
var total=(80*quantity);
cookievalue+=":Pencil Box,"+total;
}

if(document.getElementById("l1").checked)
{
```

```
check=true;
var quantity=document.getElementById("q7").value;
var total=(110*quantity);
cookievalue+=":Angry Bird Lunch box,"+total;
}

if(document.getElementById("l2").checked)
{
check=true;
var quantity=document.getElementById("q8").value;
var total=(120*quantity);
cookievalue+=":cartoon lunch box,"+total;
}

if(document.getElementById("e1").checked)
{
check=true;
var quantity=document.getElementById("q9").value;
var total=(50*quantity);
cookievalue+=":Eraser (set of 4),"+total;
}

if(!check)
{
alert("No item selected");
}
else
{
document.cookie=cookievalue;
window.location="bill.html";
}
}
  </script>
```

Code listing 3.1 : Validation JavaScript

As seen in the code listing 3.1 we have a function oncart(). The first if statement block checks whether the checkbox for item1 is checked or not. Earlier we have given an id named "b1" to the checkbox of item1. Using document.getElementById("b1").checked, we will check whether the user has checked the item. If it is checked then we will save the quantity into a variable and calculate the total price. We finally save the product name, price, quantity and amount as a string in a variable. Each of the details are separated with a delimiter "," (comma). Individual product description is separated with delimiter ":" (colon). The script continues for each item listed.

If the user has not selected any of the items then pressing the purchase button will show a message with text "No item is selected". Otherwise, the items are saved as cookie. A cookie is a variable that is stored on the user's computer. They are useful as they provide feature to remember and track preferences, purchases, commissions, and other information required for better visitor experience or site statistics. JavaScript can manipulate cookies using the cookie property of the document object. Cookies can be read, created, modified and deleted.

When the user clicks on the Purchase button, the script calls the web page "bill.html". The JavaScript in bill.html will calculate the total of all the items selected along with the total amount to be paid. Code listing 3.2 shows the JavaScript that calculates the total amount.

Here we have defined a function myfun(). The function is accepting data from cookie and storing it in an array. Then we are converting the data into number by using function parseFloat() and add it to total. Finally we print the values of the total.

```
function myfun()
{
var cookiearr=new Array();
cookiearr=document.cookie.split(":");
document.writeln("<center><h1>Welcome To Billing</h1></center>");
document.write("<center><table
border=3><thead><tr><th>Description</th><th>Price</th><th>Quantity</th><th>Total</th></tr></
thead>");
var total2=0;
var total3=0;

for(var i=1;i<cookiearr.length;i++)
{
var cookiedata=cookiearr[i].split(",");
document.writeln("<tr><td>"+cookiedata[0]+"</td><td>"+cookiedata[1]+"</td><td>"+cookiedata[2]
+"</td><td>"+cookiedata[3]+"</td></tr>");

total2+=parseFloat(cookiedata[2]);
total3+=parseFloat(cookiedata[3]);
}
document.write("<tr><td>Total</td><td>"+" "
+"</td><td>"+total2+"</td><td>"+total3+"</td></tr>");
document.write("</tbody></table></center>");
}
 </script>
```

Code Listing 3.2 : JavaScript to calculate total amount

Finally, the user clicks on the Proceed to Checkout button on the bill webpage. This will ask the user to login in case he/she has not logged in or Sign-up to create an account if he/she is not a registered user. The registration form can be created as discussed in the earlier chapters. Later the payment details can be entered as decided by the owner. Generally, to handle such websites we need to create database which stores the details of the product, registered user, login name, password and other relevant details.

Publishing a Website

Till now we learnt how to design a simple website using KompoZer. Now, let us see how to publish our website. To publish a website means to transfer the web pages, images and stylesheets related to the site, to a web server from which they may be accessed by the users. This process is generally called 'Uploading'.

We can upload the website if we have an account on the web server. The Internet Service Provider (ISP) offers limited free space on the web server. We can also buy space from professional hosting providers. Alternatively, to upload the website, there are many web hosts that provide limited free space to the users. The free space can also be for a limited time period. We can register on these sites and can avail the free space. In our sample website example we have opted one of these free space provider and uploaded the website. To publish a site, we need to know the settings for the space to set up the system. Open the index.html page and go to **File → Publish.** This will open a Publish Page dialog box as shown in figure 3.14.

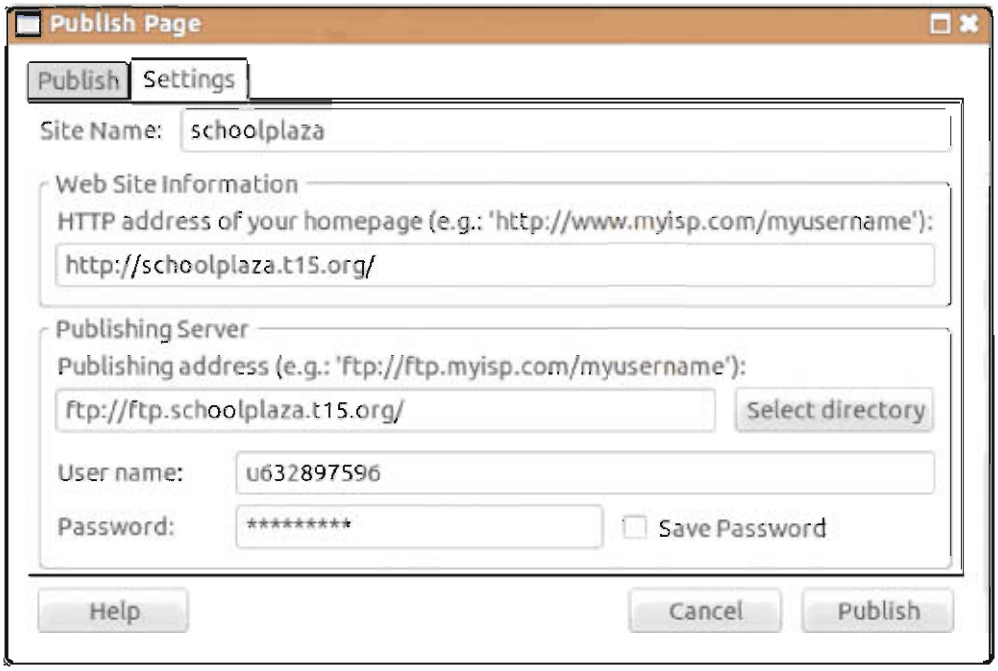


Figure 3.14 : Publish Page dialog box

The following details are to be provided in settings tab of Publish Page dialog box.

1. In the site name field, enter the name of the website. This name is only used for internal purpose by KompoZer to refer to the website.

2. In the website information, "HTTP address of your homepage" field specify the actual web address or URL of the website.

   Note : We can register a domain on Internet. There are many websites on Internet wherein we can register a domain name, for example: www.schoolplaza.com, for our website by paying nominal cost.

3. Publishing server details are provided by the ISP or the web host from where we have purchased the space. Type the publishing address. If connecting via FTP then type in FTP address. File

Transfer Protocol (FTP) is the means by which we can transfer the web pages from our computer to the web host. As discussed earlier, this is known as uploading or publishing. Provide the FTP username and password.

Note : The detail shown in figure 3.14 has been provided by the web host providing the free space.

Now click on Publish tab of figure 3.14. This will open a dialog box as shown in figure 3.15.

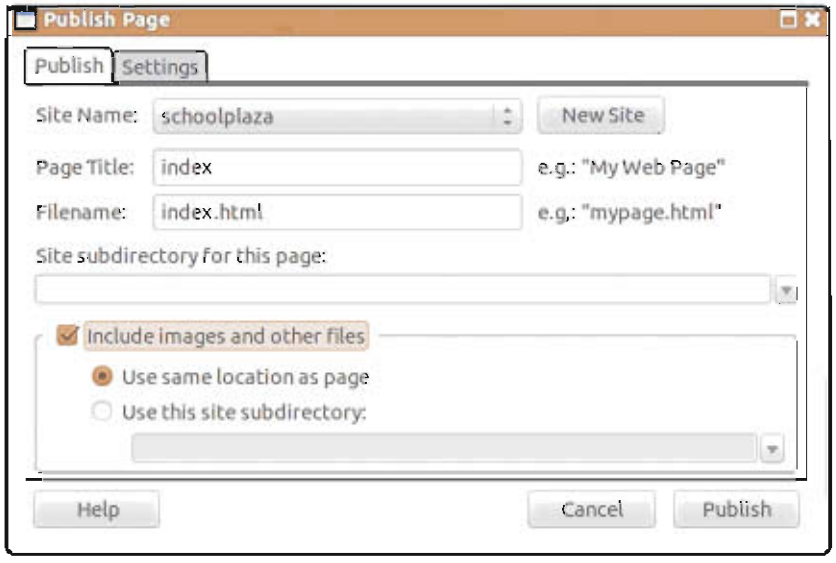


Figure 3.15 : Publish tab of Publish Page dialog box

Perform the steps mentioned below :

1. Select the site name from the drop down box if it is not selected.
2. Enter Page Title, it refers to the title of our index.html page.
3. Provide the filename. Here, as index.html file is open, it will take the filename automatically.
4. If this web page needs to be in a sub-directory of your site, enter in the "Site subdirectory" field. Remember that some hosting providers require that you keep the web pages in a separate directory like public_html.
5. Specify the directory for the images. Generally, keep the images in a folder in the same directory.
6. Click Publish button. This will open Publishing dialog box as shown in figure 3.16.

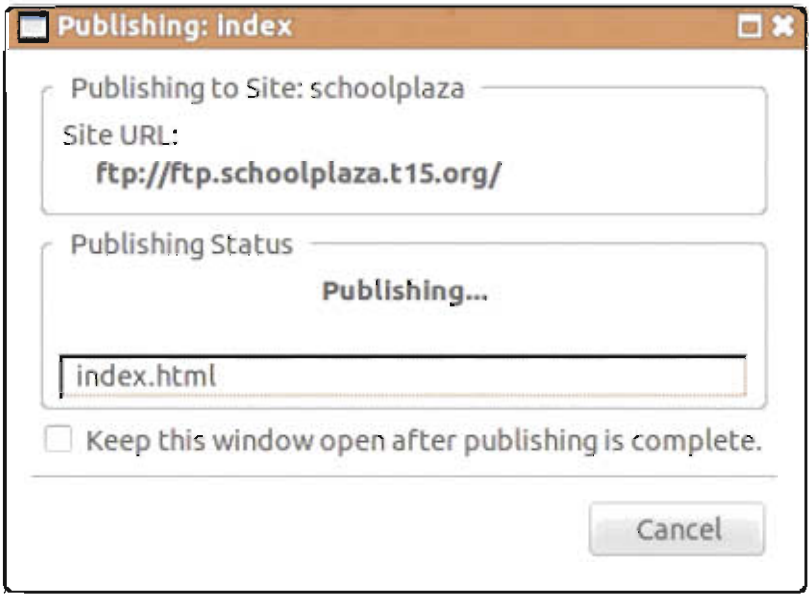


Figure 3.16 : Publishing dialog box

KompoZer will now upload the files and a popup message will be shown when finished. After uploading, type the URL of the website in the browser and check the website.

Other Open source web development tools

We have learnt how easy and fast it is to design web pages using KompoZer. As we have discussed earlier, KompoZer is a free open source web development IDE. Just like KompoZer, we have many other open source web development tools which are freely available on Internet.

Let us discuss about some of the open source web development tools that are available easily on the net and are used by web developers to develop websites with ease.

Aptana studio

Aptana studio is a powerful open source IDE (Integrated Development Environment) for building web applications. It is a complete web development environment which provides language support for HTML, CSS, JavaScript, Ruby, Rails, PHP, Python and many others. It also comes along with a large number of additional plugins. Aptana Studio 2.0.5 (also known as Aptana studio 2) is currently used to develop web applications using HTML, CSS and JavaScript. It can be easily downloaded from its website www.aptana.com. Figure 3.17 shows the Aptana studio interface. As seen in the figure it can create projects in PHP, Rails, Ruby or Web. Figure 3.18 shows the HTML code written in this interface.

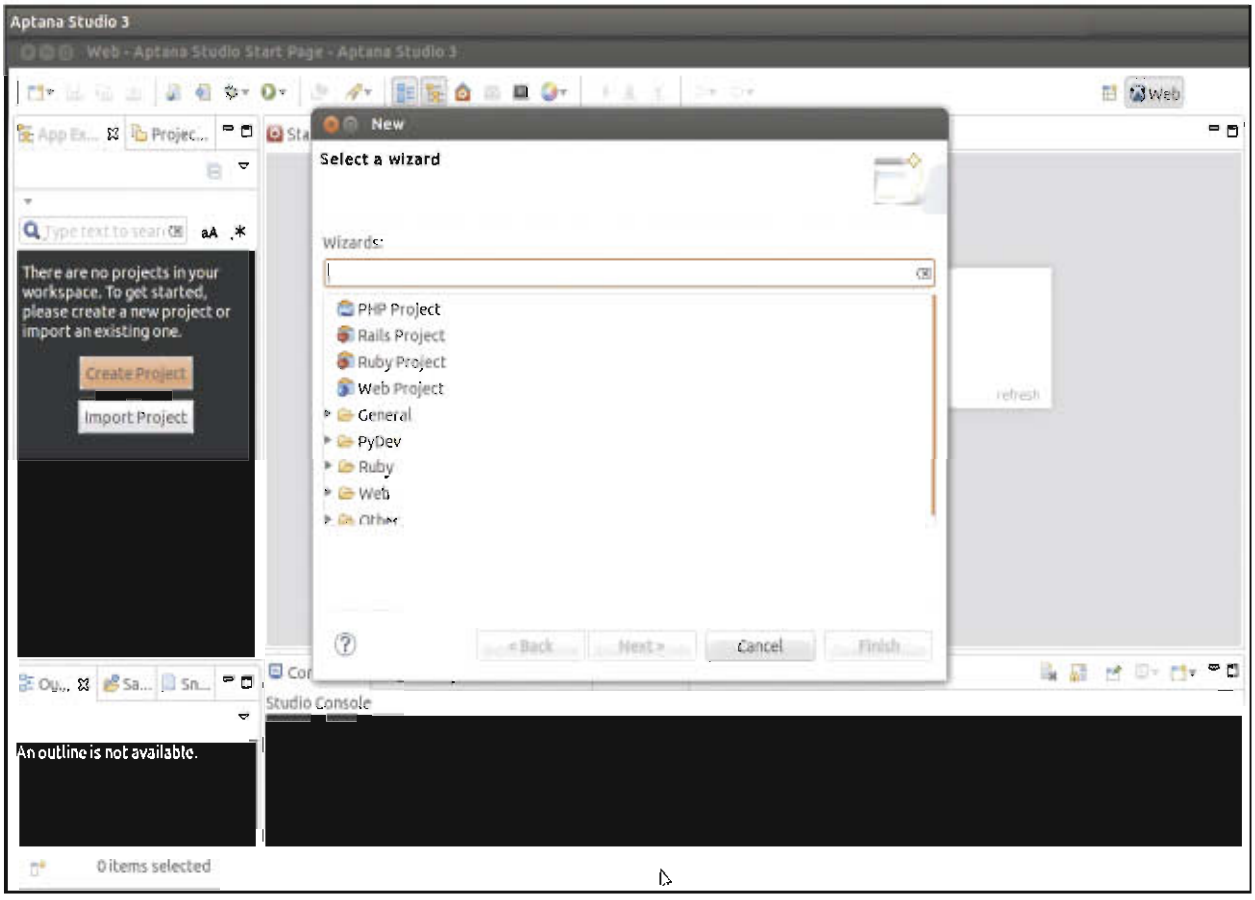


Figure 3.17 : Project creation using Aptana Studio

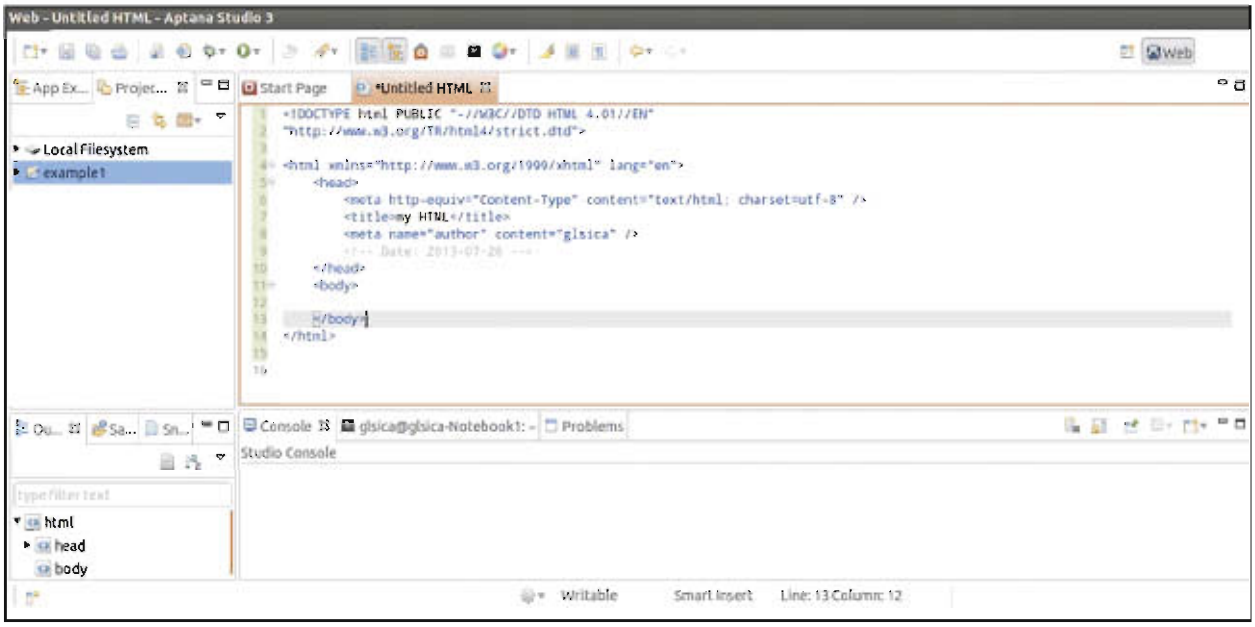


Figure 3.18 : Aptana Studio Interface

BlueGriffon

BlueGriffon is another free open source WYSIWYG HTML editor. It can be easily downloaded from www.bluegriffon.org. It supports languages like English, Dutch, German, Chinese and many more. It is an intuitive application that provides web developers with a simple user interface which allows them to create attractive websites without requiring in depth technical knowledge about web standards. Figure 3.19 shows the interface of BlueGriffon version 1.6.2. We have created a simple form using the tools.

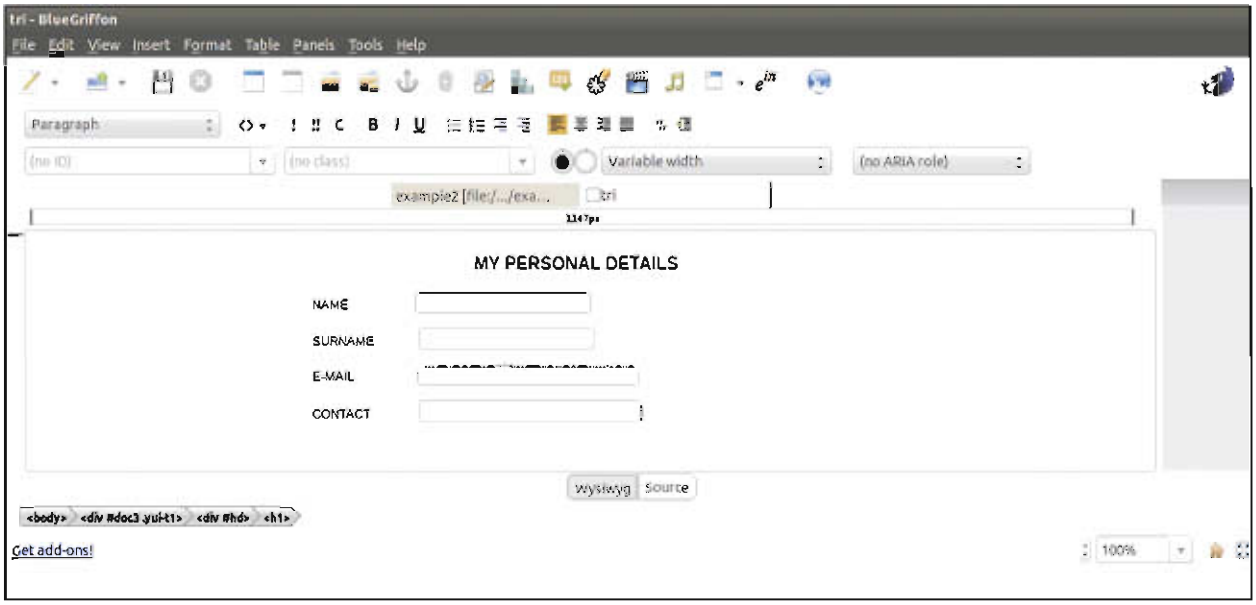


Figure 3.19 : Webpage created using BlueGriffon

Amaya

Amaya is another free open source WYSIWYG web editor developed by the World Wide Web Consortium (W3C). It was initially started as an HTML/CSS editor and has now grown into an

editor for many XML-based systems. It can be easily downloaded from www.w3.org/Amaya. Figure 3.20 shows a web page created in Amaya. The source for the web page can be seen in the window just below the design of the web page.

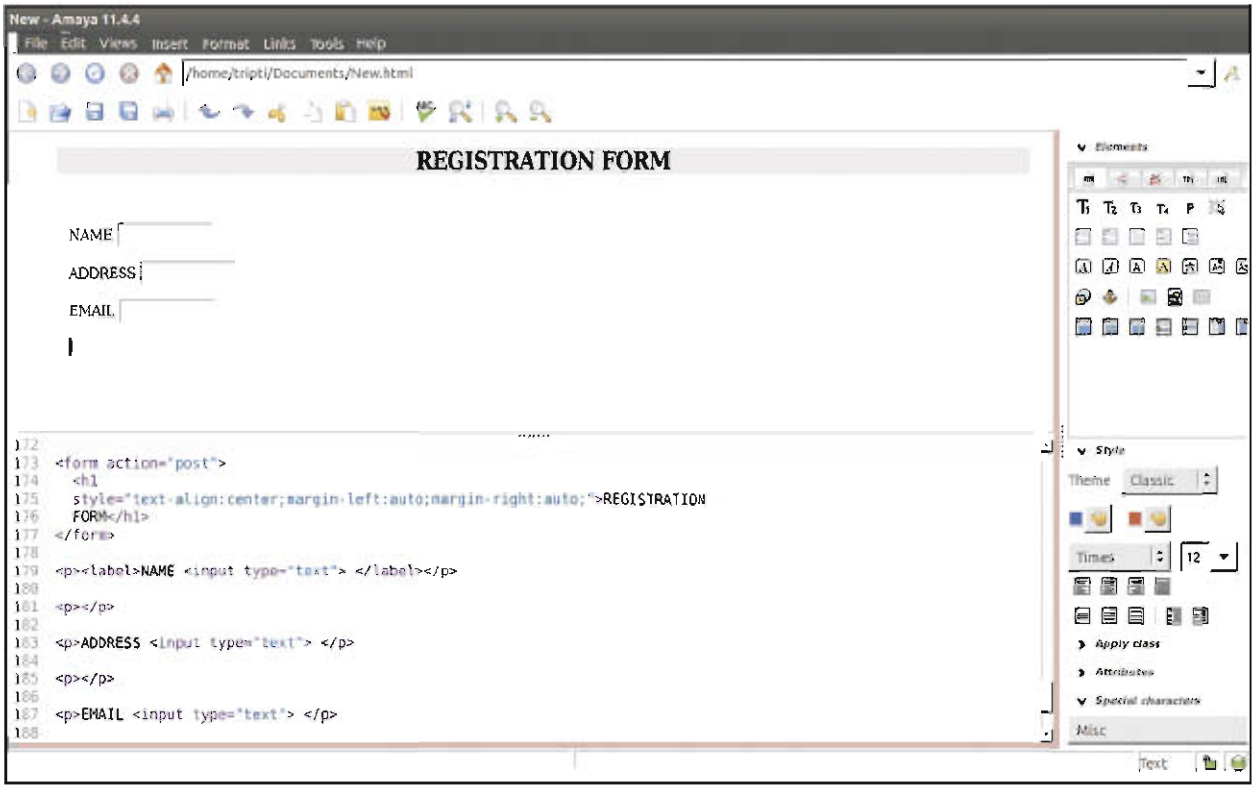


Figure 3.20 : Web page created using Amaya

Summary

In this chapter we discussed how we can create a simple website using KompoZer. After creating the website we need to upload it on the Internet. Publishing a website means transferring the site, i.e. the pages, images and stylesheets, to a web server from which they can be accessed by the users. This process is called 'Uploading'. We can upload the website if we have an account on the web server. The Internet service providers (ISP) offer limited free space on the web server. We can also buy space from professional hosting providers. We also discussed about the different free open source web development IDE available on Internet like Aptana studio, Blue Griffon and Amaya.

EXERCISE

1. How does a website play an important role in business ?
2. List the important points to be considered for developing a good website as part of planning process.

3. Why should the purpose of the website be clearly defined ?

4. How does knowing the expected audience help in the design of the website ?

5. What content should be placed in the website while designing it ?

6. Why is the homepage of the website named as index.html ?

7. Which are the options available to set the table properties ?

8. What is a cookie ?

9. What do you mean by publishing a website ?

10. Name some of the open source web development tools available in the market.

11. Choose the most appropriate option from those given below :

(1) Which of the following helps in promoting the business, selling the products and attracting a large number of customers ?

(a) Website (b) Webpage (c) Form (d) CSS

(2) Which of the following is not an important point to be considered for developing a good website as part of planning process ?

(a) Purpose (b) Audience (c) Content (d) Input

(3) Which of the following information should a website contain ?

(a) Complete, Relevant (b) Complete, Irrelevant

(c) Incomplete, Irrelevant (d) Incomplete, Relevant

(4) Which of the following content provides an overview of the site, organization, products and services, and other items ?

(a) Detailed (b) Long (c) General (d) Short

(5) Which of the following is a collection of interlinked web pages ?

(a) Webpage (b) Form (c) Kompozer (d) Website

(6) Which of the following is the first page that opens when the user enters the URL address in the address bar of the browser ?

(a) Home page (b) Last page (c) Web page (d) First page

(7) Which of the following filename is the home page of the website saved as ?

(a) first.html (b) index.html (c) home.html (d) one.html

(8) Which of the following is a variable that is stored on the user's computer ?

(a) Integer (b) HTML (c) Cookie (d) Java

(9) Which of the following stands for FTP ?

(a) File Truncate Protocol (b) File Transfer Process

(c) Fine Tune Protocol (d) File Transfer Protocol

LABORATORY EXERCISE

1. Using Kompozer create a simple website on your school.
2. Using Kompozer create a website on the theme "Garvi Gujarat".
3. Using Kompozer create a website on "Gujarat Tourism".
4. Using Kompozer create Ecommerce website on "Sports Shop".

Introduction to E-Commerce 4

We are living in the era of information. Information is available from sources like radio, television, newspaper and Internet. The availability of Internet and mobile devices has changed the way we used to access information. People have started using Internet almost every hour of the day with mobile devices. Earlier websites were used mainly to disseminate information about the product or organization. For example, website of an educational institute provides information regarding various courses, course content, students and its faculties. Similarly, business organizations provide information regarding their products, features of the product, their suppliers, how to order the products and so on.

Last few decades, due to tremendous development in telecommunication infrastructure and software technologies, Internet has become popular among the individuals as well business organizations. Internet has revolutionized the way business is conducted nowadays. Today, Internet is being used by people for various purposes like paying bills, banking and even shopping. Business organizations conduct activities like marketing of the product, selling of products, providing catalogue, trading of the stocks and customer service. The use of Internet for conducting such business activities is known as E-commerce.

E-commerce can also be defined as buying and selling of products, services and information using electronic media like Internet. It is a modern business methodology that addresses to the needs of organizations, merchants and consumers in cutting costs while improving the quality of goods and services. It also increases the speed of delivery. It allows people to rise above the barriers of time and distance and take the benefit of global markets and business opportunities.

Selling the products through websites is the fastest growing method of trading worldwide. Many different types of products and services such as books, electronic gadgets, cars, holiday packages and many more are traded online. Many businesses have now set up their own websites for E-commerce related activities. E-commerce creates a whole world of global village from where anyone can buy anything, anytime and from anywhere.

Applications of E-commerce

The major areas of business and commercial activities where E-commerce is widely used today includes trading of goods which comprises of marketing and selling, auctions of goods, financial services like banking and insurance. Let us discuss these areas briefly. E-commerce is not limited to the areas discussed here but spreads across many other developing areas for doing business and other activities.

Internet bookshops

This was one of the first applications of E-commerce on Internet. Customers prefer purchasing books on Internet as they do not require to be physically checked and can be easily described. Books can be easily shipped to the customers place little or no damage to them. Online bookstores need

a good website displaying all the books category wise, picture of the cover page, description of the books including the number of pages, price of the book, discounts and reviews of other customers. The search of a book can be made by title of the book, author's name or publication name. The online bookstores may also record the customer's interests and inform them about new arrivals to attract and retain their interest. Figure 4.1 shows the home page of one of the first online bookstore www.amazon.com.

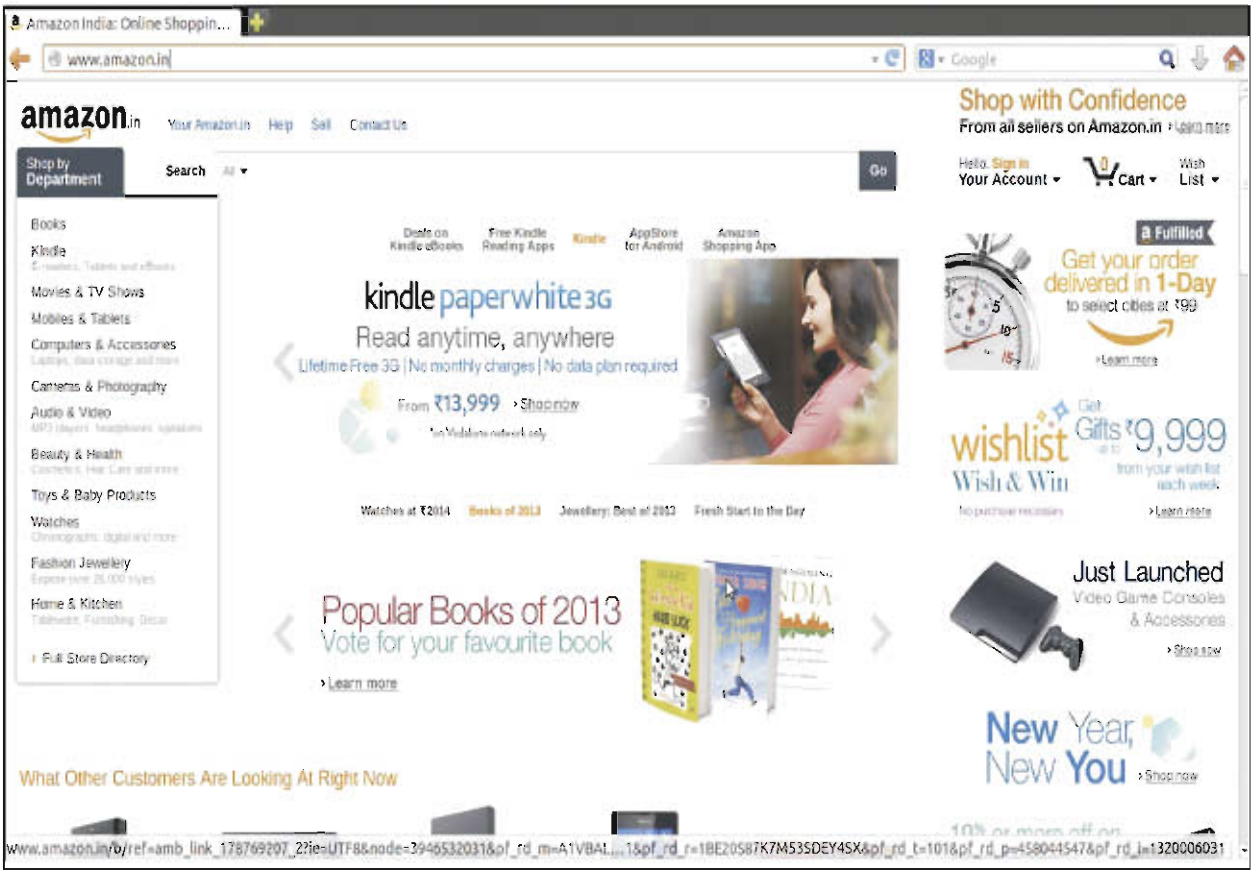


Figure 4.1 : Online bookstore

Some of the large online bookstore websites are as mentioned :

- www.amazon.com
- shopping.indiatimes.com
- www.buybooksindia.com
- www.bookshopofindia.com

Electronic newspaper

An electronic newspaper also known as E-newspaper is a newspaper that exists on the Internet in digital form. It has advantages over the printed newspaper and the news broadcasted on television and radios. It can give us up-to-date news on the issues that are happening worldwide. With the advent of digital technologies, the browser can be set to select the news as per the interest of the reader. It also removes the hassle of printing process and further help in reducing the costs. Majority of the leading newspaper now provide E-newspaper to the readers. Figure 4.2 shows the home page of E-newspaper Washington Post.

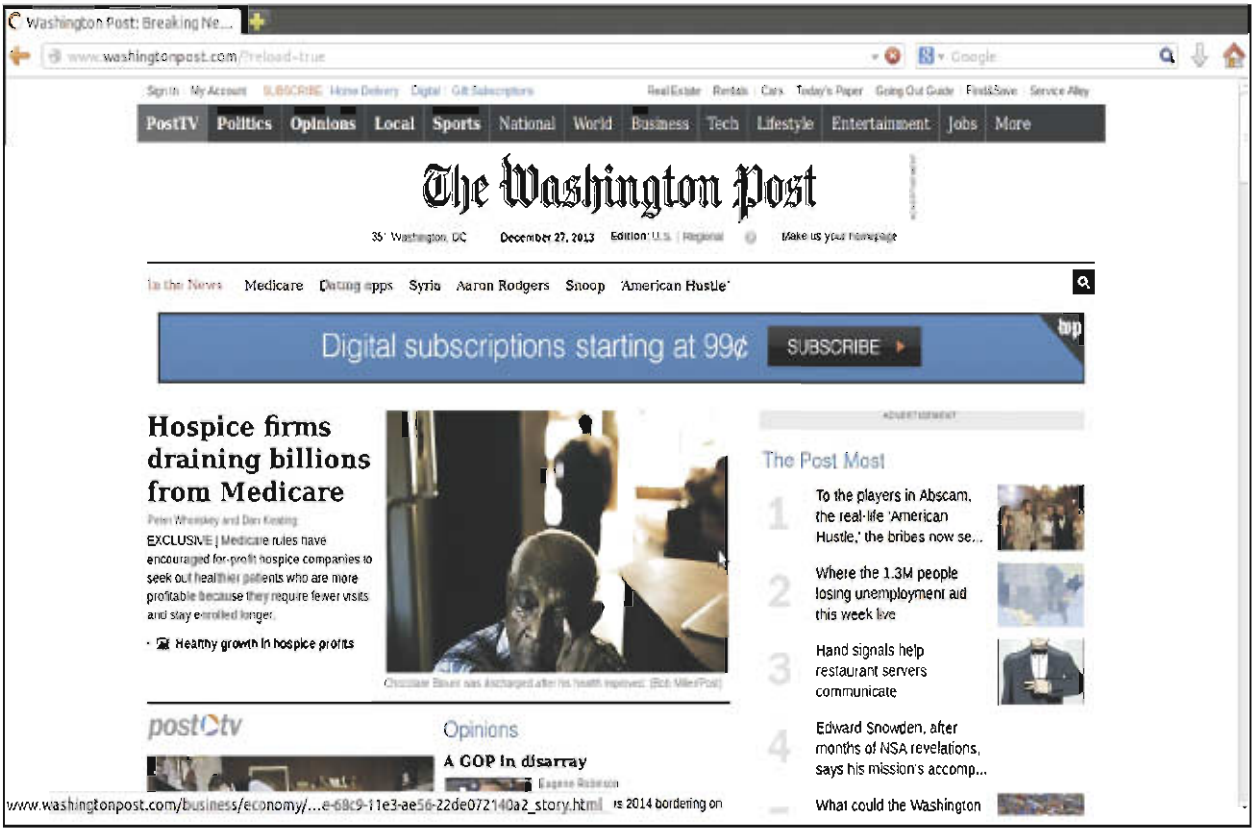


Figure 4.2 : E-newspaper

Online Auctions

An auction is the process of buying and selling products or services by offering the customers to bid the price and selling the product to the highest bidder. The traditional auctions have limited participation of people. Today, the same auction mechanism can be implemented using E-commerce technologies which allow the people to bid on Internet. This is known as Online Auction. There are many websites providing live auctions of goods. These websites provide the platform for both the seller and the bidders. When you place a product for auction on these sites, you are a seller. At the same time you can also bid for a product which is placed by other sellers on the site, in this case you are a bidder.

To sell an item through an online auction site, you need to first register with the site. This is required to track the items you sell or bid on, determine the winning bids and build a database of seller and bidder feedback. Members are also required to provide their basic contact information before they are allowed to sell. Then, the member must go through the site's steps to sell each item that they want to put up for auction. The member generally puts a digital picture of the item and writes a brief description about the item.

By placing goods for auction, a person can get a good price through bidding and once the target price is reached or the time limit is over, the item is transferred to the bidder. Various payment options are also provided to the bidder. The sellers can get the advantage of getting best price for their products and the bidders save the time to get the product of their choice. Figure 4.3 shows the home page of online auction site www.ebay.com.

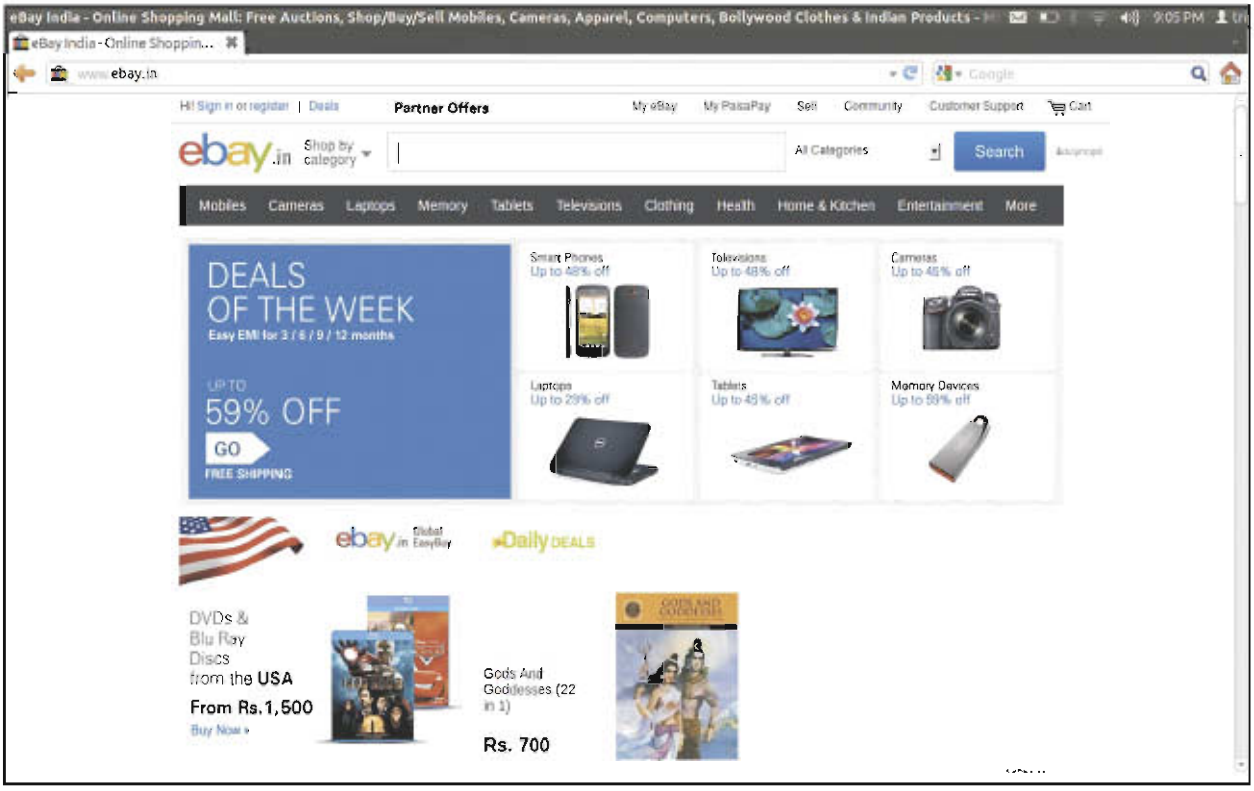


Figure 4.3 : Online auction site

Some popular websites for online auctions are listed below :

- www.ebay.com
- www.onlineauction.com
- www.mybids.in
- www.ubid.com

Marketing and Selling

Many companies now-a-days conduct their business of marketing and selling of goods and services by their websites. They provide their product catalogue online over Internet for better marketing. The catalogue displays different categories of products with images, videos in some cases, brief description and features of the product. The customers can view the catalogue and select the products of their choice by adding them to the shopping cart. An online shopping cart is similar to the original store shopping cart. At the store, the customer selects the products and puts it in the shopping cart. When finished, the final billing is done for all the products in the shopping cart purchased by the customer. Similarly, in online shopping cart the customer can add the product, review what he/she has selected, make necessary modification or addition and finally order the products while checking out. The user needs to provide shipping detail. The payment of the purchase can also be made through Internet.

Today you can purchase groceries, toys, computer accessories, mobiles and many more products from Internet. Some websites provides holiday packages along with airline tickets. We can easily buy an airline or railway tickets online. For example, the website of Indian railways www.irctc.co.in gives all the information related to various trains and provides facilities for booking tickets and payment for it online. Once the ticket is booked the E-copy of the ticket is send on your mail as well as sent as an SMS on your mobile. Figure 4.4 shows the online booking of railway ticket.

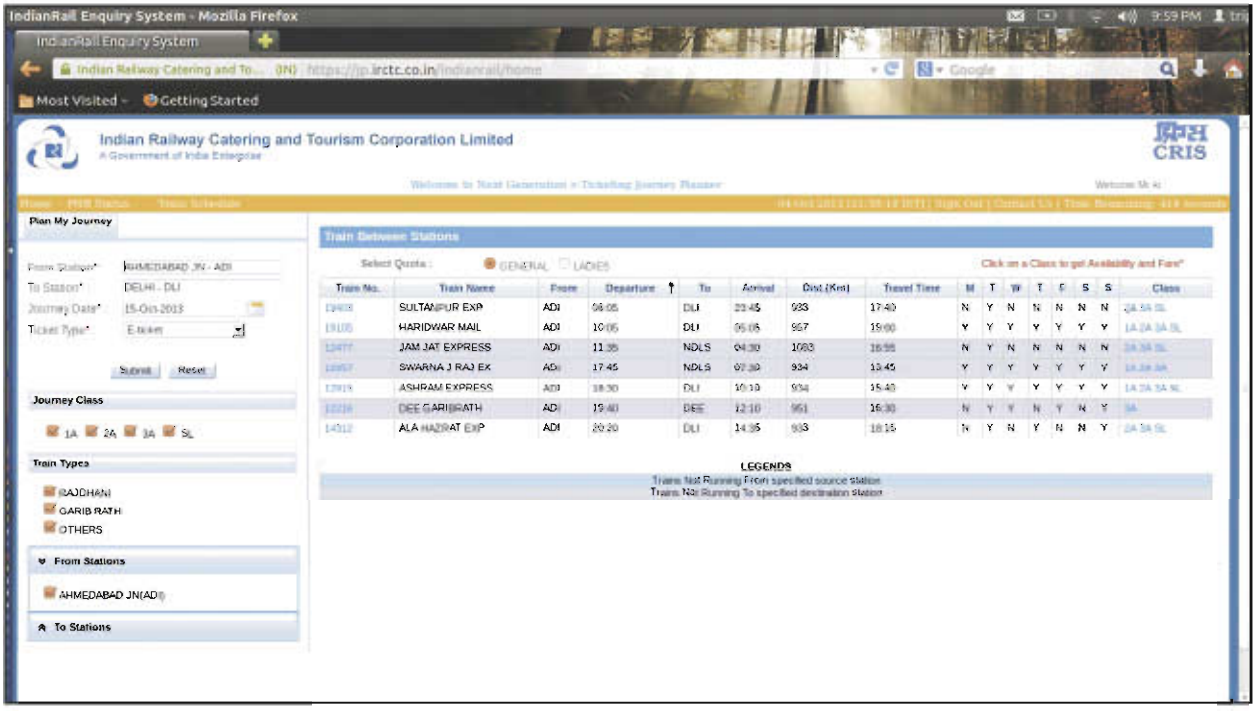


Figure 4.4 : Online railway ticket booking facility

Some popular sites used for marketing and selling are :

- www.homeshop18.com
- www.flipkart.com
- www.myntra.com
- www.makemytrip.com

Online billing

In online billing, companies send their bills to customers through E-mail. Once the customer receives the bills, he/she can pay online on the company's website using credit card or net banking facility. Companies who need to send the bills to a large number of customers periodically can use these facilities. For example, BSNL sends it customers online bills and the customers can also do online payment using Internet.

Information services

Many organizations use the Internet to provide latest information to their users or members. This includes educational institutes and universities, which provides the examination results, online enrolment forms, examination schedule and seating arrangement and important notices. Students can view their results using the website from anywhere. Another example of information services is the notices and reminders sent to the customer by the companies or banks.

Many organizations provide various forms to be downloaded so that the customer can easily download and use it. Companies can also get the information or details about the customers by filling the forms online.

Support services

Support services have become increasingly important due to the huge technological changes that have taken place in the last decade. Today, even the simplest products utilise sophisticated electronics

which requires specialised knowledge and technical ability for maintenance and support when any problem arises. After selling the products, companies are providing online support to the customers. For example, a company selling electronic product provides online complaint registration to their customers, which is then forwarded to the support engineer. Customers can also track the status of their complaint placed online. Software companies provide online support to their customers for any problem in installation, configuration or use. Software vendors also allow their licensed customers to download the recent updates of the software. Hardware vendors put software drivers for their devices so that the customer can download them by choosing proper product type and model.

Net Banking

Net banking or electronic banking is getting more popular day by day. Sometimes a customer may want to make an urgent payment or check his account balance without visiting the bank due to some problems. Online banking can help the customers solve these problems. Online banking is the process of conducting the banking transactions over the Internet. Today majority of the leading banks have started providing online banking facility to its customers. With the help of online banking customers can avail the following services :

- Check account balance at any time.
- Transfer the money from one account to other.
- Obtain statements for any credit or debit.
- Find status of transactions.
- Pay various bills online like telephone, electricity and many more without going to the bank.

The customer is provided a password for online banking services with which he logs in to the bank site and performs all the banking activities from his computer or mobile. Figure 4.5 shows the home page of State Bank of India website. The URL of the same is www.onlinesbi.com.

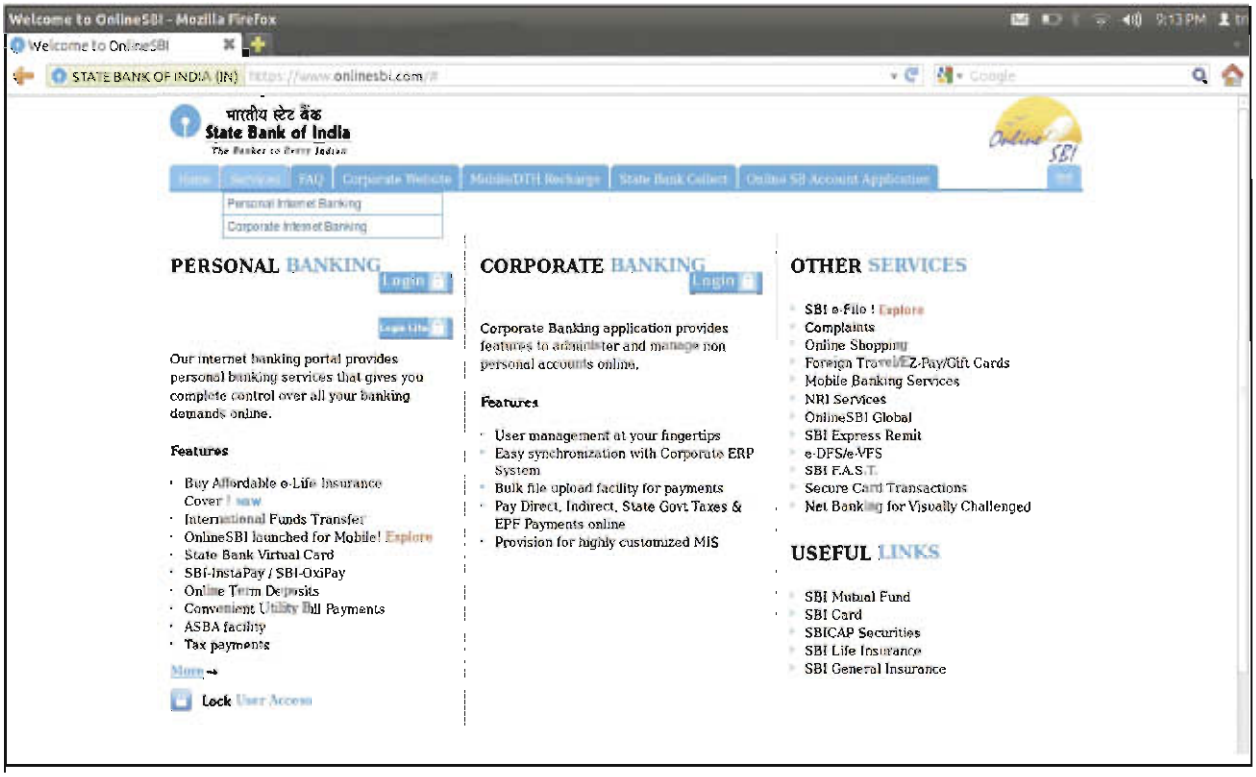


Figure 4.5 : Online banking facility

Most of the banks today provide online banking; a list of few of them is given below :

- www.centralbankofindia.co.in
- www.bankofbaroda.co.in
- www.iob.in
- www.pnbindia.in
- www.denabank.co.in

Traditional commerce vs. E-commerce

Due to the growth of Internet, the nature of competition in traditional way of doing business and using E-commerce has changed significantly. In traditional commerce, the businesses have to compete within a single industry and limited geographical area. In many cases business processes use traditional commerce activities very effectively and they cannot be improved through technology. The products that customers prefer to touch, smell or examine precisely are difficult to sell using E-commerce. For example, buyers will hesitate in buying perishable food items, expensive jewellery and high-fashion cloth as they cannot be examined closely before purchase. In traditional commerce, the retail merchants with their years of experience create the store design that helps to convince the customer to purchase a product. This arrangement of products, store design and layout is called merchandising. The merchant develops the skills to identify customer needs and find products and services that meet those needs. But this art of merchandising can be difficult to implement over Internet. Some of the common features of traditional commerce are :

- Operates within a certain period of time or during business hours.
- No sharing of information with competitors.
- Hiring of sales persons, sales manager and many more.
- Location renting or purchasing, advertising, inventory, shipping of products.

However, in today's fast paced world it is very important to break through these conventional rules. We need to adapt to the new ways of doing business worldwide using Internet. Some of the features of E-commerce are :

- Advertising of the product is done electronically.
- Customers can browse through products catalogue and available offers.
- E-payments systems are used for receiving payment. However, in many cases payment on delivery option is also available.
- Goods are delivered to the customer within few days.
- Reduces the per transaction cost.
- Reduces the time taken to perform an overall transaction.

E-commerce in India

The Internet users in India are growing day by day. E-commerce in India is experiencing a remarkable growth and successfully changing the way people transact. From high end designer apparel to second hand books, everything is now available to Indian online buyers at the click of a button. It has opened up new horizons and opportunities for millions of people.

A greater variety of goods and services that can be bought over the internet is making online buying more attractive and convenient. As internet has reached the rural parts of India, a drastic change is observed in their lifestyles, the quantity and quality of consumption is helping change mindsets in the rural areas.

Many factors are playing role for the growth of E-commerce in India. Some of the key factors are listed below :

- The growth of technology facilitators such as Internet connections, broadband and third generation (3G) services, laptops, Smartphones, tablets and dongles.
- Increase in use of mobiles devices.
- Availability of much wider product range.
- Busy lifestyles, traffic congestion and lack of time for traditional shopping.
- Lower prices compared to brick and mortar retail driven by disintermediation and reduced inventory and real estate costs
- Increased usage of online classified sites, with more consumers buying and selling second-hand goods.
- Evolution of the online marketplace model with sites like eBay, Flipkart, Snapdeal and many more.

Though the penetration of E-commerce is low in India as compared to other countries, but it is growing at a much faster rate with a large number of new entrants. There are a lot of Online stores in India which are becoming quite popular like Flipkart, eBay India, Snapdeal, Amazon India, Myntra, Dominos, PayTM, Jabong and many more.

Retailers have also started offering a cash-on-delivery option, it is the most preferred payment method in India. Almost 80% of Indian E-commerce business uses cash-on-delivery mechanism. Though, from the online retailer point-of-view this is an expensive proposition, as they have to finance the sale till the order is delivered. For the consumer, cash-on-delivery makes it easy to reject products at the point of delivery.

Industry experts are optimistic about the future potential of E-commerce industry in India. Though there are challenges to be faced, they can be easily overcome with well-planned and pre-decided strategies which will give a boost to the growth of E-commerce in India.

Advantages of E-commerce

With the growth of Internet, companies are finding new and exciting ways to expand their business. Today we can hardly find any successful company which do not use computers in their everyday business activities. E-commerce provides multiple benefits to the customers in the form of availability of goods at lower cost, wider choice and saves time. E-commerce sales would rise in the coming years with the availability of faster broadband services and new applications. Some advantages of E-commerce are listed below :

- **Conduct business 24×7**

  Using E-commerce a business can operate anywhere anytime. The business activities do not remain time bound. A customer can purchase a product any hour of the day; hence an order can also be accepted at any hour of the day.

- **Lower cost**

  Using Internet for business activities can reduce the cost of marketing, distribution, phone, postage, printing and many other such activities. In E-commerce requirement of physical store space or infrastructure investment is minimal. The cost of opening a physical store is very high as compared to doing business on Internet. Customers are also benefited by getting a wide range of products for selection at lower cost due to competition amongst the vendors and elimination of distributors or intermediaries for selling the products.

- **No boundaries or geographical limitations**

  A physical store is limited by its geographical area of service. With E-commerce, businesses can reach out to millions of customers in an instant which is not possible in any conventional mode of marketing. It has the ability to reach potential buyers easily further creating new markets. Global coverage can be reached through creating a website and uploading it on Internet.

- **Improved and better customer service**

  As there is a direct communication with the customer, it is possible to solve their queries related to the price, quality, features of the product and others thus resulting in a better customer service. For companies that do business with other companies, adding customer service online is a competitive advantage. Online customer service makes customers happier. E-commerce gives better and quicker customer service. Information can be shared easily on Internet. A merchant can email its customers about the new product and can also solve the product related problems.

- **Teamwork**

  E-commerce helps organizations to work together. One good example is the use of e-mail which helps people to exchange information. It has changed the way organizations interact with suppliers, vendors, business partners, and customers. More the interaction better is the result.

- **Eliminate Travel Time and Cost**

  The customers do not have to travel long distances to reach their preferred physical store. E-commerce allows them to visit the same store virtually, with a few mouse clicks.

- **Speed**

  Doing business electronically is much faster than using the traditional methods. The speed of business transactions increases significantly as delays for communications is avoided by conducting transactions online. Business organizations can generate purchase orders and send it to suppliers online without delay. Suppliers can receive and process the order quickly. This also helps in reducing the shipping time. Information appearing on the Internet can be changed rapidly. This gives business organizations the ability to inform the customers of any changes in the product or services offered by the company.

E-commerce also provides the following benefits to the society :

- Buy from home, office or any place.
- Less travelling to purchase a product which further reduces pollution and traffic.
- Health care services.
- Distance learning and education.

Limitations of E-commerce

Although E-commerce has a list of advantages, it also has some limitations and disadvantages. However, with passage of time many of the limitations may disappear. Following are some of the limitations of E-commerce.

- **Resistance to change**

  Business organizations need to change from traditional business to E-commerce which is faced by a lot of resistance. People need to get used to paper-less and faceless business transactions.

- **Initial Cost**

  E-commerce requires significant initial investment in new technologies which can change the company's core business processes. It also requires investment in hardware and software technologies. The staff training is needed to understand the way of conducting the business using E-commerce.

- **Security**

  A primary concern in E-commerce is security. The data or information travelling on the Internet related to the company or an individual should be protected by unauthorised access. Internet provides universal access but companies should protect their assets from malicious or accidental misuse.

- **Privacy**

  The privacy of the customer's information is a serious issue. Customers hesitate to share their personal information on the Internet due to fear of misuse. Many times, the companies sell their database information to marketing companies and they in turn send unwanted (spam) mails to the customers. The hackers also might intercept the information on the Internet and misuse the data. Credit card frauds result into financial loss to the customer. The protection should be provided from hackers, viruses, transfer of data and transaction risks.

- **Lack of trust**

  Sometimes, business frauds like non-delivery of products, incorrect information of the product, lack of security for payment transactions creates dissatisfaction amongst the customers. Also returning defective goods purchased online can be difficult issue. The common concerns here are who pays the return postage, will full refund be made by the merchant and whether the product will reach to their original source.

- **Time for delivery of products**

  In a physical store, we select the product and leave with the product in our hands. In E-commerce we often buy products that are not available locally, which means the product needs to be delivered which takes time and cost money. The cost of shipping charges is an overhead in case the product is ordered from a distant place.

- Perishable products like fruits, vegetables and others are not preferred to be purchased online. Online store doesn't provide the opportunity to touch, wear or feel the product. Therefore online selling of such products like clothes and furniture might be tricky.

- Payments on E-commerce are most often conducted using credit card facilities and as a result very small and very large transactions are not preferred to be conducted online. Timely updates

of information and services are often a problem in case of small organizations trying to establish their business online. E-commerce is dependent on Internet and any problem in the Internet connectivity causes business risk.

E-commerce Business Models

The business models of E-commerce are defined based on the parties involved and the type of business activities or services provided. Many business models have been proposed for E-commerce, but the following are the most popular amongst all. Generally, the classification is done on the basis of who is selling to whom.

- Business to Consumer (B2C)
- Business to Business (B2B)
- Consumer to Consumer (C2C)
- Consumer to Business (C2B)

Let us understand each of them in detail.

Business to Consumer

Business to Consumer (B2C) refers to business and organizations that sells products or services to consumers over the Internet using websites. Consumers from anywhere can browse and order the products or services anytime. The sellers can sell products directly to the consumers. And the buyers are individual customers. This is the type of E-commerce that the consumers are most likely to see on Internet. Lately, it has gained a high popularity amongst consumers due to simplified and fast way to buy products. Apart from online retailing, B2C also includes services like online banking, real estate services, travel services and many more. Some examples of B2C websites are: amazon.com, rediff.com, fabmart.com, flipkart.com etc. Figure 4.6 shows the home page of Flipkart website. It is a popular website for purchasing variety of goods.

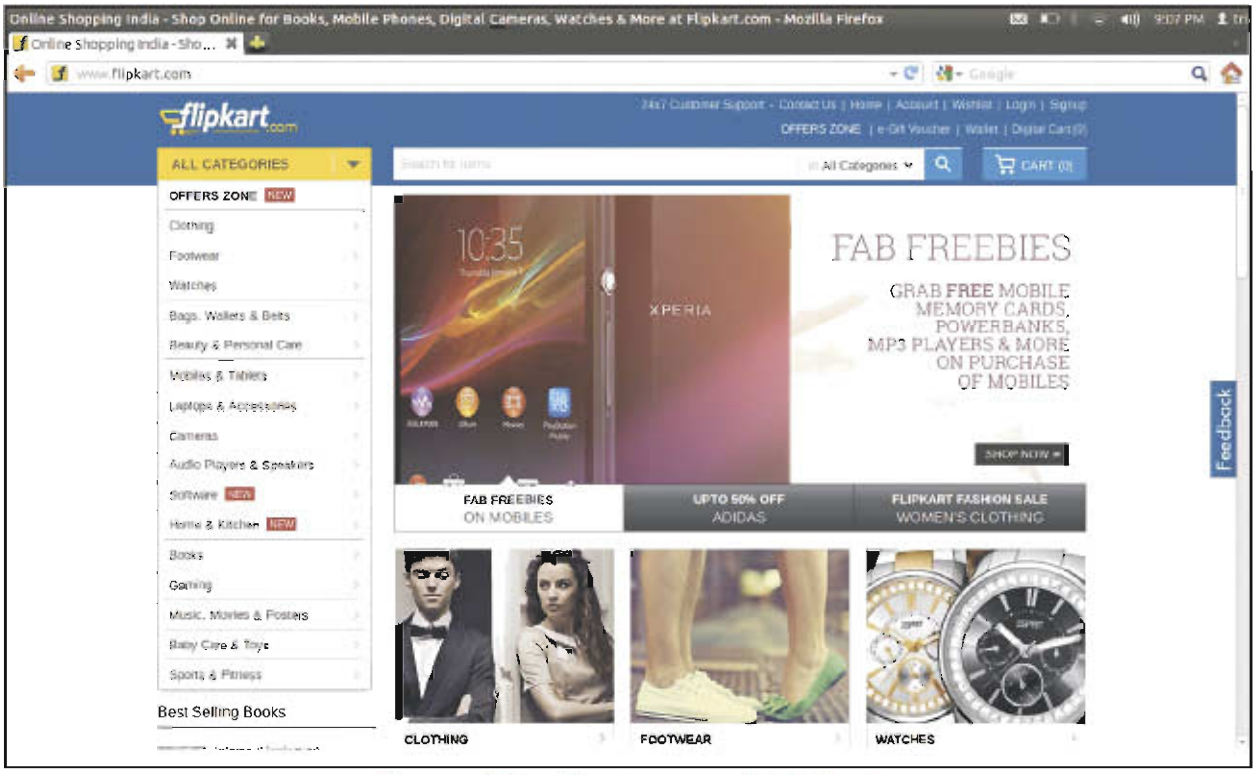


Figure 4.6 : Home page of Flipkart

As seen in the figure 4.6, the products are displayed category wise on the left side of the window. Select the category as per your choice. Here we have selected the eBooks category as shown in the figure 4.7. This will display all the eBooks in the catalog.

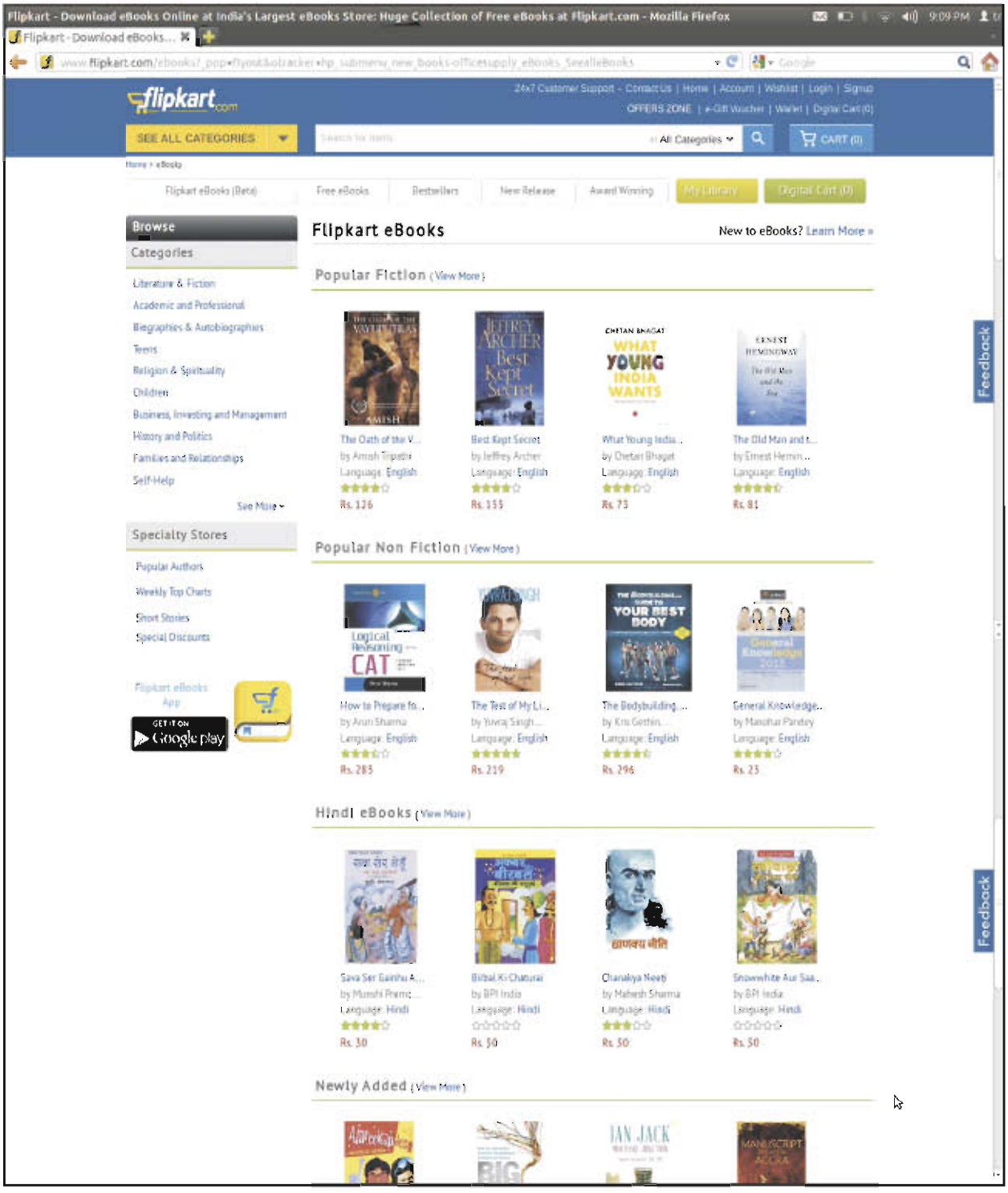


Figure 4.7 : Selecting eBooks category in Flipkart.com

When we click on icon of any of the eBook it shows the details like authors name, price, readers rating, book review, publisher and ISBN number as shown in the figure 4.8. We can also search a particular book from the different categories using the search tool provided at the topmost portion of the window.

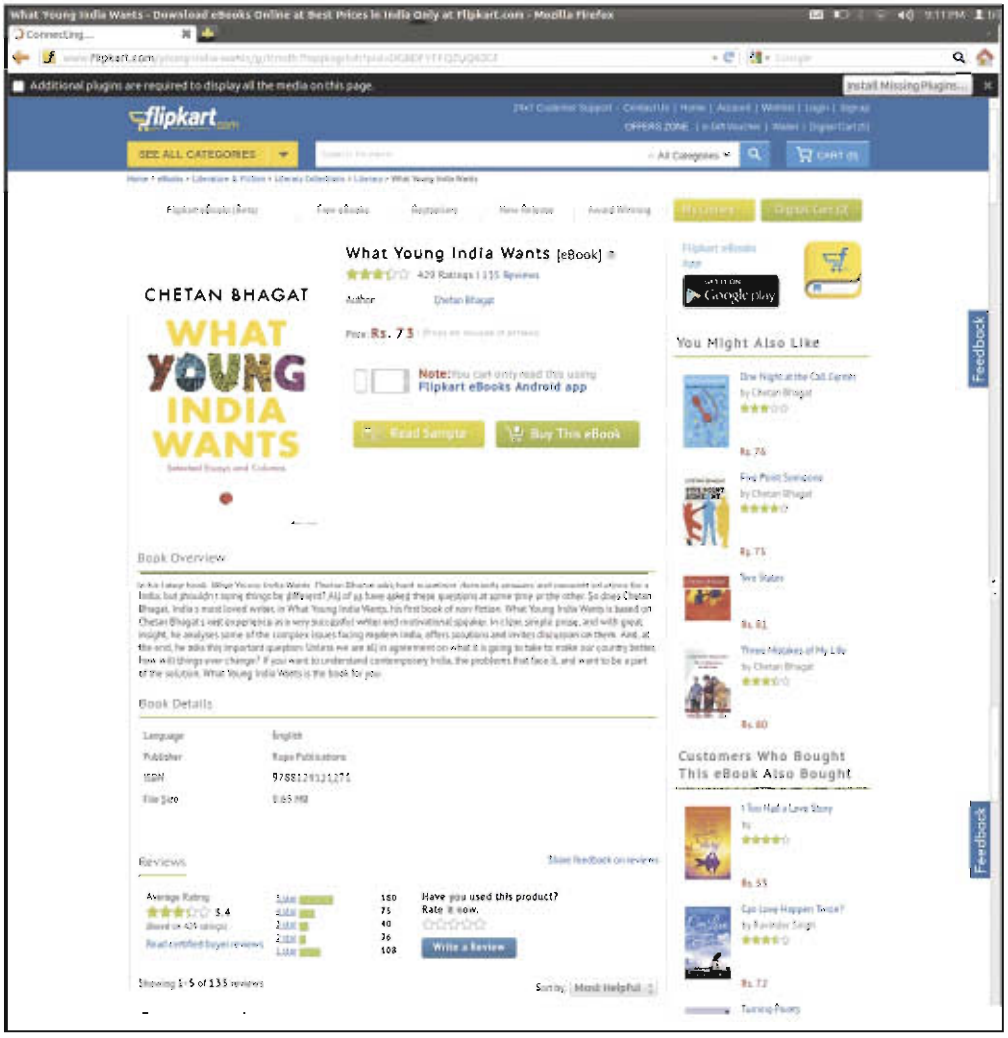


Figure 4.8 : Selecting an eBook

Pressing the button "Buy this eBook" adds the book in the shopping cart as shown in the figure 4.9.

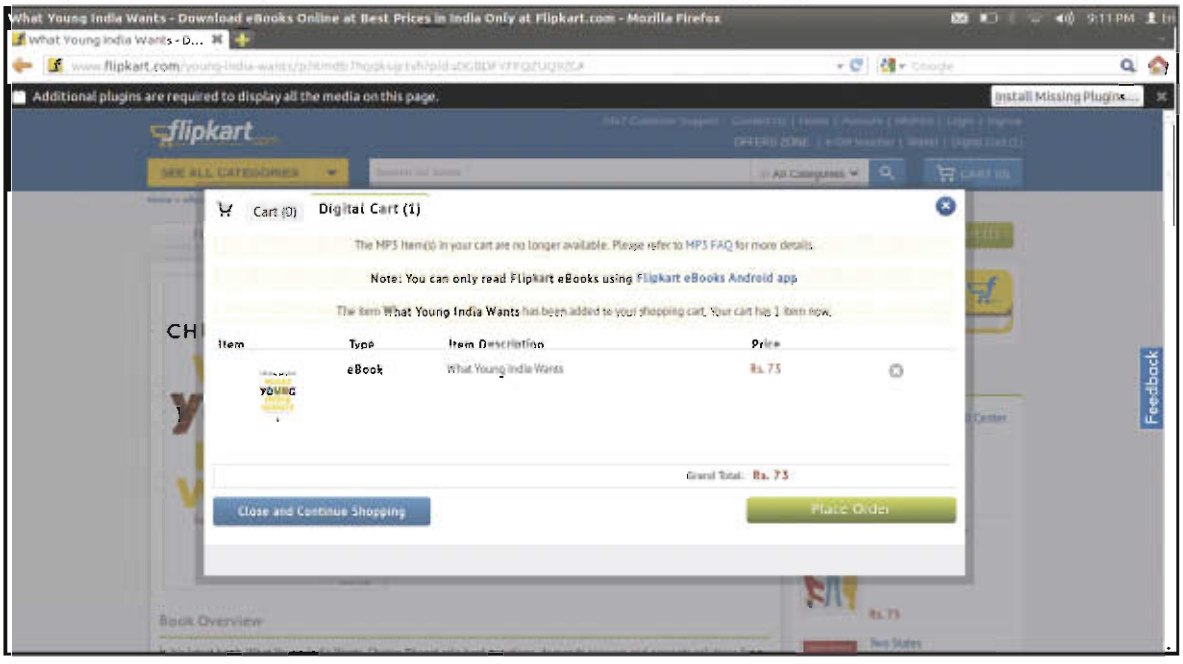


Figure 4.9 : Shopping cart showing the selected eBook

In similar way, you can add any number of products to the shopping cart from various categories you want to purchase. You can also delete a selected product; change the quantity of the product in your shopping cart.

To confirm the order, press the button "Place Order" as shown in figure 4.9. This will take you to the final screen where you are asked to create a login in case you have not created; provide the billing address, shows order summary and payment options. Once all this process is over, the purchase order is made and the eBooks are shipped to your address provided in the shipping details. Now-a-days you also have an option of "Payment on Delivery" option where you can pay for the product purchased when it is delivered at your address.

Business to Business

Business to Business (B2B) refers to E-commerce activities between different business partners. In B2B both the seller and the buyer are business entities. It enables the businesses to form E-relationship with their suppliers, distributors and other agents. This allows more transparency among the business entities involved and results in more efficiency. For example, a manufacturers deals with a supplier, a distributor and a wholesaler. The manufacturer using B2B can quickly communicate to the supplier about diminishing stock product, the supplier in turn can respond quickly to it.

With the help of B2B, the companies can improve the efficiency of common business activities like supplier management, inventory management, payment management and many more. It is an effective media for managing telemarketing, supply chain, procurement, just-in-time delivery, online services. Some examples of B2B websites are commodity.com and tradeindia.com.

Consumer to Consumer

Consumer to Consumer (C2C) refers to E-commerce activities involving transactions between and among the consumers. It enables the consumers to directly deal with each other through online auctions and classified advertisements without the involvement of third party. Any Internet user can become a vendor or purchaser at C2C websites. Auctions sites are a good example of C2C model. If we have any product to sell, we can get it listed at an auction site and others can bid for it. Some example of C2C websites are Ebay.com, OLX.com and Quikr.com

Consumer to Business

Consumer to Business (C2B) involves reverse auctions where the consumers determine the prices of the products or services. In this type of E-commerce, consumers have a choice of a wide range of products and services, along with the opportunity to specify the range of the prices that they can afford or are willing to pay for a particular product or service. The companies bid to offer the products or services to the consumer. This helps in reducing the bargaining time while increasing flexibility for both the consumer and the company. C2B uses Internet to reverse the normal buying process wherein the consumers decide what they are willing to pay and business decides whether to accept it or not. Some examples of C2B websites are bidstall.com, JeetLe.in

There are other variations to E-commerce business models. If we consider Government as a separate entity, then we can have :

- Government to Business (G2B)
- Government to Citizen (G2C)
- Government to Government (G2G)

Government to Business

Government to Business (G2B) refers to the services and information provided by the government to the business organizations through vast network of government websites. A business organization can get all the information related to business policies, approvals for starting the business, setup requirement and other specifications from these websites. Various forms can be filled and submitted online to the related government office. For example, income tax department of Government of India has their website www.incometaxindia.gov.in wherein all the rules related to tax; different forms and facility for submission of online tax returns are provided. This business model is a part of E-governance initiative.

Government to Citizen

Government to Citizen (G2C) is also a part of E-governance. The objective of this business model is to provide good and effective services to individual citizen. The website provides information regarding various government departments, various welfare schemes, different application forms to be used by the citizens etc. The Gujarat Government has developed its own network called Gujarat State Wide Area Network (GSWAN) for this purpose and is available online on www.gswan.gov.in. Figure 4.10 shows the home page of GSWAN.

Figure 4.10 : Home page of GSWAN website

Government to Government

Government to Government (G2G) refers to online non-commercial communication between the Government agencies, organizations and departments with other Government agencies, organizations and departments. The sharing of information helps in reducing IT costs, streamline procedures and government offices can be more efficient.

From all the E-commerce models described above, B2C and B2B are the two most widely used models. The major difference between the two models is the consumer. In B2B model, the consumers are other companies while in B2C model the consumers are individuals.

Summary

In this chapter we discussed about E-commerce, its advantages and disadvantages, applications of E-commerce and its business models. E-commerce can be defines as the use of Internet for conducting business activities. E-commerce is used for marketing and selling, purchasing of products, online newspapers, information services, support services, net banking and many more. The business models of E-commerce are classified based on the parties involved and the type of business activities or services provided. We studied various models like, Business to Consumer (B2C), Business to Business (B2B), Consumer to Consumer (C2C), Consumer to Business (C2B) and Government to Business (G2B).

EXERCISE

1. Define E-commerce. List some of the applications of E-commerce.
2. What is online auction?
3. What is electronic newspaper? List some of the features of the website.
4. List some online bookstores and discuss the features of any one.
5. What is net banking? Name some of the websites for online banking.
6. State the difference between traditional commerce and E-commerce.
7. State the advantages of E-commerce.
8. Write about the following E-commerce business models. Also give example for each.
   (1) Business to Consumer (B2C) (2) Business to Business (B2B)
   (3) Consumer to Consumer (C2C) (4) Consumer to Business (C2B)
   (5) Government to Business (G2B)
9. Choose the most appropriate option from those given below :
   (1) Which of the following is an example for online bookstore ?
   (a) Amazon (b) irctc (c) Gmail (d) yahoo
   (2) Which of the following is newspaper that exists on the Internet in digital form ?
   (a) l-newspaper (b) Internet-newspaper
   (c) www-newspaper (d) E-newspaper

(3) Which of the following is the process of buying and selling products by offering the customers to bid the price ?

(a) Marketing (b) Auction (c) Bookshop (d) Booking

(4) Which of the following is known as the process of conducting the banking transactions over the Internet ?

(a) Auction (b) Bidding (c) Net banking (d) www-banking

(5) Which of the following is a feature of traditional commerce ?

(a) Operates within a certain period of time or during business hours.

(b) Advertising of the product is done electronically.

(c) E-payments systems are used for receiving payment.

(d) Customers can browse through products and offers.

(6) Which of the following is a feature of E-commerce ?

(a) Operates within a certain period of time or during business hours.

(b) No sharing of information with competitors.

(c) Location renting or purchasing.

(d) Advertising of the product is done electronically.

(7) Which of the following is not an advantage of E-commerce ?

(a) Lower cost (b) Conduct business 24×7

(c) Security (d) No geographical limitations

(8) Which of the following is a disadvantage of E-commerce ?

(a) Privacy (b) Improved customer service

(c) Speed (d) Conduct business 24×7

(9) Which of the following E-commerce business model refers to business and organizations that sell products or services to consumers over the Internet using websites ?

(a) Business to Consumer (B2C) (b) Business to Business (B2B)

(c) Consumer to Business (C2B) (d) Government to Business (G2B)

(10) Which of the following E-commerce business model refers to activities between different business partners ?

(a) Government to Business (G2B) (b) Consumer to Business (C2B)

(c) Business to Business (B2B) (d) Business to Consumer (B2C)

(11) Which of the following is a good example of C2C model ?

(a) Auction sites (b) E-newspaper

(c) Online purchasing (d) Information services

(12) Which of the following E-commerce business model refers to E-commerce activities involving transactions between and among the consumers ?

(a) Government to Business (G2B) (b) Consumer to Consumer (C2C)

(c) Business to Business (B2B) (d) Business to Consumer (B2C)

(13) Which of the following E-commerce business model involves reverse auctions where the consumers determine the prices of the products or services ?

(a) Consumer to Business (C2B) (b) Business to Business (B2B)

(c) Consumer to Consumer (C2C) (d) Government to Business (G2B)

(14) Which of the following e-commerce business model is also a part of E-governance ?

(a) Business to Business (B2B) (b) Consumer to Business (C2B)

(c) Consumer to Consumer (C2C) (d) Government to Citizen (G2C)

(15) Which of the following E-commerce business model refers to online non-commercial communication between the Government agencies, organizations and departments with other Government agencies, organizations and departments ?

(a) Business to Business (B2B) (b) Consumer to Business (C2B)

(c) Government to Government (G2G) (d) Consumer to Consumer (C2C)

Introduction to M-Commerce 5

Mobile Commerce also known as M-commerce. It refers to buying and selling of goods or services through the use of Internet enabled wireless devices such as a Mobile phone, Personal Digital Assistant (PDAs), Smartphone, Tablet, Palmtop or any other mobile device.

M-commerce provides the user with the advantage of flexibility and ubiquity. By using the mobile phone, consumers can conduct business transactions without being fixed at a computer terminal or being physically present at the shop. These devices are carried by the user wherever he/she goes, making it possible to access the Internet from any place. It allows real time transactions while on the move. As the popularity of smart phone's and tablets is increasing day by day, more users are moving towards the use of M-commerce. Some of the examples of M-commerce are:

- Purchasing airline tickets
- Purchasing movie tickets
- Restaurant booking and reservation
- Hotel booking and reservation
- Stock market analysis

Banks and other financial institutions are increasingly using M-commerce to retain their business. They allow their customers to access account balance, stock quotes, make transactions via mobile phones. This service is known as Mobile Banking or M-Banking. The stock market services offered via mobile devices is also becoming popular and known as Mobile Brokerage. News information, sports, entertainment, shopping and reservation areas have also grown with the demand for mobile related services.

Benefits of M-Commerce

Mobiles are being used more and more on daily basis and today it is not merely used to make or receive a call. Mobile companies are coming up with new features for their smart phones, which offer consumers ease, flexibility and security at the same time. The ease of availability and faster speed has made M-commerce more popular now-a-days. The web design and development companies have also optimized the websites, such that it can be viewed correctly on mobile devices.

M-commerce is the integration of wireless networks accessed through handheld devices and Internet. The benefits of Internet and E-commerce are offered by M-commerce also. Some of the advantages of M-commerce as listed :

- It provides convenience to the user. In just a few clicks on the mobile device, the user can do shopping, banking and download media files while on the move.

- Mobile device enables the user to be contacted at virtually anytime and anywhere.
- Reduces transaction cost.
- Reduces the time to order. The user does not need to be on the PC or laptop to order.
- Streamline business processes.
- Provides global reach.
- Conduct business 24*7.
- Flexibility of accessing the information through any mobile devices.
- Payment can be done using the mobile devices itself just the same way as we do it on personal computer.
- Useful to deliver time critical and emergency information.
- Easily identifies the physical location of the handheld device. The emergence of location based applications enable the user to receive relevant information. We will discuss about location based services later in the chapter.
- Customized alerts can be easily received on the mobile device.
- Instant connectivity and availability of faster 3G services has made M-commerce more popular these days.
- Timely information can reach the user. The information like flight or train schedule, delay or cancellation can be given to the user on his mobile device on real time basis.

Limitations of M-commerce

Though the list of advantages of M-commerce is large it has a number of limitations. Some of the limitations of M-commerce are listed below :

- The handheld devices commonly used today offer a limited screen size. This further limits the types of file and data transfer. At times it is difficult to display videos.
- User interface is less convenient when compared to personal computers.
- Mobile devices have limited computing power, memory and storage capacity.
- It operates over wireless networks which are less secured as compared to wired network.
- It offers a limited bandwidth.
- High cost of establishing mobile and wireless broadband infrastructure.

Applications of M-Commerce

Applications of M-commerce are increasing rapidly. It is used today in many areas not limited to the ones that are discussed here. More and more applications will be introduced in the near future. Let us discuss some of the applications of M-commerce.

Mobile Marketing and Advertising

Today, companies are using M-commerce to expand their services from marketing to advertisement. Mobile advertising is one of the most popular ways for companies to reach a large audience. Advertising on Internet has become a major source of revenue for most of the portals. Many retailers are offering location based mobile advertising in order to target consumers and increase their sales. An advertisement placed on the mobile device of the user can thus be made on personal requirements and location-specific. It can update the users about the various discounts and schemes available in the nearby areas of the current location of the user.

Mobile Ticketing

Users can easily buy tickets for air or rail travel, movies etc. The tickets can be sent to the user's mobile device. Users can further show these tickets on their mobile devices at the respective place. Tickets can also be easily cancelled on the mobile phones using the application or accessing the portals of travel agents. This helps in reducing the traffic and the parking problems which are increasing day-by-day as the user need not travel to the place for buying tickets.

Figure 5.1 shows the Indian Railway Catering And Tourism Corporation limited (IRCTC) login page.

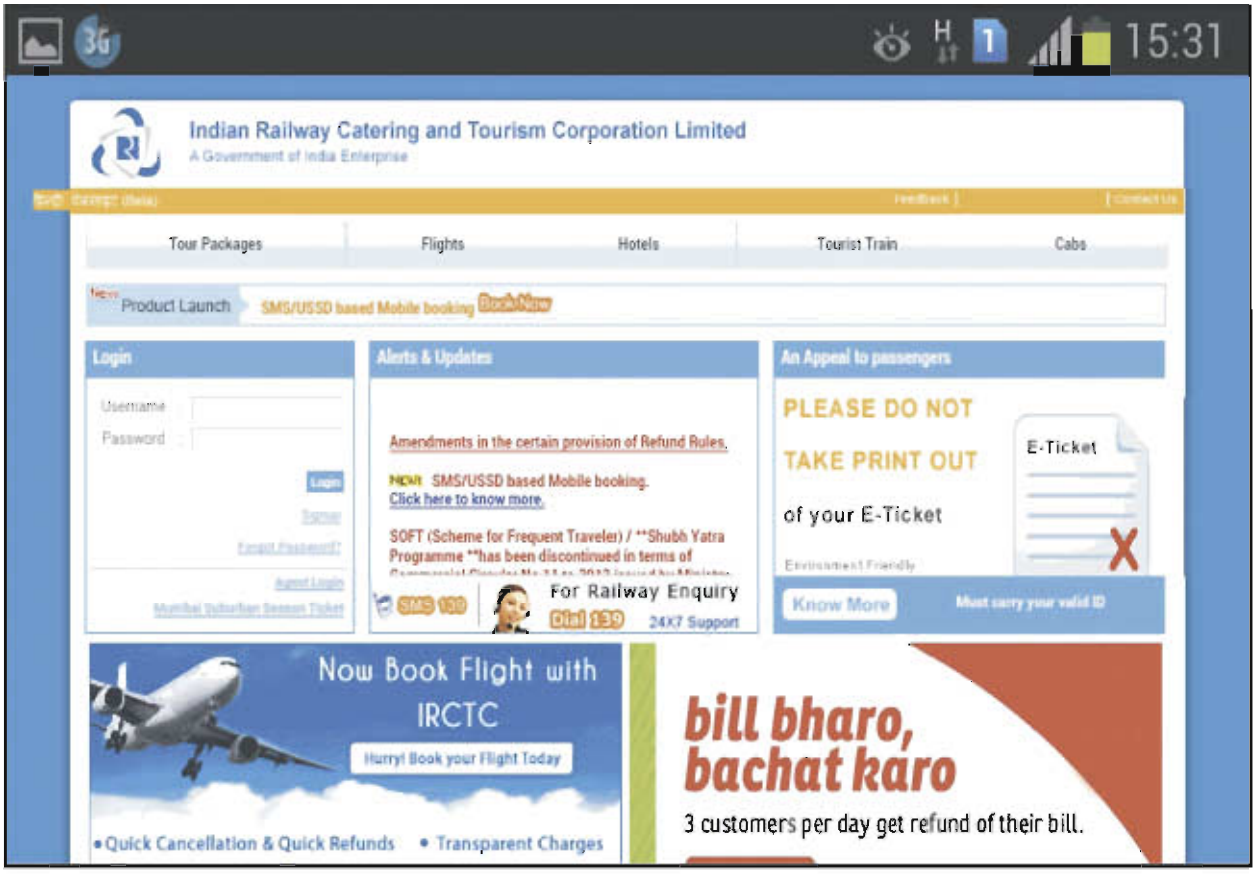


Figure 5.1 : IRCTC login page

Let us try to book an online ticket using the mobile device. After providing the username and password (note: the user already has an account), the user can give his plan travel details like source and destination name, date of travel, ticket type and quota as shown in figure 5.2.

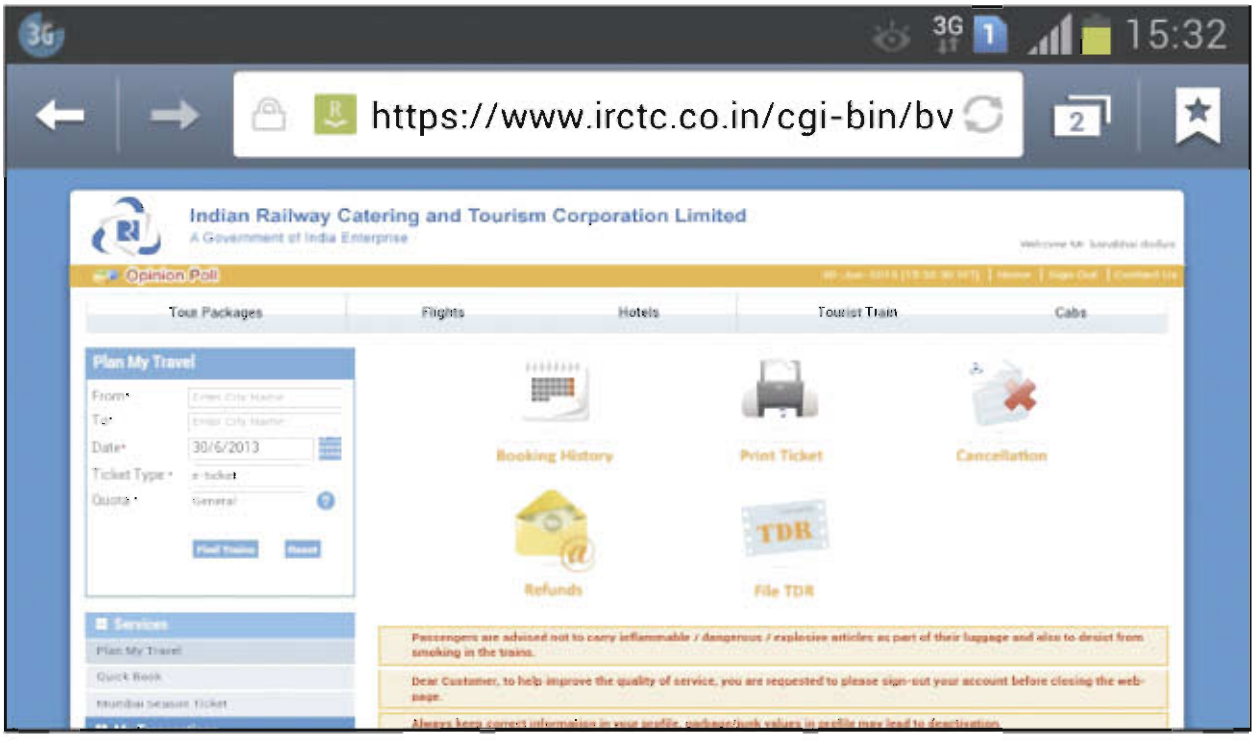


Figure 5.2 : Details of Journey

When the user clicks on Find Trains button he/she is provided with the list of trains available on the given date as per his/her requirement as shown in figure 5.3.

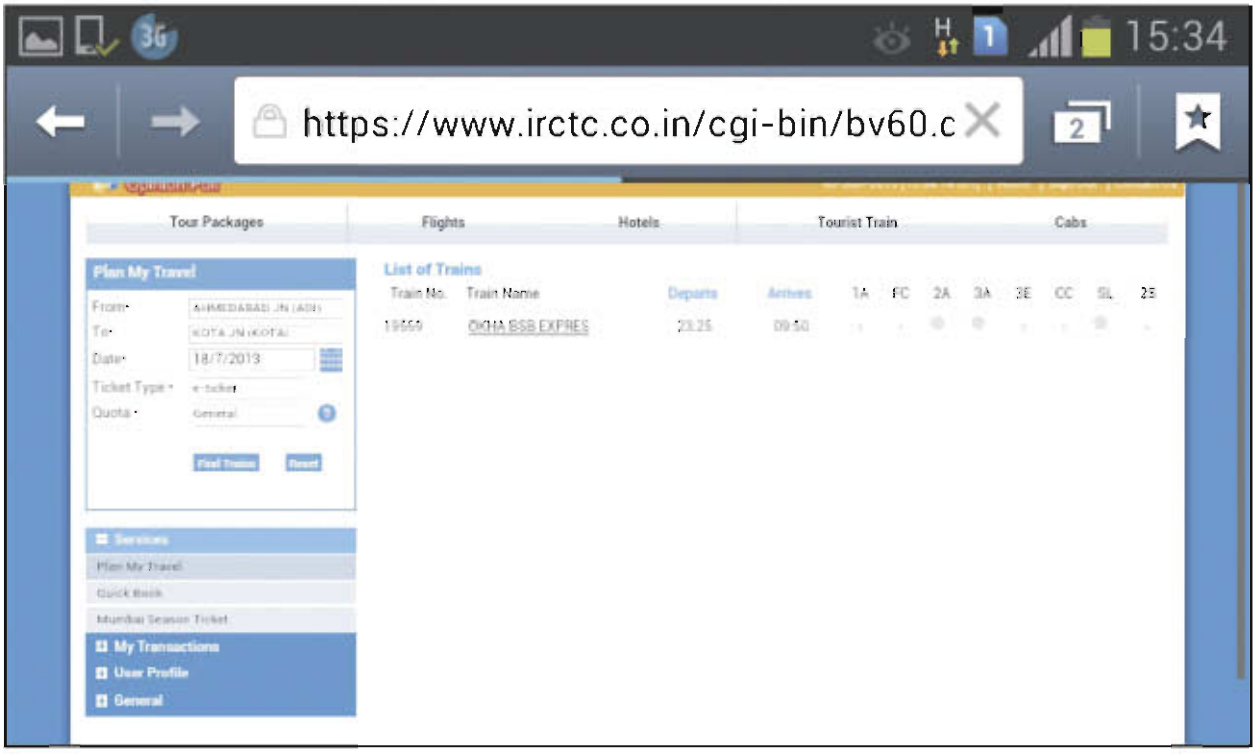


Figure 5.3 : Details of the journey with available train list

The user can click on the name of the train to get more details. In figure 5.3 we can see name of only one train. It is possible to get list of more trains also. Later, he/she can proceed to book the ticket and provide the details of the passengers travelling as shown in figure 5.4.

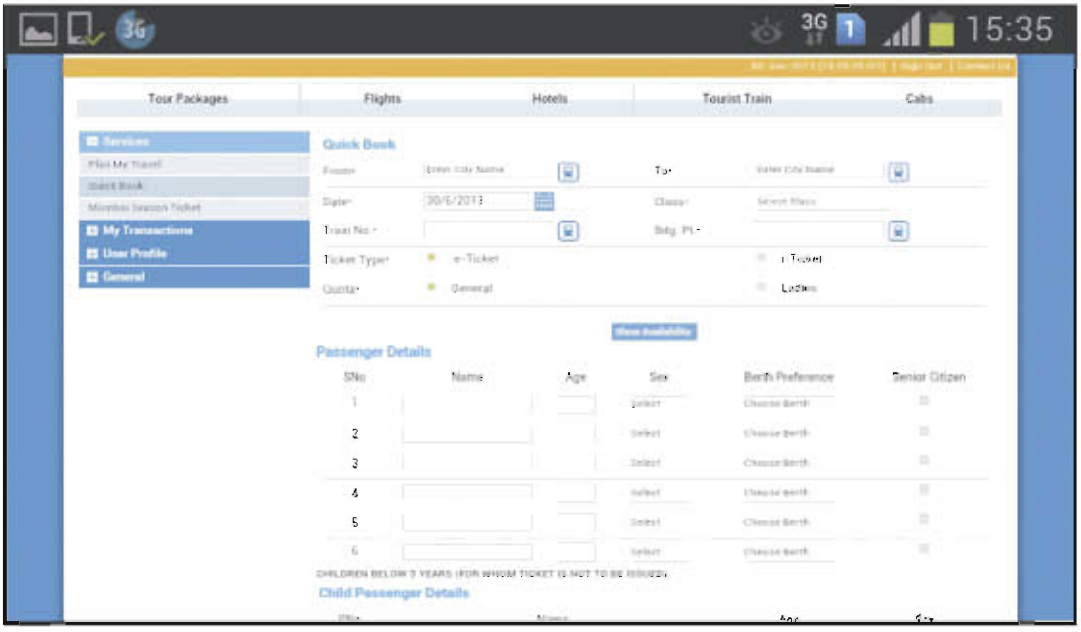


Figure 5.4 : Enter details of passenger for ticket booking

After booking, the user gets an e-ticket message from IRCTC on the mobile device which can be presented at the time of travelling. As you can see, without being fixed at the computer terminal or being physically present at the railway station, the customer can access the services using the mobile device whenever and wherever he/she goes.

Mobile Auctions

The auctions sites are becoming more popular these days. Mobile devices further help in increasing the reach of these auction sites. A user while on the move can access these sites, make a bid, monitor bids and take a timely action on the bidding process. Many of the auction sites have built gateways and interfaces to provide access to mobile devices through wireless networks. Figure 5.5 and 5.6 shows the auction site ebay as viewed on a mobile device. The user can bid for the products using the mobile device.

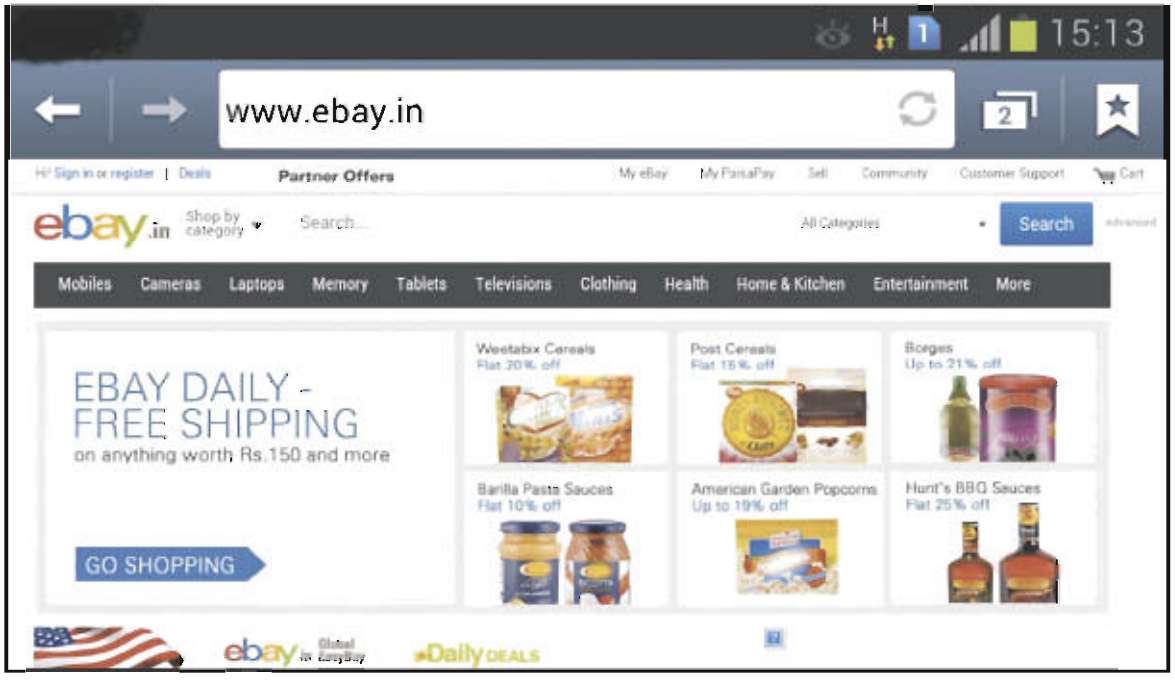


Figure 5.5 : Home page of ebay

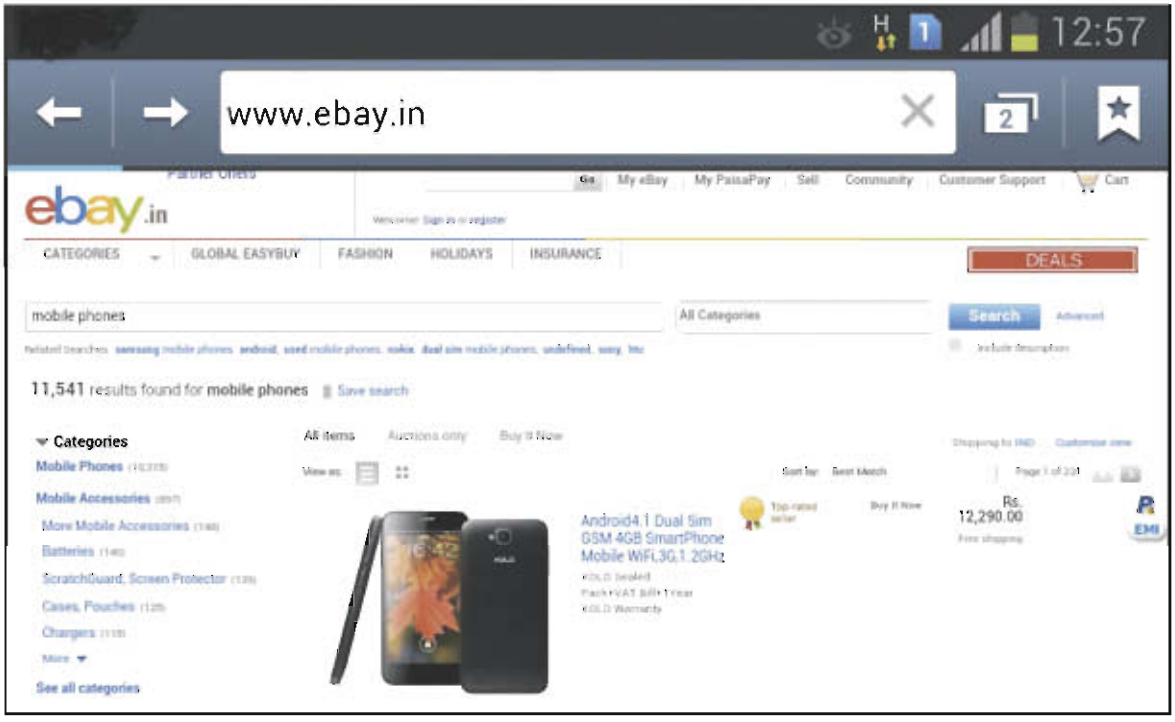


Figure 5.6 : Bid for a product

Mobile Entertainment

Mobile devices are used extensively for listening to audio, viewing video and playing games. The mobile users can subscribe to entertainment libraries where they can search for songs, videos or games and easily download them in their device for playing later. Entertainment services such as pay-per-download, pay-per-event or on subscription basis can cater to a large number of mobile users and are willing to pay for the services.

Mobile Purchase

Mobile purchase allow customers to shop online anytime anywhere. Customers can browse and order products while using a secure payment method. Instead of using paper catalogue, retailer can send a list of products that a customer would be interested in, directly to their mobile device. Alternatively, the consumers can also visit a mobile version of a retailer's E-commerce site. The retailers can also track the customers and notify them of discounts at local stores that the customer would be interested in. Figure 5.7(a) shows the home page of flipkart as seen on mobile device. Figure 5.7 (b to f) shows the purchase process from selecting a category, choosing the product, viewing its detail, placing the order and provide the shipping details on the mobile device.

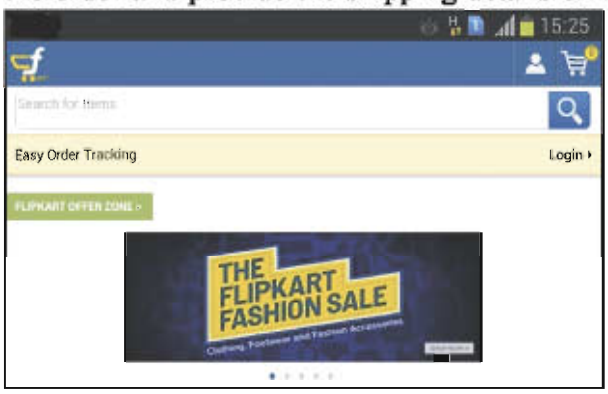


Figure 5.7(a): Flipkart home page

Figure 5.7(b): Select a category

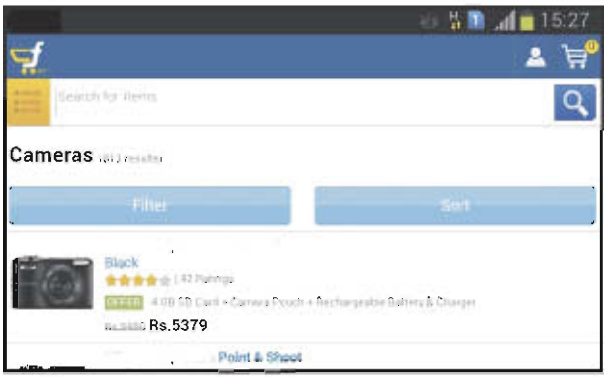


Figure 5.7(c): Products in the category

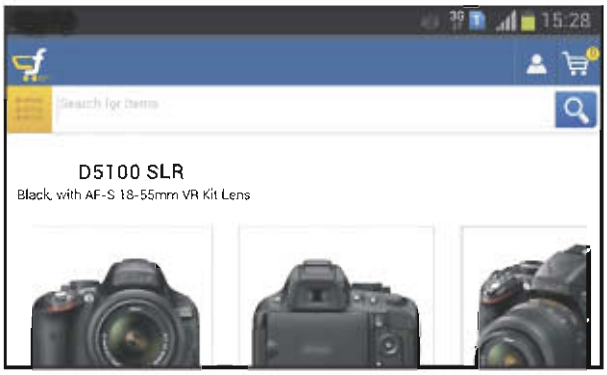


Figure 5.7(d): Select a product

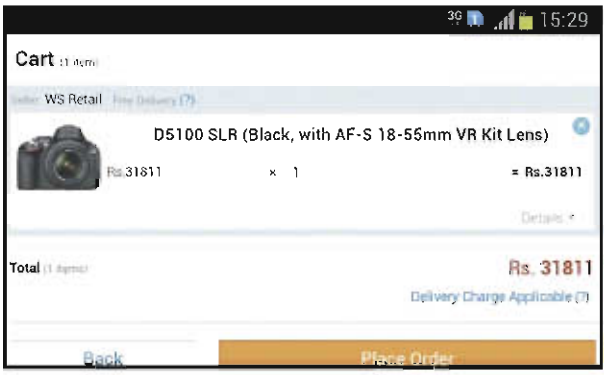


Figure 5.7(e): Place order

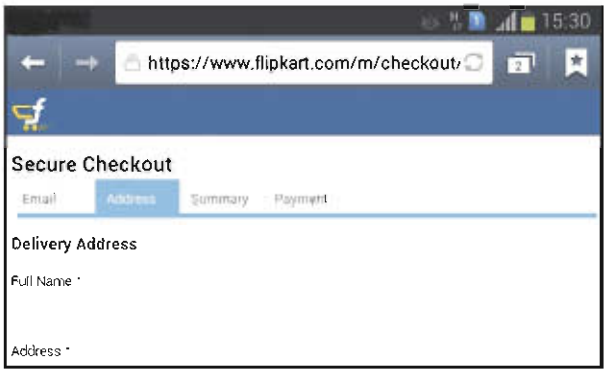


Figure 5.7(f): Provide customer details

Mobile Financial Services

Today, many popular banks and financial institutions use M-commerce. They allow their customers to access account information; perform transactions like stock purchase, remit money, via mobile phones and other mobile equipments. Figure 5.8 shows the mobile services provided by Union bank. The customer can download the application "umobile" from the android market. After getting registered, the customer can avail the services that are provided by the bank on his mobile as shown in figure 5.9. In figure 5.10, you can see the various main menu options. The customer can check his/her balance, transfer the funds to any other bank account, place request for a cheque book and can avail many other services from his/her mobile device.

Figure 5.8 : Mobile Financial services

Figure 5.9: Login using password

Figure 5.10 : List of various services

Mobile Information Services

A wide variety of information services can be delivered to mobile phone users in much the same way as it is delivered to personal computers. The services include :

- News service
- Stock market data
- Sports news
- Financial records
- Traffic information

Location and search service

The location of the mobile phone user is an important piece of information used during mobile commerce transactions. For example, a consumer wanting to buy a Smartphone within a certain price limit and specifications would be interested in knowing the nearest location of the store where he can get the desired product. In this case, it is important that the location and search service should be able to provide the list of stores in the city or nearby areas of the mobile user's current location. Knowing the location of the mobile user allows vendors to provide location based services such as local maps, local offers, local weather, people tracking and monitoring. Mobile devices can also be used to get directions to particular place, movie theatre, restaurant, hospital or other such amenities. Let us learn about location based search which is known as L-commerce.

L-Commerce

With more and more wireless handheld devices like PDA's, Cellular phone and pocket PC's there are significant opportunities for the growth of M-commerce. Although mobile commerce enables access to goods and services regardless of the location of buyer or seller, in many situations the specific location of the buyer and seller is important to the transaction. Today various location-specific applications and services are emerging. These applications track the user's location in order to deliver a service or product. The use of technologies which provide the location information for business purposes is known as L-commerce.

The technology uses the geographical location of the mobile device to determine which applications are appropriate based on that area. It enables users to log their locations, track the location of another person, and find places such as a bank or restaurant. The technology works by using signals

from GPS, cellular and Wi-Fi sources. The Global Positioning System (GPS) is the most accurate in determining a mobile device's position. It is based on a worldwide satellite tracking system where the GPS signals are generated by a group of satellite that orbits around the Earth. To locate a point, a mobile device will utilize three satellites to create an intersecting point that locates the device within 500 meters. This is known as triangulation. If the GPS signal is poor, weak or blocked, the mobile device can use the signals from cell towers and Wi-Fi hot spots. These signals do not broadcast their own locations but Smartphone companies use databases that store the locations of these sources.

Figure 5.11 shows the application which tracks the user's location on the mobile device using GPS. (Note: GPS must be enabled on the device). The advertisements seen on the mobile device in the figure 5.11 are also location specific. In the figure the user's current location is CG road in Ahmedabad. If the user wants to search the restaurants that are close to his current location, he selects the restaurant option in the figure 5.12. This will show a list of restaurants that are near to his/her current location. The user can also specify the distance as shown in figure 5.13 to search within a specific range of distance only. This will further help in filtering the search results as shown in figure 5.14.

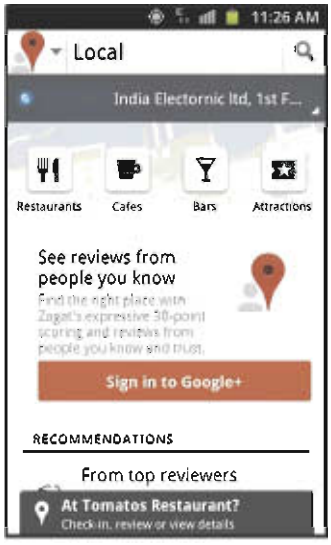


Figure 5.11 : Location specific application

Figure 5.12 : Search nearest restaurants according to user's location

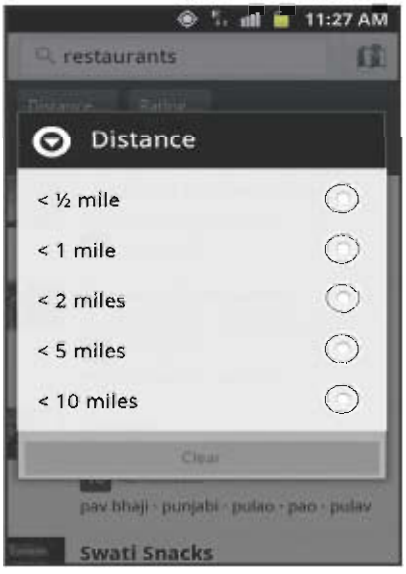


Figure 5.13 : Filtering the search result

Figure 5.14 : Search result after filtering

Let us discuss another example of location based services, where we can use maps to find the path to the destination. For example, the user wants to reach a particular destination whose path is not known to him. Using the maps, the location based services tracks the user's current location as seen in figure 5.15. The current location is indicated using blue color pointer. Now to find the destination we can either type or speak the destination name as shown in figure 5.16.

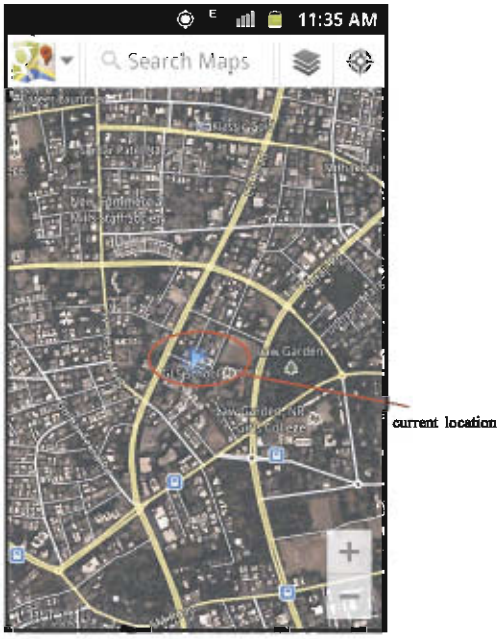


Figure 5.15 : Use of Maps to find the user location

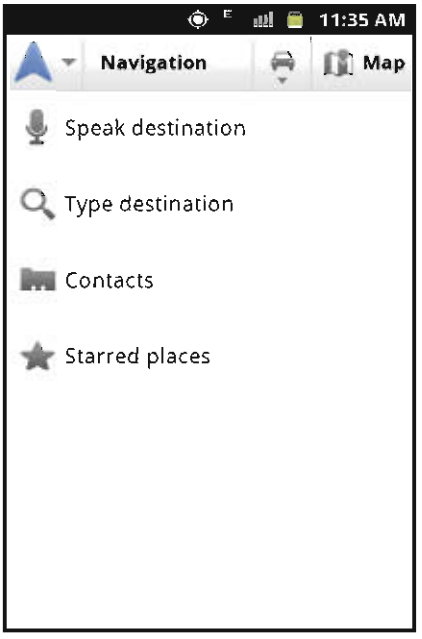


Figure 5.16 : Options to enter destination

Figure 5.17 : Select the destination

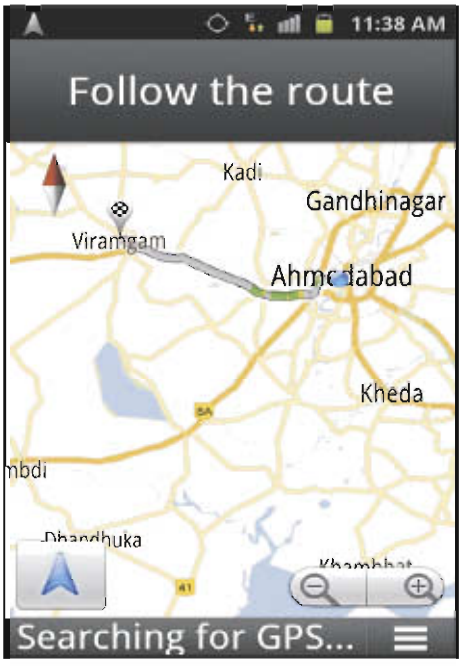


Figure 5.18 : Route to the destination

As seen in figure 5.17, the destination "Viramgam" is selected. The map shows the route from source to destination as shown in figure 5.18.

Figure 5.19 : Distance and approximate time to reach the destination

Figure 5.20 : Driving directions

The distance to the destination and the approximate time to reach the destination is also shown in figure 5.19. We can also see the driving direction to the destination as shown in figure 5.20.

Location based services offers convenience and opportunity to provide services that are more quick or precise and can meet a customer's needs. Some examples of location-based applications are:

- Information or directory services: dynamic yellow pages automatically inform the users about the nearest restaurants, parking facility, traffic updates.
- Tracking services: tracking of assets, locating friends in a geographic location, tracking stolen cars, tracking of children by parents.
- Emergency services: emergency medical ambulance, search and rescue mission, roadside assistance, police and fire response.
- Advertising promotion: targeted ads, promotional messages, customer identification in a store.
- Mapping: Creating maps of specific geographical location
- Navigation: plotting route from one place to another

Security Issues in E-commerce & M-commerce

Internet is a public network system that consists of thousands of private computer networks connected together. These private computer networks are exposed to potential threats from anywhere on the public network. Internet provides a good opportunity to the businesses but along with the convenience also come new risks. The valuable data or information that travels on the Internet may be misused, stolen, corrupted or lost. For example, while making online purchase on the E-commerce website, customer needs to provide the credit card number and the personal details. This information is transmitted to the merchant server. The merchant server sends it to the issuing bank for authorization through payment gateway. All these transmissions occur on the public network i.e. Internet. An

unauthorised user may read credit card number during the transmission and misuse it later on. Also, there are possibilities that order information might be changed in between. If the customer has ordered 10 items, and somehow the merchant receives order of 100 items, he would ask us to pay for 100 items. An intruder can steal or tamper information anywhere in the world while sitting on his computer. He can create new programs and run them on remote computers causing it to malfunction or break down in worst cases while hiding his identity.

E-commerce/M-commerce sites have to keep their online data such as customer's personal details, their bank details and many more safe. They have to be aware of all the frauds that are taking place now-a-days. As E-commerce deals with payments such as online banking, electronic transactions, using debit cards, credit cards and many others; the E-commerce/M-commerce websites have more security issues. They are at more risk of being targeted than other normal websites. Thus, it becomes very important to secure the data on Internet. The E-commerce/ M-commerce security must meet four important aspects as mentioned below:

- **Confidentiality**

  It refers to the secrecy of the information so that unauthorized user cannot read it. It is achieved by using cryptography in which all the messages transmitted are encrypted and only the receiver can read it after decrypting the message using appropriate key. This protects the data from private attacks and ensures that the message is not revealed or leaked to anyone as it travels to the destination. It helps in protecting the confidential data like credit card number.

- **Integrity**

  It ensures that the information must not be accidentally or maliciously altered or tampered in transit. The receiver should receive the same message as was sent by the sender. If the message is altered in between the transition, it should be detected. This removes the problem of modifying the order quantity in between and later creating the payment problems.

- **Authorization**

  It ensures that only authentic users are allowed to use the system. The login and password is one of the ways to achieve authentication.

- **Non-repudiation**

  It ensures that the sender of the message cannot deny that he/she has sent the message. It prevents sender or receiver from denying a transmitted message when in fact they did send it. For example, if the customer denies of sending a purchase order for any reason, then it can be proved that the customer has send the message. It is usually accomplished via digital signatures or a Trusted Third Party (TTP).

Internet Security Threats

The most common threats that are faced on Internet are :

- **Malicious code**

  Malicious code is one that causes damage to a computer or system. Malicious code can either activate itself or be like a virus requiring a user to perform an action, such as clicking on something or opening an email attachment. It can also affect a network, send messages through email and steal information or cause even more damage by deleting files.

- **Sniffing**

  A sniffer is a program that uses Internet to record information that passes through a computer or router in transit from sender to receiver. Using a sniffer program is like tapping the telephone wire and recording the conversation. Sniffer programs can read e-mail messages, user login, password and credit card numbers.

- **Denial of service attack**

  A Denial-of-Service (DoS) attack is an attack used to shut down a machine or network, making it inaccessible to its intended users. By targeting the user's computer and its network connection, or the sites which the user tries to access, an attacker may be able to prevent the user from accessing email, websites, and online accounts like banking or other services that rely on the affected computer. The users are flooded with hundreds and thousands of messages that create traffic problem on the network.

- **Cyber Vandalism**

  Cyber vandalism is the electronic defacing of an existing website page. An attacker replaces the website's original content with his/her own content. It is an example of integrity violation. It is the electronic equivalent of destroying property or placing graffiti on someone's photograph. Today, there are so many cases of cyber vandalism where the business content is replaced by offensive material.

- **Spoofing**

  Spoofing or masquerading is pretending to be someone you are not, or representing a website as authentic when it is actually a fake. It is a technique where the attacker tries to assume the identity of another person or system for transacting with victim site. For example, an attacker can create a fake website as www.gswan.co.in and substitute his IP address for the real website IP address. All the user's visiting to the real site will then be redirected to the fake website.

Security measures

To prevent the various security threats many security measures are taken. Let us discuss some of them.

- **Antivirus software**

  Antivirus software is a computer program that detects, prevents and takes action to remove the malicious codes like viruses, worms and trojan horses from the infected system. To protect your computer you need good antivirus software. A system without antivirus software can easily be targeted by malicious code within a short span of time on Internet. The problems and damage that are caused by an infection can be extremely varied. The infection may be simple as causing strange noises, pop-ups and other annoying things on the system. It may delete the files and slow down the system or also can damage the hardware or destroy the entire computer system. Once a system is infected by virus, it will spread by attaching to other programs and files within the system. Viruses not only replicates itself within the system but can also spread to other systems by taking control of the users email and sending out copies of itself to those in the users contacts list. The most common way a system is attacked is through infected attachments

to email. These attachments can be in the form of pictures, videos, sound clips or any other type of file that can be attached to an email. Infections can also be spread through Internet downloads.

To prevent harm to the system, the antivirus software scans the downloaded files and the entire computer system in order to detect the presence of an infection. Today, we have antivirus software for mobile devices also due to prevalent use of Internet on these devices. You can find a large range of antivirus software in the market. Antivirus software is critical to be installed and kept updated regularly on the computer.

- **Firewall**

Companies having their own websites have to control the access to the network services both inside and outside the company network. The most commonly used network protection barrier between the company's network and the outside world is a Firewall. As shown in Figure 5.21, firewall is a device (a computer or a router) placed between the network and the Internet to monitor and control the traffic between the company's local network and the outside world. The primary goal of a firewall is to keep intruders away from the company's E-commerce infrastructure. It ensures that the company's crucial data is kept safe and not accessed by the intruders.

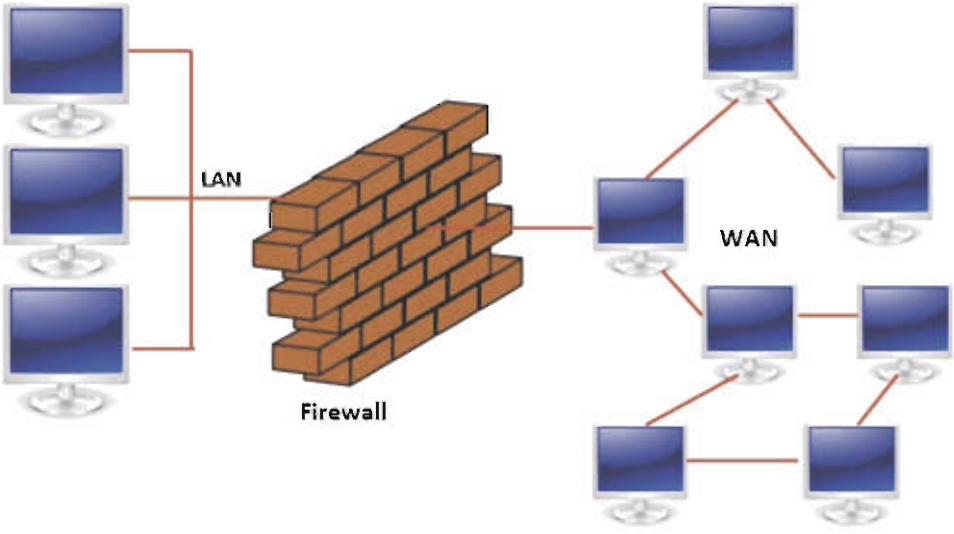


Figure 5.21 : Firewall placed between local and public network

A firewall protects the local network against the following :

- Email services that sometimes create problems.
- Undesirable material like photos, videos entering into local network.
- Unauthorized persons gaining access to local network.
- Unauthorized data or information leaving the company's network.
- Blocks the traffic from outside world to the local network.
- Protects from any type of network attack.

- **Digital Certificate**

Digital Certificates or Digital ID are used for proving our identity in electronic transactions. Just as we have a driving license or a passport to prove our identity in the real world. With a Digital Certificate, we can assure the business organisations, online services and friends that the electronic information they receive from us are authentic. Digital certificate is issued by a trusted third party to establish the identity of the holder. The third party who issues certificates is known as a Certification Authority (CA). Digital Certificate contains the holder's name, a serial number, expiration dates, a copy of the certificate holder's public key which is used for encrypting messages and digital signatures, and the digital signature of the certification authority so that a receiver can verify that the certificate is real.

- **Cryptography**

Cryptography is an art of protecting the information by transforming it into an unreadable form. Encryption is the transformation of normal text known as "plain text" into unreadable or secret text known as "cipher text" using encryption algorithm. A secret key is used to encrypt and decrypt a message. It does not hide the text but converts it into other text that does not make any meaning. Its purpose is to ensure privacy by keeping the information hidden from anyone on the Internet except the receiver of the message. Messages are encrypted just before they are sent on the Internet or network. When the encrypted message is received by the receiver, it needs to be decrypted. Decryption is the reverse of encryption. It is the transformation of encrypted text back into normal text. There are number of encryption algorithms available in the market today.

In recent years, number of cases has been reported where the data in transit was intercepted. Encryption is used to protect data in transit, for example data being transferred via networks like Internet or E-commerce, mobile telephones, Bluetooth devices and bank Automatic Teller Machines (ATMs). Assume that you want to send a message "HOW ARE YOU?" to your friend. However, to protect the message you want to encrypt it. Using encryption we create a coded message to be sent to your friend. Here the encryption mechanism substitutes each alphabet with the alphabet that comes after it. This means that "A" becomes "B," and "B" becomes "C" and so on. You have also told your friends that the code to decrypt is "Shift by 1". Your message will now be encrypted as shown in the figure 5.22 :

H	O	W	A	R	E	Y	O	U	← Plain text
I	P	X	B	S	F	Z	P	V	← Cipher text

Figure 5.22 : Encryption using key

Your friend when he gets the message can decode it. Anyone else who sees the message in transit will only see weird characters. The key kept here is simple, but usually they are quite long.

For E-commerce security various protocols are also used. Let us understand the SSL protocol.

- **Secure Socket Layer (SSL)**

Now-a-days every user sends various types of data from email to credit card details. The user

wants this data to be protected when in transit over a public network. SSL protocol is used for securing web transactions on the Internet. It was developed by Netscape. During the E-commerce transaction, all the information is exchanged in secured manner using SSL by encrypting all the messages. It also provides the authentication of the merchant or shopper using a digital certificate so that the customer knows that they are communicating with a valid owner. To identify whether a site is secured, a security logo is present on the screen. If a site is secured by VeriSign, then the security logo of VeriSign is displayed on the login screen of the secured site. On clicking, you will get the owner information and the validity of the certificate. This indicates that the communication with this site is secured and the owner of the website is valid who is certified by the authority like VeriSign. The other indication of the security is that whenever connection is made to the secured site its address starts from https:// rather than http:// as shown in figure 5.23.

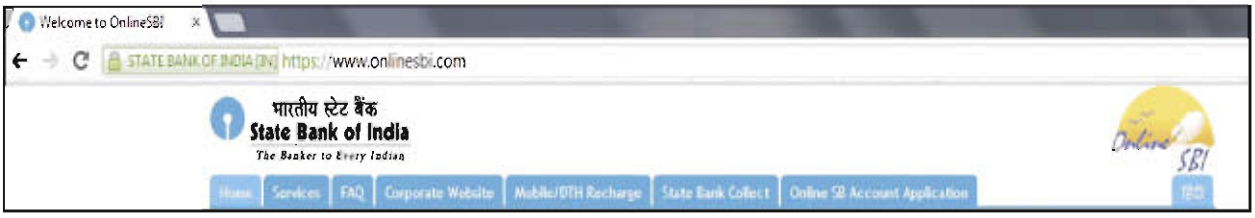


Figure 5.23: Example of https://

Legal Issues in E-commerce/M-commerce

E-commerce as well as M-commerce presents a world of opportunity for doing businesses, reaching global markets and providing facility of online purchasing. It provides opportunities to improve the business processes. However, just as any new business has some issues and risks so does E-commerce and M-commerce. Both of these pose many legal challenges, as it is the activity performed on global Internet without observing national boundaries. Every nation has its own rules and regulations. The major challenges are related to the intellectual property rights, copyrights, privacy and many times the disputes among the parties. The legal framework is necessary to resolve these issues. Many countries have already established their legal framework for electronic commerce. Indian Government has also established the IT laws (Information technology laws) under the IT act.

Enforcement of legal rules and regulations provides confidence to the customers that their personal information remains secret and will not be misused. If it is misused, responsible party can be punished. This type of security is a must in E-commerce and M-commerce as customers provide their sensitive information like credit card details. Companies making online business are getting legal support in case of misuse of their logos or any copyright materials like digital content. Any dispute between two parties can be resolved under these laws.

Securing Intellectual property

The intellectual property includes books, software's, music, video, copyrights, trademarks and web pages. Let us discuss some issues related to them.

- **Copyright**

  Copyright provides the author with a tool to protect his/her original work from being used or taken by others without permission. It is applicable to books, software programs and articles. Copyright law protects intellectual property in its various forms, and cannot be used freely.

It is very difficult to protect Intellectual property in E-Commerce or M-commerce. For example, if you buy software you have the right to use it and not the right to distribute it. The distribution rights are with the copyright holder. Also, most of the web pages are protected by copyright. In that case, copying contents from the website also violates copyright laws.

- **Trademark**

  It is a specific logo, mark, word, symbol, design, phrase or image which is used by an individual or a company to distinguish a product or service from that of others in the market. A trademark may be designated by the following symbols: TM, SM and ®.

- **Domain name disputes**

  The competition over domain names is another legal issue. Earlier, the domain names were given on first come first serve basis. Thus people would register domain names that were not in use but would be of importance. Later on such domain names were sold to concerned company at a very high price. This is known as cyber squatting. Another problem is Name changing where someone registers purposely misspelled variations of well known domain names. This can mislead the consumers who generally make typographical errors while entering a URL.

Protecting Intellectual property

Several new and improved methods are continually being developed to protect the intellectual property. Some of them are discussed here.

Steganography

Steganography is the process of hiding information within other information. The information in the files if not protected can be used for any malicious purpose. It works by replacing unused data in computer files such as images, sound or text with invisible information. This hidden information can be plain text, cipher text, or even images. Special software is needed for steganography, and there are freeware versions available on Internet which can be easily downloaded.

Digital Watermarking

The watermark is a digital code inserted into a digital image, audio or video file which can identify the file's copyright information. It also allows hiding information in a totally invisible manner. Earlier, artists used to creatively sign their paintings with a brush to claim copyright. But in digital world, artists can watermark their work by hiding their name within the image. Hence, the invisible embedded watermark helps to identify the owner of the work.

This concept is also applicable to other media such as digital video and audio. Currently the unauthorized distribution of digital audio over the Internet in the MP3 format is a big problem. In this case, digital watermarking may be useful to set up controlled audio distribution and to provide efficient means for copyright protection. In the field of data security, watermarks can be used for certification and authentication. For example, the photo identity card of a person can be protected by an identity number "123456" written on the card and hidden as a digital watermark in the identity photo. So, manipulating or changing the identity photo can be detected easily.

Digital watermarking can also link information on the documents. For example, the name of a passport owner is normally printed in clear text. But using digital watermark, the name can also be hidden

in the passport photo. If anyone tries to tamper with the passport by replacing the photo it would be possible to detect the change by scanning the passport and verifying the name hidden in the photo. A visible digital watermark can be added to any image using photo editor tools like GIMP.

Payment in E-commerce/M-commerce

Payment is one of the most important aspects of E-commerce as well as M-commerce. In traditional payment method we do the payment using cash, cheque or credit card. Electronic payment systems are becoming more important to the online business processes as companies are looking for different ways to serve the customers faster and at low cost. Electronic payment is a financial exchange that takes place online between the buyer and the seller. There are various payment options available for payment in the market today. The different types of electronic payment systems used are:

- **Payment cards**

  The payment cards can be classified as Credit cards, Debit cards and Smart cards, generally, businesses use the term payment card for all types of plastic cards that consumers use to make purchases.

  Credit card

  This is one of the most popular and widely accepted methods of payment on Internet. A credit card is issued to the customers by the banks known as issuing banks. The issuing banks provide the credit cards of the financial institutions which are established and reputed in the services of credit card business. Examples include MasterCard® or Visa®. Depending upon the customer's credit history and income level, credit limits are provided and up to that limit the customer can spend and pay to the issuing bank within the billing period.

  As the credit cards are linked to a bank account, when we use them to pay online, the merchant charges the goods to the issuing bank account and the bank shows the debit on your next statement. The customer simply pays the bank. For accepting payments on websites through credit card, merchant needs to open a merchant account with the banks known as acquiring banks, which provide the services of online authorization and payment processing. Authorization is the process of verifying whether the card is active; the credit limits are available to make purchase and verifying the other details of the customer like billing information. Credit cards are widely accepted by merchants and provide assurance to the customer as well as the merchant. Figure 5.24 shows some credit cards.

Figure 5.24 : Examples of Credit card

In credit card transactions over the Internet, the customer visits the merchant's website and selects goods to buy, all the information related to credit card is entered and then this information is transmitted to the merchant electronically. In this transaction four parties are involved :

- Customer with credit card
- Merchant accepting the credit card
- Issuing Bank: issues the credit card and guarantees the payment to the merchant. The bank collects the payment from the customer.
- Acquiring banks: Financial institution that establishes the account with the merchant validates the credit card information of the customer and authorises sale based on the customer's credit limit.

Two more entities play role in online payment. These are payment gateways and processors. Payment gateways are services provided by the third parties like PayPal which connects networks of all the parties involved and enables to perform authorization and payment in secured manner. Processors are data centres which perform the credit card transactions and settle funds to the merchant. Processors are connected to the E-commerce website of the merchant through the payment gateway.

The online payment through credit card on Internet is divided into two parts: Authorization and Settlement. During authorization the following steps are performed :

- Customer checkouts, provides credit card information on the E-commerce website, which along with the transaction details (like item detail, date of purchase and others) is transferred to the payment gateway.
- The payment gateway passes the information to the processor which contacts the issuing bank for the verification of the information.
- After verification, issuing bank sends the status of verification (or transaction result like accepted or rejected) to the processor which in turn passes it to the payment gateway.
- Finally, payment gateway sends the result of the transaction to the merchant's website. If the merchant accepts the transaction then the next step is the settlement during which it transfers the amount from the customer's account to the merchant's account.

During the settlement or payment processing the following steps are performed :

- Merchant sends the transaction request with all the details to the payment gateway which sends to the processor.
- Processor sends the payment details to the issuing bank of the customer. It also sends the payment details to the acquiring bank where the merchant has an account.
- The acquiring bank credits the amount to the merchant account. The issuing bank after including all the charges sends the bill to the customer which he needs to pay within the billing period.

Major credit card companies use the Secure Electronic Transfer (SET) security system to make online transactions secure.

The advantages of credit card are :

- Gives flexibility to the customer as they do not have to carry lot of cash. Customer can pay for goods and services both online and offline.
- Keeps record of the customer's purchase through the bank statement.
- Allows customers to purchase goods even when they do not have the cash available in the bank account. They can settle the cash by the end of the month.

The limitations of credit cards are :

- They are unsuitable for very small or very large payments. Also, due to security issues, these cards have a limit and cannot be used for excessively large transactions.
- Customers tend to overspend using credit cards.
- Problems arise in case lost or stolen credit cards.

Debit Cards

A debit cards looks like credit card but works differently. It is a kind of payment card that transfers fund directly from the consumer's bank account to the merchant. The amount is immediately deducted from the bank account of the consumer. The debit cards can keep the consumer purchases under a limit and do not allow him to exceed beyond his/her budget. But while using a debit card for a purchase, the consumer should always be aware of his account balance. Figure 5.25 shows an example of debit card.

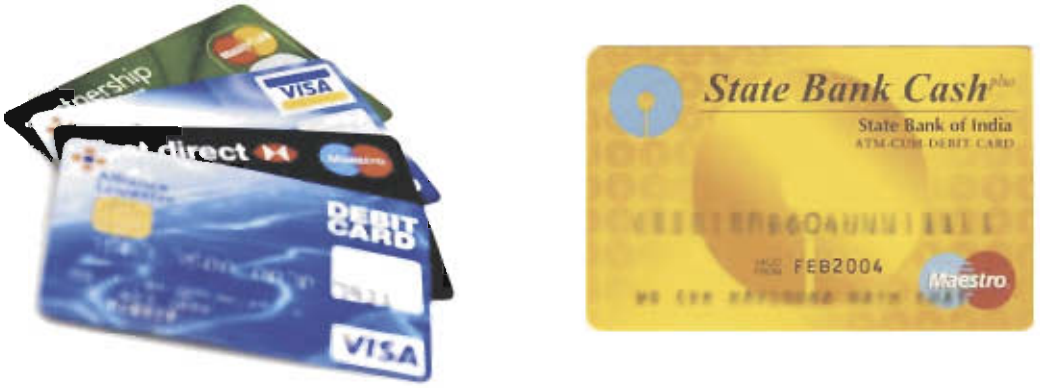


Figure 5.25 : Example of Debit card

Smart Cards

Smart cards look just like credit cards but are different as they have a microchip embedded in their surface. A smart card contains user's private information, such as account information, health insurance information, private keys etc. It can store 100 times more information than the normal cards. They are much safer than the credit or debit cards as the information stored in the smart card is encrypted. Customers can load their card with cash and then use this to pay for goods in a merchant's retail outlet or web site. Card readers are available for retail outlets as well as an attachment for PCs. This convenience gives a great advantage to smart cards. They can be used for a range of purposes like storing digital cash, storing a patient's medical records etc. Smart cards are popular in countries like U.S., Europe, Japan and some parts of Asia.

A smart card reader is required to read information or add data to it. It is small device into which the smart card is inserted. For example, when you visit your family doctor, you can give him your smart card to review the medical history and prescribe medicines. This information is also inserted into the smart card. At the medical store, you hand over your card to the pharmacist, who sees your prescription, and give you the medicine accordingly. Also, the payment can be done using smart card.

Charge cards

A charge card is another form of payment mechanism wherein the customer can pay through the card to the vendor. As compared to credit cards that have a credit limit, the charge card does not carry any spending limit. The customer has to pay the total amount used at the end of the billing period to the company that has issued the card. If the total amount is not paid back then the customer has to pay a late fee.

- **Net Banking**

  Another option which is becoming more popular is net banking or online banking. It does not involve any type of card. It can be used by customers who have bank accounts enabled with Internet banking. The bank provides the net banking password to the customer for operating the account from Internet including the payment for online purchases. Many of the E-commerce/ M-commerce websites provide the facility to make the payments using net banking. Instead of entering card details on the website, it allows one to specify the bank through which you wish to pay from. On these websites, when you proceed to make the payment, you will be asked to select your bank. Once you select the bank, you will see the screen of your bank's website where you need to log in using your account number and net banking password. You can perform the transaction and transfer the amount from your account to the account of the merchant.

  Indian railway provides the facility to book tickets online on their website www.irctc.co.in. First, the user needs to register on the site and then provide the details of the journey along with the passenger information. Once the information is filled up you can proceed to the payment where net banking option is provided. Select your bank and continue further which displays the login screen of your bank and transfer the fund to Indian railway to complete the booking. After successful payment, the user gets a message on the registered mobile number from irctc, which can be shown at the time of travel or E-copy of the ticket can also be printed. Net banking payment system is seen as being safer than using credit cards as nearly all merchant accounts in India offer it as an option.

- **Electronic Fund Transfer**

  Electronic Funds Transfer (EFT) means transferring money from one bank account to another electronically. It is safe, secure, efficient, and less expensive than paper check payments and collections. Examples of EFT are: transactions amongst various banks around the world, payment of tuition fees using an ATM, deposit of employee's salaries in their accounts, monthly bank account deductions and many more. The popularity of EFT for online bill payment is growing. The benefits of EFT include reduced administrative costs, increased efficiency, simplified bookkeeping, and greater security. However, the number of companies who send and receive bills through the Internet is still relatively small.

- **E-wallet**

  Most of the time when you make purchase on the web, you are required to fill out a form with your name, shipping address, billing address, credit information, and so on. Doing this a few times is fine, but having to do it every time you shop on the web is an annoyance. Some merchants solve the problem by having you fill out a form once, and then saving the information stored on their servers for later use. These merchants provide e-wallet services to its customers which makes the shopping or booking of tickets much easier for its customers. Today many banks, online grocery stores, telecom services etc. provide e-wallet services. E-Wallet is an electronic card for making secure online payments towards a merchant. It works just like a credit or a debit card. While making payment through e-wallet the customer is not required to provide the credit/debit card number thus reducing the risk of credit/debit card number being exposed.

  For example, IRCTC has launched e-wallet scheme for the customers to make online booking easier. The customers having an account can deposit money in advance with IRCTC which can be used in future as payment option to book the tickets online. Currently the users have to provide the credit/debit card details for booking tickets. This payment procedure takes time as the customer is transferred to the bank's server for payment. He/she can

- **RuPay**

  The term RuPay is coined from two terms Rupee and Payment. It is a new card payment mechanism launched by National Payments Corporation of India (NPCI). Figure 5.26 shows the RuPay card which can be used the same way as we use the credit and the debit cards.

Figure 5.26 : RuPay card

Currently, as there is no domestic card Indian banks have to tie with Mastercard and Visa to connect cardholders, merchants and the issuing banks around the globe. Mastercard and Visa are the world leaders in card payments and all payment transactions are processed through them. Every transaction done using a credit or debit card issued by a domestic bank is routed through their network switches which are outside the country. These transactions involve additional charges for providing the services. The banks have to pay for processing all debit and credit card payments. RuPay cards are the domestic alternative to the other payment cards. By using RuPay cards, all the transactions will be processed within India. As the transaction processing will be done domestically, the cost of each transaction clearing and settlement will be reduced. RuPay will benefit the customers and the banks by reducing the cost.

Summary

In this chapter we discussed about Mobile Commerce. M-commerce refers to buying and selling of goods and services through the use of Internet enabled wireless devices such as a Mobile phone, Personal digital assistant (PDAs), Smartphone, Tablet PCs and Palmtops. we also discussed about the advantages and limitations of M-commerce. M-commerce applications are becoming popular with the mobile users increasing day by day. Location-based commerce applications and services are emerging with significant implications for the future of E-commerce. These applications track the user's location in order to deliver a service or product. We discussed about the various the security issues and threats. Measures to be taken against the security threat are also discussed. Legal issues related to e-commerce are discussed. The various payment options provided in E-commerce like credit card, debit card, smart cards, Ru-Pay, net banking and E-wallet are discussed.

EXERCISE

1. What is M-Commerce ? List some of the examples of M-commerce.
2. Why is M-commerce helpful to the user ?
3. List the advantages of M-commerce.
4. What are the limitations of M-commerce ?
5. List some of the applications of M-commerce.
6. Give some website examples for the following M-commerce applications :
   (1) Mobile ticketing (2) Mobile Auctions
   (3) Mobile purchase (4) Mobile information services
   (5) Mobile financial services
7. What is L-commerce ?
8. What is GPS ? How does it locate a device ?
9. List some applications of location based services.
10. What are the four important aspects of E-commerce security ?
11. Explain the following Internet security threats :
    (1) Malicious code (2) Sniffer (3) Denial of service attack
12. What is cyber vandalism ?
13. Explain the term spoofing ?
14. List the various security measures taken for security threats.
15. What does a digital certificate contain ?
16. What is cryptography ? Using encryption mechanism of each alphabet with the alphabet that comes before it, encrypt the message "Gandhi Ashram".

17. What is the purpose of SSL ?

18. List the issues related to intellectual property.

19. Which are the different ways of protecting Intellectual property ?

20. List the different types of electronic payment systems.

21. List the advantages and limitations of credit card.

22. How is a smart card different from credit cards ?

23. What is electronic fund transfer ?

24. Choose the correct option from the following :

(1) Which of the following refers to buying and selling of goods or services through the use of Internet enabled wireless devices ?

(a) Internet (b) M-commerce (c) M-banking (d) WWW

(2) Which of the following is the use of technologies which provide the location information for business purpose ?

(a) E-commerce (b) M-commerce

(c) L-commerce (d) Traditional commerce

(3) Which of the following stands for GPS ?

(a) Global Positioning System (b) Global Postal System

(c) Grand Positioning System (d) Google Positioning System

(4) Which of the following security aspect refers to the secrecy of the information so that unauthorized user cannot read it ?

(a) Confidentiality (b) Integrity

(c) Non-repudiation (d) Authorization

(5) Which of the following security aspect ensures that the information must not be accidentally or maliciously altered or tampered in transit ?

(a) Confidentiality (b) Integrity

(c) Non-repudiation (d) Authorization

(6) Which of the following security aspect ensures that only authentic users are allowed to use the system ?

(a) Authorization (b) Confidentiality (c) Non-repudiation (d) Integrity

(7) Which of the following security aspect ensures that the sender of the message cannot deny that he/she has sent the message ?

(a) Authorization (b) Confidentiality (c) Non-repudiation (d) Integrity

(8) Which of the following is a program that uses Internet to record information that passes through a computer or router in transit from sender to receiver ?

(a) Sniffer (b) Denial of service attack

(c) Malicious code (d) Spoofing

(9) Which of the following is an attack used to shut down a machine or network, making it inaccessible to its intended users ?

(a) Malicious code (b) Denial-of-Service

(c) Spoofing (d) Cyber vandalism

(10) Which of the following is known as electronic defacing of an existing website page ?

(a) Cyber vandalism (b) Denial-of-Service

(c) Spoofing (d) Malicious code

(11) Which of the following is pretending to be someone you are not, or representing a website as authentic when it is actually a fake ?

(a) Cyber vandalism (b) Malicious code

(c) Denial-of-Service (d) Spoofing

(12) Which of the following is a computer program that detects, prevents and takes action to remove the malicious codes like viruses, worms and trojan horses from the infected system ?

(a) Antivirus software (b) Digital certificate

(c) Firewall (d) Cryptography

(13) Which of the following is the transformation of normal text known as "plain text" into unreadable or secret text known as "cipher text" using encryption algorithm ?

(a) Firewall (b) Encryption

(c) Antivirus software (d) Digital certificate

(14) Which of the following is the transformation of encrypted text back into normal text ?

(a) Firewall (b) Digital certificate

(c) Decryption (d) Virus

(15) Which of the following is a protocol used for securing web transactions on the Internet ?

(a) TCP/IP (b) HTTP (c) Bluetooth (d) SSL

(16) Who developed SSL protocol ?

(a) Google (b) Netscape (c) Yahoo (d) Firefox

(17) Which of the following starting address indicates that site is secured by SSL protocol?

(a) http:// (b) ssl:// (c) https:// (d) http-ssl://

(18) Which of the following provides the author with a tool to protect his/her original work from being used or taken by others without permission ?

(a) Trademark (b) Copyright (c) Digital watermarking (d) Steganography

(19) Which of the following is a specific logo, mark, word, symbol, design, phrase or image which is used by an individual or a company to distinguish a product or service from that of others in the market ?

(a) Trademark (b) Copyright

(c) Digital watermarking (d) Steganography

(20) Which of the following is the symbol for trademark ?

(a) TM, MT and © (b) TM, MS and ®

(c) TM, SM and ® (d) TM, SM and (C)

(21) Which of the following is the process of hiding information within other information ?

(a) Squatting (b) Steganography (c) Name changing (d) Copyright

(22) Which of the following is a digital code inserted into a digital image, audio or video file which can identify the file's copyright information ?

(a) Image mark (b) Digital mark

(c) Code mark (d) Watermark

(23) Who issues the credit card and guarantees the payment to the merchant ?

(a) Merchant (b) Issuing bank (c) Acquiring bank (d) Customer

(24) Which of the following payment card has a microchip embedded in its surface ?

(a) Smart card (b) Debit card (c) Credit card (d) Charge card

Object-Oriented concepts 6

Today we are in the era of Internet, Websites and Web based operations, where rapid application development and reusability of source code is very important. Object-oriented techniques as methodology or as paradigm is playing significant role in analysis, design and implementation of software system. Software developed using object-oriented methodology are proclaimed to be more reliable, easier to maintain, reuse and enhance. This chapter explains basic object-oriented concepts. Implementation of this concepts using Java programming language is covered in the later chapters.

Introduction

Object-oriented programming concepts started originating in the 1960s. Since mid 1980s, it had become the main programming paradigm used in the creation of new software. It was developed as a way to handle the rapidly increasing size and complexity of software systems and to make it easier to modify these large and complex systems over time. Some of the popular programming languages that support object-oriented programming are C++, Java, C#, VB.net, ASP.net and PHP.

The way of programming can be divided into two categories namely structure/procedural and object-oriented. In procedural programming; the focus is on writing functions or procedures which operate on data. For example, for library application software, we will think of all processes of library application and focus on the modules like student registration, book issue, book return, fine calculation.

In object-oriented paradigm, the focus is on **objects** which contain both data and functionality together. For library application, our focus is on the objects involved in the application. Here we think of objects like student, book and librarian. We also need to think of association between such objects. For example, student returns book.

The power of object-oriented programming language enables the programmer to create modular, reusable and extendable code. As a result, programmer can formulate a program by composition and modification of existing modules. Flexibility is gained by being able to change or replace modules without disturbing other parts of code. Software development speed is gained by reusing and enhancing the existing code.

Object-oriented programming uses object as its fundamental building block. Similar objects are classified using a concept of class. A computer language is object-oriented if they support four specific object properties called abstraction, encapsulation, polymorphism and inheritance.

Object

In the "real" world, objects are the entities of which the world is comprised. Some objects can be concrete things like person, car or a coffee cup. Other objects may be abstract that do not represent things which can be touched or seen; for example, concepts like date and time.

All objects have unique identity and are distinguishable from each other. For example, every person has characteristics like name, city, gender, birth-date, profession. In object-oriented terminology, such

characteristics are known as **properties** or **attributes**. To uniquely distinguish one person from other, we use the value of these characteristics. The names 'Ram' and 'Shyam' specify two different persons. When two persons have same name, they may be distinguished using other attribute like birth-date. In this way, to identify the objects, we use the value of these attributes. These values are called **state**. Additionally, there is a **behaviour** associated with objects. For example, person takes birth, gets name, changes location. The behaviour is also known as **method**. State of the object can change due to its behaviour. Thus, any real world object can be described in terms of what it is called (identity), what it is (its state) and what it does (its behaviour).

In object-oriented programming, attributes that describe the object are also referred to as data fields. The data attributes and behavioral methods associated with an object are collectively referred to as its **members *or* features.**

When we design for any software application, objects to be considered should be meaningful to the application. For example, in railway reservation application, meaningful objects are train, passenger, ticket or station. Objects like car, computer and watch may be irrelevant here.

Class

From the example of the object person, it can be easily seen that various objects possessing same characteristics and behavior differ from each other only in the state (value of the data) that they hold. Object-oriented system uses the concept of class that enables to express the set of objects that are abstractly equivalent, differing only in the values of their attributes. Class can be considered as a blueprint for various objects.

A class is a template for multiple objects with similar features. It describes a group of objects with similar attributes and common behavior. Objects in the same class share a common semantic purpose. Thus, class is a general concept used to embody all the common features of a particular set of objects.

For example, we have a class named 'Person' describing common attributes and behavior of all persons. Individual persons are then objects and are identified by the state; that is the value of their attributes. Figure 6.1 shows difference between the class and its objects.

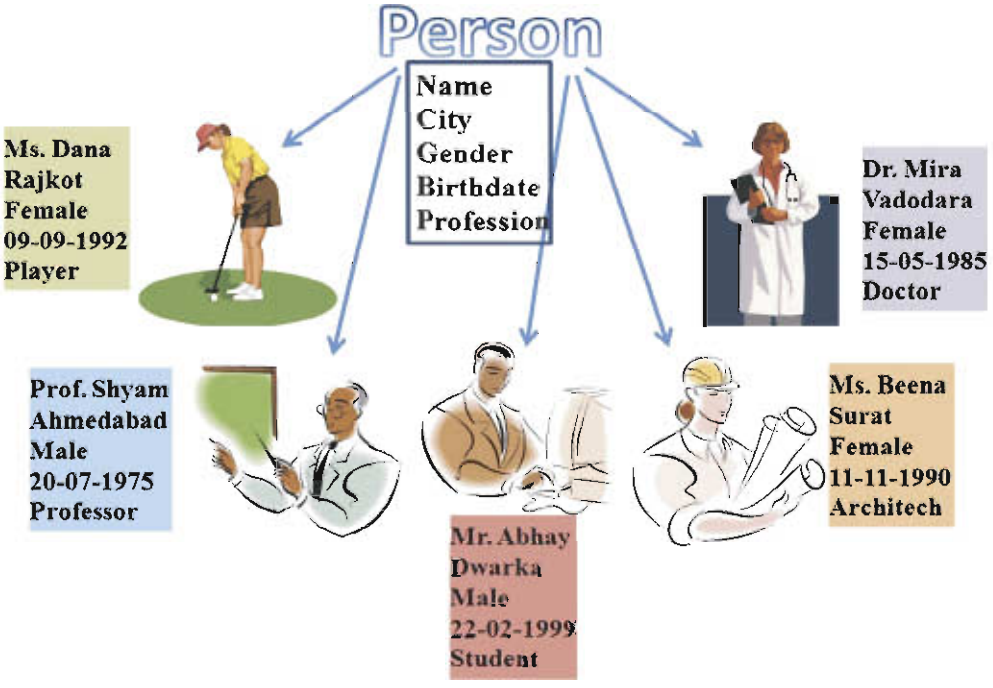


Figure 6.1 : Class 'Person' and its Objects

Before we proceed ahead to learn other object-oriented concepts, let us have brief introduction to class diagram.

Introduction to Class Diagram

The class diagram presents a collection of classes, constraints and relationship among classes.

Unified Modelling Language (UML) can be used to create models of object-oriented software to help with design of an application. UML is a visual modelling language defined and maintained by the Object Management Group (OMG). UML specifies several diagrams for representing different aspects of a software application.

The purpose of the class diagram is to model the static view of an application. The class diagrams are the only diagrams which can be directly mapped with object oriented languages and thus widely used among software developers.

In class diagram, a class is represented with an icon using a rectangle split into three sections to present name, attributes and behavior as follows :

1. Name of the class in the top section
2. Attributes or properties of the class in the middle section
3. Behavior or operations or methods of the class in the bottom section

Figure 6.2 shows the pictorial representation of a class using UML convention, while figure 6.3 shows the class diagram of Person.

Class Name
Visibility attribute : data type = initial value
Visibility operation (argument list) : return type

Figure 6.2 : Class representation in UML convention

Person
name : string city: string gender : char = 'M' -birthdate : date profession : string
setBirthdate (d:int,m:int,y:int): date changeCity (newCity:string) :string display ( )

Figure 6.3 : Diagram of class 'Person'

The class Person is intended to provide information that is useful in understanding the role of the class in the context of class diagram. It does not have to contain every attribute and operation of the class.

As seen in figure 6.2, in UML notation, an attribute is declared using following syntax :

[<visibility>] <attribute name> [: <attribute data type> [= <initial value>]]

Here, items written in pair of square brackets [ ] are optional and user is supposed to specify the values for items enclosed in angle brackets < >. Visibility can be private, protected, public or package represented using symbols -, #, + and ~ respectively. We will study about visibility in later chapters. Attribute generally refers to a variable. Data type and initial value identify the type of data stored and its value at the start of the program. As can be seen here, only attribute-name is mandatory and all other items are optional.

Some examples of declaring attribute are as follows :

name : string

- balance : float = 0.0

As seen in figure 6.2, In UML notation, an operation is declared using following syntax :

[<visibility>] <method name> (parameter list separated by comma) : <return data type>

An example of a method in the previous example is setBirthdate(d:int, m:int, y:int) : date. Here the parameter can be specified with data type and optional default value; for example gender : char = 'M'. Parameters can also be specified as input or output depending on whether it is read only or not.

UML diagrams are independent of the programming language used for coding an application. Some software developers prefer to specify attributes and operations in the more familiar format of a programming language instead of using the standard UML notation. That's fine, as long as it makes sense to everyone concerned.

Objects are presented using their state during execution of an application. Thus, objects are dynamic. Corresponding to figure 6.1, an object (also called an instance) of class person is represented as shown in figure 6.4.

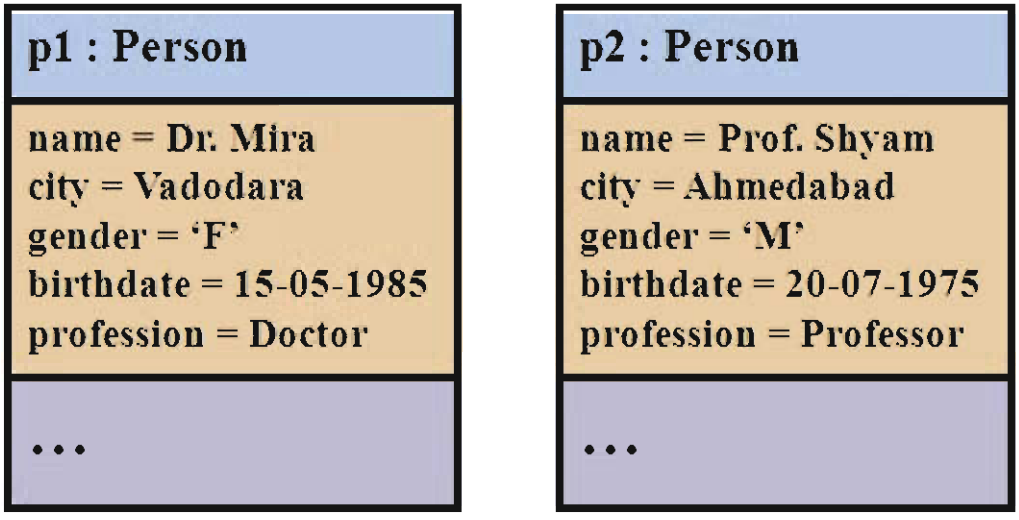


Figure 6.4 : Objects p1 and p2 of Class 'Person'

Encapsulation

For any computer program, two core elements are data and functions. Structured/Procedural programming views these two core elements as two separate entities; whereas object-oriented programming views them as single entity.

In procedural programming, data can be altered by any component of the program. It is not protected from modification.

In object-oriented programming, this problem can be solved using encapsulation. Here, data and the methods that manipulate data are guarded against modification or misuse by other components of the program. This mechanism of providing protection to data and methods of a program is called **encapsulation.** This is possible by wrapping data and methods into a single unit known as class and declaring them as private. Private members of the class are not available directly to outside world. If necessary, the data is made available via public methods. Thus, encapsulation provides data hiding capability. We will study about class and private/public data and methods in later chapters.

Encapsulation keeps the data safe from unintended actions and inadvertent access by outside objects. Unlike procedural programming where common data areas are often used for sharing information, object-oriented programming discourages direct access to common data (other than the use of global variables) by other programs. Only the object that "owns" the data can change its content. Other objects can view or change this data by sending message to the "owner".

Data Abstraction

Data abstraction is a process of representing the essential features of the objects without including implementation detail. Abstraction is a concept that hides the complexity; it says what it does, but not how it is done.

Let's take one real life example of a television set. We can turn television set on and off, change the channel, adjust the volume. But, we do not know its internal details like how it receives signals over the air or through a cable, how it translates them and finally displays them on the screen. Thus there is a clear separation of internal implementation of television set from its external interface. We can play with it via external interfaces like the power button, channel changer and volume control without having any knowledge of its internals.

Data abstraction thus is a technique that relies on the separation of interface and implementation. It is not a new concept in programming. Most people already practice it up-to some degree, probably without realizing it. For example, when we use functions like sqrt(25) and printf("Hello world"), in C programming we never thought of how they are implemented.

A user-defined function with necessary input data parameters also provides data abstraction. Let us consider a C program using structure for declaring 'date' type of data. Here date is composed of three elements; day, month and year. Let us use primitive integer data type for these elements. Consider a function named 'dateDiff' that takes two dates as parameters and returns the number of days between two dates. User of this function is not supposed to know how this difference is calculated. It is necessary to know only the signature of the function; that is the name of the function, number and type of parameters and the return type of parameters. If there is any change in the process of finding difference, it has no effect on the part of the program using this function.

Thus data abstraction provides the skeleton or templates for our use. The system hides certain details of how data is stored, created and maintained. All that is visible to the rest of the world is the

abstract behaviour of the data type; details of how that behaviour is implemented are hidden, so that they can be changed as per need without impacting others.

Abstract Data Types (ADT) or structures (struct) in C/C++, classes in C++/Java are examples for data abstraction. In ADT, we simply define a data type and a set of operations on it. We do not show the implementation of operations.

The basic difference between data encapsulation and data abstraction is: Encapsulation protects data by making them inaccessible from outside and abstraction enables to represent data in which the implementation details are hidden (abstracted).

Messaging

In object-oriented terminology, a call to a method is referred to as a message. Due to encapsulation, all method calls are handled by objects that recognize the method.

Different classes may have same methods with same name. For example, class 'date' has method named 'display' that displays the date object in specific format. Suppose there are other classes 'time' and 'person'; both having method with same name 'display'; to display time in specific format and to display the details of person respectively. When method 'display' is invoked in the program, how to know which method is to be executed? This is determined using the object that invokes the method. For example, if 'display' is called by an object of 'person' class, it executes the display method defined in 'person' class.

Polymorphism

Polymorphism means 'many forms'. There may be different forms of single method or operation.

Assume that we have written a function named 'max' that finds maximum of two numbers. It takes two integers as parameters and returns maximum as integer value. Now suppose we want to add one more function named 'max' that finds maximum of integer elements stored in an array. It takes array and its size as parameters and return maximum integer. Is it possible to define more than one function with the same name ? In some programming languages, the answer to this question is 'no'. But in object-oriented programming, the answer is 'yes' as long as the methods differs in signatures (number and type of parameters).

Object-oriented programming allows defining more than one method having same name but different signatures in a single class. This feature is known as **function** or **method overloading.**

Object-oriented programming also allows writing expression using operators on objects. For example, we can use expression 'date1-date2' where both operands are objects of class 'date'. Here operation '-' is performed in a way different than performing subtraction on two numbers; say n1-n2. Here same operation is given different meanings depending upon the data type of operands used. This type of polymorphism is achieved through **operator overloading.**

Thus, polymorphism is achieved using two types of overloading: function overloading and operator overloading. In general, the capability of using same names to mean different things in different contexts is called overloading.

Aggregation and Composition

When objects of one class are composed of objects of other class, it is called aggregation or composition. It represents 'has-a' or 'a-part-of' relationship between classes. For example, motherboard is a part of computer.

As we know, computer is composed of many parts like motherboard, screen, keyboard and mouse. A screen itself may be a class with attributes like length, width and model. Similarly motherboard may be defined as a class with attributes like model and company. When we define class 'computer', it will contain attributes that are objects of class 'motherboard' and 'screen'. Thus it implies containment.

Let us modify class 'Person' discussed before. First of all, we will define two new classes 'Name' and 'Address' as shown in figure 6.5. Class 'Name' has attributes first name, middle name and last name. Class 'Address' has attributes house (that refers to details of house), street, city, state and pin code. Now let us modify class 'Person' and change the name of the attributes name and address to nm and addr respectively. The data type of attributes nm and addr is class 'Name' and 'Address' respectively. Thus class 'Person' contains objects of class 'Name' and 'Address'.

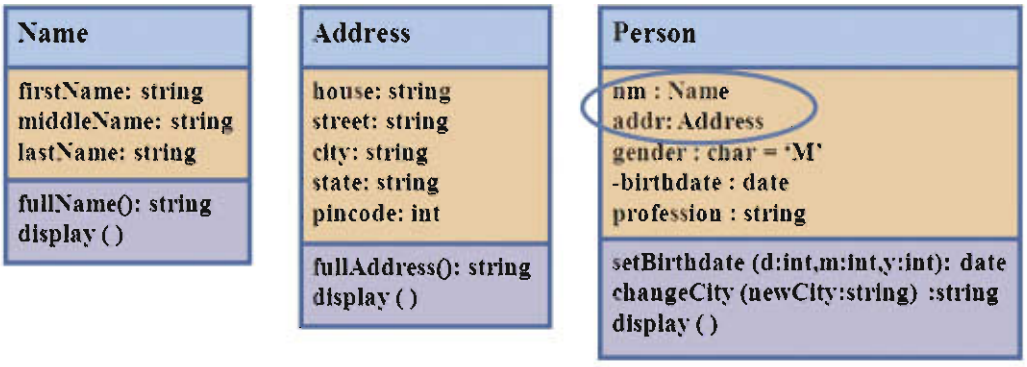


Figure 6.5 : Class 'Person' with attributes of class 'Name' and 'Address'

Aggregation vs. Composition

Aggregation represents non-exclusive relationship between two classes. In aggregation, the class that forms part of the owner class can exist independently. The life of an object of the part class is not determined by the owner class. For example, although the motherboard is part of the computer, it can exist as a separate item independent of the computer. Thus motherboard is not exclusively associated with computer. Similarly object address may be shared by two or more persons as can be seen in figure 6.6. So, address is not exclusive to any one person.

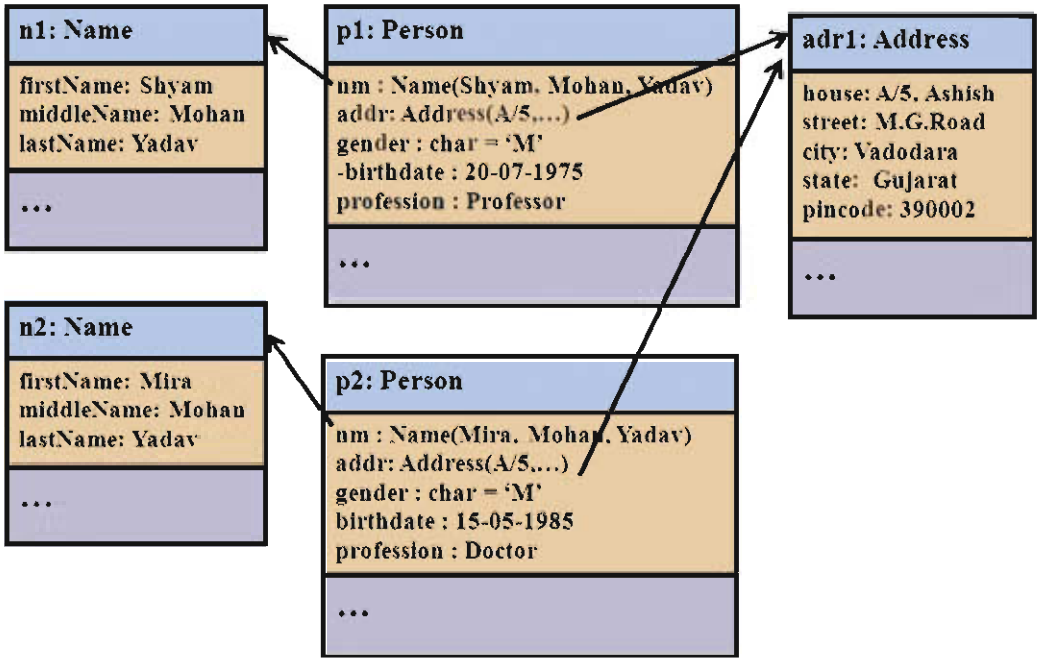


Figure 6.6: Two 'Person' objects having different names but same address

Basic aggregation is represented using an empty diamond symbol next to the whole class as shown in figure 6.7.

Composition represents exclusive relationship between two classes. Composition is a strong type of aggregation where the lifetime of the part class depends on the existence of the owner class. If an object of aggregating class is deleted, its part class object also will get deleted. For example, when an object of class Person is deleted, the object of class Name is also deleted. Name is associated exclusively with single person as shown in figure 6.6. Composition relationships are represented using a filled diamond symbol next to the whole class as shown in figure 6.7.

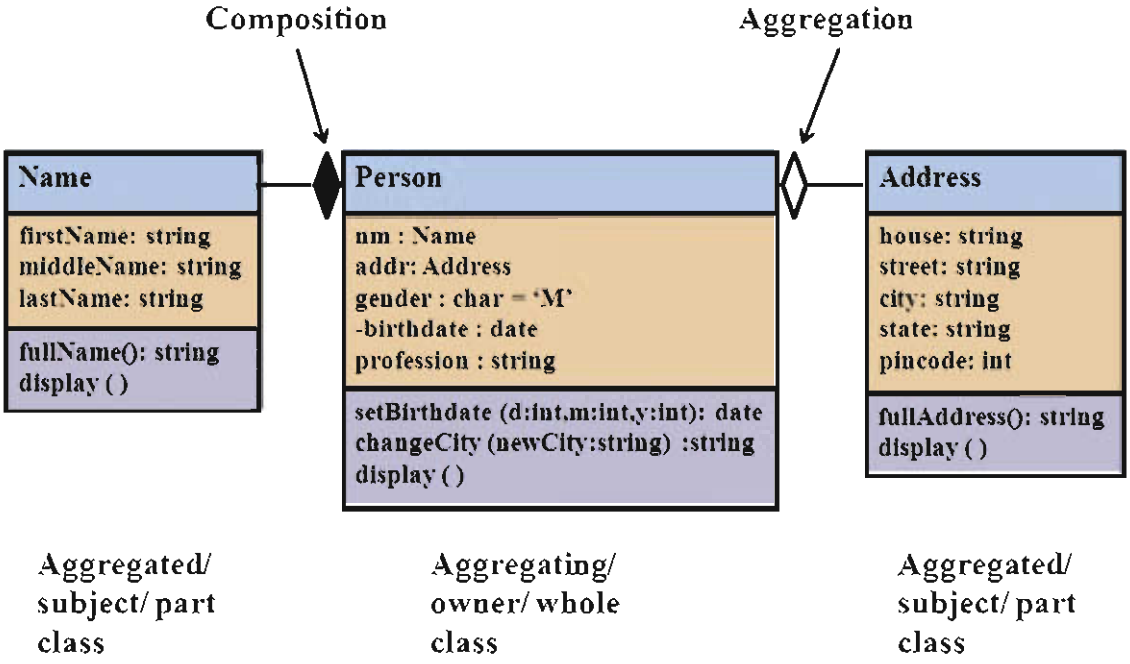


Figure 6.7 : Composition and Aggregation

In the given example, a relationship between class 'Person' and class 'Name' is composition relationship; whereas a relationship between class 'Person' and class 'Address' is aggregation relationship. Address may be shared by more than one person. So, when a person is deleted, the corresponding name object is deleted but address cannot be deleted.

Note : The class that contains objects of other class is known as owner class or whole class or aggregating class. For example, the class 'Person' shown in figure 6.7 is an aggregating class.

The class that is contained in owner class is known as subject class or part class or aggregated class. For example, the classes 'Name' and 'Address' shown in figure 6.7 are aggregated class.

Inheritance

Inheritance is generally referred to as 'is-a-kind-of' relationship between two classes. It is appropriate when one class is 'a kind of' other class. For example, teacher is a kind of person. So, all the attributes and methods of class 'Person' are applicable to class 'Teacher' also. In other words, class 'Teacher' inherits all attributes and behavior of class 'Person'. Class 'Teacher' may have additional attributes like subject and methods like taking lectures of the subject. In such scenario, class 'Teacher' can be defined using class 'Person'.

Inheritance refers to the capability of defining a new class of objects that inherits the characteristics of another existing class. In object-oriented terminology, new class is called sub class or child class or derived class; whereas the existing class is called super class or parent class or base class. The data attributes and methods of the super class are available to objects in the sub class without rewriting their declarations. This feature provides reusability where existing methods can be reused without redefining. Additionally new data and method members can be added to the sub class as an extension. It also allows the methods to be redefined in subclass as per need.

In class diagram, inheritance is represented using an arrow pointing to superclass as shown in figure 6.8. In this example, 'Person' is a superclass and 'Teacher' is a subclass.

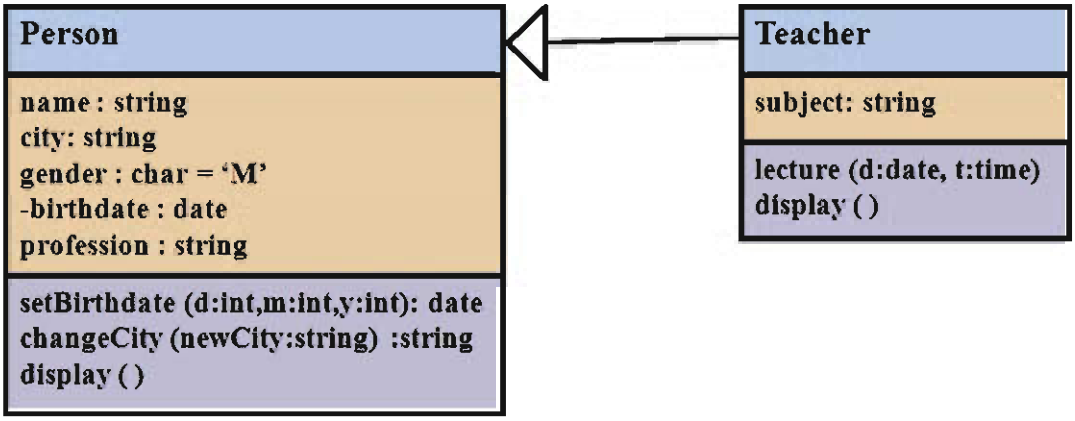


Figure 6.8 : Example of Inheritance

Generalization is another name for inheritance or "is a" relationship. It refers to a relationship between two classes where one class is a specialized version of another. Common attributes and methods are defined in superclass. Subclass is a specialized version with additional attributes and methods.

Sometimes, there may be a classical hierarchy of inheritance between classes. For example, class 'Employee' can be derived from class 'Person', then class 'Teacher' can be derived from 'Employee'. Here employee is a kind of person and teacher is a kind of employee. Such type of inheritance is known as multilevel inheritance. An example of multilevel inheritance is shown in figure 6.9.

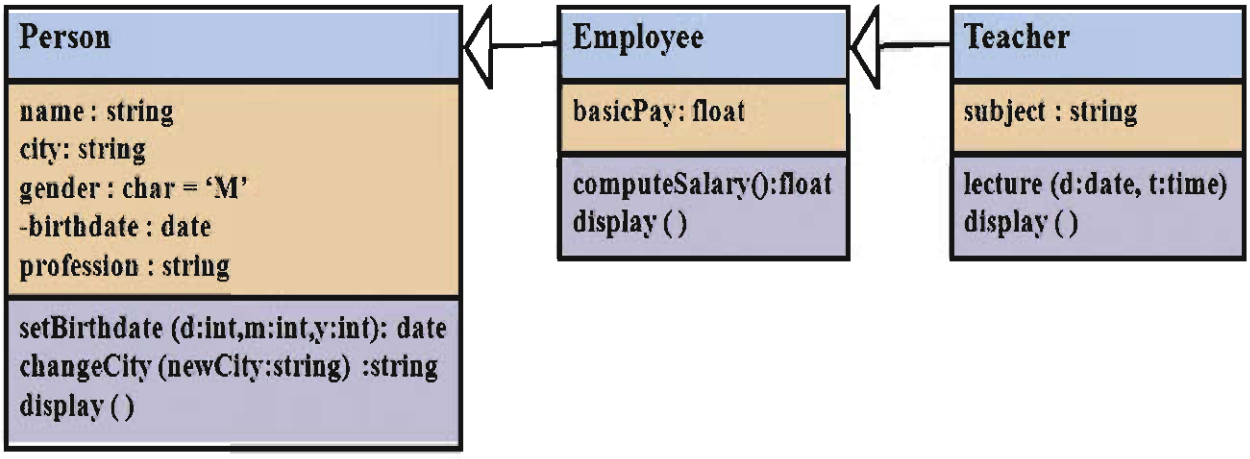


Figure 6.9 : Example of multilevel inheritance

A class can also be derived using more than parent classes. For example, a child inherits the characteristics of both mother and father; airplane is a kind of vehicle as well as flying object. When a class is derived from two or more classes, it is known as multiple inheritance.

Composition vs. Inheritance

In inheritance, class inherits from other classes in order to share, reuse or extend functionality. Here there exists 'a kind of' relationship between super class and subclass.

In composition, classes do not inherit from other classes, but are 'composed of' other classes. Class contains the attributes where some attributes are of objects of other class types.

Note that there other types of relationships and also constraints that can be represented in class diagram. The study of all the concepts is out of scope of this book.

Summary

Object-oriented as methodology is playing significant role in analysis, design and implementation of software system. In this paradigm, the focus is on objects which contain both data and functionality together. A class encapsulates attributes (data) and methods (behavior or functionality) together as a template that can be shared by all objects. Objects are then distinguished by their state; that is value of these attributes. More than one class may be associated with each other. When there is 'has-a' or 'a-part-of' relationship between two classes, the relationship is called aggregation or composition. When a class contains objects of other class, the container class is called owner class or whole class or aggregating class. The class that is contained in owner class is known as subject class or part class or aggregated class. Aggregation represents non-exclusive relationship between two classes. Composition represents exclusive relationship between two classes. When there is 'is-a' or 'a-kind-of' relationship between two classes, there is inheritance relationship. General or common features of two classes are implemented in the superclass and special features are implemented in subclass.

EXERCISE

1. List the features supported by object-oriented programming languages.
2. Distinguish between class and object.
3. Differentiate between 'Encapsulation' and 'Data Abstraction'.
4. What do you mean by polymorphism ? Name two type of overloading that may be supported to achieve polymorphism.
5. Explain the use of aggregation and composition.
6. When should one use inheritance ? Give example.
7. Explain different types of inheritance.
8. Choose the most appropriate option from those given below :

   (1) In Object-oriented methodology, the focus is on which of the following entities ?

   (a) Data (b) Functions (c) Objects (d) All of the above

(2) Which of the following best suits to Java ?

(a) A procedural programming language

(b) An Object-oriented programming language

(c) A Query language

(d) All of the above

(3) Which of the following is used to distinguish objects from each other ?

(a) Attributes (b) State (c) Behavior (d) All of the above

(4) Which of the following is used to define common features of similar objects ?

(a) Class (b) Object (c) Methods (d) All of the above

(5) Which of the following is not a visibility symbol ?

(a) ~ (b) * (c) # (d) -

(6) Which of the following is provided using encapsulation ?

(a) Data protection (b) Data sharing

(c) Separation of data and methods (d) All of these

(7) Which of the following is enabled by data abstraction ?

(a) Data protection (b) Data hiding

(c) To hide implementation details of method manipulating the data

(d) All of these

(8) With which of the following options polymorphism cannot be achieved ?

(a) Method overloading (b) Operator overloading

(c) Data hiding (d) All of these

(9) An aggregation model refers to which of the following relationships ?

(a) 'is-a' relationship (b) 'is-like' relationship

(c) 'a-part-of' relationship (d) All of these

(10) An inheritance model refers to which of the following relationships ?

(a) 'is-a' relationship (b) 'has-a' relationship

(c) 'a-part-of' relationship (d) All of these

(11) In class diagram, composition is represented using which of the following symbols ?

(a) Empty diamond symbol (b) Filled diamond symbol

(c) Empty triangle symbol (d) All of these

Java Basics 7

Java is an object-oriented programming language developed by Sun Microsystems, a company best known for its high-end Unix workstations. Modelled after C++, the Java language was designed to be small, simple, and portable across platforms and operating systems, both at the source and at the binary level. In this chapter, we will learn to start programming with Java. We will discuss simple Java statements, the basic things we can do in Java within a method main(). As we have already learnt C language, the basic constructs in Java will be very easy to understand. Most of the Java syntax is very similar to C language.

Introduction to Java

Java language was developed at Sun Microsystems in 1991. Java is small, fast, efficient, and easily portable to a wide range of hardware devices. It is considered as one of the ideal language for distributing executable programs via the World Wide Web, and also a general-purpose programming language for developing programs that are easily usable and portable across different platforms.

Java is an object-oriented language and here it differs from C. Using Java, we can take full advantage of object oriented methodology and its capabilities of creating flexible, modular and reusable code. It includes a set of class libraries that provide basic data types, system input and output capabilities, and other utility functions. These basic classes are part of the Java Development Kit (JDK). JDK has classes to support networking, common Internet protocols and user interface toolkit functions. Because these class libraries are written in Java, they are portable across platforms as all Java applications are.

Java is platform-independent at both the source and the binary level. Platform-independence is a program's capability of being moved easily from one computer system to another. At the source level, Java's primitive data types have consistent sizes across all development platforms. At binary level, platform-independence is possible due to bytecode interpreter. The designers of Java chose to use a combination of compilation and interpretation. The scenario of platform independence is shown in figure 7.1.

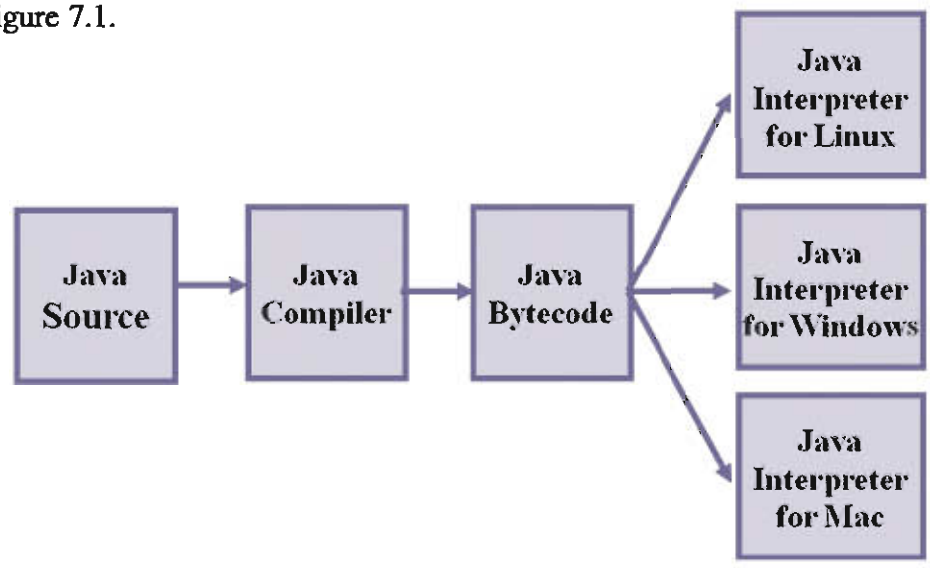


Figure 7.1 : Java : platform-independent

Programs written in Java are compiled into machine language for a computer that doesn't really exist. This so-called "virtual" computer is known as the Java Virtual Machine (JVM). The machine language for the Java Virtual Machine is called Java bytecode. Different Java bytecode interpreter is needed for each type of computer. Java binary files are actually in a form called bytecodes that is not specific to any one processor or any operating system. The only disadvantage of using bytecodes is its slow execution speed. There are tools available tools to convert Java bytecodes into native code. Native code is faster to execute, but then it does not remain machine independent.

Creating Simple Java Application

Before we learn the basics of Java programming language, let us begin exploring Java with a simple program that computes phone call charges and update the pre-paid balance amount. Here, we will learn how to create, compile and run Java programs.

A Java program is composed of classes. It should have at least one class and it must have main method in it. C programmers can think of a class as a sort of creating a new composite data type by using struct and typedef. Classes, however, can provide much more than just a collection of data. Note that typedef is not available in Java.

Let us give a name 'CallCost' to this Java application that computes call charges and update balance. To create and execute 'CallCost' application, we need to perform the following steps:

1. Create Java source file using any plain ASCII text editor.
   - Choose any text editor and type the program as given in code listing 7.1.
   - Save the source file with name 'CallCost.java'. Conventionally, Java source files are given the same name as the class they define, with an extension of .java. Note that the class name and filename are case sensitive. So, if class name is CallCost, filename should be CallCost.java (Applicable to SciTE editors).
2. Compile the source file using the Java compiler.
   - To compile the Java program, type javac followed by the name of your source file: javac CallCost.java
   - If the compiler show any errors, go back and make sure that you've typed the program exactly as it appears in code listing 7.1.
   - When the program gets compiled without errors, compiler creates a file with extension .class in the same directory as your source file. See that compiler has created file 'CallCost.class'. This is our Java bytecode file that will be executed.
3. Run the application using Java interpreter.
   - In the JDK, the Java interpreter is called simply using java. Type java CallCost.
   - To execute the program, we only need the compiled class file, not the source code.
   - Interpreter java uses the bytecode CallCost.class and executes it.

```
/**
* This class implements a simple program that
* will compute the cost of phone call and update balance
*/
public class CallCost
{
public static void main(String[] args)
{
/* declare variables */
double balance;        // balance amount in rupees
double rate;           // call rate; rupees per second
double duration;       // call duration in seconds
double cost;           // cost of last call

/* computations. */
balance = 170;
rate = 1.02;
duration = 37;
cost = duration * rate;              // compute the cost
balance = balance - cost;     // update balance amount

/* display results */
System.out.print("Call Duration: ");
System.out.print(duration);
System.out.println(" Seconds");
System.out.println("Balance: " + balance + "Rupees ");

} // end of main()
} // end of class CallCost
```

Code Listing 7.1: Sample Java Program

The process of compiling and executing a Java application is shown in figure 7.2.

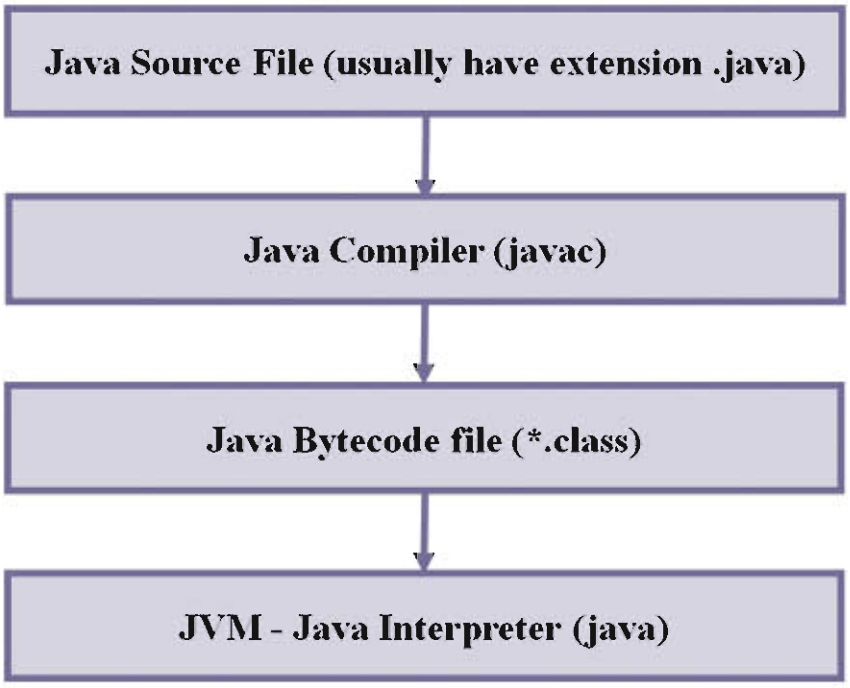


Figure 7.2 : Compilation Process

Java source program file should have extension .java. It's name should be same as that of class when class is public. Note that the name is case sensitive. The compiler 'javac' compiles source file and creates Bytecode file. The name of bytecode file is same as the class name containing main method and it has an extension .class. An interpreter 'java' interprets bytecode and executes it.

Figure 7.3 shows how to compile and execute the program in linux environment using a terminal.

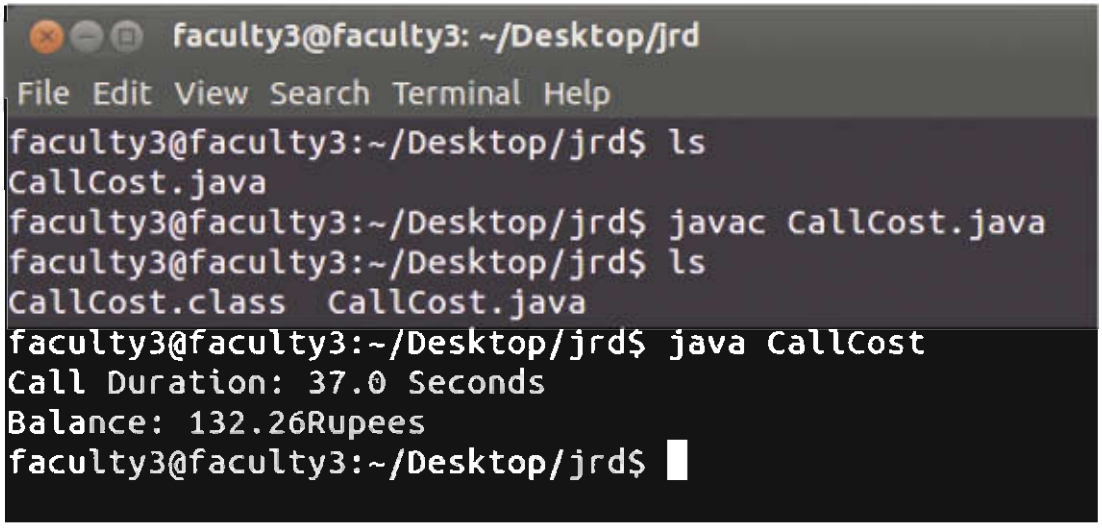


Figure 7.3 : Compilation and Execution of Java program in linux terminal

Notice two main parts in code listing 7.1.

1. The program is enclosed in a class definition; here, a class called 'CallCost'.
2. The body of the program is contained in a routine called main(). In Java applications, main() is the first routine that is run when the program is executed.

Explanation of Code Listing 7.1

The text written after // and enclosed within /* and */ are comments. As we know, comments are not compiled or interpreted.

- Variables are declared using data type followed by variable name.
- Computation part contains expressions including assignment statements.
- Here, several subroutine (also called function or method in Java) call statements are used to display information to the user of the program.
  - Methods used to display results : System.out.print and System.out.println. Both these methods take a value to be displayed as an argument.
  - Method System.out.println adds a linefeed after the end of the information that it displays, while System.out.print does not.
  - Call duration is displayed using three calls. First call displays label 'Call Duration:', second call displays the value of variable 'duration', third call displays label 'Seconds' and then bring the cursor in the next line. Note that string literal is enclosed in double quotes.
  - Balance is displayed using single call. Notice the use of + operator in Java in the parameter passed. It first evaluates the expression given as parameter and then display.

When we run the program, the Java interpreter calls the main() method and the statements that it contains are executed. These statements tell the computer exactly what to do when the program is executed. The main() routine can call other subroutines that are defined in the same class or even in other classes, but it is the main() routine that determines how and in what order the other subroutines are used.

The word "public" in the first line of main() means that this routine can be called from outside the program. This is essential because the main() routine is called by the Java interpreter, which is external to the program. The remainder of the first line of the routine is harder to explain at the moment; so for now, just think of it as part of the required syntax.

Using SciTE

Let us create one more Java application using SciTE editor. Here our application will compute simple interest and display the results. Figure 7.4 shows the code listing and its output. Perform following steps :

- Start SciTE application. Select **File → New**.
- Type the Java program as per code listing given in figure 7.4 and save this source program in a file with name Interest.java. To save the file, select **File → Save** command.
- Compile source program using **Tools → Compile** command.
- If the program is compiled without any error, execute it using **Tools → Go** command.

Interest.java * SciTE

File Edit Search View Tools Options Language Buffers Help

1 Interest.java *

```
// compute simple interest

public class Interest
{
    public static void main(String[] args)
    {
        /* declare variables */
        double principal;   // principal amount in rupees
        double rate;        // interest rate in percentage
        double duration;    // number of years
        double maturity;    // maturity amount
        double interest;    // interest amount

        /* computations. */
        principal = 17000;
        rate = 9.50;
        duration = 3;
        interest = principal * duration * rate / 100; // compute interest amount
        maturity = principal + interest;              // compute maturity amount

        /* display results */
        System.out.println("Principal amount: " + principal + " Rupees");
        System.out.println("Deposit for duration of " + duration + " years");
        System.out.println("Interest Rate: " + rate + " %");
        System.out.println("Interest amount: " + interest + " Rupees");
        System.out.println("Maturity amount: " + maturity + " Rupees");
    } // end of main()
} // end of class Interest
```

```
>javac Interest.java
>Exit code: 0
>java -cp . Interest
Principal amount: 17000.0 Rupees
Deposit for duration of 3.0 years
Interest Rate: 9.5 %
Interest amount: 4845.0 Rupees
Maturity amount: 21845.0 Rupees
>Exit code: 0
```

Figure 7.4 : Executing Java Program using SciTE

Structure of a Java program

Programming languages differ from ordinary human languages in being completely unambiguous and very strict about what is and is not allowed in a program. The rules that determine what is allowed are called the syntax of the language.

Syntax rules specify the basic vocabulary of the language and how programs can be constructed using things like variables, expressions, statements, branches, loops and methods. A syntactically correct program is one that can be successfully compiled or interpreted.

A structure of Java program is shown in figure 7.5. Here, text in angle bracket < and > is used as a placeholder that describes something actual we need to type while writing actual program. The definition of the method (function) in Java consists of function header and the sequence of statements enclosed between braces { and }.

As Java is an object-oriented language. Here, everything is defined as part of class. Thus, a method can't exist by itself. It has to be part of a class.

```
public class <class-name>
{
        <optional-variable-declarations-and-methods>
        public static void main(String[] args)
        {
                <statements >
        }
        <optional-variable-declarations-and-methods>
}
```

Figure 7.5 : Structure of a Java Program

- <class-name> in the first line is the name of the class having main method in it.

  If the name of the class is CallCost, then the program should conventionally be saved in a Java source file with a name CallCost.java. When this file is compiled, another file named CallCost.class is generated. This class file is given a name using class name. Class file, CallCost.class, contains the translation of the program into Java bytecode, which can be executed by a Java interpreter.

- Variable and method declaration after and before main() method is optional.
- Each program must have one class that contains public method main().
- Java is a free-format language. The layout (such as the use of blank lines and indentation) of the program in code listing 7.1 is not a part of the syntax or semantics of the language. The computer doesn't care about the program layout. We can write the entire program together on single line as far as it is concerned. However, layout is important to human readers.
- A program can contain other methods besides main(), as well as other variables. We will learn more about these later.

Now, let us start examining simple Java statements; the basic things we can do in Java within a method definition such as main(). We will learn the following with syntax :

- Data types
- Variables
- Literals
- Comments
- Java statements and expressions
- Arithmetic operators
- Comparisons
- Logical operators

Data Types

Data type determines the required memory size, type of values, range of values and type of operations that can be performed. Java supports eight primitive data types that handle common types for integers, floating-point numbers, characters, and boolean values (true or false).

The primitive data types are named **byte, short, int, long, float, double, char, boolean.** The first four types hold integers (whole numbers such as 17, -38477, and 0), next two hold real numbers (such as 5.8, -129.35), char holds a single character from the Unicode character set and boolean holds one of the two logical values true or false.

These data types are called primitive as they are built into the system. Note that these data types are machine-independent, which means that we can rely on their sizes and characteristics to be consistent across all Java programs on all machines. Table 7.1 lists the details of data types.

Data Type	Storage Space	Type of Value	Range of Values	Default Value
byte	1 byte	Integer	-128 and 127	0
short	2 bytes	Integer	-32768 to 32767	0
int	4 bytes	Integer	-2147483648 to 2147483647	0
long	8 bytes	Integer	-9223372036854775808 to 9223372036854775807	0
float	4 bytes	Real	10 ±38 with about 7 significant digits	0
double	8 bytes	Real	10 ±308 with about 15 significant digits	0
char	2 bytes	Character	16-bit Unicode character	0
boolean	1 byte	Boolean	true, false	false

Table 7.1 : Data Types in Java

Integer numbers with b bits precision store signed values in the range of $(-2^{b-1}-1, 2^{b-1})$. When they are preceded with keyword unsigned, the values are in the range of $(0, 2^{b-1})$.

Real numbers in Java are compliant with IEEE 754 (an international standard for defining floating-point numbers and arithmetic). Java uses the Unicode character set. The char type has 16 bits of precision and is unsigned. This allows thousands of characters from many different languages and different alphabets to be used in Java. Data type boolean is not a number, nor can it be treated as one.

Variable

If we want anything to be remembered by the computer during program execution, then it needs to be stored in the memory of a computer. Programs manipulate the data that are stored in memory. In machine language, data can only be referred to by giving the numerical address of the location in memory where it is stored. In a high-level language such as Java, names are used instead of numeric address of memory location to refer to data. The programmer has to remember only the name. A name used to refer to the data stored in memory is called a variable.

A variable can take different data values at different times during the execution of the program, but it always refers to the same memory location. A variable can be used in a java program only if it has first been declared.

One or more variables can be declared in Java using declaration statement with following syntax:

<type-name> {variable-names};

The conventions used here in the syntax are as follows:

- angle brackets < > denote the item to be specified by user
- curly brackets { } denote the list of items separated by commas

Here {variable-names} denote the list of variable names. When the list contains more than one item, items should be separated by commas.

<type-name> is to be replaced with the keyword denoting the data type of the variables. It is used to determine the size of variable, the values it can hold and the operations that can be performed on it. When the computer executes a variable declaration statement, it sets aside memory for the variable and associates the variable's name with that memory.

Some examples of variables are as mentioned :

int marks;

double amount, interest;

float rate;

char grade;

boolean isPass;

To define variable name, we need to follow certain rules as mentioned :

- It must begin with an alphabet, underscore (_) or dollar sign ($). After first character, it may contain digits, alphabets, $ and underscore.

  Some legal names are: birth_date, result, CallCost, top5students, date, amount$, $price

  Some illegal names are: 4me, %discount, birth date

- No spaces are allowed in variables. Thus, birth date is invalid variable name.
- It cannot be a reserved word. Reserved words have special use in Java and cannot be used by the programmer for other purposes. Examples of some reserved words are class, public, static, if, else and while.

Guidelines for naming variables :

- Choose meaningful variable names. Java allows variable name of any length.
- When a name includes several words, such as 'balance amount', follow one of the conventions mentioned below :
  - Capitalize the first alphabet of each word, except for the first word. This is sometimes referred to as camel case, since the upper case letters in the middle of a name are supposed to look like the humps on a camel's back. Example: balanceAmount, birthDate

- Separate words with underscore. Example: balance_amount, birth_date

- Note that Java is case-sensitive. So, upper case and lower case letters are considered to be different. Thus, variable names balance and Balance are different.
- It is customary for names of classes to begin with upper case letters, while names of variables and of methods begin with lower case letters.

Good programming style for declaring variables :

- Declare only one variable in a declaration statement.
- Include a comment with each variable declaration to explain its purpose in the program.
- Declare important variables at the beginning of the function. Declare variables which are not important to the overall logic of the function at the point where they are first used.

In Java, there are three kinds of variables: instance variables, class variables, and local variables. Function parameters and variables declared in the function are considered as local variables. We will study about instance and class variables later.

We can also give each variable an initial value while declaring. For example,

```
int marksObtained, totalMarks=100, counter=0;

boolean isPass = true;
```

In general, syntax of declaring variables is as follows: <type name> {variable [= <value>,]};

Here items in square bracket [ ] denotes optional item.

Note that local variables are not initialized with default values. It is programmer's responsibility to assign value to such variables before their first use. Figure 7.6 shows the use of variable in java program.

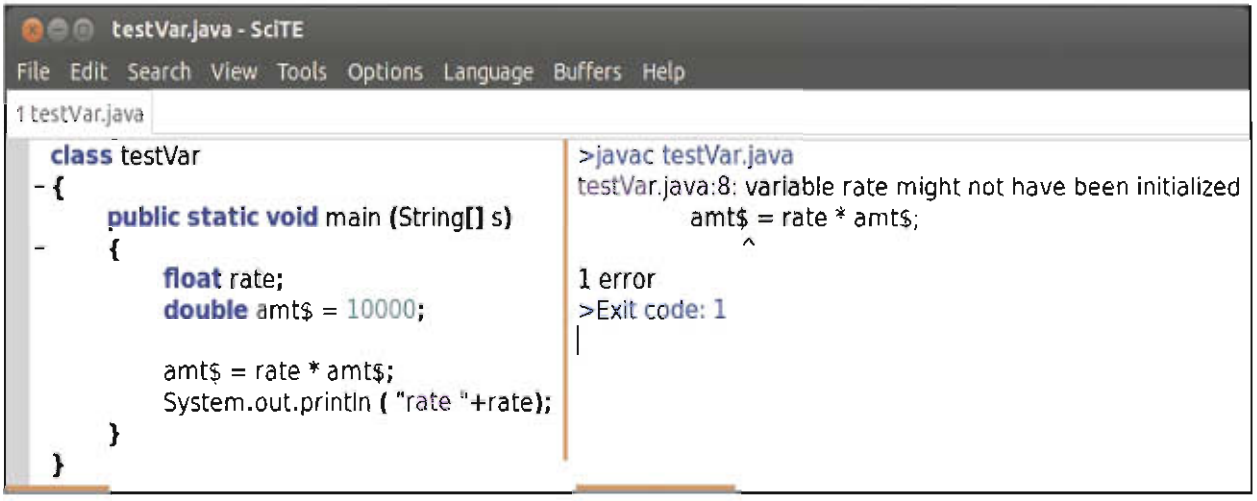


Figure 7.6 : Local Variables in Java

Observe that the local variable 'rate' is not assigned any value before its use in statement 'amt$ = rate * amt$'. When we try to compile the program the compiler will give an error as can be seen in figure 7.6.

Literals

A name used for a constant value is known as literal. There are some different kinds of literals in Java for number, character, string and boolean values.

Numeric Literals

Numeric literals are used to represent integer or real numbers for example, 157 and 17.42 are literals.

Integer literals are literals that are whole numbers. Java allows decimal (base 10), octal (base-8) and hexadecimal (base-16) and Unicode integer literals.

Ordinary integers such as 4, 157, 17777 and -32 are decimal integer literals of type byte, short or int depending on their size. A decimal integer literal larger than int is automatically of type long. We can force a smaller number to be a long by appending an L or l (upper or lower case letter l) as a suffix to that number (for example, 4L is a long integer having value 4). Negative integers are preceded by a minus (-) sign, for example, -45.

Octal numbers use only the digits 0 through 7. In Java, a numeric literal with a leading 0 (zero) is interpreted as an octal number. For example, literal 045 represents octal integer whose decimal number value is 37.

Hexadecimal numbers use 16 digits, the usual digits 0 through 9 and the letters A, B, C, D, E, and F. Upper case and lower case letters can be used interchangeably in this context. The letters A to F represent the numbers 10 to 15 respectively. In Java, a hexadecimal literal begins with 0x or 0X. Examples of hexadecimal literals are 0x45 or 0xFF7A. Hexadecimal numbers are also used in character literals to represent arbitrary Unicode characters.

A Unicode literal consists of \u followed by four hexadecimal digits. For example, the character literal '\u00E9' represents the Unicode character that is an "e" with an acute accent.

Java 7 also supports binary numbers, using the digits 0 and 1 and the prefix 0b (or 0B). For example: 0b10110 or 0b101011001011.

Real number literals are called floating point literals. These numbers can be represented using two types of notations: standard and scientific.

In standard notations, the integer part and the fractional part are separated with decimal point (.), for example 12.37.

In scientific notation, a number is followed by letter e (or E) and a signed integer exponent, for example 1.3e12 and 12.3737e-108. The "e12" and "e-108" represent powers of 10. Hence 1.3e12 means 1.3 times 1012 and 12.3737e-108 means 12.3737 times 10^{-108}. Scientific format can be used to express very large and very small numbers.

In Java, floating point literal by default is of the type double. To make a literal of type float, we have to append an "F" or "f" as a suffix to the number. For example, "1.2F" specifies literal 1.2 to be considered as a value of type float.

```
testVar.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 testVar.java

class testVar
{
        public static void main (String[] s)
        {
                float rate = 10.2;
                double amt$ = 10000;

                amt$ = rate * amt$;
                System.out.println ( "rate "+rate);
        }
}
```

```
>javac testVar.java
testVar.java:5: possible loss of precision
found   : double
required: float
                float rate = 10.2;
                             ^
1 error
>Exit code: 1
```

Figure 7.7 : Compilation error

Observe that in figure 7.7 we get a compilation error as we have assigned a literal 10.2 to variable of float type. To make the program work we have to write this statement as "float rate = 10.2f;". The suffix F can be written in either lower or upper case.

Boolean literals

For the type boolean, there are precisely two literals: true and false. These literals are to be typed without quotes. They represent values, not variables. Boolean values occur most often as the values of conditional expressions. In C, 0 is treated as false and non-zero value is treated as true. In Java, literals true and false are not associated with any numeric value.

Character literals

Character literals are expressed by a single character surrounded by single quotes: 'a', '#', '3'and so on. Characters are stored as 16-bit Unicode characters. Certain special characters have special literals that use a backslash (\) as an "escape character". Table 7.2 lists the special codes that can represent nonprintable characters, as well as characters from the Unicode character set.

Escape code	Meaning
\n	New line
\t	Tab
\b	Backspace
\r	Carriage return
\f	Form feed (New page)
\\	Back slash character
\'	Single quote character
\"	Double quote character
\ddd	Character represented by three octal digits (d: 0 to 7)
\xdd	Character represented by two hexadecimal digits (d: 0 to 9, a to f)
\udddd	Character represented by Unicode number dddd (d: hexadecimal digit)

Table 7.2 : Escape Codes

String literals

In Java all the other data types represent objects rather than "primitive" data. We are not concerned with objects for the time being. However, we will consider here one predefined object type that is very important: the type String. A String is a sequence of characters.

In code listing 7.1, we have used a string literal: "Balance:". String literal is a sequence of characters enclosed in double quotes. Within a string, special characters can be represented using the backslash notation as given in Table 7.2.

For example, to represent the string value: Many "Congratulations!", we need to type string literal: "Many, \"Congratulations!\""

Similarly in string "This string brought to you by Java\u2122", the Unicode code sequence \u2122 produces a trademark symbol (™).

Comments

Comments in a program are for human readers only; they are entirely ignored by the computer. Comments are important to make the program easy to understand for everyone.

Java supports following types of comments :

- Single-line comment: It begins with double slashes (//) and extends till the end of a line. The computer ignores the // and everything that follows it on the same line.
- Multi-line comment: It begins with /* and ends with */. This type of comment is usually used to have more than one line as comments. Actually, one may embed phrase as well as one or more entire lines as comments between /* and */. Any text between two delimiters /* and */ is considered as a comment. Comments cannot be nested; that is, we cannot have a comment inside a comment.
- Documentation comment: These type of comments begin with /** and end with */. They are used for creating API documentation from the code. These are special comments that are used for the javadoc system. Discussion of javadoc is out of scope of this book.

Other than comments, everything else in the program is required to follow the rules of Java syntax.

Expressions

Expressions are an essential part of programming. The basic building blocks of expressions are literals (such as 674, 3.14, true, and 'X'), variables, and function calls. Recall that a function is a subroutine that returns a value.

Simple expression can be a literal, a variable, a function call. More complex expressions can be built up by using operators to combine simpler expressions.

Operators include arithmetic operators (+, -, *, /, %), comparison operators (<, >, =, ...), logical operators (and, or, not...). When several operators appear in an expression, there is a question of precedence, which determines how the operators are grouped for evaluation. For example, in the expression "A + B * C", B*C is computed first and then the result is added to A. We say that multiplication (*) has higher precedence than addition (+). If the default precedence is not what

we want, we can use parentheses to explicitly specify the grouping. For example, in expression "(A + B) * C", it adds A and B first then multiplies the result by C.

Operators

Operators are special symbols used to build an expression. Java supports many types of different operators. In this chapter, we will discuss the following operators :

- Arithmetic operators
- Comparison operators
- Logical operators
- Conditional operator
- Assignment operator

Arithmetic operators

In Java, basic arithmetic operators are addition (+), subtraction (-), multiplication (*), division (/) and modulus (%). All these operators are binary, they take two operands. Operators + and - can be used as unary (taking only one operand) also. All operators can be applied on any type of numeric data: byte, short, int, long, float, or double. They can also be used with values of type char, which are treated as integers in this context. With char type of data, its Unicode code number is considered as its value when it is used with an arithmetic operator. Arithmetic operators are described in table 7.3.

Operator	Meaning	Example	Result
+	Addition	2 + 8	10
-	Subtraction	2 - 8	-6
*	Multiplication	2 * 8	16
/	Division	8 / 2	4
%	Modulus (Gives remainder after division where quotient is an integer value)	8 % 3 25.8 % 7	2 4.8

Table 7.3 : Arithmetic operators

Data type of the result after binary arithmetic operation :

- When both operands are of same data type, the data type of the result is same as the type of operands.
  - If both operands are integers, result is an integer. For example, result of 9/2 is 4 and not 4.5. Integer division results into integer value and remainder is discarded.
  - If both operands are float, result is float. Example : 9f/2f results in 4.5.

- When both operands are of different data types,
  - First of all, lower range data type is implicitly converted to higher data type to have the same types of operands. This type of conversion is also known as promotion.
  - Now, the result of an expression will be same as higher range operand.
  - Example: 4 + 3.5 results in 7.5, 9/2.0 results in 4.5, 9f/2 results in 4.5.

Points to remember :

- With modulus operator %, if first operand is negative, the result is negative.
- In Java, % operator can be used with floating point data types also. The result is the remainder after integer quotient. Thus for 25.8 % 7, integer quotient is 3 and remainder is 25.8 (3*7) = 4.8.
- Operator + can also be used to concatenate a string.

When operator + is applied with one of the operand of type String, other operand is automatically converted into type String. This is also an example of implicit type conversion. This type of conversion is seen in code listing 7.1 in an expression ' "Balance: " + balance ' while printing balance amount. It is also used in code listing shown in figure 7.8.

ArithOperators.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 ArithOperators.java

```
class ArithOperators
{
    public static void main (String[] args)
    {
        short x = 6;
        int y = 4;
        float a = 12.5f;  // suffix f to explicitly specify floating-point literal
        float b = 7.2f;   // suffix f for floating-point literal

        System.out.println ("x is " + x + ", y is " + y);
        System.out.println ("x + y = " + (x + y));
        System.out.println ("x - y = " + (x - y));
        System.out.println ("x / y = " + (x / y));
        System.out.println ("x % y = " + (x % y));
        x = -6;
        System.out.println ("x % y = " + (x % y));
        y=-4;
        System.out.println ("x % y = " + (x % y));
        x=6; y=-4;
        System.out.println ("x % y = " + (x % y));

        System.out.println ("a is " + a + ", b is " + b);
        System.out.println ("a / b = " + (a / b));
        System.out.println ("a / x = " + (a / x));
        System.out.println ("a % x = " + (a%x));
        System.out.println ("a % b = " + (a%b));
    } // end main()
} // end class ArithOperators
```

```
>javac ArithOperators.java
>Exit code: 0
>java -cp . ArithOperators
x is 6, y is 4
x + y = 10
x - y = 2
x / y = 1
x % y = 2
x % y = -2
x % y = -2
x % y = 2
a is 12.5, b is 7.2
a / b = 1.7361112
a / x = 2.0833333
a % x = 0.5
a % b = 5.3
>Exit code: 0
```

Figure 7.8 : Java program showing use of arithmetic operators

Explanation of program shown in figure 7.8 :

- We initially define four variables in main() method: x and y, which are integers (type short and int); a and b, which are floating point real numbers (type float).

- Remember that default type for floating-point literals (such as 12.5) is double. So, f is suffixed with literals 12.5 and 7 to treat them as float type.
- The System.out.println() method merely prints a message to the standard output device. This method takes a single argument, a string, but we can use + to concatenate values into a string. As pointed before, Java converts numbers into string and then concatenates the two.
- The result of x%y is 2 when x=6 and y=-4 because first operand is positive.
- The result of a%x is 0.5 where a=12.5 and x is 6. Here, integer quotient is 2 and the remainder is 0.5. Similarly in case of a%b where a=12.5 and b=7.2; the integer quotient is 1 and the remainder is 5.3.
- See the mixed type of operands in the call: System.out.println ("a / x = " + (a / x));. Here variable a is float and x is int, the result after division is float. While evaluating an expression, values are promoted to the higher data type operand.

Experiment with the same program after removing parenthesis of expression x+y in 2nd call of System.out.println method (i.e. in line 11 of code listing in figure 7.8, replace an expression "x + y = " + (x + y) with an expression "x + y = " + x + y) and analyse the results shown in figure 7.9.

ArithOperators.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 ArithOperators.java

```
class ArithOperators
{
    public static void main (String[] args)
    {
        short x = 6;
        int y = 4;
        float a = 12.5f;  // suffix f to explicitly specify floating-point literal
        float b = 7.2f;   // suffix f for floating-point literal

        System.out.println ("x is " + x + ", y is " + y);
        System.out.println ("x + y = " + x + y);
        System.out.println ("x - y = " + (x - y));
        System.out.println ("x / y = " + (x / y));
        System.out.println ("x % y = " + (x % y));
        x = -6;
        System.out.println ("x % y = " + (x % y));
        y=-4;
        System.out.println ("x % y = " + (x % y));
        x=6; y=-4;
        System.out.println ("x % y = " + (x % y));

        System.out.println ("a is " + a + ", b is " + b);
        System.out.println ("a / b = " + (a / b));
        System.out.println ("a / x = " + (a / x));
        System.out.println ("a % x = " + (a%x));
        System.out.println ("a % b = " + (a%b));
    } // end main()
} // end class ArithOperators
```

```
>javac ArithOperators.java
>Exit code: 0
>java -cp . ArithOperators
x is 6, y is 4
x + y = 64
x - y = 2
x / y = 1
x % y = 2
x % y = -2
x % y = -2
x % y = 2
a is 12.5, b is 7.2
a / b = 1.7361112
a / x = 2.0833333
a % x = 0.5
a % b = 5.3
>Exit code: 0
```

Figure 7.9 : Effect of removing ( ) from (x+y) in line 11of code listing in figure 7.8

Increment and Decrement operators

Unary operators ++ and -- are called the increment operator and the decrement operator respectively. Operator ++ adds 1 to a variable and - - subtracts 1 from the variable.

These operators can be used on variables belonging to any of the integer types and also on variables of type char.If x is an integer variable, we can use x++, ++x, x--, --x as expressions, or as parts of larger expressions.

When ++ or -- operator is used after variable name, it is known as post-increment or post-decrement operator. The old value of variable is used while evaluating the expression and thereafter the value of variable is incremented or decremented. For example, let the value of variable x be 3 before executing statement y = 4 + x++;. Here old value of x is used while evaluating expression on right-hand side and thereafter x is incremented. As a result, value of x will be 4 and y will be 7.

When ++ or -- operator is used before variable name, it is known as pre-increment or pre-decrement operator. Here the value of variable is incremented or decremented first and then this new value is used in expression. For example, in statement y = 4 + ++x;, if old value of x is 3, then value of x is incremented first and then this new value is used in evaluating an right-hand side expression. So value of x will be 4 and that of y will be 8.

When this operator is used in a standalone statement, use of pre or post does not make any difference. For example, statements x++; and ++x; are standalone statements.

Comparison operators

Comparison operators are also known as relational operators. The comparison operators in Java are: ==, !=, <, >, <=, and >=. The meanings of these operators are:

A == B	Is A "equal to" B ?
A != B	Is A "not equal to" B ?
A < B	Is A "less than" B ?
A > B	Is A "greater than" B ?
A <= B	Is A "less than or equal to" B ?
A >= B	Is A "greater than or equal to" B ?

These operators can be used to compare values of any of the numeric types as well as of char type. For characters, their unicode numeric values are used in comparison.

After applying comparison operator, the result of expression is boolean; either true or false. So, such expressions are also called boolean-valued expression.

We can also assign boolean-valued expressions to boolean variables, just as assigning numeric values to numeric variables.

Operators == and != can also be used to compare boolean values. For example :

```
boolean bothPositive;  bothPositive = ((x > 0) == (y > 0));
```

Usually, relational operators are used in if statements and loops.

Logical operators

Logical operators are also called boolean operators, as they operate on Boolean operands. In Java,

logical operations AND, OR, XOR and NOT are performed using operators &&, ||, ^ and ! respectively.

Boolean operator && is used to combine two boolean values for logical AND operation. The result of expression A && B is true only if both the operands A and B are true. For example, (x == 0) && (y == 0) is true if and only if both x is equal to 0 and y is equal to 0.

Boolean operator for logical OR is || (two vertical line characters). Expression A || B is true if either A is true or B is true, or if both are true. The result is false only if both A and B are false. For example, (x == 0) && (y == 0) is false only if both x and y are not equal to 0.

Boolean operator for logical NOT is denoted by ! and it is a unary operator. It results in complemented result. If the operand is true, result is false and vice versa.

In addition, there is operator ^ to perform logical XOR (exclusive OR). It returns true only if its operands are different (one true and one false) and false otherwise. Thus result is false when both operands have same boolean value. For example, (x == 0) ^ (y == 0) is true if only one of the x or y is zero.

Short circuiting

When using the conditional AND and OR operators (&& and ||), Java does not evaluate the second operand unless it is necessary to resolve the result. If first operand is false in case of &&, there is no need to evaluate second operand. Similarly, if first operand is true in ||, there is no need to evaluate second operand.

For example, consider expression (x != 0) && (y/x > 1). If the value of x is zero, first sub-expression (x !=0) results in false. As logical operator && results in false when any one of the operands is false, there is no need to evaluate second sub-expression (y/x > 1). Here, the evaluation has been short-circuited and the division by zero is avoided. Without the short-circuiting, there would have been a 'division by zero' error at runtime.

Conditional operator

The conditional operator in Java is a ternary operator using three operands. It uses two symbols ? and : in the expression to delimit three operands. It takes the following form:

<boolean-expression> ? <expression1> : <expression2>

Here, the first operand is a boolean expression and is evaluated first. If its value is true, value of the entire expression is the value of second operand expression1; otherwise it evaluates to third operand expression2.

For example, consider statement: next = (N % 2 == 0) ? (N/2) : (3*N+1);

Suppose value of N is 8. The value of first operand (N % 2 == 0) is true, so value of right-hand side expression is the value of expression (N/2), i.e. 4. If N is odd, first expression results in false and it will assign the value (3*N+1) to next. Note that the parentheses in this example are not required, but they do make the expression easier to read.

Assignment

In Java, an expression containing assignment (=) operator is generally referred to as assignment statement.

Once a variable has been declared, we can assign a value to that variable by using the assignment operator '='. In Java, one of the ways to get data into a variable is with an assignment statement. An assignment statement takes the form :

< variable > = < expression >;

where <expression> represents anything that refers to or computes a data value.

When an assignment statement is executed, it first evaluates the expression on right side of = sign and then put the resulting data value into the variable on left side of = sign.

For example, consider the simple assignment statement: rate = 10.02f;

Here the variable in this assignment statement is 'rate', and the expression is the number '10.02f'. Execution of this statement replaces the value of float variable 'rate' by 10.02.

Now, consider another assignment statement: balance = balance - cost;

Here expression 'balance-cost' is evaluated first using the existing values of variables 'balance' and 'cost' in memory. There after the resulting value is placed in variable 'balance' replacing its older value.

When a variable is used in an expression on right-hand side, it refers to the value stored in the variable. When a variable is used on the left-hand side of an assignment statement, it refers to memory location that is named by the variable to place the value. So, variable on left-hand side of expression is called lvalue, referring to location in memory.

In general, the type of the expression on the right-hand side of an assignment statement should be the same as the type of the variable on the left-hand side. However, if they are not matching, the value of expression is automatically converted to match the type of the variable. If the data type of expression is larger than the variable on left-hand side, it may result in an error due to precision problem. For example, there may not be automatic conversion from int to short or double to float.

Shorthand assignment operators

Java also support shorthand version of assignment. It saves the typing time. It takes the form <variable> <operator> = <expression>. Its effect is same as <variable> = <variable> <operator> <expression>. Here the operator should be a binary operator using two operands.

Some examples of shorthand assignment operators are; a += b and q &&= p. Here a += b is same as a = a + b, while expression q &&= p is same as expression q = q && p.

Type cast

In some cases, we may want to force a conversion that wouldn't be done automatically. For this, we can use what is called a type cast. A type cast is indicated by putting a type name in parentheses before the value we want to convert. Thus it takes a form :

(<data-type>) <expression>

For example,

int a; short b;

a = 17; b = (short)a;

Here variable a is explicitly converted using type cast to a value of type short

Precedence and associativity of Java operators

Java has well-defined rules for specifying the order in which the operators in an expression are evaluated when the expression has several operators. The operators are evaluated as per their priority (or precedence).

Precedence Order

When two operators are having different priority, then an operator with the higher precedence is operated first. For example, in expression a + b * c, multiplication is having higher precedence than addition, so b*c is evaluated first and then the result is added to a. Thus it is as good as writing a + (b*c). Precedence rules can be overridden by explicit parentheses. So, instead of creating confusion, use parenthesis freely.

Associativity

When two operators with the same precedence appear in the expression, the expression is evaluated according to its associativity. Associativity determines the direction (left-to-right or right-to-left) in which operations are performed. In most of the cases, associativity is from left to right. For unary operations and assignment, it is from right-to-left.

Table 7.4 gives the list of operators discussed in this chapter, listed in order from highest precedence (evaluated first) to lowest precedence (evaluated last):

Operations	Operators	Associativity
Unary operations	++, --, !, unary - and +, type-cast	Right-to-left
Multiplication, division, modulus	*, ?, %	Left-to-right
Addition and subtraction	+, -	Left-to-right
Relational operators	<, >, <=, >=	Left-to-right
Relational operators (Equality and inequality)	==, !=	Left-to-right
Logical AND	&&	Left-to-right
Logical OR	||	Left-to-right
Conditional operator	? :	Right-to-left
Assignment operators	=, +=, -=, *=, /=, %=	Right-to-left

Table 7.4 : Operators and their precedence

For example x = y = z = 7 is treated as x = (y = (z = 7)), leaving all three variables with the value 7. This is due to right-to-left associativity of assignment (=) operator. Remember that an assignment is an operator, so assignment statement evaluates to the value on the right hand side expression. Thus, in x=y=z=7, it evaluates z=7 first. This assigns value 7 to variable z and value of expression z=7 is also 7, which is making the expression as x=y=7.

On the other hand, 72 / 2 / 3 is treated as (72 / 2) / 3 since the / operator has left-to-right associativity. It is to be noted that there is no explicit operator precedence table in the Java Language Specification and different tables on the Web and in textbooks disagree in some minor ways.

Control Structures

In general, the statements are executed sequentially, one by one. Sometimes, program logic needs to change the flow of this sequence. The statements that enable to control the flow of execution are considered as control structures.

There are two types of control structures: loops and branches. Loops are used to repeat a sequence of statements over and over until some condition occurs. Branches are used to choose among two or more possible courses of action, so also called selective structure.

In Java, control structures that are used to determine the normal flow of control in a program are: if statement, switch statement, while loop, do..while loop and for loop. Each of these structures is considered to be a single statement, may be a block statement.

Block

A block statement is a group of statements enclosed between a pair of braces, "{" and "}".

The format of a block is:

```
{
        <statements>
}
```

Block can be used for various purposes as follows :

- To group a sequence of statements into a unit that is to be treated as a single statement, usually in control structures. (will be discussed soon)
- To group logically related statements.
- To create variables with local scope for statements within a block.

For example,

```
{ // This block exchanges the values of x and y
        int temp;       // temporary variable for use in this block only
        temp = x;       // save a copy of x in temp
        x = y;          // copy y into x
        y = temp;       // copy temp into y
}
```

When we declare variables inside a block, they are local in that block and such variables will cease to exist after the block. We cannot use temp outside the block in which it is declared.

A variable declared inside a block is completely inaccessible and invisible from outside that block. When the variable declaration statement is executed, memory is allocated to hold the value of the variable. When the block ends, that memory is released and is made available for reuse. The variable is said to be local to the block.

There is a general concept called the "scope" of a variable. The scope of a variable is the part of the program in which that variable is valid. The scope of a variable defined inside a block is limited to that block only.

When we try to declare a variable with the same name as of the variable in scope, there will be an error. See code listing and the result shown in figure 7.10. Here, we have used labelled block only for better understanding. It is not must to use in the given example. We will see the use of labelled block later in this chapter.

In code listing shown in figure 7.10, we have tried to declare variable x in block labelled blk2. Block blk2 is in the scope of main method block. Variable x that is declared in the block of main method is also having its scope in blk2. Thus, declaring variable x in blk2 conflicts with variable x in the scope of main method block and compiler shows an error.

```
Block.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 Block.java

class Block
{
    public static void main (String[] args)
    {
        int x = 10;

        blk1:
        { // start of block1
            int y = 50;
            System.out.println("inside the block1:");
            System.out.println("x: " + x);
            System.out.println("y: " + y);
        } // end of block1

        blk2:
        { // start of block2
            int y = 20;
            int x = 30;  // conflict with x in main
            System.out.println("inside the block2:");
            System.out.println("x: " + x);
            System.out.println("y: " + y);
        } // end of block2

    System.out.println("outside the block: x is " + x);
    }  // end main()
} // end class Block
```

```
>javac Block.java
Block.java:18: x is already defined in main(java.lang.String[])
                int x = 30;  // conflict with x in main
                    ^
1 error
>Exit code: 1
```

Figure 7.10 : Conflicting variable names in the same scope

Figure 7.11 shows the modified code. Here variable x is declared in the scope of entire main method. As blk1 and blk2 are in the same block, x is available in blk1 and blk2 as well. Variable y declared in labelled block blk1 ceases to exist outside that block. After execution of blk1, variable y is discarded. In blk2, when variable y is declared, it may be allocated any memory location from available memory. If we do not initialize value of variable y in blk2, it will return an error when we refer it in blk2.

This means that variable y of blk2 has nothing to do with variable y of blk1. Actually variable y of blk1 is not accessible in blk2.

```
Block.Java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 Block.java

class Block
{
        public static void main (String[] args)
        {
                int  x = 10;

                blk1:
                { // start of block1
                        int y = 50;
                        System.out.println("inside the block1:");
                        System.out.println("x: " + x);
                        System.out.println("y: " + y);
                } // end of block1

                blk2:
                { // start of block2
                        int y = 20;
                        //int x = 30;  // conflict with x in main
                        System.out.println("inside the block2:");
                        System.out.println("x: " + x);
                        System.out.println("y: " + y);
                } // end of block2

        System.out.println("outside the block: x is " + x);
        }  // end main()
} // end class Block
```

```
>javac Block.java
>Exit code: 0
>java -cp . Block
inside the block1:
x: 10
y: 50
inside the block2:
x: 10
y: 20
outside the block: x is 10
>Exit code: 0
```

Figure 7.11 : Program showing scope of variable

if Statement

The if statement when used in a program enables to take one of two alternative courses of action, depending on whether the value of a given boolean-valued expression is true or false. It is an example of a "branching" or "decision" or "selective" control structure. The form if statement is as shown in figure 7.12.

```
if ( <boolean-expression> )
{
        <statement1>
}
else
{
        <statement2 >
}
```

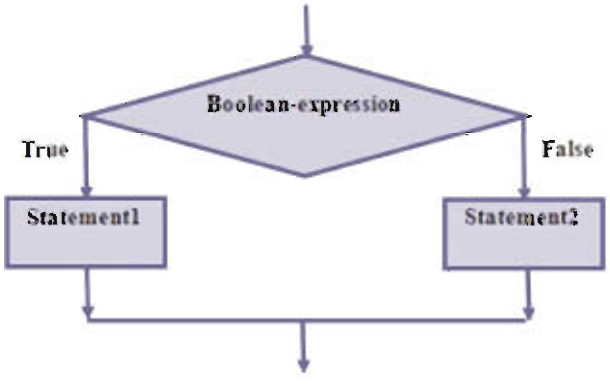


Figure 7.12 : Structure of if statement

When if statement is executed, it first evaluates boolean expression. If its value is true, it executes statement1 (all statements in a block); otherwise it executes a block of statements written after keyword else. <statement> in if statement is usually a block statement. It may be any single statement, but

it is advisable to use block to make it easy to insert other statements later. Refer following code snippet that uses if statement to determine whether an integer is even or odd.

```
if ( x%2 == 0 ) // assume int x
  { // divisible by 2
        System.out.print(x);
        System.out.println (" is even");
  }
else
  { // not divisible by 2
System.out.println (x + " is odd");
  }
```

Keyword else and a block after else are optional. So, if statement can be without else part as shown in the following form:

```
if ( <boolean-expression> )
{
        <statement1>
}
```

When an if statement is used in another if statement, it is called nested-if statement. See following example which determines grade based on marks obtained.

```
if ( marks >= 70 ) // int marks
  { grade = 'A';       // char grade
  }
else
  {
   if (marks >= 60)
        grade = 'B';
   else if (marks >= 50)
                grade = 'C';
        else
         grade = 'F';
 }
```

Let us consider one more example of nested-if statement.

```
if ( x > 0 )
        if (y > 0)
                System.out.println("both x and y are greater than zero");
else
        System.out.println("x <= 0");
```

Here, it seems that the else part is corresponding to "if (x > 0)" statement, but actually it is attached to "if (y > 0)", which is closer. Thus it is executing the true part of (x>0).

If we want to attach else to "if (x>0)", we should enclose the nested if in a block as shown here.

```
if ( x > 0 )
{
        if (y > 0)
                System.out.println("both x and y are greater than zero");
}
else
System.out.println("x <= 0");
```

Switch Statement

A switch statement is used when there are many alternative actions to be taken depending upon the value of a variable or expression. Here, the result of the test expression must be of the type taking discrete values. The form of switch statement is shown below :

```
switch (<expression>)
{
        case <constant-1>:
                <statements-1>
                break;
        case <constant-2>:
                <statements-2>
                break;

        // more such cases

        case <constant-n>:
                <statements-n>
                break;

        default:
                <statements-n1>
}
```

In the switch statement, the test expression should be of the type byte, char, short or int. It can also be of enum data type (not discussed here).

When executing switch statement, the value of the test expression is compared with each of the case values in turn from case1 onwards. If a match is found, the respective case statements (not necessarily a block) are executed. If no match is found, the default statement is executed. The default

is optional, so if there is no match in any of the cases and default doesn't exist, the switch statement completes without doing anything.

Note that break statement used after each case is not mandatory. Use of break statement is to break the switch statement, i.e. to jump at the first statement after the end of switch.

Repetitive control structures

Java supports three types of looping constructs: for, while and do...while. Figure 7.13 shows the syntax and structure of all these loops.

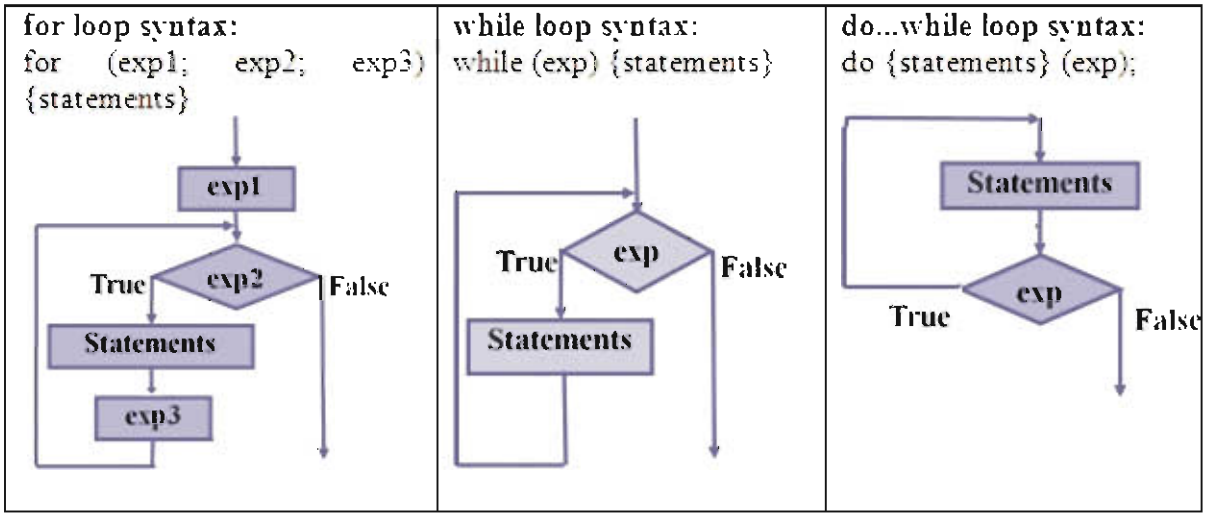


Figure 7.13 : Repetitive Control Structures

In for and while loop, test expression is evaluated first and the statements in the loop are executed if condition is true. These loops are entry-controlled or pre-test loop constructs. Here, it is possible that statements in a loop are not executed at all.

When number of iterations are pre-determined, usually for loop is used. In this loop, all the three expressions are optional. First expression is initializer, second expression is condition and third expression is an iterator. Thus for(;;) is valid; but requires some control statements to break the loop. If break statement is not executed, it will result in infinite loop.

In do...while loop, it evaluates the test expression after executing the statements in a loop. It repeats the loop if the test condition is true. Thus, do...while loop is exit-controlled or post-test loop construct. Here, statements of loop are executed at least once.

Following examples print integer values from 0 to 9 using all the three loop constructs.

Use of for loop :

```
for (int i = 0; i < 10; i++)
{
        System.out.println(i);
}
```

Use of while loop :

```
int i = 0;
```

```
while(i < 10)
{
        System.out.println(i++); //prints i before applying i++
}
```

Use of do...while loop :

```
int i = 0;
do
{
System.out.println(i++);
} while(i < 10);
```

Use of break and continue statement

The break statement is used to transfer the control outside switch or loop structure.

When break is used in switch structure, it skips all the following statements and control is transferred at the first statement after the end of switch statement.

In a loop, break statement is used to exit the loop. When break is executed in a loop, all the following statements in a body of the loop are skipped and no further iteration takes place. The control is transferred at the first statement after the end of loop structure. Remember that break jumps outside the nearest loop containing this statement.

Use of continue statement is used to skip the following statements in a loop and continue with the next iteration.

Both the statements break and continue are used the same way as in C language.

Nested loops

Loops of same or different types can be nested in Java. Figure 7.14 shows the use of nested loops, break and continue statement.

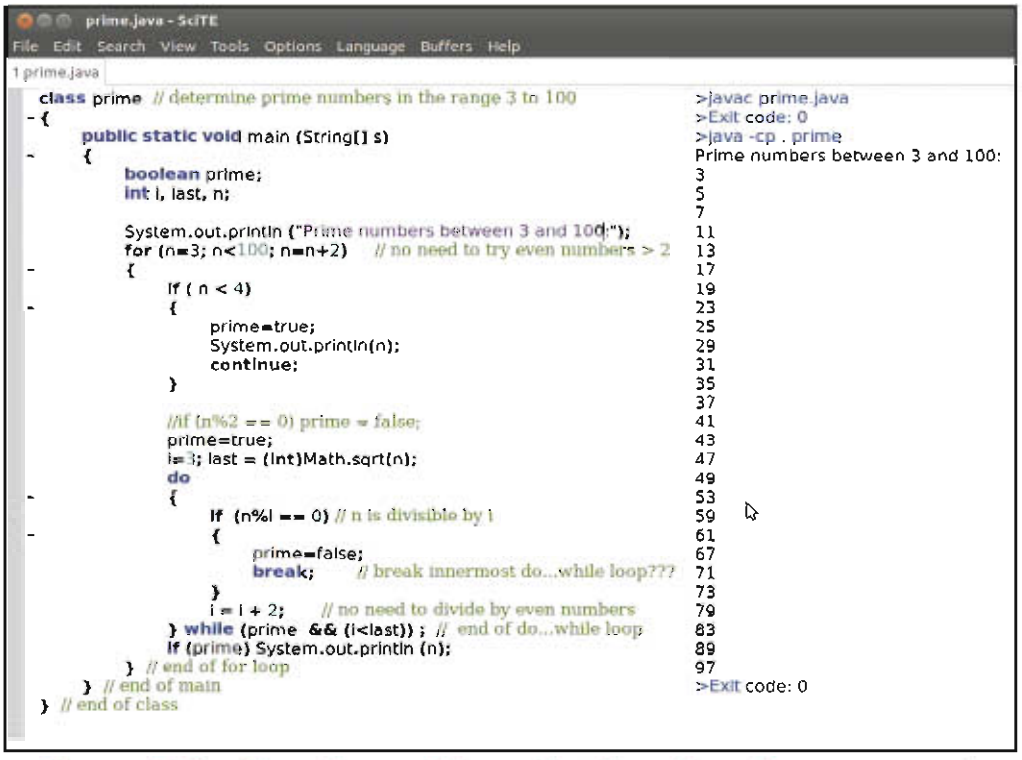


Figure 7.14 : Use of nested loops, break and continue statements

In figure 7.14 we have written a program that prints prime numbers between 3 and 100. Additionally, it also shows the use of sqrt function. The Java class libraries include a class called Math. The Math class defines a whole set of math operations. Function sqrt() is one of the static method member of the class Math and is invoked as Math.sqrt().

Labelled loops and labelled break

When we use nested loops, break statement breaks the nearest enclosing loop and transfers the control outside the loop. Similarly continue also restart the enclosing loop.

If we want to control which loop to break and which loop to reiterate, we can use labelled loop. To use a labelled loop, add the label followed by colon (:) before the loop. Then, add the name of the label after the keyword break or continue to transfer control elsewhere other than enclosing loop. Figure 7.15 shows a program that uses labelled loops.

When code in figure 7.15 is executed, the value of i*x is 350 when i=5 and x = 70. As the if condition in while loop is evaluated to true, it executes 'break out;' statement. This will terminate the outer for loop labelled as 'out'. If only break statement would have used, it would have exited an enclosing while loop and continued with next iteration in for loop. Here, it does not execute outer for loop for i=6 onwards.

LblBlock.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 LblBlock.java

```
class LblBlock  // Use of labelled block to break the outer loop
{
    public static void main (String[] s)
    {
        int x=0;
        out: for (int i = 4; i <10; i++)
        {
            x=10;
            while (x < 100)
            {
                System.out.println ( "Inside while loop: i is "
                                + i + ", x is " + x);
                if ( i * x == 350 )
                    break out;
                x = x+20;
            } // end while
            System.out.println ( "\nOutside while loop: i is "
                                + i + ", x is " + x + "\n");
        } // end out for loop
        System.out.println ( "\nOutside for loop: x is " + x);
    } // end of main
} // end of class
```

```
>javac LblBlock.java
>Exit code: 0
>java -cp . LblBlock
Inside while loop: i is 4, x is 10
Inside while loop: i is 4, x is 30
Inside while loop: i is 4, x is 50
Inside while loop: i is 4, x is 70
Inside while loop: i is 4, x is 90

Outside while loop: i is 4, x is 110

Inside while loop: i is 5, x is 10
Inside while loop: i is 5, x is 30
Inside while loop: i is 5, x is 50
Inside while loop: i is 5, x is 70

Outside for loop: x is 70
>Exit code: 0
```

Figure 7.15 : Use of labelled loop and labelled break

Let us consider another example of labelled loop and labelled break in a program that stops execution when first odd non-prime number in the range 40 to 100 appears. See figure 7.16.

LblLoop.Java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 LblLoop.java

```
class LblLoop  // determine first non-prime odd number in the range 41 to 100
{
    public static void main (String[] s)
    {
        boolean prime=true;
        int i, last, n;

        // break the loops as soon as first non-prime odd number is found
        forLoop: for (n=41; n<100; n=n+2) // no need to try even numbers >2
        {
            if ( n < 4)
            {
                prime=true;
                System.out.println(n);
                continue;
            }

            prime=true;
            i=3; last = (int)Math.sqrt(n);
            do
            {
                if  (n%i == 0) // n is divisible by i
                {
                    prime=false;
                    break forLoop;      // break for loop labelled 'forLoop'
                }
                i = i + 2;      // no need to divide by even numbers
            } while (prime  && (i<last)) ;  //  end of do...while loop
            if (prime) System.out.println (n + " is prime");
        }  // end of for loop
        if (!prime) System.out.println (n + " is not prime");
    }  // end of main
}  // end of class
```

```
>javac LblLoop.java
>Exit code: 0
>java -cp . LblLoop
41 is prime
43 is prime
45 is not prime
>Exit code: 0
```

Figure 7.16 : Program to print prime number between 40 and 100

Summary

In this chapter, we have discussed about the basics of Java. There are eight basic data types supported in Java. Character in Java is a 2-byte Unicode character. Various types of literals can be used in Java. By default, numeric literal is assumed to be of the type double. Control structures like if, switch, for loop, while loop and do...while loop are very similar as in C language. Block can be specified by enclosing the statements in a pair of curly braces { }. Scope of variables is in the block where they are defined.

EXERCISE

1. What is the size of character data type in Java ?
2. Write the number of bytes taken by int and long data types in Java.
3. Explain if and switch statement available in Java.
4. Discuss about various repetitive structures available in Java.
5. How can one use labels for a block or loop in Java ?

6. Choose the most appropriate option from those given below :

(1) How many basic (primitive) data types are supported in Java ?

(a) 2 (b) 4 (c) 8 (d) 16

(2) What is the default data type of floating point literal ?

(a) int (b) long (c) float (d) double

(3) Which character set is used for char data type in Java ?

(a) Unicode (b) ASCII (c) EBCIDIC (d) All of these

(4) Which of the following is compiled error free ?

(a) for(;;){int i=7}; (b) while (1){int i=7};

(c) while (True){int i=7}; (d) All of these

(5) Which of the following is not allowed as first character in valid variable name ?

(a) Underscore (_) (b) Digit (c) Letter (d) Dollar ($)

(6) Which of the following is not a basic data type in Java ?

(a) char (b) long (c) byte (d) String

(7) What is the default value of boolean type data ?

(a) null (b) true (c) false (d) 0

(8) What will be the result of arithmetic expression 7/2 ?

(a) 3 (b) 3.5 (c) 1 (d) 0

(9) What will be the result of arithmetic expression -7%2 ?

(a) -3 (b) -1 (c) 1 (d) -3.5

(10) What will be the result of arithmetic expression -7.5%2 ?

(a) -3 (b) -1.5 (c) 1.5 (d) Error

LABORATORY EXERCISE

Write a Java program for the following :

1. During a sale at a store, a 10% discount is applied to purchases over Rs. 5000. Write a program that assigns any value to variable 'purchase' and then calculates the discounted price. Display purchase amount and discount offered.

2. Write a program that determines the price of a movie ticket based on customer's age and the time of the show (normal or matinee). The normal show and matinee show ticket price for adult is Rs. 100 and Rs. 50 respectively. Adults are those over 13 years. The children's ticket price is Rs. 60 and Rs. 40 for normal show and matinee show respectively. Declare variables of suitable data types for age and show time, assign the values to these variables and print age, show time and the ticket price.

3. Write a program to display the grade based on percentage of marks using switch statement.

4. A bank gives loan to customers at a simple interest of 12% per annum. Interest for the whole term is charged at the beginning. Compute the monthly installments considering the term of 36 months for loan amount of Rs. 10000, 20000, 30000, 40000, 50000, 60000, 70000, 80000, 90000 and 100000.

5. Write a program that prints square root of integer numbers starting from 5 till the square root is at 50 or less.

Classes and objects in Java 8

As learnt earlier object and class are fundamental parts of object-oriented programming. Java is an object-oriented Language. This chapter explains how to create a class and object in Java programming. After studying this chapter, we will be able to get a clear picture as to what are objects and what are classes in Java.

Introduction

A class contains both data (referred to as attributes), and program code (functions referred to as methods).

In the previous chapter, we wrote programs using a class that contained a single method named 'main'. When the class was executed, the main method was called, and the application ran as a normal program. We have not used any data members in those examples.

While it is possible to use only a single class in a Java project, this is not a good practice for large applications. When designing software, we should divide the entire application into simpler components that perform logically related tasks. For each such component or module we may create a class.

Let us understand the concept of class and object in Java programming through an example of 'Room'. A room of a house, hostel, hotel or school possess common characteristics. Each room can be uniquely identified by its properties like length, width, height, number of windows, number of doors and direction. For simplicity, let us consider here length, width, height and number of windows.

We will now write a Java code as shown in code listing 8.1 to create a class named 'Room' with the attributes length, width, height, nWindows and three methods. The first method will be used to assign values to attributes. The second method will calculate the area, while the third will display its attributes. Thereafter, we will write code to use this class.

```
/* Class Room */
class Room
{
        float length, width, height;
        byte nWindows;

        void setAttr (float l, float w, float h, byte n)
        {
                        length = l; width = w; height = h;
                        nWindows = n;
        }  // end setAttr () method

      double area ( )  // area =  length * width
```

```
        {
                return (length * width);
        } // end area() method

      void display ( )
        {
                System.out.println ("\nLength: " + length);
                System.out.println ("Width: " + width);
                System.out.println ("Height: " + height);
                System.out.println ("Number of Windows: " + nWindows);
        } // end display() method
}  // end Room class

/* using Room class to create objects and run application */
class RoomDemo
{
      public static void main (String args[])
        {
                // Create a room object, assigned default values to attributes
                Room r1;  // reference variable with null value by default
                r1 = new Room();
                // both declare and create in one statement
                 Room r2 = new Room();

            // Display two room objects with initial default values
                r1.display();
                r2.display();

                // Assign values of attributes of objects
                r1.setAttr (18, 12.5f, 10, (byte)2);
                r2.setAttr (14, 11, 10, (byte)1);

                // Display updated contents
                r1.display();
                r2.display();

                // Display area
                System.out.println ("\nArea of room with length " + r1.length
                   + " width " + r1.width +  " is " + r1.area());
                System.out.println ("\nArea of room with length " + r2.length
                   + " width " + r2.width +  " is " + r2.area());
        }  // end main()
}  // end RoomDemo
```

Code Listing 8.1 : Creating and using class and objects

Here, first of all, code is written to create a class named 'Room'. Thereafter, code is written to create objects of class 'Room' and use its methods. To do so, we have created another class named 'RoomDemo' containing main() method. In a source file, these classes may appear in any order.

Here, we have separated creating 'Room' class performing logically related tasks and using this class in application class 'RoomDemo'.

When a program contains two or more classes, only one class can contain the main() method. Figure 8.1 shows the execution of the code in SciTE editor.

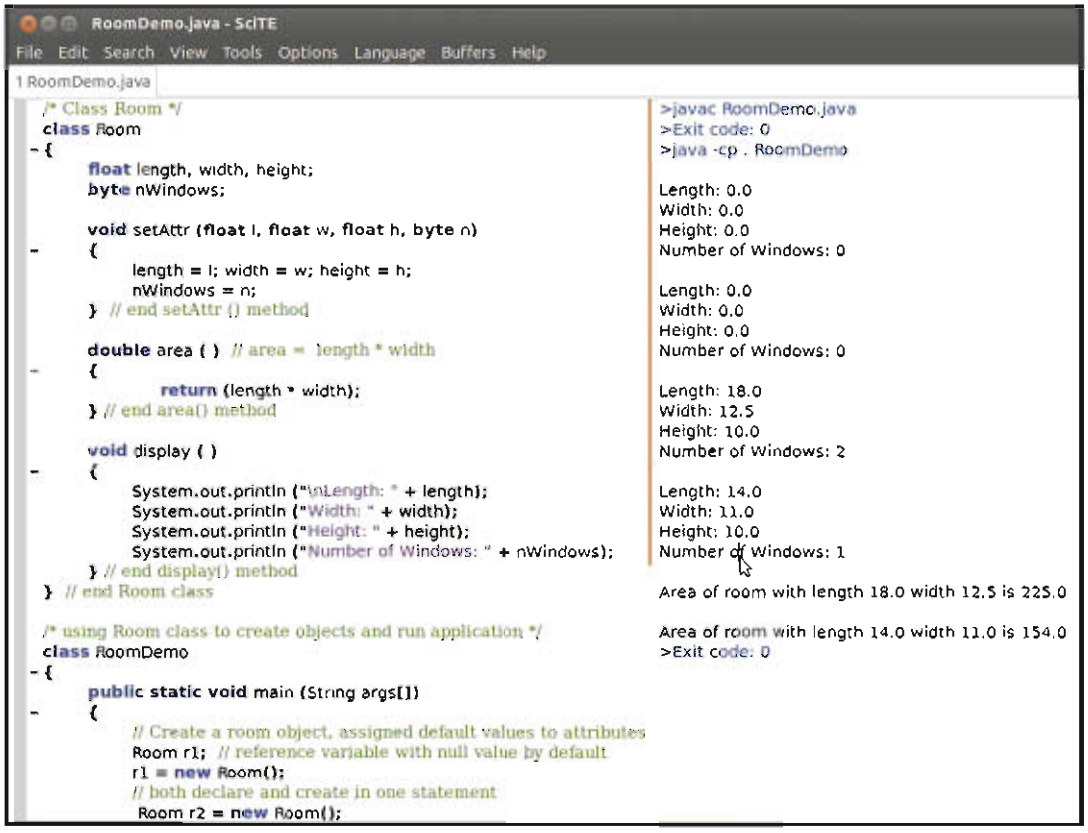


Figure 8.1 : Java Program to demonstrate use of multiple classes

In the given example of RoomDemo application as shown in figure 8.1, following operations are performed.

- Two objects r1 and r2 of class Room are created first. They are initialized with default values (Numeric values are zero by default).
- Contents of these two objects are displayed using display method.
- Invoking setAttr() method with numeric literals as parameters, attributes of room objects are modified for both the objects r1 and r2.
- The updated contents of two objects are displayed.

Finally area of both Room objects is displayed. Here area() method is invoked to get area.

Class in Java

We have already written one program that uses class. Let us now learn the syntax and other details of how to create class, objects and use them in an application.

A class is a template for multiple objects with similar features. Classes embody all the features of a particular set of objects. For example, class 'Room' is a template for all rooms with their common properties.

In Java, a class is defined using **class** keyword as follows :

```
class <ClassName >
{
        <Variables>
        <Methods>
}
```

Every class we write in Java is generally made up of two components: attributes and behaviour. Attributes are defined by variables in a class. Behaviour is defined by methods in a class. Methods are used to access or modify attributes.

In code listing 8.1, we have defined a class named 'Room' with attributes defined by variables named length, width, height and nWindows; and behaviour by methods named setAttr(), display() and area(). Another class is created just to create an application using 'Room' class.

Creating objects

Creating an object from a class requires the following steps :

- **Declaration :** A variable (reference variable) name of type class is declared with syntax <class name> <variable name>
- **Instantiation :** Keyword **new** is used to create the object by allocating memory
- **Initialization :** Constructor (a special type of method) is called to initialize the newly created object

In Java, a class is a type, similar to the built-in types such as int and boolean. So, a class name can be used to specify the type of a variable in a declaration statement, the type of a formal parameter, or the return type of a function.

To declare an object of class 'Room', use following statement :

```
Room r1;
```

Declaring a variable does not create an object. This is an important point to be remembered. In Java, no variable can ever store an object. A variable declared using class type can only store a reference to an object. So variables of class type are also referred to as reference variables. Here r1 is a reference variable.

Next step is to create an object. Using new keyword, we can create an object. Operator new allocates the memory for an object and returns the address of the object for later use. Reference is the address of the memory location where the object is stored. In fact, there is a special portion of memory called the heap where the objects live.

When an object is created, in addition to allocating memory, a special method called 'constructor' is executed to perform initial task. We will learn about constructors later in this chapter.

Let us create an object of type Room and assign its address to variable r1 as follows :

```
r1 = new Room();
```

Here, parentheses are important; don't leave them off. With empty parentheses without arguments, a default constructor is called. It initializes the attributes (variables) of the object using default values. The parentheses can contain arguments that determine the initial values of variables. This is possible by using user-defined constructor.

When statement "r1=new Room();" is executed, remember that object is not stored in variable r1. Variable r1 contains only address of an object.

Both the above steps used to declare and create an object can be combined into a single statement as follows :

```
Room r2 = new Room();
```

Variable r2 contains a reference or address of memory location where a new object is created.

It is also to be noted that the class determines only the types of the variables. The actual data is contained inside the individual objects and not in the class. Thus, every object has its own set of data. In code listing 8.1, we have created two objects r1 and r2 using new operator. These objects are allocated different memory space to hold their data values as seen in figure 8.2 .

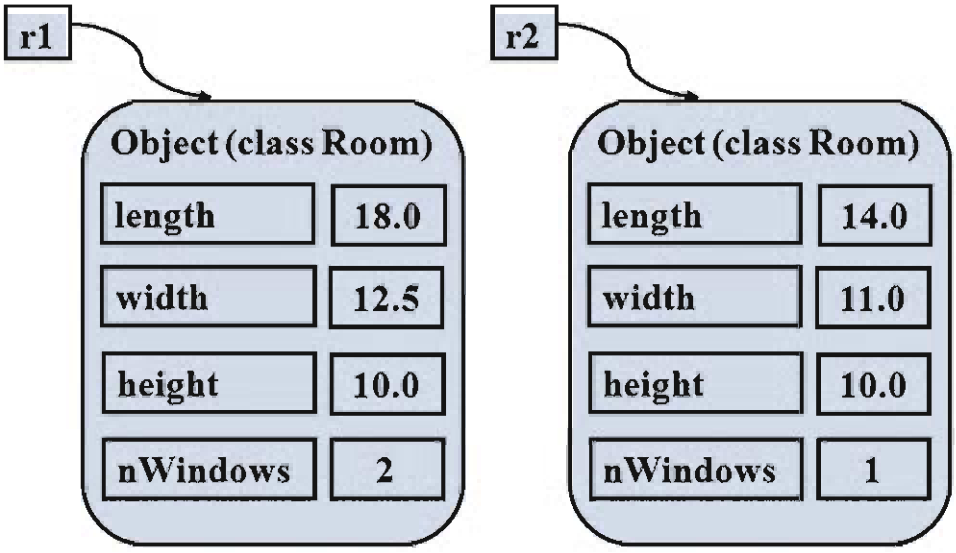


Figure 8.2 : Two instances of class Room

In Java, when objects are no more needed, the memory is claimed back for reuse. Java has a garbage collector that looks for unused objects and reclaims the memory that those objects are using. We do not need to do any explicit freeing of memory.

In object-oriented programming (OOP) languages, creating an object is also called object instantiation. An instance for an object is created by allocating memory to store the data for that object. An object that belongs to a class is said to be an instance of that class.

Thus, an instance of a class is another word for an actual object. Class is an abstract representation of an object; whereas an instance is its concrete representation. In fact, the terms instance and object are often used interchangeably in OOP language.

Each instance of a class can hold different values for its attributes in variables declared in a class. Such variables are referred to as instance variables. Instance variables are created at the time of creating an object and stay throughout the life of the object.

Instance variables define the attributes of an object. The class defines the kind of attribute. Each instance stores its own value for that attribute.

To define an object's behaviour, we create methods. In Java, methods can be defined inside a class only. These methods can be invoked using the objects to access or modify the instance variables. Such methods are known as instance methods.

Instance methods are used to define behaviour of an object. Invoking a method is to ask the object to perform some task.

In code listing 8.1, we have defined a class named Room with instance variables length, width, height and nWindows; and instance methods setAttr(), display() and area().

To summarize,

- Objects are created with the new keyword.
- The new keyword returns a reference to an object that represents an instance of the class.
- All instances of class are allocated memory in data structure called heap.
- Each object instance has its own set of data.

Accessing instance variables and calling instance methods

Instance variables and instance methods are accessed via objects. They can be referred by using dot (.) operator as follows :

<object refernce>.<instance variable or method>

For example, we can refer to length of room r1 using r1.length in a program and invoke method r1.display() that displays attribute values of room r1 as can be seen in code listing 8.1. Note that here dot (.) is an operator and associativity of dot operator is from left to right.

Remember that data should be protected from such direct access from anywhere in the program. Such protection is possible with the use of access modifiers that we will see later.

When the instance variables are referred within the methods of the same class, there is no need to use dot (.) operator. For example, in code listing 8.1, in method display, instance variables are accessed without using dot operator. This is possible because method is invoked using reference variable r1 and thus referred instance variables are considered to be of the corresponding object stored at r1.

As studied, when we only declare an object using class, object is not created. In this case, reference variable does not refer to any object and its initial value is null by default. Use of such null reference

or null pointer is illegal and may raise an exception. So, referring instance variable or invoking method with null reference will give an error. Try following code:

```
Room r1; // null value assigned to reference variable by default

System.out.println (r1.length); // illegal

r1.display();  // illegal
```

Class variables and class methods

As discussed earlier, when an object is created using new keyword, memory is allocated from heap area to store the value of its attributes. Thus every object has its own instance variables occupying different space in memory.

Now suppose we want to have a total number of windows of all Room objects created so far. To store this value, it requires only one variable per class. It is meaningless to keep this variable as an attribute of each object. Actually, it is not an attribute of an object; it is an attribute of a class.

Thus, when there is a need to have some variable that is shared by all object instances of the same class, the variable should be allocated the memory only once per class. It means that variable belongs to a class and not to an object. Such variables can be declared within a class using static keyword before data type and these static variables are called class variables.

Values of instance variables are stored in instance (memory allocated for the object); whereas values of class variables are stored in class itself.

Let us declare class variable totWindows by adding following statement in class with instance variables:

```
static int totWindows;
```

Static variables can be accessed without creating an instance of a class. For example, if we try to display the value of totWindows without creating any object of the class 'Room', it will display 0.

Just like class variable, class method can be defined using static keyword in front of the method definition. Try following method in 'Room' class to display total windows.

```
static void displayTotalWindows()
{
    System.out.println("Total Windows: " +totWindows);
}
```

Class variables and class methods can be accessed outside the class using

<classname>.< class variable/method name>

For example : Room.totWindows, Room.displayTotalWindows()

Note that class members (variables and methods) can be referred without class name by the methods of the same class as shown in the code listing of figure 8.3.

```
RoomDemoStatic.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 RoomDemoStatic.java

/* Class Room */
class Room
{
        float length, width, height;
        byte nWindows;
        static int totWindows;  // class variable

        void setAttr (float l, float w, float h, byte n)
        {
                setWindows(n);
                length = l; width = w; height = h;
        }  // end setAttr () method

        void setWindows ( byte n)
        {
                totWindows = totWindows - nWindows + n;
                nWindows = n;
        }
        double area ( )  // area =  length * width
        {     return (length * width);      } // end area() method

        void display ( )
        {
                System.out.println ("\nLength: " + length + "\nWidth: " + width);
                System.out.println ("Height: " + height);
                System.out.println ("Number of Windows: " + nWindows);
        } // end display() method
}  // end Room class

/* using Room class to create objects and run application */
class RoomDemoStatic
{
        public static void main (String args[])
        {
                Room r1 = new Room();   Room r2 = new Room();

                r1.setAttr (18, 12.5f, 10, (byte)2);   r1.display();
                r2.setAttr (14, 11, 10, (byte)1);   r2.display();
                System.out.println("\nRoom2 Number of windows modified from 1 to 2");
                r2.setWindows((byte)2);
                System.out.println ("\nTotal number of Windows: " + Room.totWindows);
        }  // end main()
}  // end RoomDemo
```

Figure 8.3 : Use of class variable totWindows

The output of the code shown in figure 8.3 is given in figure 8.4.

```
RoomDemoStatic.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 RoomDemoStatic.java

>javac RoomDemoStatic.java
>Exit code: 0
>java -cp . RoomDemoStatic

Length: 18.0
Width: 12.5
Height: 10.0
Number of Windows: 2

Length: 14.0
Width: 11.0
Height: 10.0
Number of Windows: 1

Room2 Number of windows modified from 1 to 2

Total number of Windows: 4
>Exit code: 0
```

Figure 8.4 : Output of code listing given in figure 8.3

Here, we have written an additional instance method setWindows() that updates total windows by subtracting the old number of windows and adding new number of windows in a room. Method setWindows() is defined in the same class. So it can access class variable totWindows without class name. In main() method (defined in RoomDemoStatic class) that is defined outside 'Room' class, class variable totWindows is to be accessed as Room.totWindows using class name.

Class methods are global to the class itself and available to any other classes or objects. Therefore, class methods can be used anywhere regardless of whether an instance of the class exists or not.

Now, when should we use class methods ? The methods that operate on a particular object, or affect that object, should be defined as instance methods. Methods that provide some general utility but do not directly affect an instance of the class are better declared as class methods.

For example, consider a function that determines whether a given number is prime or not. This function should be defined as class method. The code listing and its output is shown in figure 8.5.

primeClassMethod.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 primeClassMethod.java

```
// Static method (class method)
// isPrime (int) returns true if given integer is prime
class Prime
{
    static boolean isPrime (int n)
    {   //n>1 is prime if it is not divisible by any number except 1 and itself
        int i, last;

        If (n <= 1) return false;
        if (n < 4) return true;
        //if (n%2==0) return false;   // divisible by 2, so not prime

        last = (int) Math.sqrt(n);
        i=3;
        do
        {
            if (n%i == 0) return false; // n is divisible by i
            i = i + 2;      // no need to divide by even numbers
        } while (i<last) ;  // end of do...while loop

        return true;
    } //end of method isPrime
} // end class primeFunc

class primeClassMethod
{
    public static void main (String[] s)
    {
        int i, n;

        System.out.println ("Prime numbers between 3 and 100:");
        for (n=3; n<100; n=n+2)
        {
            if (Prime.isPrime(n)) System.out.println(n);
        }
    } // end of main
} // end classs ClassMethodDemo
```

```
>javac primeClassMethod.java
>Exit code: 0
>java -cp . primeClassMethod
Prime numbers between 3 and 100:
3
5
7
11
13
17
19
23
25
29
31
35
37
41
43
47
49
53
59
61
67
71
73
79
83
89
97
>Exit code: 0
```

Figure 8.5 : Using static method to determine whether a given integer is prime or not

The designers of Java have already provided a large number of built-in classes with such class methods. In chapter 7 we have used static method 'sqrt()' without creating any object of class Math.

It is to be noted here that the main() method that we have defined till now is also a class method. This is the reason we use keyword static while defining main() method.

As seen here, the static and the non-static portions of a class serve different purposes. The static definitions in the source code specify the things that are part of the class itself; whereas the non-static definitions in the source code specify the things that will become part of every instance object belonging to a class.

Points to Remember :

- Class variables and class methods can be accessed using a class name or reference variable. To increase readability, it is advised to access with class name.
- Class variables and class methods can be accessed from instance methods also.
- Instance variables and instance methods can't be accessed from class methods, as class methods do not belong to any object.

Classification of variables declared in a class

- **Local variables :** Variables defined inside methods or blocks are called local variables. Formal parameters of the methods are also local variables. They are created when the method or block is started and destroyed when the method or block has completed. Local variables are not initialized by default values.
- **Instance variables :** Instance variables are variables defined within a class but outside any method. These variables are allocated memory from heap area when an object is instantiated (created). Instance variables are initialized by default values.
- **Class variables :** Class variables are variables defined with in a class, outside any method, with the static keyword. These variables are allocated memory only once per class and is shared by all its objects. Class variables are initialized with default values.

Polymorphism (Method Overloading)

We have seen first two steps of creating objects: Declaration and Instantiation. The third step is Initialization. It requires the use of constructors. Before we learn about constructors, let us see the way to implement polymorphism in Java.

The word polymorphism means "many forms"; different forms of methods with same name. In Java, we can have different methods that have the same name but a different signature. This is called 'method overloading'. The method's signature is a combination of the method name, the type of return value (object or base type), a list of parameters.

For example, to find maximum of two integers, maximum of three integers, maximum of three double precision real numbers and so on one requires to perform similar task but on different set of numbers. In such scenario, Java facilitates to create methods with same name but different parameters. Finding maximum does not need creation of any object, so it can be defined as static class method. For example :

```
static int max(int x, int y) {...}

static int max(int x, int y, int z) {...}

static double max(double x, double y, double z) {...}
```

You may try to define above forms of max methods and use in the application after looking the example of code listing given in figure 8.6. Here it uses polymorphism to print a line as per specified parameters. Call printline() without any parameter prints 40 times '=' character, printline(int n) prints n times '#' character, whereas printline(int n, char ch) prints n times specified character ch.

polyDemo.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 polyDemo.java

```
// polymorphism: method printline
class PrintLine
{
    static void printline()
    {
        for (int i=0; i<40; i++)
            System.out.print('=');
        System.out.println();
    }

    static void printline(int n)
    {
        for (int i=0; i<n; i++)
            System.out.print('#');
        System.out.println();
    }

    static void printline(char ch, int n)
    {
        for (int i=0; i<n; i++)
            System.out.print(ch);
        System.out.println();
    }
} // end class PrintLine

public class polyDemo
{
    public static void main(String[] s)
    {
        PrintLine.printline();
        PrintLine.printline(30);
        PrintLine.printline('+',20);
    } // end main
} // end PolyDemo class
```

```
>javac polyDemo.java
>Exit code: 0
>java -cp . polyDemo
========================================
##############################
++++++++++++++++++++
>Exit code: 0
```

Figure 8.6 : Method Overloading

Constructors

Constructor is a special kind of method that is invoked when a new object is created. Constructor can perform any action; but it is mainly designed to perform initializing actions.

Till now, we have used class without user-defined constructors. Every class is having its default constructor; sometimes referred to as no-argument constructor. Default constructor does not take any argument. It initializes the attributes of a newly created object using default values based on their data types.

Constructor differs from general methods in the following ways :

- Constructor must have the same name as class name.
- Constructor does not have return type.

- Constructor is invoked implicitly only when an object is constructed using new operator.
- Constructor cannot be invoked explicitly elsewhere in the program.

Like other methods, constructor can also be overloaded. It is possible with varying list of parameters. For example, we may write our own constructors to initialize the attributes of 'Room' object with different parameters as follows:

Room(float l, float w, float h, byte n) : To initialize length to l, width to w, height to h and nWindows to n

Room(float l, float w) : To initialize length to l, width to w, height to 10, nWindows to 1

Program seen in figure 8.7 shows the use of constructors.

RoomConstructorDemoA.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 RoomConstructorDemoA.java

```
/* Using Constructors */
class Room
{
    float length, width, height;
    byte nWindows;
    static int totWindows;  // class variable

    Room (float l, float w, float h, byte n)
    {
        length = l; width = w; height = h;
        nWindows = n;   totWindows+=n;
    }
    Room (float l, float w)
    {
        length = l; width = w; height = 10;
        nWindows = 1;     totWindows++;
    }

    double area ( ) // area =  length * width
    {   return (length * width);     } // end area() method
    void display ( )
    {
        System.out.println ("\nLength: " + length + "\nWidth: " + width);
        System.out.println ("Height: " + height);
        System.out.println ( "Windows: " + nWindows);
    } // end display() method
} // end Room class

class RoomConstructorDemoA  /* Application using Room */
{
    public static void main (String args[])
    {
        Room r1 = new Room(16, 12.5f);
        Room r2 = new Room(20, 14, 12, (byte)2);
        r1.display();  r2.display();
        System.out.println ("\nTotal number of Windows: " + Room.totWindows);
    } // end main()
} // end RoomConstructorDemoA
```

```
>javac RoomConstructorDemoA.java
>Exit code: 0
>java -cp . RoomConstructorDemoA

Length: 16.0
Width: 12.5
Height: 10.0
Windows: 1

Length: 20.0
Width: 14.0
Height: 12.0
Windows: 2

Total number of Windows: 3
>Exit code: 0
```

Figure 8.7 : Using constructors

In absence of user-defined constructors in a class, objects are constructed using default no-argument constructor. It initializes the attributes using default values.

In presence of user-defined constructors in a class, default constructor is no more available. An attempt to create an object using constructor without arguments, compiler returns an error. To solve this problem, we need to provide a user-defined no-argument constructor as follows: <classname> ( ) { };

In code listing given in figure 8.7, try to create a Room object in main method as follows and observe an error occurred during compilation: Room r3 = new Room();

To solve this error, add user-defined no-argument constructor 'Room () {};' and execute. See the successful execution as given in figure 8.8.

RoomConstructorDemoB.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 RoomConstructorDemoB.java

```
/* Using Constructors */
class Room
{
    float length, width, height;
    byte nWindows;
    static int totWindows;  // class variable

    Room ( ) { }; // user-defined no-argument constructor
    Room (float l, float w, float h, byte n)
    {
        length = l; width = w; height = h;
        nWindows = n;   totWindows+=n;
    }
    Room (float l, float w)
    {
        length = l; width = w; height = 10;
        nWindows = 1;     totWindows++;
    }

    double area ( )  // area =  length * width
    {    return (length * width);     } // end area() method
    void display ( )
    {
        System.out.println ("\nLength: " + length + "\nWidth: " + width);
        System.out.println ("Height: " + height);
        System.out.println ( "Windows: " + nWindows);
    } // end display() method
} // end Room class

class RoomConstructorDemoB  /* Application using Room */
{
    public static void main (String args[])
    {
        Room r1 = new Room( );
        Room r2 = new Room(20, 14, 12, (byte)2);
        r1.display();   r2.display();
        System.out.println ("\nTotal number of Windows: " + Room.totWindows);
    } // end main()
} // end RoomConstructorDemoB
```

```
>javac RoomConstructorDemoB.java
>Exit code: 0
>java -cp . RoomConstructorDemoB

Length: 0.0
Width: 0.0
Height: 0.0
Windows: 0

Length: 20.0
Width: 14.0
Height: 12.0
Windows: 2

Total number of Windows: 2
>Exit code: 0
```

Figure 8.8 : Using user-defined no-argument constructor

Visibility Modifiers for Access Control

Access control is about controlling visibility. So, access modifiers are also known as visibility modifiers. If a method or variable is visible to another class, then only it can be referred in another class. To protect a method or variable from such references, we use the four levels of visibility to provide necessary protection.

The Four P's of Protection are public, package (default protection), protected, and private. Access modifiers public, protected and private are used before the type of variable or method. When no modifier is used, it is the default one having visibility only within a package that contains the class.

Package is used to organize classes. To do so, package statement should be added as the first non-comment or non-blank line in the source file. When a file does not have package statement, the classes defined in the file are placed in default package. Package statement has following syntax: package <packageName>;

In this book, we will use default package. In all our programs, we have used the default access modifier till now. Table 8.1 shows the type of access modifier and its visibility.

		Type		
Access Modifier	public	(default: package)	protected	private
Visibility	widest	→ → →	→ →	narrowest

Table 8.1 : Type of access modifier and its visibility

We will now see examples of default and private modifier.

public

Any method or variable is visible to the class in which it is defined. If we want to make it visible to all the classes outside this class, declare the method or variable to have public access. This is the widest possible access. It provides visibility to classes defined in other package also. To provide public access, use access modifier public before type of variable or method. For example,

```
public float length

public double area ( )
```

Note that public variables and methods are visible anywhere and thus can be accessed from other source files and packages also.

We have used public keyword with main() method to make it available to everyone.

```
public static void main(String[] args) { ... }
```

Package (without any modifier)

This is the next level of access that has no precise name. It is indicated by the lack of any access modifier keyword in a declaration. This is the default level of protection. The scope is narrower than public variables. The variable or method can be accessed from anywhere in the package that contains the class, but not from outside that package. Note that a source file without package statement is considered as a package by default. So, in our programs till now, it is as good as public.

Refer the program shown in figure 8.9. Here 'Rectangle' class has two attributes: length and width. It also has various methods. We have not used any modifier keyword, so they have package protection by default. Due to this reason, they are directly accessible in another class 'RectangleDemo' defined in the same source file (default package).

RectangleDemo.Java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 RectangleDemo.Java

```
class Rectangle
{
    double length, width;

    void setAttributes(double x, double y)
    {
        length = x;  width = y;
    }

    double area ()
    {
        return length * width;
    }

    void display()
    {
        System.out.println ("Rectangle with length = " + length
                            + " width = " + width );
    }
}// end class Rectangle

class RectangleDemo
{
    public static void main (String[] s)
    {
        Rectangle rect1;
        rect1 = new Rectangle();
        Rectangle rect2 = new Rectangle();

        rect1.setAttributes (10.5, 20);
        rect1.display();
        System.out.println ("Area of rectangle is " + rect1.area());
        rect2.setAttributes(10,15);
        System.out.println ("Area of rectangle with length " +
                            rect2.length + ", width = " + rect2.width
                            + " is " + rect2.area());
    } //end main()
} // end class RectangleDemo
```

```
>javac RectangleDemo.java
>Exit code: 0
>java -cp . RectangleDemo
Rectangle with length = 10.5 width = 20.0
Area of rectangle is 210.0
Area of rectangle with length 10.0, width = 15.0 is 150.0
>Exit code: 0
```

Figure 8.9 : Default visibility modifier, available everywhere in a package

protected

This level of protection is used to allow the access only to subclasses or to share with the methods declared as "friend". Thus the visibility is narrower than previous two levels; but wider than full privacy provided by fourth level "private".

Use of protected protection will become more relevant when we use inheritance concept of object-oriented programming.

private

Highest level of protection can be achieved by using "private" protection level. This provides the narrowest visibility. The private methods and variables are directly accessible only by the methods defined within a class. They cannot be seen by any other class.

This may seem extremely restrictive, but it is, in fact, a commonly used level of protection. It provides data encapsulation; hiding data from the world's sight and limiting its manipulation. Anything that shouldn't be directly shared with anyone including its subclass is private.

The best way to provide data encapsulation is to make as much data as private as possible. It separates design from implementation, minimizes the amount of information one class needs to know about another to get its job done.

Let us modify code listing given in figure 8.9 and declare length and width as private. As private members are directly available only within the same class, it does not give any error when accessed in area() or display() method. When they are accesses in main method() of RectangleDemo class, it will show an error.

The modified code is shown in figure 8.10. Here, we have used constructors in addition to private instance variables. Note the error in the output pane that says 'length has private access in Rectangle'. This means that it is not accessible in class 'RectangleVisibility1'.

```
RectangleVisibility1.java * SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 RectangleVisibility1.java *

class Rectangle
{
    private double length, width;

    Rectangle (double x, double y)
    {
        length = x;  width = y;
    }

    Rectangle ( ) { };

    double area () {  return length * width; }

    void display()
    {
        System.out.println ("Rectangle with length = " + length
                            + " width = " + width );
    }
}// end class

class RectangleVisibility1
{
    public static void main (String[] s)
    {
        Rectangle rect1;
        rect1 = new Rectangle();
        Rectangle rect2 = new Rectangle(10, 15);

        rect1.display(); rect2.display();
        System.out.println ("Area of rectangle with length " +
                            rect2.length + ", width = " + rect2.width
                            + " is " + rect2.area());
    } //end main()
}// end class

>javac RectangleVisibility1.java
RectangleVisibility1.java:31: length has private access in Rectangle
                              rect2.length + ", width = " + rect2.width
                              ^
RectangleVisibility1.java:31: width has private access in Rectangle
                              rect2.length + ", width = " + rect2.width
                                                          ^
2 errors
>Exit code: 1
```

Figure 8.10 : Error while accessing private instance variable from another class

The problem now is how to access private variables from another class? It can be made available indirectly by the methods that are accessible by another class. See code listing given in figure 8.11.

Here we have added two methods getLength() and getWidth(). As no modifier is used, they have package visibility and so directly available in another class, 'VisibilityPrivateB' here. Through these methods, we can get the values of private 'length' and 'width' data fields (instance variables). See the use of getLength() and getWidth() method calls in a last output statement in main() method of class 'VisibilityPrivateB'.

```
VisibilityPrivateB.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 VisibilityPrivateB.java

class Rectangle
{
        private double length, width;

        Rectangle (double x, double y)
        {
                length = x;  width = y;
        }
        Rectangle ( ) { };

        double area () {  return length * width;   }
        void display()
        {
                System.out.println ("Rectangle with length = " + length
                                + " width = " + width );
        }
        double getLength() { return length; }
        double getWidth() { return width; }
}// end class

class VisibilityPrivateB
{
        public static void main (String[] s)
        {
                Rectangle rect1;
                rect1 = new Rectangle();
                Rectangle rect2 = new Rectangle(10, 15);

                rect1.display(); rect2.display();
                System.out.println ("Area of rectangle with length " +
                        rect2.getLength() + ", width = " + rect2.getWidth();
                                + " is " + rect2.area());
        }  //end main()
}// end class
```

```
>javac VisibilityPrivateB.java
>Exit code: 0
>java -cp . VisibilityPrivateB
Rectangle with length = 0.0 width = 0.0
Rectangle with length = 10.0 width = 15.0
Area of rectangle with length 10.0, width = 15.0 is 150.0
>Exit code: 0
```

Figure 8.11 : Accessing private variables through public or package methods

Accessor and Mutator Methods

When we restrict access to data by declaring them as private, our purpose is to protect them from getting directly accessed or modified by methods of other class. If we want to allow such data to be used by others, then we write "accessor" methods. If we want to allow such data to be modified by others, then we write "mutator" methods.

Conventionally, naming of accessor and mutator methods is to capitalize the first letter of variable name and then prepend the variable name with the prefixes get and set respectively. Due to this convention, accessor methods are also known as "getter" and mutator methods as "setter".

Observe that in code listing shown in figure 8.11 we have used "getter" methods getLength() and getWidth() methods. If we change the type of variable 'length', it will be hidden from other users. It affects only the implementation of accessor method 'getLength()'.

If we want to allow other methods to read only the data value, we should use "getter" methods.

If we want to allow other methods to modify the data value, we should use "setter" methods. For example, 'setLength()' method can be defined to set the value of 'length' attribute using passed argument as follows:

```
void setLength(float l) { length = l; }
```

Use of accessor and mutator methods will prevent the variables from getting directly accessed and modified by other users of the class. It may seem a little difficult to get used to this as we need to write get and set method for each and every instance variable as per need. But, this minor inconvenience will reward us with an ease of reusability and maintenance.

Passing object as a parameter in a method

Just like variables of primitive data types and available built-in data types, objects can also be passed as parameters in a method.

For example, we want to determine whether the area of invoking rectangle object is larger than another rectangle or not. For this, we may write a method where another rectangle object is passed as an argument or parameter. Let us add method 'isLarge' in 'Rectangle' class as shown in figure 8.12 and use it in main() method.

```
boolean isLarge (Rectangle rect)

{ if (area() > rect.area() ) return true; else return false; }
```

See the method call in if statement in the main() method as shown in figure 8.12: rect1.isLarge(rect2). Here 'rect1' is a calling or invoking object and 'rect2' is an object passed as parameter. In the isLarge() method, area() refers to area of calling object 'rect1' and rect.area() refers to area of an object passed as argument.

ObjectParameter.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 ObjectParameter.java

```
class Rectangle
{
    private double length, width;

    Rectangle (double x, double y)
        { length = x; width = y; }
    Rectangle ( ) { };

    double area () { return length * width;  }
    void display()
    {
        System.out.println ("Rectangle with length = " + length
                            + " width = " + width );
    }
    double getLength() { return length; }
    double getWidth() { return width; }
    boolean isLarge (Rectangle rect)
    {
        if (area() > rect.area()) return true;
            else return false;
    }
}// end class

class ObjectParameter
{
    public static void main (String[] s)
    {
        Rectangle rect1 = new Rectangle(8, 20);
        Rectangle rect2 = new Rectangle(10, 15);

        rect1.display();
        System.out.println ("Area of rectangle 1 is "+ rect1.area() + "\n");
        rect2.display();
        System.out.println ("Area of rectangle 2 is "+ rect2.area() + "\n");
        if (rect1.isLarge(rect2))
            System.out.println ("Area of rectangle 1 is larger than " +
                                "Area of rectangle 2");
    } //end main()
}// end class ObjectParameter
```

```
>javac ObjectParameter.java
>Exit code: 0
>java -cp . ObjectParameter
Rectangle with length = 8.0 width = 20.0
Area of rectangle 1 is 160.0

Rectangle with length = 10.0 width = 15.0
Area of rectangle 2 is 150.0

Area of rectangle 1 is larger than Area of rectangle 2
>Exit code: 0
```

Figure 8.12 : Passing an object as a parameter

Remember that parameters of primitive types are passed by value. The values of actual parameters are copied to formal parameters and then function is executed. Changes made to formal parameters are not affecting actual parameters.

It is to be noted that object parameters are passed by reference. So, whatever modifications are performed to the object inside the method, the original object is affected as well. Here, an address (and not value) of the actual parameter is copied to formal parameter.

Inheritance

Object-oriented programming languages provide reusability feature using inheritance. Inheritance allows us to build new class with added capabilities by extending existing class.

Inheritance models 'is-a' relationship between two classes. For example, classroom is a room, student is a person. Here, room and person are called parent class; classroom and student are called child classes. Parent class is also referred to as superclass or base class. In the same way; child class is also referred to as subclass, derived class or extended class.

Whenever two classes have 'is-a' relationship, we use inheritance. Common features are kept in superclass. A subclass inherits all instance variables and methods from superclass and it may have its own added variables and methods. Note that constructors are not inherited in subclass. For example, like room, classroom also has variables length, width, height, number of windows. In addition, it has number of benches and capacity of each bench to accommodate students. Similarly, subclass inherits all methods of superclass and it may have additional methods. Here, subclass Classroom has additional methods: show(), display(), getSeats() and its constructor methods.

Figure 8.13 shows a class diagram that has a class named 'Classroom' derived from its parent class named 'Room'. See that the arrow points from subclass towards superclass. In subclass, only additional attributes and methods are to be shown.

Note that a subclass is not a subset of superclass. In fact, subclass usually contains more information and methods than its superclass.

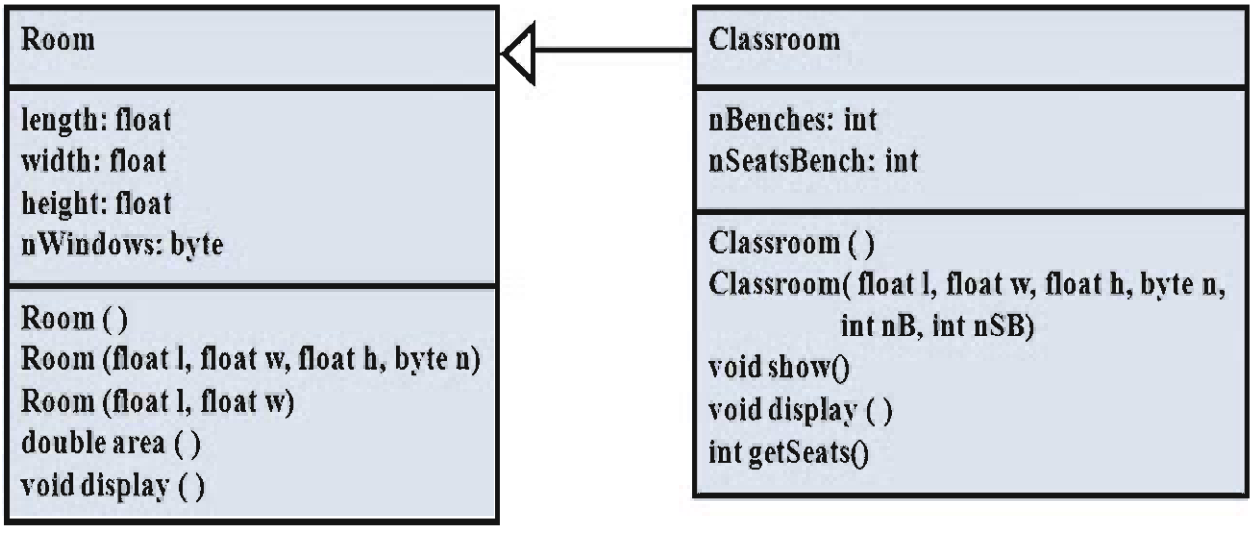


Figure 8.13 : Inheritance Class Diagram

When an object of subclass is instantiated, memory is allocated for all its attributes including inherited ones. Figure 8.14 shows the instances of superclass Room and subclass Classroom.

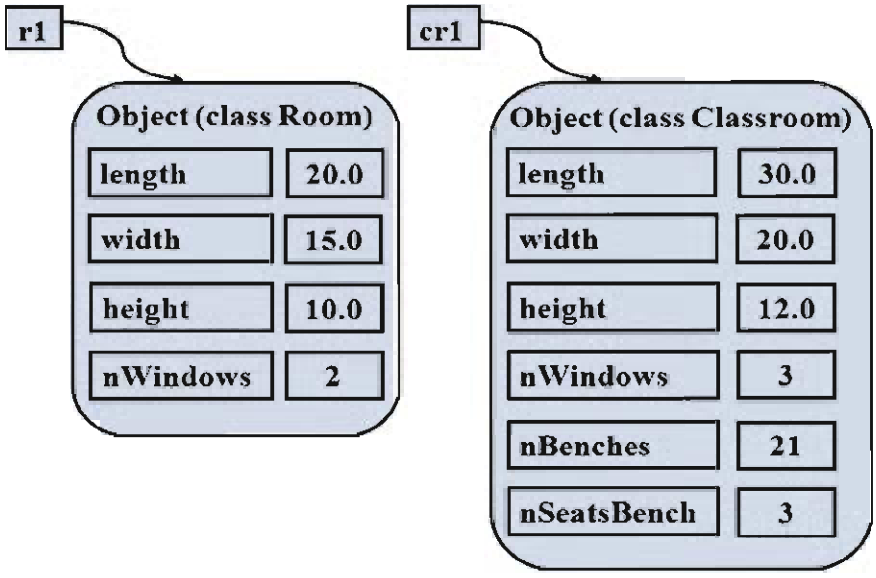


Figure 8.14 : Instances of superclass and subclass

In Java, to create a subclass, keyword 'extends' is used in the class definition. Let us create a subclass 'Classroom' using existing class 'Room' as shown in code listing 8.2.

```
class Room
{
        float length, width, height;
        byte nWindows;
        static int totWindows;  // class variable

        Room ( ) { }; // user-defined no-argument constructor
        Room (float l, float w, float h, byte n)
        {
                length = l; width = w; height = h;
                nWindows = n;   totWindows+=n;
        }
        Room (float l, float w)
        {
                length = l; width = w; height = 10;
                nWindows = 1;       totWindows++;
        }

    double area ( )  // area =  length * width
        {       return (length * width);        } // end area() method

          void display ( )
        {
                System.out.println ("\nLength: " + length + "\nWidth: " + width);
                System.out.println ("Height: " + height);
                System.out.println ( "Windows: " + nWindows);
        } // end display() method
}  // end Room class
```

Code Listing 8.2 : Example of using Inheritance

Now, let us add code to create a subclass 'Classroom' by extending superclass 'Room' as shown in code listing 8.3. Subclass has two additional instance variables nBenches and nSeatsBench. 8Variable nSeatsBench denotes the number of students that can seat on one bench. It has its own additional constructors and three methods: show(), display() and getSeats().

```
class Classroom extends Room
{
        int nBenches, nSeatsBench;
        Classroom( ) {};
        Classroom( float l, float w, float h, byte n, int nB, int nSB)
        {
                super (l,w,h,n);
                nBenches = nB; nSeatsBench = nSB;
        }
        void show()
        {
                super.display();
                System.out.println ("Benches: " + nBenches );
                System.out.println ("Seats per Bench: " + nSeatsBench );
                System.out.println ("Total Seats in a class: " + getSeats() );
        }
        void display()
        {
                System.out.println("\nClassroom with length " + length + " feet, width "
                        + width + " feet\nhas " + nBenches + " Benches, each to accomodate "
                        + nSeatsBench + " Students\nSo, Total seats in a class is  "
                        + getSeats());
        }
        int getSeats() {return nBenches * nSeatsBench; }
}  // end class Classroom
```

Code Listing 8.3 : Code for subclass named Classroom

In sub class 'Classroom', user defined constructor and method show() have reused the code written in superclass 'Room'.

As constructors of a superclass are not inherited in the subclass, keyword 'super' is used to call the constructor of superclass in the constructor of subclass. This call must be the first statement in the constructor. When there is no explicit call to constructor of superclass, no-argument constructor of superclass is implicitly called as the first statement 'super()'.

Method show() is used to display attributes of Classroom object. To display first four attributes inherited from superclass 'Room', we want to use existing code in display() method of superclass.

Now display() method is available in subclass also. In show(), our intention is call display() method of superclass. To do so, we have used super.display().

When superclass and subclass have methods with same signature, a superclass method is said to be overridden in the subclass. Method display() in subclass is used to override display() method of superclass. It means that we want to display the details in a different way without reusing the display() method of superclass. When such method of superclass is to be referred, we need to use keyword 'super' with dot operator and method name. Here we have used super.display() in show() method to invoke display() method of superclass. Method getSeats() is used to compute the seating capacity of a classroom.

Let us use these classes in an application by adding code as given in code listing 8.4. Here, we have created objects of both superclass and subclass.

```
class Inheritance  /* Application using Room, Classroom */
{
        public static void main (String args[])
        {
                Room r1 = new Room(20, 15, 10, (byte)2);
                r1.display();
                Classroom cr1 = new Classroom (30, 20, 12, (byte)3, 21, 3);
                cr1.show();
                System.out.println("Area of classroom1 is " + cr1.area() + " square feet");
                Classroom cr2 = new Classroom (30,30,10, (byte)4, 20, 4);
                cr2.display();
                System.out.println ("\nTotal number of Windows: " + Room.totWindows);
        }  // end main()
}  // end Inheritance
```

Code Listing 8.4 : Application using Room, Classroom

All instance variables and methods are inherited from super class to subclass. Thus method area() of superclass can be invoked using an object of subclass also using cr1.area(). When overridden method is referred in an application using subclass object, it calls the method of subclass as cr2.display(). The output of the code created by combining code 8.2, 8.3 and 8.4 is shown in figure 8.15.

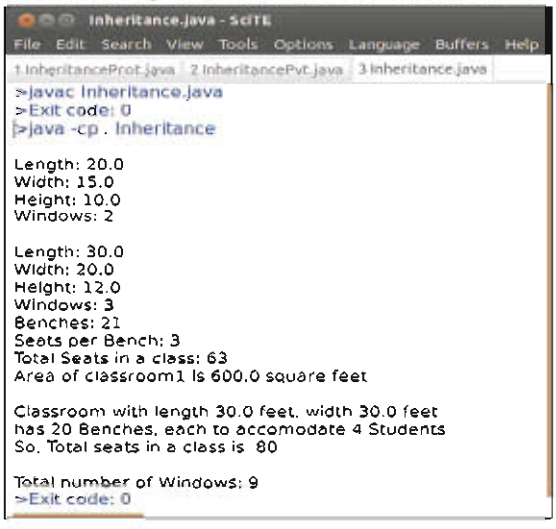


Figure 8.15 : Output of Inheritance using classes Classroom and Room

Private members of superclass

In the previous example, all instances have package visibility. So they are available for direct use in the entire package. If we try to access r1.length or cr1.width directly in main() method, there will not be any error.

If we change the visibility of instance variables of superclass to private, these variables are not directly accessible outside the class. Remember that these attributes belongs to subclass also, but still private instance variables or methods are not visible even in its subclass.

Modify declaration of instance variables in code listing 8.2 to make them private as follows and observe an error in output as shown in figure 8.16.

```
private float length, width, height;

private byte nWindows;
```

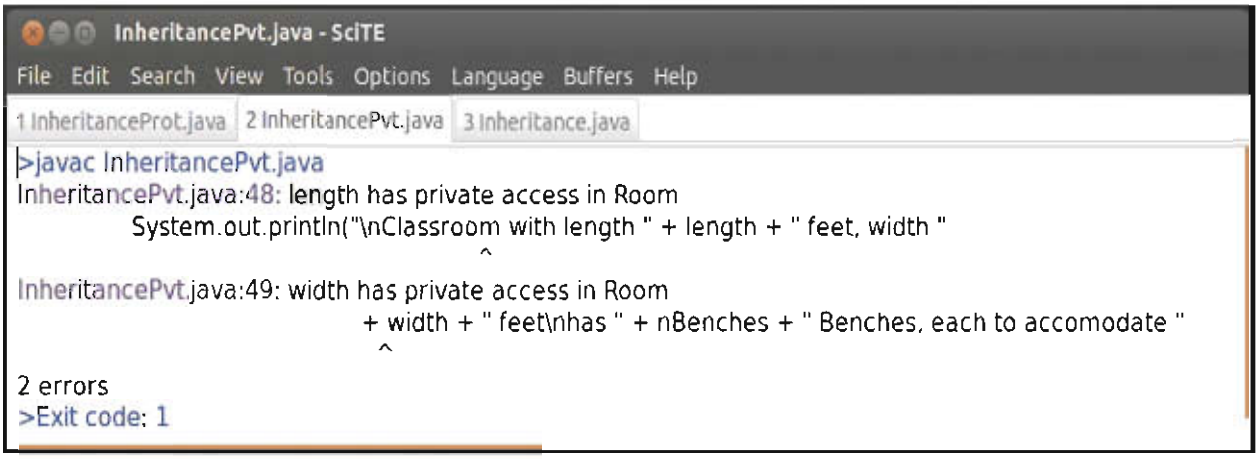


Figure 8.16 : Private members of a class are not accessible outside its class

Remember that private members are directly available only in the class in which they are defined and nowhere else. To make them available elsewhere, use public accessor and mutator methods; 'getter' and 'setter' methods.

Protected members of superclass

As studied before in the topic about 'Visibility Modifiers', 'protected' members are available as 'private' members in the inherited subclass. Modify the instance variables of class 'Room' to protected as follows.

```
protected float length, width, height;

protected byte nWindows;
```

When we execute the modified code we will get the same result as shown in figure 8.15. It is to be noted that Java does not support multiple inheritance. A subclass can be derived from only one superclass.

Composition and Aggregation

Composition and aggregation are the construction of classes that incorporate other objects. They establish a "has-a" relationship between classes.

For example, if we define class 'Library', it has a reading room. Here reading room is an object of the class 'Room'. Thus Library has a Room. When a class includes objects of other class, it is also referred to as container class.

Other examples are :

- Person has a name, address. Here, name is of the class Name with three attributes first name, middle name and last name; address is of class Address with attributes house number, apartment/ society name, area, city, state, country, pin code.
- Car has a steering, wheels and engine. Here steering is of the class Steering, wheel is of the class Wheel and engine is of the class Engine. All the three attributes of class Car are components or parts of a car.

Let us create class 'Library' that contains following attributes :

- nBooks: int, number of books in library
- nMagazines: int, number of magazines subscribed in library
- nNewspapers: int, number of newspapers subscribed in library
- readingRoom: Room

See that readingRoom is not of the basic data types. It is of the type 'Room' class.

Code listing 8.5 shows the code to create class named Room and Library and use them in application named 'Container.java'.

```
/* Using objects as data members in a container class */
class Room
{
        protected float length, width, height;
        protected byte nWindows;
        static int totWindows;  // class variable

        Room ( ) { }; // user-defined no-argument constructor
        Room (float l, float w, float h, byte n)
        {
                length = l; width = w; height = h;
                nWindows = n;   totWindows+=n;
        }
        Room (float l, float w)
        {
                length = l; width = w; height = 10;
                nWindows = 1;       totWindows++;
        }
```

```
        double area ( )  // area =  length * width
        {       return (length * width);        } // end area() method
        void display ( )
        {
                System.out.println ("Length: " + length + "\nWidth: " + width);
                System.out.println ("Height: " + height);
                System.out.println ( "Windows: " + nWindows);
        } // end display() method
}  // end Room class

class Library
{
        int nBooks, nMagazines, nNewspapers;
        Room readingRoom;
        Library( ) {};
        Library( int nB, int nM, int nN, Room r)
        {
                nBooks = nB; nMagazines=nM; nNewspapers=nN;
                readingRoom=r;
        }
        void display()
        {
                System.out.println ("\nLibrary Details:\nNumber of books: " + nBooks );
                System.out.println ("Number of subscribed magazines: " + nMagazines );
                System.out.println ("Number of subscribed newspapers : " + nNewspapers);
                System.out.println ("Reading Room:");
                readingRoom.display();
        }
}  // end class Library

class Container  /* Application using Room, Library */
{
        public static void main (String args[])
        {
                Room r1 = new Room(20, 15, 10, (byte)2);
                r1.display();
                Library lib = new Library (300, 20, 5, r1);
                lib.display();
        }  // end main()
}  // end class Container
```

Code Listing 8.5 : Example of Composition and aggregation

Observe the following points in code listing 8.5,

- In main() method of class 'Container',
  - An object lib of class 'Library' is created using a constructor with four arguments. Last argument is r1 of class 'Room'.
  - Method display() of class 'Library' is invoked using lib object.
- In 'Library' class,
  - An object readingRoom is an attribute of class 'Room'
  - Constructor with four arguments uses last argument 'r' of the class 'Room', assigns the values of Room attributes using an assignment statement 'readingRoom = r;'
  - Defines display() method which invokes display() method of 'Room' class using 'readingRoom.display();'. Note that display() method is not overridden; we have not used inheritance. Readers may use another name for this method, say show() in class 'Library' and use 'lib.show()' in main() method instead of 'lib.display()'.

In this scenario, reader may think: Why not to use inheritance? Why not to inherit 'Library' class from 'Room' class and add three additional attributes?

Note that 'Library' is not of the type of 'Room'. There is no 'is-a' relationship between 'Library' and 'Room'. There is 'has-a' relationship; 'Library' has a 'Room' used as reading room. So, we have used readingRoom of type 'Room' as an attribute of class 'Library'.

Summary

We have seen how to create class and object, how to access instance variables of class and invoke instance methods using objects. Constructors are special methods with the same name as class name, no argument and no return type. Constructors are called implicitly when an object is instantiated using new operator. Instance variables belong to each object. Class variables and methods are defined using 'static' keyword. Static members are allocated memory only once per class and shared by all objects of the class. They belong to class and not to object. We also studied about access modifiers that determine visibility of instance members. Private members are visible only within a class where they are defined, protected members are visible only in inherited subclasses, package members are visible anywhere within a package, public members are visible anywhere. For protection purpose, it is advisable to use private instance variables and supply 'getters' and 'setters' methods as public. In the last, we studied how to inherit new class using existing class; re-use, extend and override the methods. We also see the use of an object as an instance variable in class. This enables creating classes as composition and aggregation of objects.

EXERCISE

1. What do you mean by instantiation of an object ?
2. Give an example for the need of class variable.
3. Write the differences between methods and constructors.

4. Write about accessor and mutator methods.

5. How can a superclass constructor be invoked from the subclass ?

6. How can an overridden method of superclass be invoked from the subclass ?

7. Write a short note on 'Access modifiers'.

8. Explain the use of inheritance and composition or aggregation based on type of relationship between classes.

9. Explain the difference between method overloading and method overriding.

10. Choose the correct option from the following :

(1) Which of the following defines attributes and methods ?

(a) Class (b) Object (c) Instance (d) Variable

(2) Which of the following keyword is used to declare Class variables and class methods ?

(a) static (b) private (c) public (d) package

(3) Which of the following operator creates an object and returns its reference ?

(a) dot (.) (b) new (c) colon (:) (d) assignment (=)

(4) Which of the following method can be called without creating an instance of a class ?

(a) Instance method (b) Class method

(c) Constructor method (d) All of the above

(5) Which of the following refers more than one method having same name but different parameters ?

(a) Overloaded methods (b) Overridden methods

(c) Duplicate methods (d) All of the above

(6) Which method is invoked automatically with creation of an object ?

(a) Instance method (b) Constructor

(c) Class method (d) All of the above

(7) Which of the following is the keyword used to refer a superclass constructor in subclass constructor ?

(a) extends (b) super

(c) name of the superclass (d) new

(8) Which of the following is used to invoke an instance method in Java ?

(a) The name of the object, colon(:) and the name of the method

(b) The name of the object, dot(.) and the name of the method

(c) The name of the class, colon(:) and the name of the method

(d) The name of the class, dot(.) and the name of the method

(9) Which of the following is accessible by instance methods ?

(a) Only instance variables
(b) Only class variables
(c) Both instance variables and class variables
(d) All of the above

(10) When methods in the superclass and subclass have same name and signature, what are they called ?

(a) Overloaded methods
(b) Overridden methods
(c) Inherited methods
(d) All of the above

LABORATORY EXERCISE

Write Java Programs for the following :

1. Create a class named 'FixedDeposit' that contains three attributes (principal amount, annual interest rate and period (years) of deposit) and a method that returns maturity amount using compound interest. Create another class named 'FixedDepositDemo' with main() method. In main() method, create two objects, assign the values to their attributes and display them with maturity amount.

2. Add following constructors in 'FixedDeposit' class and use them to create objects.

   a. No-argument constructor that initializes principal amount with 1000, annual interest rate with 5% and period of deposit as 3 years.

   b. Parameterized constructor with 3 arguments to initialize 3 attributes.

3. Modify the visibility of instance variables in class 'FixedDeposit' as 'private' in lab exercise 1 and try to execute. Will it give an error ? If so, add display() method in the class to display the value of the instance variables.

4. Add accessor and mutator methods to get and set three attributes of the class 'FixedDeposit'. Using these methods, display the value of private instance variables in main() method.

5. Add class variable 'totDeposit' in a class that contains total amount of deposited principal amount. Modify the constructors and setter methods so as to get the total deposit amount. Write a method to display the value of 'totDeposit' variable and show its use.

6. Using 'Rectangle' class, derive a subclass 'Box' having additional attribute 'height' and method 'volume'. (Volume = height x width x length = height x area)

Working with Array and String 9

In chapter 7, we studied basic data types supported in Java. Each variable of such basic data type can store only one value at a time. Such variables are called scalar variable. In this chapter, we will discuss some composite data types that can be used to store a collection of more than one data values. For example, array and string.

Introduction to Array

An array is a variable representing a collection of homogeneous type of elements. Arrays are useful to represent vector, matrix and other multi-dimensional data. Vector is a one dimensional (1-D) data structure that can be used to store list of items like characters, numbers. Matrix is used to represent two dimensional (2-D) data structure like table of rows and columns.

Arrays are useful when same operations are to be performed on various elements of the similar type. All the elements of an array are stored in memory using contiguous storage space. Each element is identified by an index position associated with array variable.

In Java, array is an object used to manage list of items. Creating an array is a two step process:

1. Declare an array object
2. Create an array object

An array object can be created in two ways:

1. Using new operator and specifying the size
2. Directly initializing the content of array

1-D array

Array with single dimension is known as 1-D array. For example, vector which can be seen as a collection of one or more scalar variables. Instead of declaring individual variables like marks1, marks2, marks3, marks4, mark5; we can declare array marks[5] and accesses individual variable elements as marks[0], marks[1], marks[2], marks[3], marks[4].

To declare a 1-D array we use a pair of square brackets [ ] after array name or after data type. The syntax to declare array is as follows:

```
<data type> <array name> [];    or <data type> [] <array name>;
```

For example, to store the marks obtained by a student in five tests in Mathematics subject, we can use an array of five integer elements. Array object named 'marks' with size for 5 elements can be created as follows:

```
int marks[];                    // declare array object
marks = new int [5];            // create array object
```

We can combine above two steps and create array using a single statement also as follows :

int marks[] = new int [5]; or int [] marks = new int [5];

Here name of the array is 'marks'. It refers to memory location where five integer values are stored. Integer of int data type uses 4 bytes storage space. Thus, array 'marks' requires 5 x 4 = 20 bytes in contiguous locations in memory.

To refer an element of an array, we use index or subscript in square bracket [ ] after array variable name as shown in figure 9.1. Index specifies the position of an element in an array; for example, marks[2]. Index value starts from 0.

```
int marks[] = {90, 60, 70, 65, 80};
```

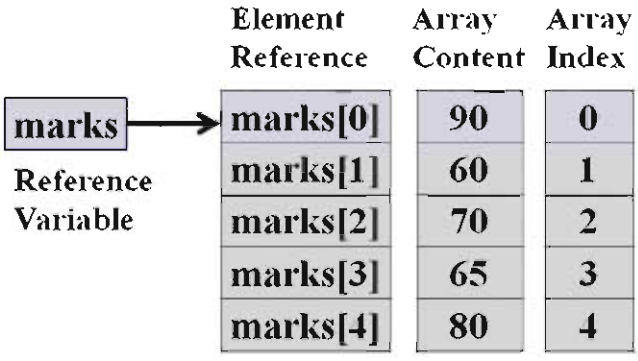


Figure 9.1 : One dimensional array

In figure 9.1 the array variable marks has an index value from 0 to 4. Here, marks[0] refers to the first element and marks[4] refers to the last element.

As explained in chapter 8, an object is a reference variable. Array being an object in Java, array name is also a reference variable. It contains reference to memory location where the array elements are stored. See figure 9.1.

As array is an object, so its elements are initialized with default values. Figure 9.2 shows an example of how to use an array in java programs.

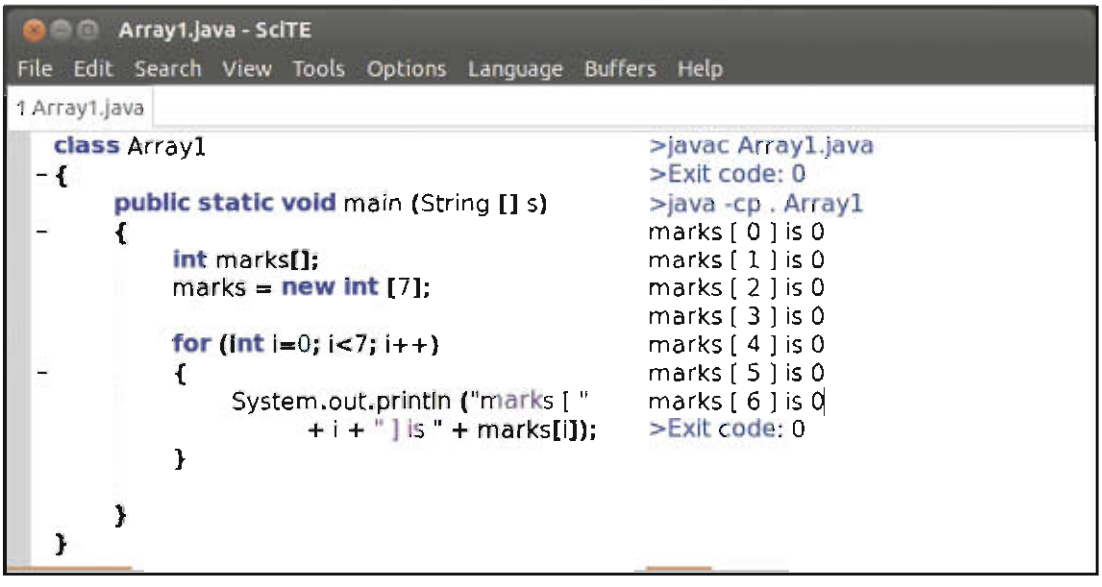


Figure 9.2 : Array object with its elements initialized by default values

One can also specify the initial values of data elements while declaring an array. 1-D array is initialized using comma separated values of data elements in braces { }. For example,

int marks[ ] = {90, 70, 77}; or int [] marks = {90, 70, 77};

Note that when an array is initialized while creating, it does not need the use of new operator. Its size is same as the number of values specified in braces.

The code shown in figure 9.3 uses different ways to create and initialize array. Note the use of class method display() used to display contents of an array. Class methods are declared as static and can be referred without any object in the class.

```
File Edit Search View Tools Options Language Buffers Help
1 Array2.java
/* creating and initializing 1-D array in different ways */
class Array2
{
    public static void main (String [] s)
    {
        int marks1[];
        marks1 = new int [3];

        int marks2[] = new int[3];
        int [] marks3 = new int [3];
        int marks4[] = {50, 60, 70};
        int [] marks5 = {70, 80, 90};

        System.out.print ("Array marks1:\t");
        display(marks1,3);
        System.out.print ("Array marks2:\t");
        display(marks2,3);
        System.out.print ("Array marks3:\t");
        display(marks3,3);
        System.out.print ("Array marks4:\t");
        display(marks4,3);
        System.out.print ("Array marks5:\t");
        display(marks5,3);
    }

    static void display(int arr[], int size)
    {
        for (int i=0; i<size; i++)
        {
            System.out.print (arr[i] + "\t");
        }
        System.out.println();
    }
}
```

```
>javac Array2.java
>Exit code: 0
>java -cp . Array2
Array marks1:   0    0    0
Array marks2:   0    0    0
Array marks3:   0    0    0
Array marks4:   50   60   70
Array marks5:   70   80   90
>Exit code: 0
```

Figure 9.3 : Different ways to create and initialize array object

In Java, we can not specify both the size of dimensions and initial values of the array elements simultaneously while declaring an array. Figure 9.4 shows the code to illustrate the statement. If we declare an array variable without initialization, only a variable is created to hold the reference as shown in figure 9.1. It does not create an array object, i.e. no memory is allocated for array elements. If we try to access the elements of an array, it will result in an error.

```
File Edit Search View Tools Options Language Buffers Help
1 Array3.java
/* creating and initializing 1-D array
   specifying size with array variable"/

class Array3
{
    public static void main (String [] s)
    {
        int marks4[3] = {50, 60, 70};
        int [5] marks5 = {70, 80, 90};

        System.out.print ("Array marks4:\t");
        display(marks4,3);
        System.out.print ("Array marks5:\t");
        display(marks5,3);
    }

    static void display(int arr[], int size)
    {
        for (int i=0; i<size; i++)
        {
            System.out.print (arr[i] + "\t");
        }
        System.out.println();
    }
}
```

```
>javac Array3.java
Array3.java:8: ']' expected
        int marks4[3] = {50, 60, 70};
                  ^
Array3.java:8: illegal start of expression
        int marks4[3] = {50, 60, 70};
                  ^
Array3.java:8: illegal start of expression
        int marks4[3] = {50, 60, 70};
                  ^
Array3.java:8: not a statement
        int marks4[3] = {50, 60, 70};
                  ^
Array3.java:8: ';' expected
        int marks4[3] = {50, 60, 70};
                  ^
Array3.java:9: ']' expected
        int [5] marks5 = {70, 80, 90};
            ^
Array3.java:9: ';' expected
        int [5] marks5 = {70, 80, 90};
            ^
Array3.java:9: <identifier> expected
        int [5] marks5 = {70, 80, 90};
            ^
Array3.java:11: <identifier> expected
        System.out.print ("Array marks4:\t");
                  ^
Array3.java:11: illegal start of type
        System.out.print ("Array marks4:\t");
```

Figure 9.4 : Illegal to specify size while initializing an array object

Every element of an array is an individual variable that can be referred by using index. Like variables, to change the value of an element, we can use assignment statement referring a variable element on left side. For example, marks[3] = 56; Let us write a program that computes an average of 10 floating point numbers and execute it. Observe the code listing and its output as given in figure 9.5.

ArrayAvg.Java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 ArrayAvg.java

```
/* compute average of 10 numbers */
class ArrayAvg
{
    public static void main (String [] s)
    {
        double numbers [] = { 10.5, 20.6, 30.8, 15.5,
                              17.3, 25.5, 27.2,
                              20, 30, 18.5};

        byte ctr;
        double sum=0, avg;

        System.out.println ("list of numbers is");
        for (ctr=0; ctr<10; ctr++)
        {
            System.out.println (numbers[ctr]);
            sum = sum + numbers[ctr];
        }
        avg = sum/10;
        System.out.println ("\nAverage of above numbers is "
                            + avg);
    } // main
} // class
```

```
>javac ArrayAvg.java
>Exit code: 0
>java -cp . ArrayAvg
list of numbers is
10.5
20.6
30.8
15.5
17.3
25.5
27.2
20.0
30.0
18.5

Average of above numbers is 21.59
>Exit code: 0
```

Figure 9.5 : Program to compute average of 10 numbers using 1-D array and loop

Here, we need to use the same operation 'add number to sum' repeatedly for different elements of same type. Use of an array helps in declaring 10 variables at a time and using a loop to perform same operation on different variable elements of an array.

We can perform various other operations on array. For example, compare two arrays, copy all the elements of one array to another, search for a specified element in array, sort elements of array and so on. For all such operations, we may write procedures as we did for finding average of elements. Instead of writing such code ourselves, we can use various static methods provided by Java using java.util.Arrays class. In Java, array is treated as an object of this Arrays class. This enables us to make use of methods of Arrays class with array type of data. The code listing and its output in figure 9.6 shows how to use methods sort() and fill() of java.util.Arrays class.

We can use sort() method to sort entire or part of array. When we use only array as an argument, it sorts an entire array. When we invoke this method with 3 arguments: array, start, last; it sorts partial array from element at index start to element at index (last-1). For example, the method call java.util.Arrays.sort (list, 1, 5) sorts elements of array list from list[1] to list[5-1]; See figure 9.6.

Method 'fill' is used to fill the whole or partial array with specified value. When the method is invoked with two arguments: array and value; it assigns the specified value to all array elements. When it is invoked with four arguments: array, start, last, value; it fills partial array from element at start to (last-1) with specified value. For example, fill (list, 7) assigns value 7 to all elements of list array; whereas fill(list, 2, 6, 5) assigns value 5 to elements list[2] to list[6-1]. See figure 9.6.

```
ArraysClassSortFill.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 ArraysClassSortFill.java

class ArraysClassSortFill
{  // sort and fill methods on whole or partial array
    public static void main (String [] s)
    {
        double list[] = {   6.4, 8, 7.8, 9.8, 9.5,
                            6, 7, 8, 8.5, 5.9        };
        int indx;

        System.out.println ("Initial Elements:");
        display(list);
        java.util.Arrays.sort (list, 3, 9);  //sort partial array
        System.out.println ("\nsort partial array: list[3] to list[8]:");
        display(list);
        java.util.Arrays.sort (list);  //sort whole array
        System.out.println ("\nsort whole array:");
        display(list);

        java.util.Arrays.fill (list,7);  //fill whole array
        System.out.println ("\nFill whole array:");
        display(list);
        java.util.Arrays.fill (list, 2, 6, 5);  //fill partial array
        System.out.println ("\nFill partial array: list[2] to list[5]");
        display(list);
    }  // end main

    static void display(double ary[])
    {
        for (int i=0; i<ary.length; i++)
        {
            System.out.print (ary[i] + "\t");
        }
        System.out.println();
    }  // end display
}  // end class
```

```
>javac ArraysClassSortFill.java
>Exit code: 0
>java -cp . ArraysClassSortFill
Initial Elements:
6.4  8.0  7.8  9.8  9.5  6.0  7.0  8.0  8.5  5.9

sort partial array: list[3] to list[8]:
6.4  8.0  7.8  6.0  7.0  8.0  8.5  9.5  9.8  5.9

sort whole array:
5.9  6.0  6.4  7.0  7.8  8.0  8.0  8.5  9.5  9.8

Fill whole array:
7.0  7.0  7.0  7.0  7.0  7.0  7.0  7.0  7.0  7.0

Fill partial array: list[2] to list[5]
7.0  7.0  5.0  5.0  5.0  5.0  7.0  7.0  7.0  7.0
>Exit code: 0
```

Figure 9.6 : Using sort() and fill() methods of Arrays class

To search an element in an array, Arrays class provides binarySearch() method. We will write our own method to search an element in array using linear search. Linear search method does element by element comparison in a serial fashion. Refer code listing and its output given in figure 9.7. It returns index position when an element is found; otherwise it returns -1.

```
LinearSrch.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 LinearSrch.java

// Search element in array using linear search
class LinearSrch
{
    public static void main (String [] s)
    {
        double list[] = {6, 5, 7, 9, 9.5, 6.5, 7.5, 8 };
        int indx;

        System.out.println ("Given Array elements are:");
        display(list);

        indx=search(list, 8);  // search 8
        if (indx < 0)
            System.out.println("\nelement 8 is not found in array");
        else
            System.out.println("\nelement 8 is found at position " + indx);
        indx=search(list,5.5);  // search 5.5
        if (indx < 0)
            System.out.println("\nelement 5.5 is not found in array");
        else
            System.out.println("\nelement 5.5 is found at position " + indx);
    }  // end main
    static void display(double ary[])
    {
        for (int i=0; i<ary.length; i++)
        {
            System.out.println ( ary[i] );
        }
        System.out.println();
    }  // end display
    static int search(double ary[], double x)
    {
        for (int i=0; i<ary.length; i++)
        {
            if (ary[i] == x ) return i;
        }
        return -1;
    }  // end search
}  // end class
```

```
>javac LinearSrch.java
>Exit code: 0
>java -cp . LinearSrch
Given Array elements are:
6.0
5.0
7.0
9.0
9.5
6.5
7.5
8.0

element 8 is found at position 7

element 5.5 is not found in array
>Exit code: 0
```

Figure 9.7 : Search an element in an array

2-D array

Two dimensional (2-D) arrays are used to store tabular data in the form of rows and columns. For example, to store 5 student's marks in 3 tests, we may use a tabular arrangement of marks in 5 rows and 3 columns as given in figure 9.8. In mathematics, we call it as a matrix of 5 rows and 3 columns where each element contains marks.

Students	Test1	Test2	Test3
1	50	60	70
2	35	30	50
3	70	75	80
4	80	85	90
5	40	50	55

Figure 9.8 : Tabular representation of 2-D array

In Java, 2-D array can be declared using array name and two pairs of square brackets [ ] [ ] to specify the size of two dimensions row and column respectively. For example, following statement declares and creates an array named marks to store 5 x 3 = 15 integer values in contiguous memory locations.

```
int marks [][] = new int [5][3];
```

Here, the logical view of array elements is a table of 5 rows and 3 columns. Physically, they are stored in memory using contiguous memory locations for 15 integers that is 60 bytes.

Java does not support multi-dimensional arrays directly. To create 2-D array, we have to create an array of array. There is no limit on number of dimensions.

In marks [5][3], it creates 1-D array object of 5 elements and each element is 1-D array object of 3 integers as shown in figure 9.9.

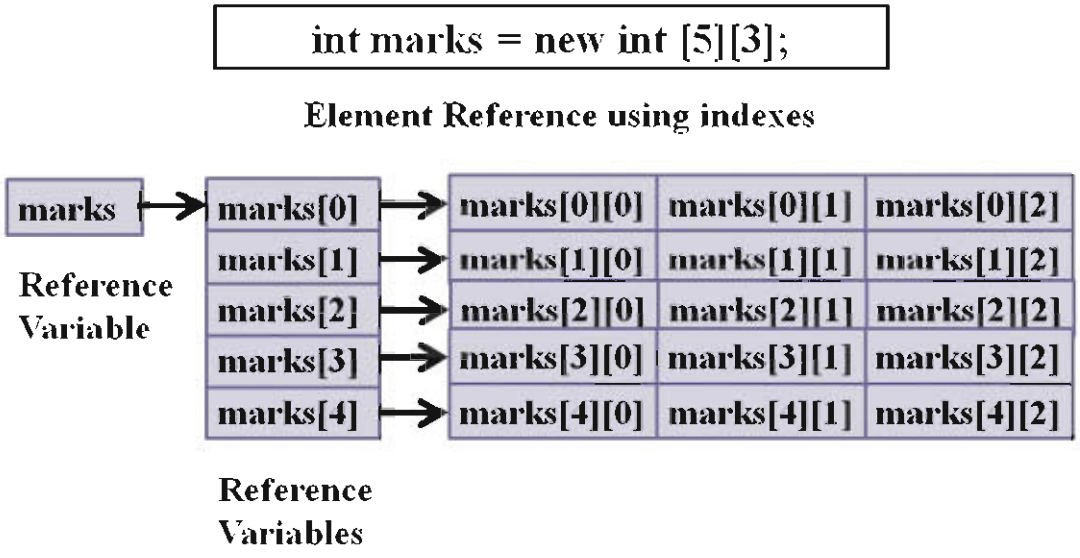


Figure 9.9 : 2-D array as an array of 1-D array

Declaration and initialization of 2-D array is very similar to 1-D array except the number of dimensions. In 2-D array, each row is considered as 1-D array element. Thus, like 1-D array initialization, each row is initialized by enclosing its elements in a pair of braces { } separated by comma. To initialize 2-D array, all these initialized rows are to be enclosed in curly braces {} separating them by comma (,). Like 1-D array, 2-D arrays can be declared in various ways as seen in code listing given in figure 9.10. Figure 9.11 shows output of the code.

```
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 Array2D.java *
/* creating and intializing 2-D array */
class Array2D
{
    public static void main (String [] s)
    {
        int marks1[] [ ];
        marks1 = new int [ 5] [3];
        int marks2[][] = new int [5][3];
        int [][] marks3 = new int [5][3];
        int marks4[] []= { {50, 60, 70}, {35, 30, 50},
                           {70, 75, 80}, {80, 85, 90},
                           {50, 50, 55}  };
        int [] []marks5 = {{50, 60, 70}, {35, 30, 50},
                           {70, 75, 80}, {80, 85, 90},
                           {50, 50, 55}  };

        System.out.print ("2-D Array marks1\n:");
        display(marks1,5,3);
        System.out.print ("2-D Array marks2:\n");
        display(marks2,5,3);
        System.out.print ("2-D Array marks3:\n");
        display(marks3,5,3);
        System.out.print ("2-D Array marks4:\n");
        display(marks4,5,3);
        System.out.print ("2-D Array marks5:\n");
        display(marks5,5,3);
    } // main
    static void display(int arr[][], int rows, int cols)
    {
        for (int i=0; i<rows; i++)
        {
            for (int j=0; j<cols; j++)
            {
                System.out.print (arr[i] [j] + "\t");
            }
            System.out.println();
        }
    } // display
} // class
```

Figure 9.10 : Example of 2-D array

```
Array2D.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 Array2D.java
>javac Array2D.java
>Exit code: 0
>java -cp . Array2D
2-D Array marks1
:0    0    0
0    0    0
0    0    0
0    0    0
0    0    0
2-D Array marks2:
0    0    0
0    0    0
0    0    0
0    0    0
0    0    0
2-D Array marks3:
0    0    0
0    0    0
0    0    0
0    0    0
0    0    0
2-D Array marks4:
50   60   70
35   30   50
70   75   80
80   85   90
50   50   55
2-D Array marks5:
50   60   70
35   30   50
70   75   80
80   85   90
50   50   55
>Exit code: 0
```

Figure 9.11: Output

In Java, 2-D array is considered as an array of 1-D array. So, rows of 2-D array can have variable number of columns. Figure 9.12 shows how to use 2-D array of characters to store five names of computer programming languages.

```
char names [ ][ ] = {   {'J', 'a', 'v', 'a'}, {'C'}, {'C', '+', '+'},
                        {'B', 'a', 's', 'i', 'c'}, {'P', 'a', 's', 'c', 'a', 'l'}   }
```

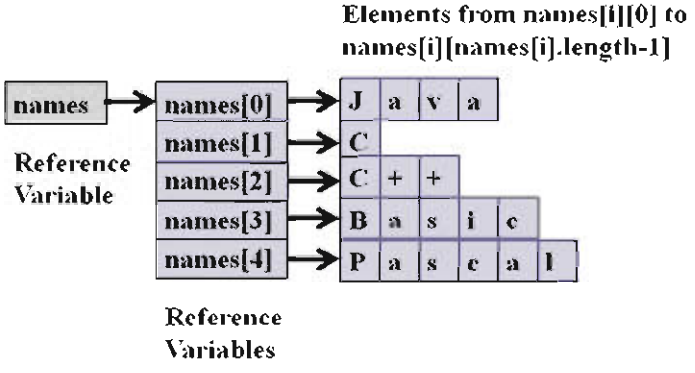


Figure 9.12 : 2-D array with variable number of columns

Each name is stored in different row and all names have different number of characters. Thus, each row is of different size. Size of each row can be known using 'length' property of 1-D array.

Figure 9.13 shows how to use 2-D array with varying column size. Here, we have used 'length' property to get the number of elements in 1-D array. For 2-D array, it returns number of rows. For 1-D array, it returns number of columns in specified row element. Remember that for 2-D array, number of elements is considered as number of rows and each row is 1-D array. In short, length property used with only array name returns the size of its first dimension.

```
Array2Dchar.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 Array2Dchar.java
/* creating and initializing 2-D array */
class Array2Dchar
{
    public static void main (String [] s)
    {
        char names[] []= {
                          {'J','a','v','a'},
                          {'C'},
                          {'C', '+', '+'},
                          {'B', 'a', 's', 'i', 'c'},
                          {'P','a','s','c', 'a', 'l'}
                         };

        System.out.println("Number of elements in 2-D array: "
                           + names.length + "\n");
        System.out.print
                 ("Five names stored in 2-D Array of characters:\n");
        display(names,5);
    } // main
        static void display(char arr[][], int rows)
    {
        for (int i=0; i<rows; i++)
        {
            System.out.print ("Row " + i + " have " + arr[i].length +
                              " character elements: ");
            for (int j=0; j<arr[i].length; j++)
            {
                System.out.print ( arr[i] [j] );
            }
            System.out.println();
        }
    } // display
} // class
```

```
>javac Array2Dchar.java
>Exit code: 0
>java -cp . Array2Dchar
Number of elements in 2-D array: 5

Five names stored in 2-D Array of characters:
Row 0 have 4 character elements: Java
Row 1 have 1 character elements: C
Row 2 have 3 character elements: C++
Row 3 have 5 character elements: Basic
Row 4 have 6 character elements: Pascal
>Exit code: 0
```

Figure 9.13 : Program using 2-D array with variable number of columns

Later we will see that 1-D array of characters can be defined using String class available in Java. In figure 9.14, we have used 2-D array of bytes to store names. Here, it shows that initial chara type data values are converted to byte data type and the characters of names are assigned their corresponding ASCII values.

```
File Edit Search View Tools Options Language Buffers Help
1 Array2Dbyte.java
/* creating and initializing 2-D array */
class Array2Dbyte
{
    public static void main (String [] s)
    {
        byte names[] []= {
                          {'J','a','v','a'},
                          {67},
                          {'C', '+', '+'},
                          {'B', 'a', 's', 'i', 'c'},
                          {'P','a','s','c', 'a', 'l'}
                         };
        System.out.print ("Five names stored in 2-D Array of bytes:\n");
        display(names,5);
    } // main
        static void display( byte arr[][], int rows)
    {
        for (int i=0; i<rows; i++)
        {
            for (int j=0; j<arr[i].length; j++)
            {
                System.out.print ( arr[i] [j] + "\t" );
            }
            System.out.println();
        }
    } // display
} // class
```

```
>java -cp . Array2Dbyte
Five names stored in 2-D Array of bytes:
74   97   118  97
67
67   43   43
66   97   115  105  99
80   97   115  99   97   108
>Exit code: 0
```

Figure 9.14 : 2-D array: Characters stored in bytes using corresponding integer values

Points to be remembered :

- array is a collection of homogeneous type of data
- array elements can be accessed using index for each dimension in [ ]
- index value starts with zero
- multi-dimensional arrays can have variable size of 2nd, 3rd ... dimensions
- attribute 'length' is used to determine the size of the dimension
- array is treated as an object
- declaring an array without initialization does not create an array object

Strings

String is nothing but a sequence of characters. So, 1-D array of characters can be considered as a string. We have already studied and used string literals, where a sequence of characters is enclosed between double quotes.

To use variables that can store strings, Java supports two types of strings that are handled by two classes namely 'String' and 'StringBuffer'. In this chapter, we will see the first one.

Some of the constructors that can be used to create an object are as follows :

- String () without arguments create a String object with no character
- String (char ary[]) creates a String object with its initial value using ary argument
- String (char ary[], int start, int len) creates a String object using 1-D ary argument starting at ary[start] with len number of characters
- String (String strObj) creates a String object which is same as object specified in argument
- String (string literal) creates a String object that refers to the literal specified in argument.

Figure 9.15 shows the use of various types of String class constructors. Do not forget to use new operator while creating object.

String1.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 String1.java

```
/* creating String Objects */
class String1
{
    public static void main (String [] s)
    {
        char name [] = { 'j', 'a', 'v', 'a'};

        String str1 = new String();
        String str2 = new String(name);
        String str3 = new String(name, 1,2);
        String str4 = new String(str3);
        String str5 = new String("Java");

        System.out.println ("str1 is " + str1);
        System.out.println ("str2 is " + str2);
        System.out.println ("str3 is " + str3);
        System.out.println ("str4 is " + str4);
        System.out.println ("str5 is " + str5);
    } // main
} // class
```

```
>javac String1.java
>Exit code: 0
>java -cp . String1
str1 is
str2 is Java
str3 is av
str4 is av
str5 is Java
>Exit code: 0
```

Figure 9.15 : Creating objects of class String using various constructors

In Java, characters are stored using two bytes. To save space, if the characters are ASCII, we should use an array of bytes instead of array of characters. We can use a constructor with array of bytes as an argument. Try the same program by replacing char array with byte array.

Note :

Literals are stored in memory. When two String objects are created using same string literals, memory space is not allocated for second object. Both objects refer to same memory location.

Separate memory is allocated when string objects are created using new operator even if strings are identical. Code listing given in figure 9.16 shows such an example.

```
String2.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 String2.java

/* String Objects */
class String2
{
    public static void main (String [] s)
    {
        String str1 = "I like Java";
        String str2 = "I like Java";
        String str3 = new String("I love India");
        String str4 = new String(str3);

        System.out.println ("str1==str2: "
                            + (str1==str2));
        System.out.println ("str3==str4: "
                            + (str3==str4));

        System.out.println ("str1: " + str1);
        System.out.println ("str2: " + str2);
        System.out.println ("str3: " + str3);
        System.out.println ("str4: " + str4);
    } // main
} // class
```

```
>javac String2.java
>Exit code: 0
>java -cp . String2
str1==str2: true
str3==str4: false
str1: I like Java
str2: I like Java
str3: I love India
str4: I love India
>Exit code: 0
```

Figure 9.16 : Identical String objects created using string literal and using new operator

In code listing given in figure 9.16, note that "str1 == str2" compares the contents of str1 and str2 and not the objects at str1 and str2. String objects str1 and str2 are created using same string literal but without using new operator. Here, both reference variables str1 and str2 refer to the same instance as created for str1. For object str2, separate memory is not allocated as seen in figure 9.17. In the same program, String objects str3 and str4 are created using new operator. The contents of both these objects are same and they refer to different memory locations. Separate memory is allocated for these two objects as seen in figure 9.17.

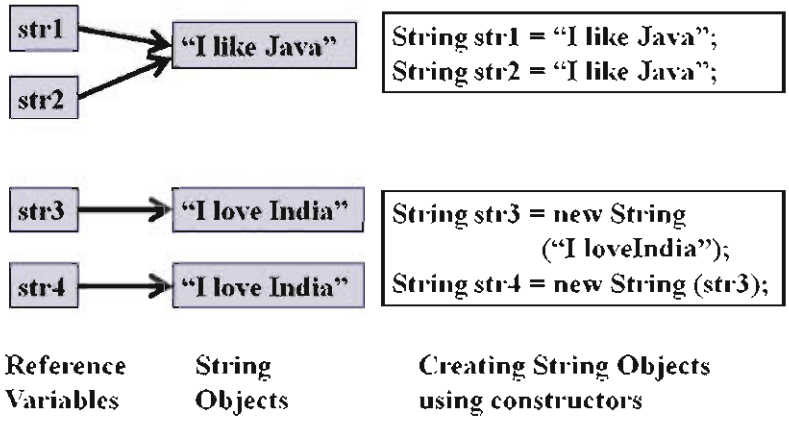


Figure 9.17: Identical strings

String class methods

The String class provides methods to compare strings, find length of string, combining strings, obtaining substrings, converting strings, splitting strings, searching for character or patterns in string etc. We will see the examples of using some of these methods through programs. The program in figure 9.18 shows different ways to compare strings or part of strings.

```
StringComparison.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 StringComparison.java

class StringComparison
{
    public static void main (String [] s)
    {
        String str1 = "India is great";
        String str2 = "INDIA is Great";

        System.out.println ("str1: " + str1);
        System.out.println ("str2: " + str2);
        System.out.println ("str1.equals(str2): "
                    + (str1.equals(str2)));
        System.out.println ("str1.equalsIgnoreCase(str2): "
                    + (str1.equalsIgnoreCase(str2)));
        System.out.println ("str1.compareTo(str2): "
                    + (str1.compareTo(str2)));
        System.out.println ("str1.compareToIgnoreCase(str2): "
                    + (str1.compareToIgnoreCase(str2)));
    } // main
} // class

>javac StringComparison.java
>Exit code: 0
>java -cp . StringComparison
str1: India is great
str2: INDIA is Great
str1.equals(str2): false
str1.equalsIgnoreCase(str2): true
str1.compareTo(str2): 32
str1.compareToIgnoreCase(str2): 0
>Exit code: 0
```

Figure 9.18 : Comparing strings using methods of class String

The syntax and description of various comparison methods is shown in table 9.1.

Method	Description
boolean equals (String str)	Returns true if invoking string is same as str
boolean equalsIgnoreCase (String str)	Returns true if invoking string is same as str after ignoring case (case insensitive)
int compareTo (String str)	Returns 0, >0, <0 integer if invoking string is equal to, greater than or less than str respectively
int compareToIgnoreCase (String str)	Same as CompareTo but case insensitive

Table 9.1 : String class methods for comparing strings

String class provides methods for other tasks as follows. Some of them are given in table 9.2.

- Extracting part of string
- Replacing characters or substrings
- Splitting string into substrings
- Getting number of characters

- Getting character at specified index position
- Converting string into an array of bytes
- Converting string into lowercase or uppercase
- Appending string
- Copy string or part of string

Method	Description
int length()	Returns number of characters in invoking string
char indexAt(int index)	Returns character at index position from the invoking string, index considered from 0
byte [] getBytes()	Returns an array of characters as bytes from invoking string
void getChars (int fromIndx, int toIndx, char target [], int targetIndx)	Copies characters of invoking string from fromIndx to toIndx-1 to target array from targetIndx onwards
String concat(String str)	Returns a string after appending str with the invoking string
String toLowerCase()	Returns a string with all characters of invoking string converted to lowercase
String toUpperCase()	Returns a string with all characters of invoking string converted to uppercase

Table 9.2 : Additional methods of String class

The code listing given in figure 9.19 shows how to use some of these methods.

StringConversion.java - SciTE

File Edit Search View Tools Options Language Buffers Help

1 StringConversion.java

```
/* String conversions */
class StringConversion
{
    public static void main (String [] s)
    {
        String str1 = "I like Java";
        String strAry[] = new String[5];
        byte byteAry[] = new byte[20];

        System.out.println ("Given string is \"" + str1 + "\"");
        System.out.println ("String in lowercase: \""
                            + str1.toLowerCase()+ "\"");
        System.out.println ("String in uppercase: \""
                            + str1.toUpperCase()+ "\"");

        System.out.println("\nString converted to array of bytes");
        byteAry = str1.getBytes();
        for (int i=0; i<byteAry.length; i++)
            System.out.println ( byteAry[i] ) ;
    } // main
} // class
```

```
>javac StringConversion.java
>Exit code: 0
>java -cp . StringConversion
Given string is "I like Java"
String in lowercase: "i like java"
String in uppercase: "I LIKE JAVA"

String converted to array of bytes
73
32
108
105
107
101
32
74
97
118
97
>Exit code: 0
```

Figure 9.19 : Using String conversion methods

Note: With array variable, length is an attribute or property of array; whereas with String object, length is a method. So we need to use () while using length method with String object.

String class of Java does not provide any method to reverse a string. So, let us write a program to reverse string. Refer code listing and its output given in figure 9.20.

```
StringReverse.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 StringReverse.java

/* Reverse String */
class StringReverse
{
    public static void main (String [] s)
    {
        String str = new String("I Like Java");
        System.out.println ("Given   String: " + str);
        System.out.println ("Reverse String: " + reverseStr(str));
    } // end main

    static String reverseStr (String str)  // reverse string
    {
        int len = str.length();
        byte byteAry[] = new byte[len];
        byteAry = str.getBytes();        // convert to byte array
        for (int i=0, j=len-1; i<len/2; i++, j--)  // exch half array
        {
            byte tmp = byteAry[i];
            byteAry[i] = byteAry[j] ;
            byteAry[j]=tmp;
        }
        return new String(byteAry);  // used constructor
    }
} // class
```

```
>javac StringReverse.java
>Exit code: 0
>java -cp . StringReverse
Given   String: I Like Java
Reverse String: avaJ ekiL I
>Exit code: 0
```

Figure 9.20 : Reversing a string

Here, we have written a user-defined method reverseStr that reverses the string as follows:

1. Determine the length of string using length() method of String class
2. Convert a string into an array of bytes using method getBytes() of String class
3. Exchange half of the elements on left (starting from first towards right) with half of the elements on right (starting from last towards left) of a byte array
4. Construct a String object from a byte array using constructor and return it

Date class

We have studied about String class and Arrays class provided by Java. Java library also provides Date class in java.util package. Date class encapsulate both date and time and represents the value using milliseconds precision. Figure 9.21 shows the use of Date class with code listing and its output.

```
Date1.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 Date1.java
import java.util.Date;

class Date1
{
    public static void main(String args[])
    {
        Date date1 = new Date();  // current date, time
        // if not used: import java.util.Date; use next statement
        java.util.Date date2 = new java.util.Date();
        System.out.println ("date1: Date & Time: " + date1);
        System.out.println ("date2: Date & Time: " + date2);

        System.out.println (
            "Elapsed time since Jan 1, 1970 is\n\t"
            + date1.getTime() + " milliseconds");
        date1.setTime(date1.getTime() + 10000000);
        System.out.println ("DateTime after 1 crore milliseconds\n\t"
            + date1.toString());
    }  // end main()
} // end class
```

```
>javac Date1.java
>Exit code: 0
>java -cp . Date1
date1: Date & Time: Thu Oct 31 08:50:59 IST 2013
date2: Date & Time: Thu Oct 31 08:50:59 IST 2013
Elapsed time since Jan 1, 1970 is
    1383189659935 milliseconds
DateTime after 1 crore milliseconds
    Thu Oct 31 11:37:39 IST 2013
>Exit code: 0
```

Figure 9.21 : Using Date class

Table 9.3 lists some of the methods of Date class.

Method	Description
Date()	Constructs Date object using current system time
Date (long elapsedTime)	Constructs Date object using specified time in milliseconds elapsed since January 1, 1970 GMT
String toString()	Returns a string representing date and time of invoking object
long getTime()	Returns number of milliseconds since January 1, 1970 GMT
void setTime (long elapsedTime)	Sets new date and time of an object using elapsed time

Table 9.3 : Methods of Date class

Calendar class

Like Date class, Calendar class is also provided in java.util package. This class can be used to extract detailed calendar information like year, month, date, hour, minute and second. Here, we will see the use of GregorianCalendar subclass of Calendar class.

Figure 9.22 shows an example program to use Calendar class along with its output. Note the use of user-defined class method display() used to display various components of date and time.

```
Calendar1.java - SciTE
File  Edit  Search  View  Tools  Options  Language  Buffers  Help
1 Calendar1.java

import java.util.*;   // for Date, Calendar, GreGorianCalendar class
public class Calendar1
{
    public static void main(String s[])
    {
        System.out.println ("Current Date & Time: " + new Date()); // current date,time
        //using current date & time
        Calendar calendar1 = new GregorianCalendar();
        display(calendar1);
        //using given year,month,day
        Calendar calendar2 = new GregorianCalendar(2013, 11, 17);
        display(calendar2);
        //using given date,year, month, day and time,hour, minute, second
        Calendar calendar3 = new GregorianCalendar(2013,11,20,210,2);
        display(calendar3);
    } // end main()
    static void display(Calendar calendar)
    {   // using Calendar constants
        System.out.println ("\nYear:\t" + calendar.get(Calendar.YEAR));
        System.out.println ("Month:\t" + calendar.get(Calendar.MONTH));
        System.out.println ("Date:\t" + calendar.get(Calendar.DATE));
        System.out.print ("Hour (12 hour):\t" + calendar.get(Calendar.HOUR));
        if (calendar.get(Calendar.AM_PM) == 1)
            System.out.println (" p.m.");
        else
            System.out.println (" a.m.");
        System.out.println ("Hour (24 hour):\t" + calendar.get(Calendar.HOUR_OF_DAY));
        System.out.println ("Minute:\t" + calendar.get(Calendar.MINUTE));
        System.out.println ("Second:\t" + calendar.get(Calendar.SECOND) );
        System.out.print ("Day of Week: ");
        switch ( calendar.get(Calendar.DAY_OF_WEEK) )
        {
            case 1: System.out.println ("Sunday"); break;
            case 2: System.out.println ("Monday"); break;
            case 3: System.out.println ("Tuesday"); break;
            case 4: System.out.println ("Wednesday"); break;
            case 5: System.out.println ("Thursday"); break;
            case 6: System.out.println ("Friday"); break;
            case 7: System.out.println ("Saturday"); break;
            default: System.out.println("Error...");
        }
    } // end display()
}// end class
```

```
>javac Calendar1.java
>Exit code: 0
>java -cp . Calendar1
Current Date & Time: Thu Oct 31 07:48:28 IST 2013

Year: 2013
Month:    9
Date:     31
Hour (12 hour):     7 a.m.
Hour (24 hour):     7
Minute:   48
Second:   28
Day of Week: Thursday

Year: 2013
Month:    11
Date:     17
Hour (12 hour):     0 a.m.
Hour (24 hour):     0
Minute:   0
Second:   0
Day of Week: Tuesday

Year: 2013
Month:    11
Date:     28
Hour (12 hour):     6 p.m.
Hour (24 hour):     18
Minute:   2
Second:   0
Day of Week: Saturday
>Exit code: 0
```

Figure 9.22 : using Calendar class and its constants

Like get methods, we can use set methods to set the value of the field constants of Calendar class. For example, if calendar is an object of class Calendar, then execution of call 'calendar.set(Calendar.DATE, 20)' will set the date to 20.

Table 9.4 lists the integer constants defined in Calendar class. These constants are given meaningful and self-explanatory names.

Constant	Description
YEAR	Year of calendar
MONTH	Month of calendar (0 for January, 11 for December)
DATE	Day of calendar month
DAY_OF_MONTH	Same as DATE
HOUR	Hour in 12-hour notation
HOUR_OF_DAY	Hour in 24-hour notation
MINUTE	Minute
SECOND	Second
AM_PM	0 for AM, 1 for PM
DAY_OF_WEEK	Day number within a week (1 for Sunday, 7 for Saturday)
WEEK_OF_MONTH	Week number within the month
WEEK_OF_YEAR	Week number within the year
DAY_OF_YEAR	Day number in the year (1 for the first day)

Table 9.4: Constants defined in Calendar class

Summary

This chapter explains how to deal with array, string and date. Array is used to have collection of variables of same data type. Java basically supports only 1-D arrays. Using 1-D array of arrays, we can practically get multi-dimensional arrays. Array is treated as an object of class Arrays, so array name is a reference variable referring to memory location where its elements are stored. All methods of class Arrays can be used with array objects. Examples of such methods are: sort, fill. String class provided in Java enables to work with a sequence of characters. Examples of operations that can be performed on strings are: comparing strings, finding number of characters in a string, converting case of letters of string. Class Date and Calendar are provided to work with date and time. Using Calendar class methods, we can get and set the fields like year, month, date, hour, minute, seconds.

EXERCISE

1. Explain array giving example.
2. Differentiate between 1-D and 2-D arrays.
3. Explain the use of the following classes :

   a. String b. Date c. Calendar

4. Choose the most appropriate from those given below :

   (1) Which of the following refer to the starting index value in arrays ?

   (a) 0 (b) 1 (c) null (d) All of these

   (2) What is the size of second dimension in an array sales[5][12] ?

   (a) 5 (b) 12 (c) 60 (d) 10

   (3) What will expression sales.length return for an array sales[5][12] ?

   (a) 5 (b) 12 (c) 60 (d) 120

   (4) When an array sales [5][12] is declared without specifying initial values, what is the initial value of sales[0][0] ?

   (a) 0 (b) default value (c) compilation error (d) 60

   (5) What does 'length' refer to for an object of String class ?

   (a) attribute (b) method (c) class variable (d) class name

   (6) If 'str' is the object of String class and its content is "Thank GOD", then what is the value of str.length() ?

   (a) 9 (b) 10 (c) 8 (d) 11

(7) What type of value is returned when we use get method of Calendar class with constant DAY_OF_WEEK as an argument ?

(a) int (b) char (c) String (d) boolean

LABORATORY EXERCISE

1. Write a program that uses an array to store 12 values of hourly temperature of a day from 6 am to 6 pm. Use either initialization or assignment statements to assign these values. Print maximum and minimum temperature of the day.

2. Write a program that stores weekly sale of five salespersons. A sales person is given a weekly incentive depending upon his/her average weekly sale. Incentive is: 10 % if average weekly sale is up to Rs. 10000, 15% if average weekly sale is above Rs.10000 but less than or equal to Rs.30000 and 20% if average weekly sale is above Rs. 30000. Compute total daily sale of all salespersons combined together. Also compute weekly incentive for each salesperson.

3. Write a program to see whether a string is palindrome or not. (String is palindrome if its reverse is same as original, example "1771", "madam")

4. Write a program to print current date in the format "DD-MMM-YYYY". For example, 17th November 2013 should be printed as 17-Nov-2013. Also print the day of week in words.

5. Write a program to set the date and time as per your birth date and time. Print the week day on your birthday.

Exception handling in Java 10

Usually we believe that a compiled program is error free and will always execute successfully, but in few cases the program can terminate while it is executing. For example, if we have written a program that connects to a particular website and download the web pages, under normal conditions the program will execute as expected but suppose if the program is executed in a computer where there is no internet connection then the program will produce unexpected output. There can be similar situations when we are dealing with files, for instance the program tries to modify a read-only file or it tries to open a file that does not exist in the system. Such cases are known as "exceptions" in Java. The exception results in an abnormal execution and it may lead to abnormal termination of the program.

An exception is an indication of a problem that occurs during a program's execution, it usually signals an error. Although exceptions occur infrequently, we must be careful to handle such cases while writing the code. Exception handling allows a program to continue executing as if no problem had been encountered or it may notify the user of the problem before terminating in an uncontrolled manner.

In this chapter we will learn techniques to handle exceptions in our programs, few standard exceptions available in Java, a technique to guarantee that a particular block of code will always be executed, even if exceptions are present in our program. Finally we will look at a technique to define and use our own type of exception

Types of Exceptions

In Java, all kinds of error conditions are called exceptions. Errors can be broadly classified into two categories namely Compile-time errors and Run-time errors.

Compile-time errors

We have already learnt in previous chapters how to compile our java programs. A compiler is used to convert source code into object code. If there is a syntax error in the program we will get a compilation error and will not be able to create the ".class" file. Examples of some common syntax errors are missing semicolon, use of undeclared variable, wrong spellings of identifier or keyword and mismatch of bracket. The java program shown in figure 10.1 does not contain a semicolon.

Figure 10.1 : A program that illustrates Compile-time Error

The above code when compiled will generate a compile-time error. This happens because we have missed writing the semicolon ';' in line number 8. The Java compiler when showing the output suggests the type of error, along with the line number where the error has occurred as shown in figure 10.2.

```
ErrorDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
ErrorDemo.java
>javac ErrorDemo.java
ErrorDemo.java:8: ';' expected
        System.out.println("Hello World")  // No Semicolon
                                         ^
1 error
>Exit code: 1
```

Figure 10.2 : Output of the program shown in Figure 10.1

Compile-time errors are usually the mistakes of a programmer and it won't allow the program to compile unless they are solved.

Note : In the field of Computer Science, "Exit code" or "Exit status" indicates whether the command or a program executed successfully or not. Code "0" indicates that the command executed successfully whereas code "1" indicates that some problem occurred while executing the command. In figure 10.2, the last line indicates "Exit Code: 1", it means that the compilation of program - ErrorDemo.java was not successful.

Run-time errors

If there are no syntax errors in the source code then the program will compile successfully and we will get a ".class" file. However this does not guarantee that the program will execute as per our expectations. Let us look at another example shown in figure 10.3 which will compile but will terminate abnormally.

```
RuntimeErrorDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
RuntimeErrorDemo.java
 1 /* A program which generates run-time error. An array of four elements is created but the program tries to
 2    access fifth element of the citylist[] array resulting in run-time error */
 3
 4 class RuntimeErrorDemo
 5 {
 6     public static void main(String args[])
 7     {
 8         /* Create an array of four elements */
 9         String citylist[] = {"Ahmedabad", "Baroda", "Rajkot", "Surat"};
10
11         System.out.println("Statement to be executed before displaying the fifth element");
12
13         /* Statement that generates run-time error  */
14         System.out.println(citylist[5]);
15
16         System.out.println("Statement to be executed after displaying the fifth element");
17     }
18 }
```

Figure 10.3 : A program illustrating run-time error

The program shown in figure 10.3 will compile as there are no syntax errors it. In the program, we have created an array "citylist[]" that contains name of four different cities. In line 14 we are trying to display the contents of element of citylist array that does not exist. This case here is an exception condition. The exception will be generated during runtime; the output of the execution of the program is shown in figure 10.4.

```
RuntimeErrorDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
RuntimeErrorDemo.java
>javac RuntimeErrorDemo.java
>Exit code: 0
>java -cp . RuntimeErrorDemo
Statement to be executed before displaying the fifth element
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 5
        at RuntimeErrorDemo.main(RuntimeErrorDemo.java:14)
>Exit code: 1
```

Figure 10.4 : Output of the program shown in Figure 10.3

From the output shown in figure 10.4, we can notice that the program execution terminated abruptly from line number 14. The output contains a phrase "ArrayIndexOutOfBoundsException" that indicates the type of exception occurred while executing the program.

Let us see few other cases that generate exceptions. For each type of exception, there are corresponding Exception classes in Java.

The java.lang and java.io package contains a hierarchy of classes dealing with various exceptions. Few widely observed exceptions are listed in Table 10.1.

Exception Class	Condition resulting in Exception	Example
ArrayIndexOutOfBoundsException	An attempt to access the array element with an index value that is outside the range of array	int a[] = new int[4]; a[13] = 99;
ArithmeticException	An attempt to divide any number by 0	int a = 50 / 0;
FileNotFoundException	An attempt to access a non-existing file	
NullPointerException	An attempt to use null in a case where an object is required	String s = null; System.out.println(s.length());
NumberFormatException	An attempt to convert string to a number type	String s = "xyz"; int i = Integer.parseInt(s);
PrinterIOException	An I/O error has occurred while printing	

Table 10.1 : List of few widely observed Exceptions

Figure 10.5 shows one more program that generates an exception.

```
RuntimeErrorDemo2.java - SciTE
File Edit Search View Tools Options Language Buffers Help
RuntimeErrorDemo2.java
1   /* A program which generates run-time error.  On dividing any number by zero generates ArithmeticException */
2
3   class RuntimeErrorDemo2
4   {
5       public static void main(String args[])
6       {
7           int numerator = 15;
8           int denominator = 0;
9           int answer;
10
11          System.out.println("Statement to be executed before performing division operation");
12
13          /* Statement that generates run-time error   */
14          answer = numerator / denominator;   // Creates ArithmeticException
15
16          System.out.println("Statement to be executed after performing division operation");
17      }
18  }
```

Figure 10.5 : A program illustrating arithmetic exception

In the program shown in figure 10.5 we try to divide an integer number by zero. A number when divided by zero leads to infinity as a result. Here the program will be compiled successfully but will generate unexpected output due to ArithmeticException. Figure 10.6 shows the output of the execution of program.

```
RuntimeErrorDemo2.java - SciTE
File Edit Search View Tools Options Language Buffers Help
RuntimeErrorDemo2.java
>javac RuntimeErrorDemo2.java
>Exit code: 0
>java -cp . RuntimeErrorDemo2
Statement to be executed before performing division operation
Exception in thread "main" java.lang.ArithmeticException: / by zero
    at RuntimeErrorDemo2.main(RuntimeErrorDemo2.java:14)
>Exit code: 1
```

Figure 10.6 : Output of the code shown in figure 10.5

As we are trying to perform 'division by zero', JVM will terminate the program immediately when it encounters the statement that performs division operation.

Exception Handling

An exception is an error condition. Exception handling is an object-oriented technique for managing errors. While performing exception handling, we try to ensure that the program does not terminate abruptly nor does it generate unexpected output. In this section, we will learn how to handle the exceptions.

Java uses keywords like try, catch and finally to write an exception handler. The keywords try, catch and finally are used in the presence of exceptions, these keywords represent block of statements.

- A try block contains the code that may give rise to one or more exceptions.

- A catch block contains the code that is intended to handle exceptions of a particular type that were created in the associated try block.
- A finally block is always executed before the program ends, regardless of whether any exceptions are generated in the try block or not.

Let us understand these blocks one after another in detail.

The try block

The try statement contains a block of statements within the braces. This is the code that we want to monitor for exceptions. If a problem occurs during its execution, an exception is thrown. Each type of problem (exception) corresponds to an object in Java. A try block may give rise to one or more exceptions. The syntax of the try block is shown below :

```
try
{
   // Set of statements that may generate one or more exceptions
 }
```

Let us see the code that is placed within a try block; it creates a single exception, which we have already seen earlier. The code shown in figure 10.7, when executed will terminate the program as there is no exception handler. The part of program that may lead to runtime error must be written within the try block.

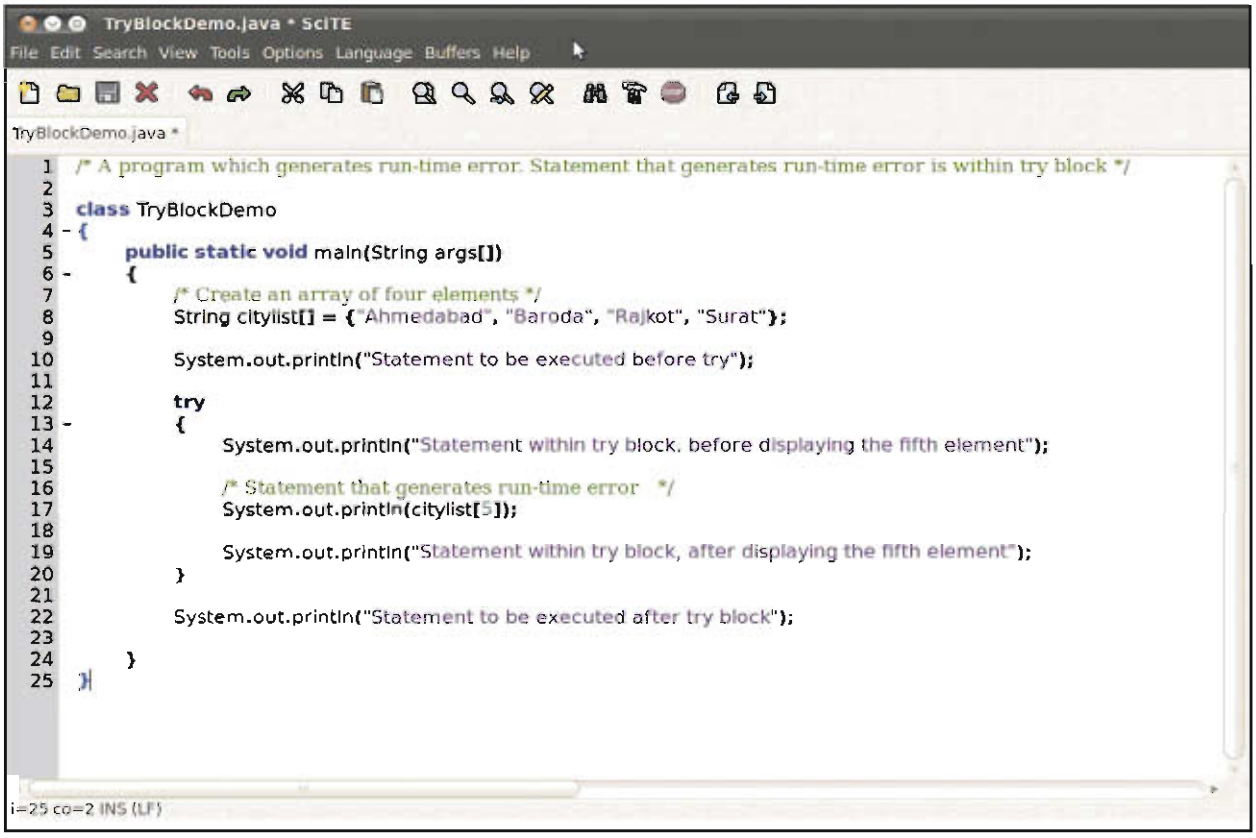


Figure 10.7 : A Program that illustrates try block

On compiling the program shown in figure 10.7, it results in compilation error as there has to be either a catch block or a finally block following the try block. The compilation error is shown in figure 10.8.

Figure 10.8 : Compilation error in program shown in figure 10.7

The catch block

The catch block must immediately follow the try block. It contains the code that is to be executed to handle an exception. The catch block is an exception handler, for a single try block there can be one or more catch blocks. The syntax of the catch block is shown below:

```
try
{
        // Set of statements that may generate one or more exceptions
  }
catch(Exception_Type Exception_object)
  {
        // Code to handle the exception
  }
```

A catch block consists of the keyword catch followed by a single parameter. The code to handle exception has to be written between parentheses. The parameter identifies the type of exception that the block is to deal with. Java supports various types of exceptions, in the later part of this topic we will see how different type of exceptions can be handled by using multiple catch blocks. Figure 10.9 illustrates mechanism of try-catch block.

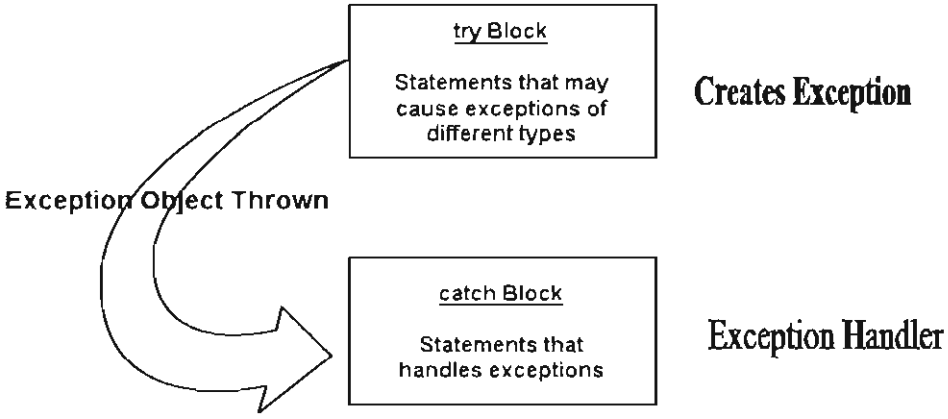


Figure 10.9 : Exception Handling Mechanism

Let us see the example program that uses appropriate exception handler. For the sake of understanding, we will handle ArrayIndexOutOfBounds exception by just catching the object within catch block. In the later sections of the chapter, we will see how the code can be handled to continue the execution of program without termination.

```
TryCatchDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
TryCatchDemo.java
1  /* A program which generates run-time error. Statement that generates run-time error is within
2     try block, there is a corresponding catch block immediately below the try block */
3
4  class TryCatchDemo
5  {
6      public static void main(String args[])
7      {
8          /* Create an array of four elements */
9          String citylist[] = {"Ahmedabad", "Baroda", "Rajkot", "Surat"};
10         System.out.println("Statement to be executed before try");
11         try
12         {
13             System.out.println("Statement within try block, before displaying the fifth element");
14
15             /* Statement that generates run-time error  */
16             System.out.println(citylist[5]);
17
18             System.out.println("Statement within try block, after displaying the fifth element");
19         }
20         catch(ArrayIndexOutOfBoundsException eobj)
21         {
22             System.out.println("Within Catch Block");
23             System.out.println("Caught Exception object of type : " + eobj);
24         }
25         System.out.println("Statement to be executed after try...catch");
26
27     }
28 }
i=28 co=2 INS (LF)
```

Figure 10.10 : A program that illustrates the use of try...catch blocks

The code shown in figure 10.10 will compile successfully and execute. In the program shown in figure 10.10, line 16 contains statement that will generate exception. At line 16, we are trying to access the fifth element of an array citylist[], however the array contains only four members. Our program tries to access array element by specifying index position that is outside the range which leads to an exception. When an exception occurs, an object of type ArrayIndexOutOfBoundsException is created and is thrown; a corresponding catch block handles the exception and does not allow the program to terminate unexpectedly. The catch block contains a reference to object "eobj" which was created and thrown by the try block. Figure 10.11 shows the output of the program.

```
TryCatchDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
TryCatchDemo.java
>javac TryCatchDemo.java
>Exit code: 0
>java -cp . TryCatchDemo
Statement to be executed before try
Statement within try block, before displaying the fifth element
Within Catch Block
Caught Exception object of type : java.lang.ArrayIndexOutOfBoundsException: 5
Statement to be executed after try...catch
>Exit code: 0
```

Figure 10.11 : Output of the program shown in figure 10.10

From figure 10.11, we can observe that the program did not terminate in the presence of exception. The statement displaying "after try…catch" gets executed.

Multiple catch blocks

In a single program multiple exceptions can occur. For instance, if we want to upload particular file to a remote computer, it may lead to two distinct exceptions - an exception may occur if the file is not present in our computer or another exception may occur if the computer is not connected to the network. There is a provision in Java to support multiple exceptions. As discussed before, the code that may generate exception should be written within the try block; apart from this there can be multiple catch blocks to handle each type of exception separately.

If the try block throws several different kinds of exceptions, we can write multiple catch blocks, each handling a specific type of exception. This helps the programmer to write separate logic for each type of exception. Figure 10.12 shows the use of multiple catch blocks.

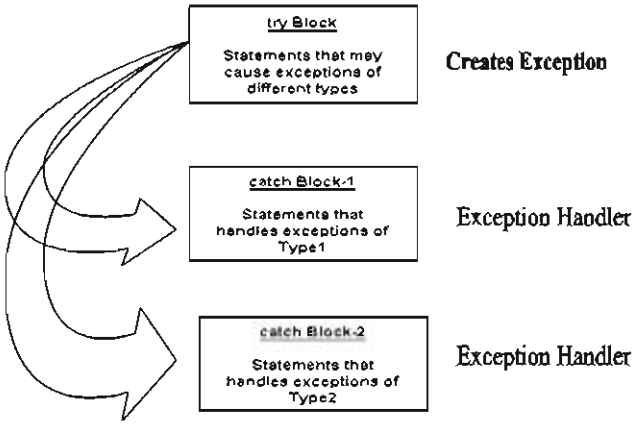


Figure 10.12 : Exception Handling Mechanism with Multiple catch Blocks

Figure 10.13 shows program where multiple catch blocks are used. Here two distinct exceptions are generated within the try block. Exception of type ArrayIndexOutOfBoundsException will be caught by the first catch block and exception of type ArithmeticException will be caught by the second catch block.

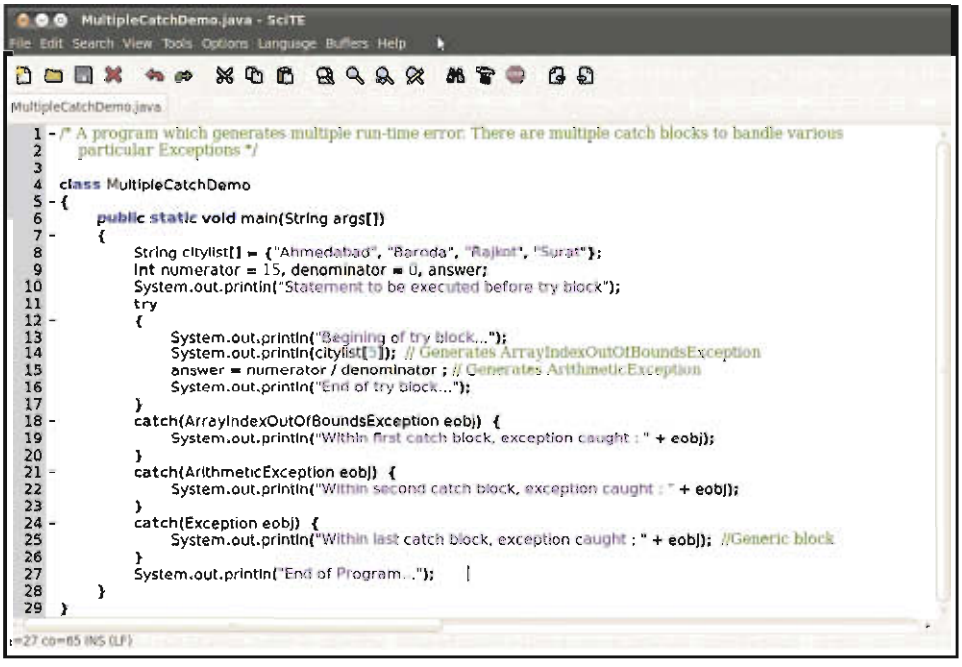


Figure 10.13 : A program that illustrates use of multiple catch blocks

The last catch block can handle any type of exception. It is a kind of default catch block and must be the last block when there are multiple catch blocks. While writing program, the order of the specific catch blocks does not matter but the default block has to be placed at the end of all catch blocks. For instance, in the program shown in figure 10.13 we can swap the occurrence of catch blocks starting at line number 18 with the one starting at line number 21; however the catch block given at line 24 must be the last block.

Multiple try blocks can be nested together but care must be taken to write a corresponding catch block for each try block.

The finally block

The finally block is generally used to clean up at the end of executing a try block. We use a finally block when we want to be sure that some particular code is to be run, no matter what exceptions are thrown within the associated try block. A finally block is always executed, regardless of whether or not exceptions are thrown during the execution of the associated try block. A finally block is widely used if a file needs to be closed or a critical resource is to be released at the completion of the program. The syntax of finally block is shown below:

```
finally
 {
    // clean-up code to be executed last
 // statements within this block always get executed even though if run-time errors
       terminate the program abruptly
 }
```

Each try block must always be followed by at least one block that is either a catch block or a finally block. Figure 10.14 shows an example of using finally block.

FinallyDemo.java * SciTE

File Edit Search View Tools Options Language Buffers Help

FinallyDemo.java *

```
1  /* A program which generates multiple run-time error. There is a finally block following try block.
2     There are no catch blocks in this program */
3
4  class FinallyDemo
5  {
6      public static void main(String args[])
7      {
8          String citylist[] = {"Ahmedabad", "Baroda", "Rajkot", "Surat"};
9          int numerator = 15, denominator = 0, answer;
10         System.out.println("Statement to be executed before try block");
11         try
12         {
13             System.out.println("Begining of try block...");
14             System.out.println(citylist[5]);  // Generates ArrayIndexOutOfBoundsException
15             answer = numerator / denominator ; // Generates ArithmeticException
16             System.out.println("End of try block...");
17         }
18         finally
19         {
20             System.out.println("This part of code will always get executed");
21         }
22
23         System.out.println("End of Program...");
24     }
25 }
```

Figure 10.14 : Program which illustrates finally block without catch block

In the program shown in figure 10.14, as there is no catch block, the program terminates abruptly due to the exception generated at line 14; statements present in line 15 and line 16 will not be executed. However, in the presence of finally block, the program executes the statements within the finally block before being terminated. The output of program is shown in figure 10.15.

```
FinallyDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
FinallyDemo.java
>javac FinallyDemo.java
>Exit code: 0
>java -cp . FinallyDemo
Statement to be executed before try block
Begining of try block...
This part of code will always get executed
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 5
        at FinallyDemo.main(FinallyDemo.java:14)
>Exit code: 1
```

Figure 10.15 : The output of program shown in figure 10.14

Let us look at an example with multiple catch blocks and a finally block all used together. The program shown in figure 10.16 contains all these blocks. It is a complete program with multiple catch blocks for corresponding exceptions being generated within the try block. Output of program is shown in figure 10.17.

```
AllBlocksDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
AllBlocksDemo.java
 1 /* A program which generates multiple run-time error. There are multiple catch blocks to handle various
 2    particular Exceptions */
 3 class AllBlocksDemo
 4 {
 5     public static void main(String args[]) {
 6         String citylist[] = {"Ahmedabad", "Baroda", "Rajkot", "Surat"};
 7         int numerator = 15, denominator = 0, answer;
 8         System.out.println("Statement to be executed before try block");
 9         try {
10             System.out.println("Begining of try block...");
11             System.out.println(citylist[5]); // Generates ArrayIndexOutOfBoundsException
12             answer = numerator / denominator ; // Generates ArithmeticException
13             System.out.println("End of try block...");
14         }
15         catch(ArrayIndexOutOfBoundsException eobj) {
16             System.out.println("Within first catch block, exception caught : " + eobj);
17         }
18         catch(ArithmeticException eobj) {
19             System.out.println("Within second catch block, exception caught : " + eobj);
20         }
21         catch(Exception eobj) {
22             System.out.println("Within last catch block, exception caught : " + eobj); //Generic block
23         }
24         finally {
25             System.out.println("This part of code will always get executed");
26         }
27         System.out.println("End of Program...");
28     }
29 }
i=19 co=1 INS (LF)
```

Figure 10.16 : Program illustrating try, catch and finally blocks

```
AllBlocksDemo.java - SciTE
File Edit Search View Tools Options Language Buffers Help
AllBlocksDemo.java
>javac AllBlocksDemo.java
>Exit code: 0
>java -cp . AllBlocksDemo
Statement to be executed before try block
Begining of try block...
Within first catch block, exception caught : java.lang.ArrayIndexOutOfBoundsException: 5
This part of code will always get executed
End of Program...
>Exit code: 0
```

Figure 10.17 : Output of the program shown in figure 10.16

From the output it is clear that the control of program execution switched from line 11 to the first catch block, later it switched to the finally block. The last two catch blocks did not execute. Although the program did not execute completely, it terminated gracefully.

A finally block is associated with a particular try block, and it must be located immediately following any catch blocks for the corresponding try block. If there are no catch blocks, then the finally block can be positioned immediately after the try block. If the finally block or catch blocks are not positioned correctly, then the program will not compile.

The throw statement

The throw keyword is used to explicitly throw an Exception object. In the example programs that we have seen so far, the JVM created an exception object and was throwing it automatically. For example, an object of ArithmeticException was created when we tried to perform a divide by zero operation and it was thrown automatically by the JVM.

Java does provide mechanism to create an Exception object and throw it explicitly. The object that we throw must be of type java.lang.Throwable, (object of Throwable class or any of its sub-classes) otherwise a compile error occurs. The syntax to throw an exception object is as follows:

throw exception_object;

When a throw statement is encountered, a search for matching catch block begins. Any subsequent statements in the try or catch block are not executed. The code in figure 10.18 shows the use of throw statement, its output is given in figure 10.19.

```
ThrowDemo.java * SciTE
File Edit Search View Tools Options Language Buffers Help
ThrowDemo.java *
1  /* A program which uses throw keyword to throw an exception object explicitly */
2
3  class ThrowDemo
4  {
5      public static void main(String args[])
6      {
7          try
8          {
9              System.out.println("Before throwing an exception object...");
10
11             /* Create an Exception object */
12             Exception myobject = new Exception("Demonstration of throw..");
13
14             throw myobject;  // throw the exception object explicitly
15
16             /* Statements written below throw will generate compile time error */
17
18         }
19         catch(Exception eobj)
20         {
21             System.out.println("Exception caught : " + eobj);
22         }
23     }
24 }
```

Figure 10.18 : Program which illustrates the use of throw keyword

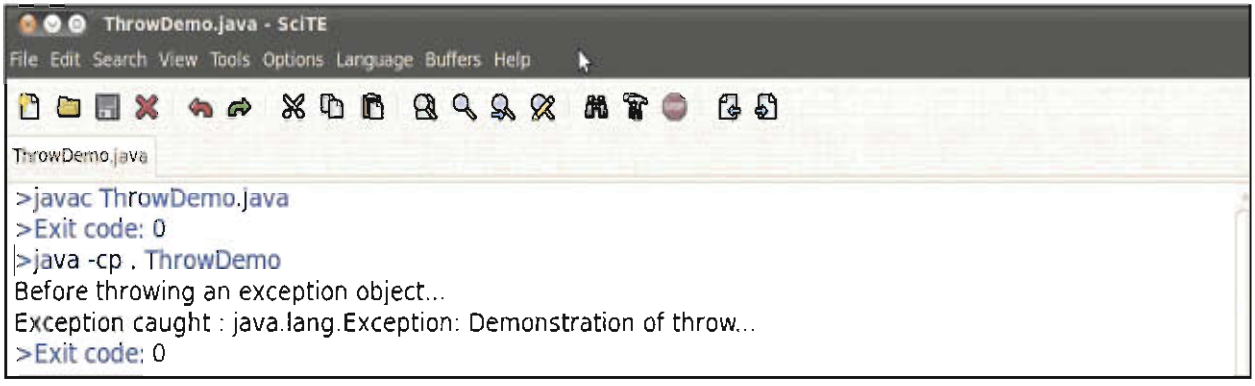


Figure 10.19 : Output of the program in figure 10.18

In the program shown in figure 10.18, we have created an object "obj" of the class Exception, the same object is thrown using throw statement. There has to be catch blocks to handle the exception object thrown explicitly.

The throws Clause

We have explored the try-catch-finally blocks, the programs we discussed so far were simple programs that didn't involve the use of methods. Few questions may arise like what will happen if an exception occurs in a method or a constructor, where will we place the try-catch blocks. There are two alternate approaches to handle exceptions created by a method :

- Write a try-catch block within the method or a constructor that may generate an exception
- Invoking a method (that may generate exception) or constructor within a try block

A throws clause can be used in a method declaration or constructor declaration to inform that the code within the constructor or method may throw an Exception. It also signifies that there is no catch block within the method that can handle the exception. When we write a constructor or a method that can throw exceptions to its caller, it is useful to document that fact. The throws keyword is used with the declaration of method.

A throws clause can be used in a method declaration as follows :

method_Modifiers return_type method_Name(parameters) throws Exception list... {

.....

// body of the method

.....

}

A method can throw multiple exceptions. Each type of exception that a method can throw must be stated in the method header. For example, a method header can be like:

performDivision() throws ArithmeticException, ArrayIndexOutOfBoundsException

{

.....

// body of the method

.....

}

The program in figure 10.20 demonstrates the use of throws clause in the presence of user defined methods. In the following code, it must be noted that if the method throws an Exception object, there must be a matching catch handler. However if we are catching an exception type within the method, there is no need to throw it.

```
ThrowsDemo.java * SciTE
File Edit Search View Tools Options Language Buffers Help
ThrowsDemo.java *
1  /* A program which uses throws keyword to throw an exception from any method */
2
3  class ThrowsDemo
4  {
5      public static void main(String args[])
6      {
7          try
8          {
9              performDivision();   // This method throws exception
10         }
11         catch(ArithmeticException eobj)
12         {
13             System.out.println("Exception caught : " + eobj);
14         }
15     }
16
17     /* Method that throws ArithmeticException object */
18
19     public static void performDivision() throws ArithmeticException
20     {
21         int ans;
22         ans = 15 / 0;
23     }
24 }
```

Figure 10.20 : Program which illustrates the use of throws keyword

In the program of figure 10.20, we have written a method performDivision(), an ArithmeticException object will be generated in this method. The method performDivision() is called in the main() method of our program. It is quite obvious that as there is no exception-handling mechanism within the performDivision() method, the exception will be caught by calling method and there has to be a handler in the calling method. Similarly, there can be exceptions within the Constructors of Java class.

Creating Custom Exceptions

Java allows creating our own exception classes according to application-specific problems. For instance, we are writing a program to generate a mark sheet in which the user is asked to enter marks of different subjects. The marks must be in the range of 0 to 100, suppose if the user enters negative marks or the value is above 100 then the program must generate an Exception. Such kind of exceptions are application specific, Java does not provide built-in exception classes for application specific exceptions.

We can create user-defined exceptions by creating a subclass of Exception class. These exceptions can be thrown explicitly using the throws statement. However it is required to catch this exception and handle it accordingly. Let us see a program code that creates a custom exception to validate the marks.

The program shown in figure 10.21 accepts input from the user. We have used java.util.Scanner class to accept input from the keyboard in line number 21. The "nextInt()" method of the Scanner class helps in reading integer input from the console. We will discuss the functionality of Scanner class in the next chapter that deals with Files and I/O.

Here we have implemented two classes. An additional class is required to create a custom exception. The class "InvalidMarksException" extends Exception class of java.lang package; it contains a single parameter constructor that accepts string to describe the type of error.

```
CustomExceptionDemo2.java - SciTE
File Edit Search View Tools Options Language Buffers Help
CustomExceptionDemo2.java
1  /* A program that creates Custom Exceptions */
2  /* Class InvalidMarksException is a user defined class which is inherited from java.lang.Exception class
3     This program wont proceed untill valid marks are entered*/
4
5  import java.util.Scanner;
6
7  class InvalidMarksException extends java.lang.Exception
8  {
9      public InvalidMarksException(String message)
10     {
11         super(message);
12     }
13 }
14
15
17 class CustomExceptionDemo2
18 {
19     public static void main(String args[])
20     {
21         Scanner kbinput = new Scanner(System.in);
22         int marks;
23         boolean continueLoop = true;
24         do {
25             System.out.println("Enter the marks : ");
26             marks = kbinput.nextInt();
27             System.out.println("You entered " + marks);
28             try
29             {
30                 if(marks < 0 || marks > 100) {
31                     throw new InvalidMarksException("Wrong marks..."); // Throw Customized Exception obj
32                 }
33                 else {
34                     System.out.println("Marks are Valid");
35                 }
36                 continueLoop = false;
37             }
38             catch(InvalidMarksException eobj)
39             {
40                 System.out.println("Exception caught : " + eobj);
41             }
42         }
43         while(continueLoop);
44     }
45 }
```

Figure 10.21 : User defined exception class

In the main method, the business logic is coded (application specific) that ensures whether the marks are in range or not. If the marks are not in range, we create an object of type "InvalidMarksException" and throw it. There has to be a catch block to handle this exception. This program will not proceed unless valid marks are entered. In case if the user enters invalid marks, he/she is asked to keep on re-entering until the input is correct. It must be noted that try-catch blocks are used within a do-while loop. The output of the code is shown in figure 10.22.

```
Terminal
File Edit View Terminal Help

$ javac CustomExceptionDemo2.java
$ java CustomExceptionDemo2
Enter the marks :
145
You entered 145
Exception caught : InvalidMarksException: Wrong marks...
Enter the marks :
-47
You entered -47
Exception caught : InvalidMarksException: Wrong marks...
Enter the marks :
78
You entered 78
Marks are Valid
$
```

Figure 10.22 : Output of the program shown in figure 10.21

Note: SciTE is an editor to type the programs. If the program accepts data from the keyboard, it is advisable to execute the program at command prompt; however, a simple program that does not require user interaction can be executed from within SciTE editor.

Advantages of Exception Handling

Throughout the chapter, we have advocated the use of exception handling in Java programs. By now, it must be clear that a good program must always handle exceptions rather than the program being terminated abruptly. Let us briefly look at the advantages of using exception handling in our Java programs. Few advantages of using exception-handling in Java programs are listed below:

- It allows us to maintain normal flow of program. In the absence of exception handling, the flow of program is disturbed.
- It allows writing separate error handling code from the normal code.
- Error types can be grouped and differentiated within the program.
- Assertions can be used to debug the program before deploying it to the clients.
- It provides an easy mechanism to log various run-time errors while executing the program.

Summary

In chapter we learnt that a program can use exceptions to indicate that an error occurred. A program can catch exceptions by using a combination of the try, catch, and finally blocks.

To throw an exception, we use the throw statement and provide it with an exception object (subclass of java.lang.Throwable class). A method that throws an uncaught, checked exception must include a throws clause in its declaration. The try statement should contain at least one catch block or a finally block and may have multiple catch blocks. Assertions are used to check that something should never happen; they are used by programmers for debugging purpose.

EXERCISE

1. What is the difference between compile-time error and run-time error ?
2. What is an Exception ? Give examples of few Exceptions found in Java.
3. How are the exceptions handled in java ?
4. What is the significance of try, catch and finally block ?
5. What is the use of throw and throws keyword ?
6. How do you create a custom Exception ?
7. Choose the most appropriate option from those given below :

   (1) Which of the following refers to an error condition in object-oriented programming terminology ?

   (a) anomaly (b) abbreviation

   (c) exception (d) deviation

(2) Which of the following is a correct word for all Java Exceptions ?

(a) Errors (b) Runtime Exceptions

(c) Throwables (d) Omissions

(3) Which of the following statements is true ?

(a) Exceptions are more serious than Errors.

(b) Errors are more serious than Exceptions.

(c) Errors and Exceptions are equally serious.

(d) Exceptions and Errors are the same thing.

(4) Which of the following elements is not included in try block ?

(a) the keyword try (b) the keyword catch

(c) the curly braces (d) statements that might cause Exceptions

(5) Which of the following block handles or takes appropriate action when an Exception occurs ?

(a) try (b) catch

(c) throws (d) handles

(6) Which of the following should be within a catch block ?

(a) finally block (b) single statement that handles Exception

(c) any number of statements to handle Exception

(d) throws keyword

(7) What will happen when a try block does not generate an Exception and you have included multiple catch blocks ?

(a) they all execute (b) only the first matching one executes

(c) no catch block executes (d) only the first catch block executes

(8) Which of the following is an advantage of using a try…catch block ?

(a) Exceptional events are eliminated

(b) Exceptional events are reduced

(c) Exceptional events are integrated with regular events

(d) Exceptional events are isolated from regular events

(9) Which of the following methods can throw an Exception ?

(a) methods with throws clause (b) methods with a catch block

(c) methods with a try block (d) methods with finally block

(10) Which of the following is least important to know if you want to be able to use a method to its full potential ?

(a) the method's return type

(b) the type of arguments the method requires

(c) the number of statements within the method

(d) the type of Exceptions the method throws

LABORATORY EXERCISE

1. Write a java program that uses a method - "add()", to add the elements in an array, include a try block within the method, so that the method must handle ArrayIndexOutOfBoundsException.

2. In the above example, remove the try...catch block from the method. By using the throws keyword, the method that invokes "add()" method must handle the exception.

3. Write a java program that throws and catches an ArithmeticException when you attempt to take the square root of a negative value. Prompt the user for an input value and try the Math.sqrt() method on it. The application either displays the square root or catches the thrown Exception and displays an appropriate message. Save the file as SqrtException.java.

4. Write a java program to validate the birth date. Create a custom exception "InvalidBirthDateException". Ask the user to enter any date, if the birth date is after the current date then throw the "InvalidBirthDateException", also write a suitable code to handle such Exceptions.

5. Write a java program to simulate bank transactions. Take two variables, balanceAmount and withdrawAmount. Program must assert if the withdrawAmount is less than balanceAmount.

6. Write a java program to simulate bank transactions. Take two variables, balanceAmount and withdrawAmount. Create a custom exception, "InvalidTransaction", your program must throw the "InvalideTransaction" exception if the withdrawAmount is less than balanceAmount. Write appropriate handlers for this Exception.

7. Write a java program to validate the birth date. Create three different custom exceptions like "InvalidDateException", "InvalidMonthException" and "InvalidYearException". Throw "InvalidDateException" if the date is negative or greater than 30, throw "InvalidMonthException" if the month is negative or greater than 12, throw "InvalidYearException" if the year is below 1950 or after the current year.

File handling 11

In this chapter, we will explore how to access, identify and manipulate files and directories on the hard disk through java programs. Through out the chapter we will focus on input/output processing and file handling. We will use the most common classes available in java.io package and few classes of java.util package. Let us begin the discussion with an overview of files and directories.

Understanding Computer Files

Storage devices of a computer system can be broadly classified into two categories: volatile storage and non-volatile Storage.

Volatile storage is temporary; values stored in variables are lost when a computer is shutdown. A Java program that stores a value in a variable uses Random Access Memory (RAM). Apart from variables, objects and their references are generally stored in RAM, once the program terminates or the computer shuts down, the data is lost.

Non-volatile storage is permanent storage; data is not lost when a computer loses power. When a Java program is saved on a disk, we are using permanent storage. A computer file is a collection of data stored on a non-volatile device. Files exist on permanent storage devices, such as hard disks, USB drives, optical disks and compact discs. Data stored in files is often called persistent data.

Files can be further classified broadly into two categories: text files and binary files.

Text files contain data that can be read in a text editor because the data has been encoded using a scheme such as ASCII or Unicode. Text files can be data files that contain facts, such as a payroll file that contains employee numbers, names, and salaries; or some text files can be program files or application files that store software instructions. Files created through editors like gedit, vi, pico are example of text files. They may have extensions like txt, java or c.

Binary files contain data that has not been encoded as text. Their contents are in binary format, which means the data is accessed in terms of bytes. Some example extensions of binary files are jpeg, mp3 and class.

Java language supports various operations that can be performed on file or on directories. Few operations that can be performed on files using java programs are listed below :

- Determining the path of a file or a directory
- Opening a file
- Writing to a file
- Reading from a file
- Closing a file

- Deleting a file
- Querying the attributes of a file

Java provides built-in classes that contain methods to help us with these tasks. These classes are present in java.io package. Java uses the concepts of streams and it provides two different categories of java classes to perform I/O operations on bytes and characters. A detailed discussion of reading from files and writing to files will be covered in the subsequent sections.

File Class in Java

The java.io.File class encapsulates information about the properties of a file or a directory. File class can be used to access attributes of files and directories. We can create/rename/delete a file or a directory. We can also access the attributes of a file like its permissions, length of a file, or last modification time. The creation of a file object that belongs to File class does not imply that the file or directory exists. A File object encapsulates a pathname or reference to a physical file or a directory on the hard disk.

There are nearly 30 methods of File Class that can be used to perform various operations on a file or a directory. However, File class does not provide any method to read from a file or write into a file, there are several stream classes to perform such operations. So, let us begin our study of few widely used constructors and methods of the File class.

Constructors of File Class

By using the File class, we can create a reference to any file by providing its absolute path in string format or by providing the relative path. The File class provides following constructors to refer a file or a directory.

File(String path)
File(String directory_path, String file_name)
File(File directory, String file_name)

Let us take an example for the above mentioned constructors. In Linux, "passwd" file present in "/etc" directory stores the information of the users existing in the system. Suppose we want to display its attributes, then its java file object can be created using three ways as mentioned below:

- By specifying the path as File fileobj = new File("/etc/passwd");
- By specfying directory and filename as two separate arguments

  File fileobj = new File("/etc", "passwd");

- By using the reference to directory encapsulated in dirobj object

  File dirobj = new File("/etc");

  File fileobj = new File(dirobj, "passwd");

Methods of File Class

The Table 11.1 summarizes few widely used methods of File class.

Method	Description
boolean exists()	Returns true if the file or directory exists, otherwise returns false
boolean isFile()	Returns true if the file exists, otherwise returns false
boolean isDirectory()	Returns true if the directory exists, otherwise returns false
boolean isHidden()	Returns true if the file or directory is hidden
String getAbsolutePath()	Returns the absolute path of the file or directory
String getName()	Returns the name of the file or directory referred by the object.
String getPath()	Returns the path to the file or directory
long length()	Returns the number of bytes in that file
String[] list()	Returns the name of files and directories in a directory
File[] listFiles()	Returns an array of abstract pathnames denoting the files in the directory.

Table 11.1 : Few widely used method of File Class

Let us now try to understand a program that lists the file names in a given directory. After creating an object of file class that refers to a particular directory, we can use the list() method to list all the files present in that directory. The program given in code listing 11.1 demonstrates the listing of files present in "/home/Akash/programs/files" directory. In the program, we have used a variable to count the number of files and directories for a given directory, an array, listOfFiles of the File class is used to store the file objects present in the directory.

```
//List the contents of a Directory
import java.io.File;
public class ListFiles
{
        public static void main(String args[])
        {
                //Provide a Directory Path
                String path = "/home/Akash/programs/files";
                String files;
                int countOfFiles;
                try
                {
```

```
                File folder = new File(path);
                //Store the list of files in an array of File[] objects
                File[] listOfFiles = folder.listFiles();
                //count the number of files in the folder
                countOfFiles = listOfFiles.length;

                System.out.print("List of files in the Directory : ");
                System.out.println(folder.getAbsolutePath());

                //Iterate to display the name of each file
                for(int i=0; i<countOfFiles; i++)
                {
                        if(listOfFiles[i].isFile())
                        {
                                files = listOfFiles[i].getName();
                                System.out.println(files);
                        }
                }
        }
        catch(Exception eobj)
        {
                System.out.println(eobj);
        }
    }
}
```

Code Listing 11.1 : Program to list files in a given directory

The output of code listing 11.1 is shown in figure 11.1

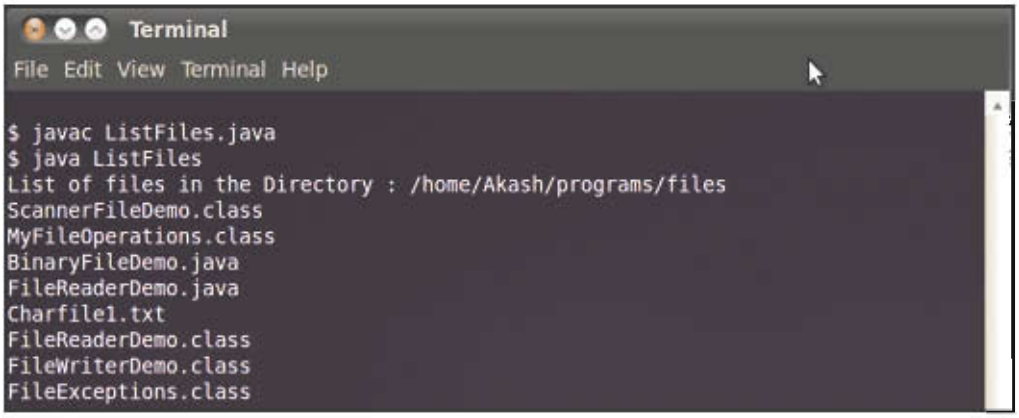


Figure 11.1 : Output of Code Listing 11.1

Note : The above code lists files and directories present in the "/home/Akash/programs/files" directory. You may use appropriate file name and path to see the output.

Introduction to Streams

Till now, we haven't seen how to modify a file or display the contents of a file. To perform such operations we need to understand the concept of streams. Java uses stream classes to carry out read and write operations on files.

We have already studied the types of devices used for input and output. For instance, keyboard is an input device while monitor is an output device. Hard disk can be classified as both input and output device as we can store and read data from the files. There are various devices available in the market from different manufacturers and different capabilities. For example, hard disk are manufactured by various companies and come with different storage capacities like 500GB or 1 TB. Apart from this, hard disk can be connected using different cables like USB or SATA. However, a Java programmer does not need to worry about the technical details like type of hard disk or its capacity while developing a program to perform read/write operations over the files. This is possible because java language provides functionality of streams.

A stream is an abstract representation of an input or output device that is used as a source or destination for data. We can visualize a stream as a sequence of bytes that flows into the program or that flows out of our program. We can write data or read data using streams.

When we write data to stream, the stream is called an output stream. The output stream can transfer data from the program to a file on a hard disk or a monitor or to some other computer over the network. An input stream is used to read data from an external device to the program; it can transfer data from keyboard or from the file on a hard disk to the program.

The main reason for using streams for input or output operations is to make our program independent of the devices involved. Two advantages of streams are:

- Programmer does not need to worry about the technical details of the device
- The program can work for a variety of input/output devices without any changes to the source code.

Stream Classes in Java

To understand byte streams and character streams, let us first try to differentiate between character and byte representation. Let us take an example of number "5"; we can represent the number in two different ways as shown in table 11.2.

Representation	Particulars	Binary Representation
Character 5	ASCII Value : 53	00110101
Binary Number 5	Binary Value : 5	00000101

Table 11.2 : Character and Binary Representation

A character is generally stored using ASCII or Unicode format but when it is used for calculation purpose; its binary value is meaningful. Let us take one more example, the statement int i = 32; declares 'i' as an integer type variable that stores number 32. However 32 can be represented as two seperate characters '3' and '2'. The character representation is advisable when we write sentences like "Human beings have 32 teeth", where we are not performing any kind of operations over the number. But when we apply numerical calculations, we use data types like int, float or double that allows us to store the numbers in binary format.

Java supports two types of streams, byte stream and character stream. Streams that transfer data in the form of bytes to the file or devices are known as byte stream or binary stream. The files that are created using byte stream are known as binary files. Suppose if we wish to store variables like integer, double or boolean into a file, then we must use binary files. Binary files can also be used to store arrays or objects. Similarly text files and program codes are created using character stream. They can be opened in text editors like vi or SciTE.

Java provides two set of classes, character stream class and binary stream class. Character Stream classes present in java.io package deal with the character/text data while byte stream classes present in java.io package deal with binary data. Figure 11.2 shows the classification of stream classes.

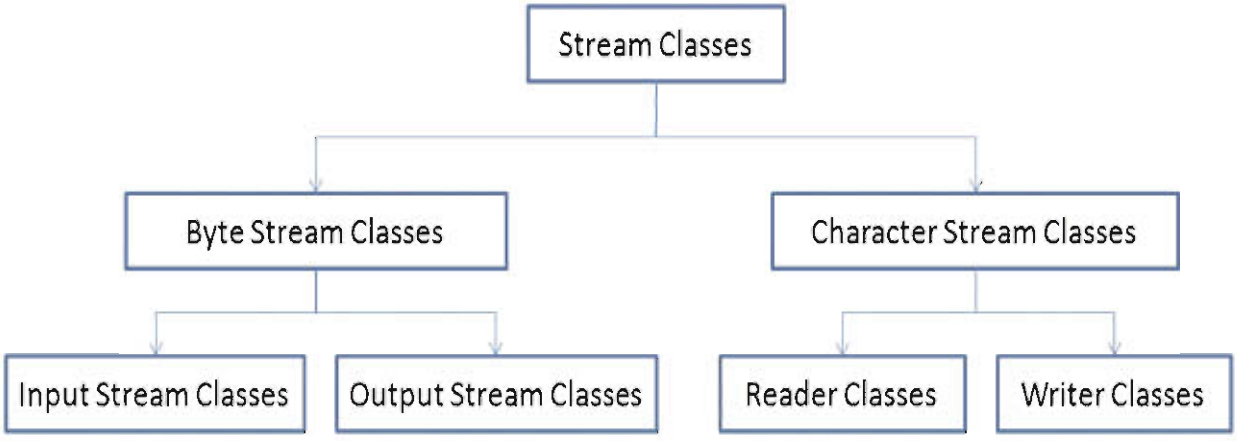


Figure 11.2 : Classification of Stream Classes

Java streams can be classified into two basic types, namely input stream and output stream. An input stream reads data from the source (file, keyboard) while the output stream writes data to the destination (file, output device).

The java.io package contains a collection of stream classes that support reading and writing in a file. To use these classes, a program needs to import the java.io package. Although there are many classes to perform character and byte operations, we will discuss the most widely used classes. Discussion of each class belonging to java.io package is out of the scope for this textbook.

Character Stream Classes

Character stream classes are a group of classes available in java.io package. They can be used to read and write 16-bit Unicode characters. Character stream classes can be further classified into Reader and Writer classes. Reader classes are a group of classes designed to read characters from files. The Writer classes are a group of classes designed to write characters into a file. Figure 11.3 shows the hierarchy of Character Stream Classes.

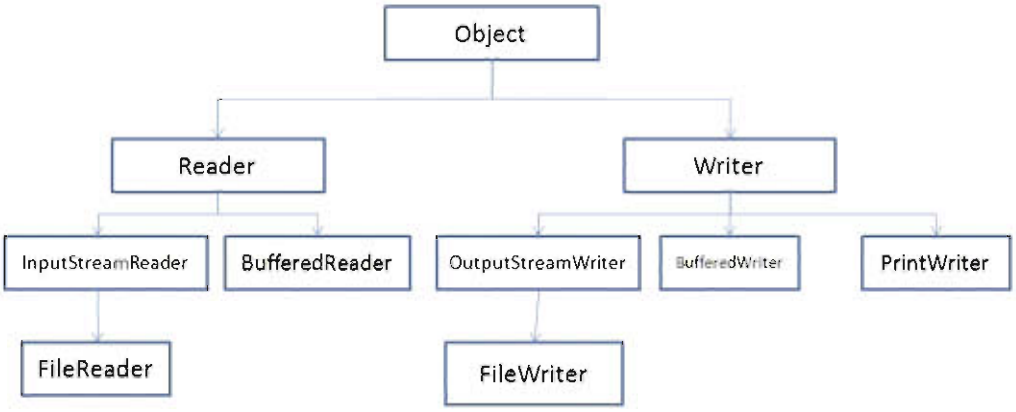


Figure 11.3 : Hierarchy of Character Stream Classes

As seen in the figure 11.3, java.io.Reader class and java.io.Writer class are inherited from the Object class. They are abstract classes (classes that cannot be used to create an object) and come with set of methods to be implemented by its subclasses. InputStreamReader and BufferedReader are the subclass of Reader Class. FileReader class is the subclass of InputStreamReader class. Similarly, OutputStreamWriter, BufferedWriter and PrintWriter are the subclasses of Writer class. FileWriter class is the subclass of OutputStreamWriter class.

Now, let us understand the constructor and methods of some of the above mentioned classes. A detailed description of methods, constructors can be obtained from the online Java documentation at http://docs.oracle.com/javase/6/docs/api/.

Writer Classes

Writer class is the base class for writing a character stream. The abstract Writer class defines the functionality that is available for all character output streams. We will be using the FileWriter class in our program to perform write operations.

The methods of Writer class can throw IOException. An IOException occurs when there is a failed I/O operation. IOException is a checked exception, so we must take care of it otherwise there will be a compiling error. Table 11.3 lists few of the methods of Writer class; these methods are used by its subclasses.

Method	Description
void close()	Closes the stream
void write(int c)	Writes the lower 16 bits of 'c' to the stream
void write(String s)	Writes string 's' to the stream

Table 11.3 : Few methods of FileWriter Class

The OutputStreamWriter class extends Writer class. It converts stream of characters to a stream of bytes. The FileWriter class extends OutputStreamWriter and outputs characters to a file. Some of its constructors are :

FileWriter(String filepath) throws IOException

FileWriter(File fileobj) throws IOException

FileWriter(String filepath, boolean append) throws IOException

In the above constructors, the parameter filepath is the full path name of a file and fileobj is a File class object that describes the file. In the last constructor, if append is true, characters are appended to the end of file, otherwise the existing contents of the file are overwritten. For example we can create an object of FileWriter as shown below:

```
FileWriter  fwobject = new FileWriter("/java/files/Charfile1.txt");
```

Let us take an example that illustrates how to write to a file. We will assume that the files are saved in the working directory. The code listing 11.2 shows how to create a file "Charfile1.txt" that does not exist. Further, we write few lines into that file using the write() method. Methods like write() and close() used in this program are inherited from the Writer class. Output of the code listing is shown in figure 11.4, after execution of the program, we use the cat command to display contents within the newly created file "Charfile1.txt".

```
// Write to a file using character stream
import java.io.*;
class FileWriterDemo
{
        public static void main(String args[])
        {
                FileWriter fwobject = null;
                try   {

                        // Create an object of FileWriter
                        fwobject = new FileWriter("Charfile1.txt");

                        //Write strings to the file
                        fwobject.write("File writing starts...");

                        for(int i = 1; i < 11 ; i++)
                                fwobject.write("Line : " + i + "\n");

                        fwobject.write("File writing ends...");
                }
                catch(Exception eobj)
                {
                        System.out.println(eobj);
```

```
                }
                finally
                {
                        try {
                        // Close the filewriter
                        fwobject.close();
                        }
                        catch(Exception eobj)
                        {
                                System.out.println(eobj);
                        }
                }
        }
}
```

Code Listing 11.2 : Program to illustrate file write operation

It is important to close the stream object after writing to a file is accomplished. Open files consume system resources, and depending on the file mode, other programs may not be able to access them. It's important to close files as soon as the operations are over.

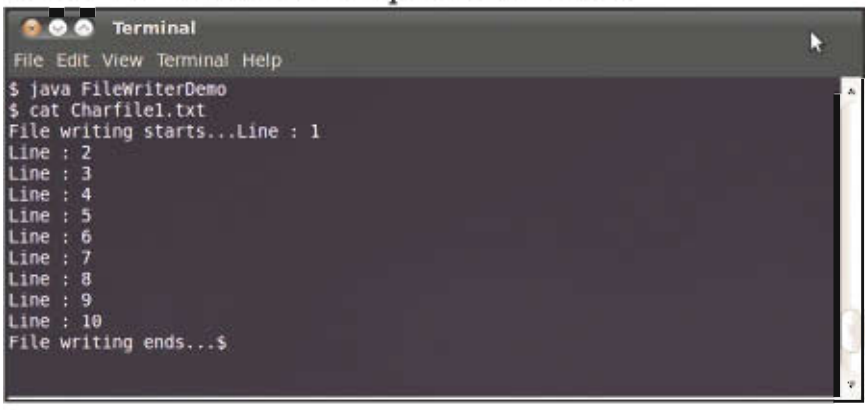


Figure 11.4 : Output of code listing 11.2

Reader Classes

Reader class is the base class for reading a character stream. The abstract Reader class defines the functionality that is available for all character input streams. We will be using the FileReader class in our program to perform read operations. Table 11.4 lists few of the methods of Reader class; these methods are used by its subclasses.

Method	Description
void close()	Closes the stream
int read()	Reads next available character from the stream, it returns "-1" to indicate the end of stream.

Table 11.4 : Few methods of FileReader Class

The InputStreamReader class extends Reader class. It converts a stream of bytes to a stream of characters. The FileReader class extends InputStreamReader class and reads characters from a file. Some of its constructors are:

FileReader(String filepath) throws FileNotFoundException

FileReader(File fileobj) throws FileNotFoundException

We can create an object of FileReader as shown below:

FileReader frobject = new FileReader("/java/files/Charfile1.txt");

In the program shown in code listing 11.2, we attempted to write to a file "Charfile1.txt". Now, let us write a program that reads from the file that we have already created. The program shown in code listing 11.3 reads data from the file using read() method of FileReader class. This read method is inherited from the Reader class.

```
// Reading from a file using character stream
import java.io.*;
class FileReaderDemo
{
        public static void main(String args[])
        {
                FileReader frobject = null;
                try  {

                        // Create an object of FileReader
                        frobject = new FileReader("Charfile1.txt");
                        int i;
                        char ch;
                        while ( ( i = frobject.read()) != -1)
                        {
                                ch = (char) i ;
                                System.out.print(ch);
                        }
                }
                catch(Exception eobj)
                {
                        System.out.println(eobj);
                }
                finally
                {
```

```
                    try {
                    // Close the filewriter
                    frobject.close();
                }
                catch(Exception eobj)
                {
                    System.out.println(eobj);
                }
            }
        }
    }
```

Code Listing 11.3 : Program to illustrate file read operation

Figure 11.5 shows the output of code listing 11.3, here it can be noticed that the execution of program displays the output on the screen. It displays all the contents of the file "Charfile1.txt".

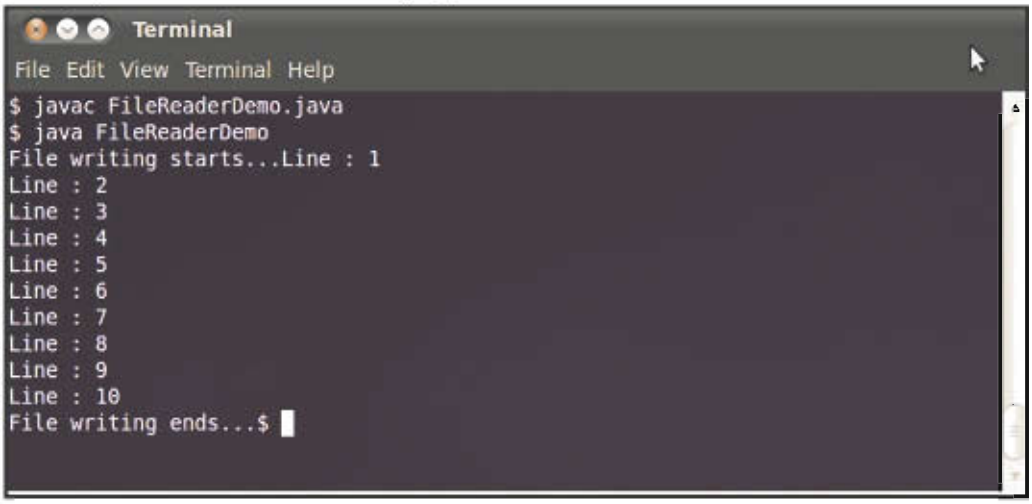


Figure 11.5 : Output of code listing 11.3

There is a special symbol to identify End of File (EOF). While reading from a file, program must identify that the file has ended. However the program reads from input stream so java read() method returns "-1" to identify the end of data in the stream.

Byte Stream Classes

Till now, we have seen how to process characters using java.io package. Most of the real life applications require numeric calculations like storing the details of inventory in a file and calculating the costs involved or like storing the employee details and their salaries in a file. For such kind of processing, java provides set of binary streams and their associated classes.

The FileInputStream and FileOutputStream classes in the java.io package give us the ability to read and write bytes from and into any files in the disk. These classes are the sub-classes of InputStream and OutputSream classes.

FileOutputStream

The FileOutputStream is a subclass of OutputStream and is used to write bytes to the file or some output stream. To use this class and its methods, first we need to create a file object, then we could use the write method derived from the abstract class OutputStream to write byte data into a file. Few widely used methods of FileOutputStream classes are shown in table 11.5:

Method	Description
void close()	Closes this file output stream and releases any system resources associated with the stream.
void write(int b)	Writes the specified byte to this file output stream.
void write(byte[] b)	Writes b.length bytes from the specified byte array to this file output stream.

Table 11.5 : Few methods of FileOutputStream Class

The constructors of FileOutputStream can accept either a string containing the path to the file location or an object of the File class. Let us see few constructors that are normally used :

FileOutputStream(String name) throws FileNotFoundException

Or

FileOutputStream(File file) throws FileNotFoundException

Examples of creating instances of FileOutputStream are as shown below:

```
FileOutputStream fosobject = new FileOutputStream("/home/Akash/myfile.txt");
```

Alternatively we can also use

```
File fobj = new File("/home/Akash/myfile.txt");
FileOutputStream fosobject = new FileOutputStream(fobj);
```

The above listed constructors can throw FileNotFoundException, if the file name refers to directory rather than a regular file, file does not exist, or file cannot be opened for some reason.

FileInputStream

FileInputStream is a subclass of InputStream and is generally used to read byte data from the files. It provides set of methods to perform write operations over the files as shown in table 11.6.

Method	Description
void close()	Closes this file input stream and releases any system resources associated with the stream.
int read()	Reads a byte of data from this input stream.
int read(byte[] b)	Reads up to b.length bytes of data from this input stream into an array of bytes.

Table 11.6 : Few methods of FileInputStream Class

Let us now see a program that uses the above described classes to perform write and read operations over a file as shown in code listing 11.4.

```
//Program to read and write bytes to binary file
import java.io.*;
class BinaryFileDemo {
        public static void main(String args[])
        {
                FileOutputStream outobject = null;
                FileInputStream inobject = null;
                String cities = " Rajkot \n Ahmedabad \n Vadodara \n Vapi \n";

                //Convert cities into byte array
                byte citiesarray[] = cities.getBytes();

                try {
                        // Create object of Binary output stream
                        outobject = new FileOutputStream("Binaryfile.dat");
                        //Write the array of bytes into file
                        outobject.write(citiesarray);
                        outobject.close();

                        //Create object of Binary input stream
                        inobject = new FileInputStream("Binaryfile.dat");
                        //Variable to read each byte
                        int i;
                        //Read each byte from the file and display
                        while((i = inobject.read())!=-1)
                        {
                                System.out.print((char)i);
                        }
                        inobject.close();

                }
                catch(Exception eobj)
                {
                        System.out.println(eobj);
                }
        }
}
```

Code Listing 11.4 : Program to read and write bytes to binary file

The program shown in code listing 11.4 writes a string to a file "Binaryfile.dat", later it creates an object of FileInputStream to read the data from the same file and display it on the screen. We must note that in the above program, typecasting is applied to convert integer to character after read operation is performed.

Note: to observe the difference between text file and binary file, open the files created using the programs given in code listing 11.2 and 11.4.

Processing Input from Keyboard

In this section, let us explore the different ways to input the data to java program through the keyboard. As we know, a program can get input of data from live interaction through keyboard/GUI or it may take input as command line arguments or from files. Although there are many classes that facilitate the input from keyboard, here we will discuss two of the most widely used technique, using Scanner class of java.util package, and using the Console class of java.io package.

Scanner Class

Scanner class belongs to the java.util package; it provides various methods to read input from the keyboard or from the file. A special feature of this class is that it breaks the input string into tokens (words) using a delimiter. (White space is the default delimiter). Each token can be of different type, for example a string like "India-1947" can be read as "String-int" values. Let us explore the constructors and few methods of the Scanner class.

Scanner(String str)

Scanner(InputStream isobject)

Scanner(File fobject) throws FileNotFoundException

A Scanner object can be created from a string, file object or InputStream object. For instance we may use the constructor in the following way to read from a file and keyboard respectively:

```
Scanner fileinput = new Scanner(new File("Students.dat"));

Scanner kbinput = new Scanner(System.in);
```

Some of the important methods available with the Scanner class are listed in table 11.7.

Method	Descrption
void close()	Closes the Scanner
String next()	Returns the next token
boolean hasNext()	Returns true if there is a token in input
int nextInt()	Scans the next token of the input as Int.
float nextFloat()	Scans the next token of the input as Float.
String nextLine()	Scans the next token of the input as Line

Table 11.7 : Methods of Scanner Class

Let us now see a program that reads two numbers interactively from the user and displays the addition of those two numbers. Code listing 11.5 shows the program.

```
//Accepts input at command prompt
import java.io.*;
import java.util.*;
class ScannerInputDemo
{
        public static void main(String args[])
        {
                Scanner kbinput = null;
                int number1;
                int number2;
                int sum=0;
                try
                {
                        // Create an object of Scanner class
                        // that reads from Standard Input
                        kbinput = new Scanner(System.in);
                        System.out.println("Enter the first number : ");
                        //Read the integer number from console
                        number1 = kbinput.nextInt();
                        System.out.println("Enter the second number : ");
                        //Read the integer number from console
                        number2 = kbinput.nextInt();
                        sum = number1 + number2;
                        System.out.println("Sum is : " + sum);
                }
                catch(Exception eobj)
                {
                        System.out.println(eobj);
                }
                finally {
                        try {
                                kbinput.close();
                        }
                        catch(Exception eobj)
                        {
                                System.out.println(eobj);
                        }
                }

        }
}
```

Code listing 11.5 : Program to add two numbers

In the code listing 11.5, we have created an object of Scanner class, the constructor of Scanner class accepts "System.in" as an argument so that it reads from the standard input (keyboard). Both the numbers are scanned as integer numbers using the nextInt() method of the Scanner class. Figure 11.6 shows the output of the program.

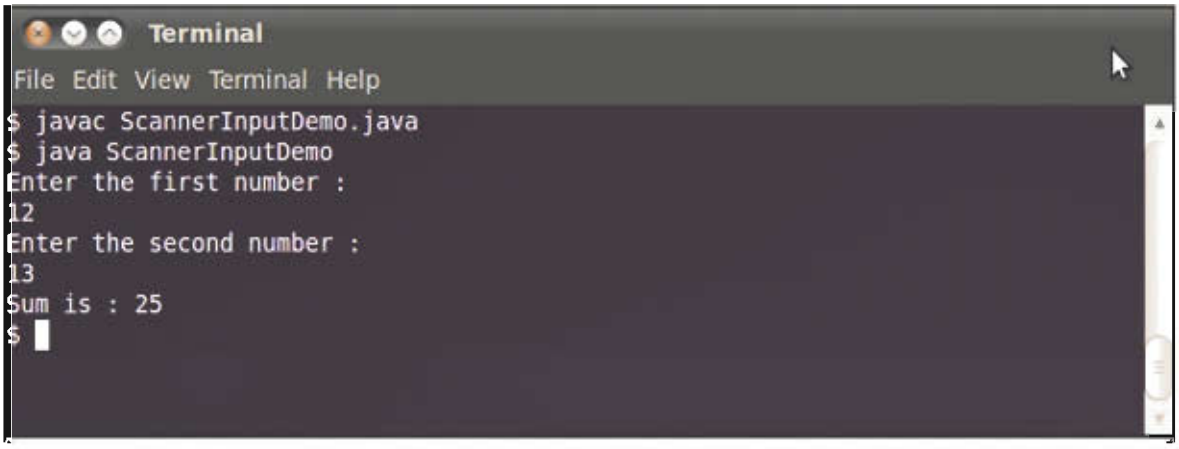


Figure 11.6 : Output of Code Listing 11.5

Scanner class can also be used to read from a file. Let us see an example where we use the Scanner class to read the data from a file that contains information about few students. It is assumed that the file "Students.dat" already exists. It contains five fields, student_no, student_name, marks of three subjects. The format and data of the file Students.dat is as shown below:

1 Akash 45 65 55

2 Badal 10 20 30

3 Zakir 45 40 60

4 David 65 50 75

We will write a program as shown in code listing 11.6 to read the data of each student and perform operations like calculating the total marks and displaying them on the output.

```
//Accepts input from a file "Students.dat" and calculate the total marks of each student
import java.io.*;
import java.util.*;
class ScannerFileDemo
{
        public static void main(String args[])
        {
                Scanner fileinput = null;
                int rollno, mark1, mark2, mark3, totalmarks;
                String name = null;
                File fobject;
        try
        {
```

```
                // Specify the file from where data is to be read
                fobject = new File("Students.dat");
                // Create an object of Scanner class that reads from File
                fileinput = new Scanner(fobject);
                //Display the default Delimiter to separate fields within a file
                System.out.println("Default delimeter is : " + fileinput.delimiter() + "\n");
                // Iterate to read the values of each record
                while(fileinput.hasNext()) {
                rollno = fileinput.nextInt();
                name = fileinput.next();
                mark1=fileinput.nextInt();
                mark2=fileinput.nextInt();
                mark3=fileinput.nextInt();
                totalmarks = mark1 + mark2 + mark3;
                System.out.println("Total marks of Rollno " + rollno + " ," +name+ " are
                                                                    : " + totalmarks);
                }
                fileinput.close();
        }
        catch(Exception eobj)
        {
                System.out.println(eobj);
        }
    }
}
```

Code Listing 11.6 : Program to calculate the total marks of each student

The output of the program is displayed on the monitor as shown in figure 11.7.

Figure 11.7 : Output of code Listing 11.6

Console Class

In the previous example, we used the Scanner class to read user input. Apart from the Scanner class, there is another class java.io.Console which can be used to get the input from the keyboard. We use this class especially when the input is to be typed in hidden form (characters must not be echoed on screen).

The Console class provides a method for reading password. When reading the password the user input will be hidden or not shown in the console screen. And it will return an array of character as the return type. The Console class belongs to java.io package. Few widely used methods of Console class are listed in table 11.8:

Methods	Description
String readLine()	This method reads a single line of text from the console.
char[] readPassword()	This method reads a password or passphrase from the console with echoing disabled.
Console printf(String format, Object args)	This method is used to write a formatted string to this console's output stream using the specified format string and arguments.

Table 11.8 : Few Methods of Console Class

The program shown in code listing 11.7 demonstrates the use of Console class to read username and password, further, it validates whether the username and password are correct or not.

```
// Program to reading passwords
import java.io.Console;
import java.util.Arrays;
public class ConsoleDemo {

    public static void main(String[] args) {
     Console console = System.console();
    String username = console.readLine("Username: ");
     char[] password = console.readPassword("Password: ");

   if (username.equals("admin") &&  String.valueOf(password).equals("secret")) {
      console.printf("Welcome to Java Application \n");
     } else {
     console.printf("Invalid username or password.\n");
      }
   }
}
```

Code Listing 11.7 : Program to read username and password

The output of the program is shown in figure 11.8. We have used two attempts to input the username and password. In the first attempt the username is false and that is why we use invalid username message. In the second attempt, we entered the correct username and password.

```
Terminal
File Edit View Terminal Help
$ javac ConsoleDemo.java
$ java ConsoleDemo
Username: Akash
Password:
Invalid username or password.
$ java ConsoleDemo
Username: admin
Password:
Welcome to Java Application
$
```

Figure 11.8 : Output of Code Listing 11.7

The program can be extended to read list of users and match their passwords from a file in the presence of multi-user environment.

Apart from various classes discussed in this chapter, java provides classes to store and retrieve objects using files. It also provides classes and methods to access a file randomly. Till now, we have seen the methods that perform sequential operations; however, there are classes which allow us to directly jump to nth record instead of accessing it sequentially. Discussion of these classes being out of scope for this text, we leave it to the students for exploring various other interesting features of java.io package.

Summary

In this chapter we learnt about file handling operations. We saw how to use the java.io.File class to perform file operations. We learnt the concept of stream and saw how input and output streams of different types can be used. We also learnt how to use the Scanner class to access data from a keyboard or a file. Finally we learnt how to use the Console class to input data from keyboard.

EXERCISE

1. Why is a file important in java programming? Under what circumstances will you store the data in files ?
2. State various operations that can be performed on a file and directory.

3. Why is the concept of Streams introduced in java, give the advantages of using streams.

4. Choose the most appropriate option from the following :

(1) Which of the following statements is true ?

(a) Volatile storage lasts only a few seconds.

(b) Volatile storage is lost when a computer is shutdown.

(c) Computer disks are volatile storage devices.

(d) All of the above are true.

(2) Which of the following refers to a collection of data stored on a nonvolatile device in a computer system ?

(a) file (b) application (c) volatile data (d) hard-disk

(3) The data hierarchy occurs in which of the following order from the smallest to largest piece of data ?

(a) file:character:field:record (b) file:character:record:field

(c) character:field:file:record (d) character:field:record:file

(4) Which of the following is true about streams ?

(a) Streams always flow in two directions

(b) Streams are channels through which the data flow

(c) Only one stream can be open in a program at a time

(d) All of the above are true

(5) Which of the following is used as a separator between fields of a record ?

(a) path (b) delimiter (c) variable (d) space

(6) Scanner class can be used to for performing which of the following operations ?

(a) accept input from the keyboard (b) read from the file

(c) parse a string separated by delimiters (d) All of the above

LABORATORY EXERCISES

1. Write a java program to list all the files in a given directory with ".txt" extension.

2. Write a java program to enter a filename, your program must check whether the file exists or not, if it exists then display the properties of that file.

3. Write a java program to create a file "friends.dat"; write the name of your friends in that file using Writer classes.

4. Write a java program in which a user enters the filename, further copy the file into another file.

5. Write a java program to copy the file /etc/passwd into your directory with the name mypasswd.dat.

6. Write a java program to count the number of characters in a given file.

7. Write a java program to display the size of file in terms of bytes.

8. Write a java program to calculate the simple interest, the user must enter principal amount, rate of interest and period in years.

9. Write an interactive program to convert inches into centimeter and reverse by entering the data at console.

Publishing documents using LaTeX 12

We have learned how to use OpenOffice.org Writer to create documents. In this chapter we shall discuss the TeX and LaTeX typesetting software. We shall discuss the advantages of using LaTeX. Then we will learn how to use LaTeX using TeX Live, a software package of the TeX/LaTeX software combination along with the SciTE text editor.

Using LaTeX

To use LaTeX, we need any LaTeX distribution (software). Most include TeX and some additional software in them. TeX Live is a very popular LaTeX distribution available in the standard Ubuntu repositories. We also need a plain text editor and software to view the output file. LaTeX can produce output in different file formats, and depending on the output file format, corresponding viewer software is needed.

LaTeX documents are typically created using any plain text editor (like gedit or SciTE). The different parts of the text are marked using LaTeX commands that associate a meaning to them. For example, **\title** is used to define the document's title, **\author** to specify the author(s) of the document and **\date** to indicate the date the document was created. Similarly, **\chapter, \section, \subsection, \paragraph** can be used to explicitly specify the logical structure of the document.

LaTeX comes with built-in ways to format these document elements in a pleasant-looking professional style. However, when typing the document, we will only see it as plain unformatted text with these commands written as part of the text. The document is then compiled (processed) using the LaTeX system and an output file is produced. LaTeX may also produce some additional files. In most cases, these additional files can be deleted safely without losing any information.

When we view the output file using appropriate software or print it on a printer, we see the formatted document. If we are not satisfied with the looks of the document, we may customize the built-in styles or define our own styles. Every time we make some modification in the source text, we need to compile it again for seeing the effect of our change on the output document.

Both TeX and LaTeX use the file extension .tex; LaTeX now has a command called pdflatex that produces the popular PDF (Portable Document Format) format files. PDF files can be viewed on screen as well as printed to printers and the printout looks exactly same as the monitor display. PDF files are very popular for sharing printable documents on the Web. PDF documents can be viewed in Ubuntu's default document viewer, evince. Hence the edit-compile-view cycle becomes

- Edit the document using any plain text editor like gedit
- Compile the document by issuing the command ***pdflatex filename*** at the command prompt (in the directory where the tex file was saved)
- View the generated PDF file by either opening it from the GUI or by issuing the command ***evince pdffilename*** at the command prompt (the terminal will display the next prompt only when you close the PDF file)

We may also use the SciTE editor for editing LaTeX files. While both gedit and SciTE have syntax highlighting (displaying different language elements in different colors for easy identification and readability), SciTE has one advantage over gedit - one can compile and view the document from within the SciTE program itself. As you are already familiar with SciTE, we shall use SciTE as the editor for learning LaTeX.

To use SciTE with pdflatex, we need to make changes to its configuration file as shown at the end of this chapter. The user needs to perform the operation only once, after installing the required software.

The LaTeX Language

LaTeX is essentially a markup language. The LaTeX source consists of plain text, with some parts of the text marked up using markers known as commands. Some commands are independent commands - they do not mark any specific part of the text. These commands can perform a variety of tasks when the document is processed by the LaTeX system - they may provide information about the text or the document, they may indicate the role of the marked text in the overall structure of the document (and hence cause LaTeX to format the text in certain way), they may directly specify formatting, or they may instruct LaTeX to process the document in a certain way (for example, use a certain page size, start a new chapter only on an odd-numbered page, etc.).

LaTeX commands start with a \ (backslash) character followed by the command name. The command name may be a string of alphabetic letters only, or may be a single non-letter. LaTeX commands are case-sensitive (capital and small letters are treated as being different). Some commands may accept additional information (for example, the \textcolor command expects the color in which the text is to be displayed). This additional information is called arguments.

There are two types of arguments. Optional arguments, as the name suggests, are not mandatory. We may or may not provide them. If we want to provide one or more optional arguments, we write them after the command name, enclosed in [ ] (square brackets) and separated by comma. These are followed by mandatory arguments (if there are any) in { } (curly braces) with each mandatory argument written in its own set of curly braces. For example, if we issue a command \documentclass[12pt]{article}; then documentclass is the name of the command, 12pt is an optional argument while article is a compulsory argument.

LaTeX treats all whitespace characters (the space, tab and newline characters) as the same. It converts all occurrences of multiple consecutive whitespace into a single space character. The white spaces at the beginning of a line are generally ignored and one or more consecutive blank lines are considered to mark the beginning of a new paragraph. This means that even if you type your text as several lines, it will appear in the output as a continuous flow unless there is a blank line in it. To insert a break in lines use \\ (the line break command) at the end of each line except the last line in the paragraph. Figure 12.1 shows an example of continuous text as well as how to insert explicit line breaks (We shall discuss the \textsf command later). Figure 12.2 shows the output of the LaTeX file.

```
\documentclass[12pt]{article}
\title{Line handling in \LaTeX}
\date{May 2013}
\begin{document}
 \maketitle
 \section{Continuous Text}\textsf{
 We have no wings,  we cannot fly
 But we have legs to sail and climb
 By slow degrees and by and by
 The cloudy summits of our time}
 \section{Text with Seperate Lines}\textsf{
 Heights by great men reached and kept \\
 Were not attained by a sudden flight \\
 But they, while their companions slept, \\
 Were toiling upwards in the night}
\end{document}
```

Figure 12.1 : Line Handling Example Source File

1 Continuous Text

We have no wings, we cannot fly But we have legs to sail and climb By slow degrees and by and by The cloudy summits of our time

2 Text with Seperate Lines

Heights by great men reached and kept
Were not attained by a sudden flight
But they, while their companions slept,
Were toiling upwards in the night

Figure 12.2 : Line Handling Example Output

The following characters are reserved characters in LaTeX;

$ % & _ (underscore) { } ^ ~ \

They have a special meaning in LaTeX. These characters cannot be used directly in our text in LaTeX. If we want to use them in our text, we must use the following forms.

\# \$ \% \& _ \{ \} \^{} \~{} \textbackslash{}

Note the special cases of the last three characters. The symbols < and > print very differently by default (except in math mode). Hence they must be written as **\textless** and **\textgreater.** The ` (grave accent or backquote) and ' (apostrophe or straight quote) are used around text to put it in single quotes, like `Book Code'. Double quotes are produced by repeating them twice, like ``Book Code'' (these are two straight quotes, not a single double quote). These may look odd in the source file, but are typeset properly in the output file.

LaTeX uses groups to mark portions of text. A group is enclosed between curly braces { and }. Any command in a group applies only to text following the command within that group. Also, some commands are followed by a group and apply to the whole group. Groups are useful for applying few commands to a small amount of text such as a part of a line, few lines or a paragraph.

For the cases where a multitude of commands must be applied (for example, for formatting a table or mathematical equation properly) or where some command(s) have to be applied to large portions of the text (like several paragraphs, whole sections), LaTeX provides a facility called environments; an environment begins with a \begin{environment-name} command and ends with a \end{environment-name} command.

All the formatting characteristics of the environment are applied to the entire text inside the environment. Environments can be nested; we can have one environment inside another. There are several standard environments meant for specific types of content, like equation, quotation, table and list that come with nice ready-made formatting commands for these specific content types.

While LaTeX documents are meant to be displayed and printed, LaTeX has several advanced features, including programming and automatically generating parts of the documents or multiple documents (mail merge). Often a complicated LaTeX template or package developed by one person or team is used by many others. Sometimes those people also need to tweak the LaTeX code.

In such cases, providing explanation for the complicated parts makes it easy for others to understand the code. Such explanation is provided in the form of comment. In LaTeX, the % character marks the beginning of a comment and everything from the % character up to the end of the line is treated as a comment. Comments are meant for the humans who read the LaTeX source code in a text editor for understanding and possibly modifying it. They are completely ignored by the compilation process and hence never make it into the output.

The Structure of a LaTeX Document

A LaTeX document has two parts namely preamble and content. The preamble contains metadata (data about data). In this case, the metadata is information about the document (for example, what kind of document it is, who is the author, when was it created.) and instructions on how LaTeX should process the document. The actual content is always inside the environment document, written between \begin{document} and \end{document}.

The Preamble

As LaTeX supports creation of a wide variety of documents, each with different characteristics and format, LaTeX needs to know what type of a document used as a source file is. The very first element in the preamble must be **\documentclass**(document-class-name} specifying the type of the document. Some common document classes are as shown in Table 12.1. Many document classes have options. Table 12.2 lists some common options.

Document Class	Purpose
article	For writing individual articles.
book	For writing entire books.
slides	For creating presentation slides. It automatically sets larger font size.
letter	For writing letters.
beamer	For generating presentations similar to office suites using the beamer package.

Table 12.1: Some Common Document Classes

Options	Function
10pt, 11pt, 12pt	Sets the size of the main font in the document to 10 points (the default), 11 points and 12 points respectively.
a4paper, letterpaper, legalpaper	Defines the paper size. These are several international standard paper sizes. The most common paper sizes in regular office use are A4, letter and legal.
fleqn	Displayed formulas and equations are flushed left (left-aligned) rather than centered (the default).
landscape	Changes the layout of the document to print in landscape mode.

Table 12.2 : Some Common Options of Some Document Classes

The document class declaration is followed by optional package declaration. While the LaTeX system itself provides for many common typesetting requirements, it also recognizes that it cannot provide everything that users may need. Hence LaTeX allows users to write packages that provide additional functionality. There is a large community of LaTeX users who develop new LaTeX packages or enhance existing ones to cater to their own needs and then share them with others over the Comprehensive TeX Archive Network (CTAN).

LaTeX distributions themselves usually come with a large number of such packages preinstalled. To use one or more packages in our document, we need to declare them in the preamble as \usepackage{package-name}. Some packages may also have options to customize their behaviour. If we are not going to use options for any of the packages, multiple package names can be declared in a single \usepackage command, separated by commas. Table 12.3 shows only a few of the commonly used packages.

Package	Description
amsmath	It contains the advanced math extensions for LaTeX originally developed for the American Mathematical Society.
color	It adds support for colored text.
easylist	Adds support for multilevel lists.
geometry	For page layout tasks like setting paper size, orientation, margins, etc.
listings	Has special features for including programming code within the document
setspace	Lets you change line spacing

Table 12.3 : Some Commonly Used Packages

Three more pieces of information are typically supplied. They are as mentioned :

\title{the-title-of-the-document}

\author{author(s) of the document}

\date{date of creation / last update of the document, in any format}

It is necessary to provide the title and the author if you want LaTeX to create an automatic title for you. Providing date is optional; if it is omitted then the date of compilation is used in the title. These pieces of information can be supplied in the preamble or as the first thing in the document environment also.

The Document Environment

The document environments for articles and slides will only have a title followed by the main content of the document. A title is automatically generated by LaTeX when it sees the \maketitle command. Of course, the \title, \author and \date commands must precede the \maketitle command because their information is used in creating the title.

A book can have a far more elaborate structure, though most elements are optional. The document environment of a book is divided into three main parts - the front matter, the main matter and the back matter, denoted by the commands \frontmatter, \mainmatter and \backmatter respectively. This structure has things like the title, the table of contents and the preface in the front matter, the bibliography, index and references in the back matter and the primary content in the form of chapters, sections and subsections in the main matter.

The main contents of a book has a hierarchical structure, where a book is divided into parts, parts are divided into chapters, chapters are divided into sections which are divided into subsections which are divided into sub subsections which are divided into paragraphs which are divided into subparagraphs. These are marked with the commands \part, \chapter, \section, \subsection, \subsubsection, \paragraph and \subparagraph respectively.

Each accepts one compulsory argument, the title; and one optional argument, the title to be displayed in the table of contents. They start a new part, chapter or section respectively. The title to be displayed in the main text may be longer and may have internal formatting (like boldface or italics applied to a part of the title), while the title to be displayed in the table of contents is expected to be short and devoid of any special formatting to maintain consistency in looks. These 7 elements nest inside one another and have an integer level assigned to them, with part having a level of 1, chapter a level of 0, section a level of 1 and so on.

The parts, chapters, sections are numbered automatically by LaTeX. Hence the author does not have to worry about it. The author can also move chapters, sections and subsections without having to bother about renumbering. The parts are numbered in Roman numerals (I, II, III, and so on.), while the chapters, sections, subsections, etc. are numbered in Arabic numerals (1, 2, 3, and so on.).

All chapters after the special \appendix command (to be used only once) are treated as appendices and get uppercase alphabetic numbering (A, B, C, and so on.). Pages in the front matter are numbered

using Roman numerals, while the pages in the main matter and back matter are numbered in Arabic numerals with the numbering restarting from 1. While both the front matter and back matter can have chapters (like preface, acknowledgments, bibliography), those chapters usually do not have sections or other sub-elements. The commands defining different elements of the document also have their starred equivalents that are not numbered. For example, \section* can be used to create a section that is not automatically numbered.

By default elements are assigned number up to level 2 that is up to subsections. Sub subsections and further divisions are not assigned numbers. This can be changed by modifying one of the built-in counters of LaTeX in the preamble. For example, the command

\setcounter{secnumdepth}{3} ensures that elements up to level 3 (subsubsection) are assigned numbers. Elements are assigned a number formed by appending a period (.) and the element number to the number of the parent element. Chapters are an exception as their assigned number does not have the part number and period in front of them. Chapters are assigned simple Arabic numeral numbers. If part II of a book has chapter 5 in it, which, in turn, has section 4 in it that has subsection 1 in it, the subsection would be numbered 5.4.1.

A well-formatted table of contents (TOC) is generated automatically from the element titles by LaTeX when it encounters the command \tableofcontents. Again, by default a TOC has entries up to level 2 (subsection), but this can be changed by altering the value of another built-in counter tocdepth.

There is one important point to be noted. LaTeX processes the source file sequentially from beginning to end in a single pass. It produces the output file also sequentially. It cannot move back and forth in either of the file. This poses a problem. The TOC comes early in the document and must output the entries for chapters or sections along with their page numbers. However, at that point LaTeX has no knowledge about chapters or sections that would follow. In this kind of situation, LaTeX must be run multiple times.

In the first run LaTeX collects information about the document structure and stores it in supplementary files. The TOC in the output files would be empty at this point or may have information from old supplementary files. In the second run, it will pick up the correct information from the supplementary files in the beginning to produce the correct TOC.

Just like the TOC, LaTeX can also maintain list of figures, list of tables, cross references, bibliography, glossary or index automatically. This offloads a great burden from the author's back and is a major reason for the popularity of LaTeX.

Example

To have an idea of how LaTeX typesets books, let us create one source file and see its output.

We will use the SciTE editor with LaTeX to work on the example.

- Create a new file in SciTE using the **File → New** menu option.
- Type the content shown in Listing 12.1 in a SciTE editor.

```
\documentclass[12pt]{book}
\usepackage{amsmath}
\title{\huge Mathematics \\[3\baselineskip]
\Large Standard 12}
\author{Gujarat State Board of School Textbooks}
\date{2013}
\setcounter{secnumdepth}{2}
\setcounter{tocdepth}{1}
 \begin{document}
 \frontmatter
  \maketitle
  \chapter{\MakeUppercase{Fundamental Duties}}
  \tableofcontents
  \chapter{\MakeUppercase{About This Textbook...}}
\mainmatter
 \part{Semester I}
 \chapter{Set Operations}
 \section{Introduction}
 \section*{Exercise 1.1}
 \section{Properties of the Union Operation}
 \subsection{Union is a Binary Operation}
 \section{Properties of the Intersection Operation}
 \subsection{Intersection is a Binary Operation}
 \subsection{Associative Law}
 \chapter{Number Systems}
 \section{Introduction}
 \section*{Exercise 2.1}
 \section{Irrational Numbers}
 \chapter{Polynomials}
 \chapter{Coordinate Geometry}
 \chapter{Some Primary Concepts in Geometry : 1}
 \chapter*{Answers}
 \markboth{\MakeUppercase{Answers}}{}
 \addcontentsline{toc}{chapter}{Answers}
 \part{Semester II}
 \chapter{Quadrilaterals}
```

```
\section{Introduction}
\section{Plane Quadrilateral}
\chapter{Areas of Parallelograms and Triangles}
\section{Introduction}
\section{Interior of a Triangle}
\chapter{Circle}
\chapter{Surface Area and Volume}
\chapter*{Answers}
\markboth{\MakeUppercase{Answers}}{}
\addcontentsline{toc}{chapter}{Answers}
\appendix
\chapter{Terminology}
\backmatter
\end{document}
```

Listing 12.1 : A Sample Book in LaTeX

- Save the file using the **File → Save** menu option. Note that the extension of the file should be .tex.
- Now select the **Tools → Build** menu option (shortcut key: F7) to compile your LaTeX file. The output window will show several messages. If the last line (in blue color) reads Exit code: 0 then compilation was successful. Otherwise the error messages may point to the error(s), but often they are difficult to interpret.
- If the compilation was successful, you may select the **Tools → Go** menu option (shortcut key: F5) to view the file in the default document viewer. Don't forget to close the document viewer before returning to SciTE.

Note that we have done some customization of the default book style in listing 12.1. Some of the features used are not discussed in this text. LaTeX typesetting is heavily customizable.

Text Formatting

In LaTeX documents, we type a paragraph in a continuous flow without pressing the ENTER key. LaTeX then adjusts the text automatically. It decides how much text should go in the first line, how much text should go in the second line, and so on according to the page width, font size, alignment option used. LaTeX usually avoids breaking a word in two parts. Where it must do so, it has a hyphenation algorithm to decide the most appropriate way of breaking the word with a hyphen (-). For example, if the word formatting cannot be accommodated on a single line, the first line may end with formatt- and the next line may begin with ing.

On the other hand, there are some situations where some text, even though technically forming multiple words, should not be split into multiple lines. For example, up to are two words, and yet, it is not desirable to have one line ending in up and the next line beginning with to because up is an independent word with different meaning; so the reader is surprised for a moment on seeing to on the next line. This is an obstacle

to smooth reading experience. To avoid it, both up and to should be on the same line, either the first line or the second. In LaTeX, this can be taken care of by inserting a non-breaking space between up and to. LaTeX uses the ~ (tilde) character to denote a non-breaking space.

LaTeX divides font families into three categorie; Roman (also called serif) fonts have a tiny line or curve called serif at the end of the strokes (lines), sans serif fonts do not have serifs while monospace fonts use equal width for all characters. Monospace fonts are typically used for computer code listings. The default fonts are Roman. These three types of fonts can be used for any text by employing the commands \textrm{text}, \textsf{text} and \texttt{text} respectively. You can see the difference between serif and sans serif fonts in figure 12.2 where the section titles are in the default serif fonts while the body text is in sans serif.

The font size can be changed using the commands \tiny, \scriptsize, \footnotesize, \small, \normalsize, \large, \LARGE, \huge and \Huge. Notice that the commands are case-sensitive. The commands \textbf, \textit and \emph can be used to add the bold, italic and emphasis (generally same as italics) effect to text. \textsc provides small capital letters. Superscripts and subscripts can be created in text mode using the commands \textsuperscript and \textsubscript respectively from the package fixltx2e.

Paragraph Formatting

In LaTeX, the setspace package provides the singlespace, onehalfspace, doublespace and spacing{amount-of-spacing} environments for setting the line spacing. By default, body text is fully justified in LaTeX. To achieve left alignment, right alignment or center alignment, one may use the environments flushleft, flushright and center respectively. The first line of a paragraph is indented, except for the paragraphs immediately following a heading.

The \indent and \noindent commands can be used immediately before a paragraph to explicitly make the first line indented and unindented respectively. The verbatim environment outputs everything inside it (including special characters, spaces, newlines and LaTeX commands) as it is without any processing. The moreverb package provides a listing environment with one mandatory argument line-number-of-first-line for program code listing with line numbers.

Page Layout

In LaTeX, the geometry package can be used for page layout. The paper size and margins can be passed as optional arguments with the \usepackage command itself. E.g. the command

\usepackage[a4paper, top=1in, bottom=2in, left=1.5in, right=1in]{geometry}

sets the page size to A4, top margin to 1", bottom margin to 2", left margin to 1.5" and right margin to 1". Page sizes have been standardized internationally. The page sizes commonly used with regular printers are A4, letter and legal. These can be specified by a4paper, letterpaper and legalpaper respectively. The page orientation can also be specified using the portrait (default) and landscape options.

Documents can be either one-sided or two-sided. Articles are by default one-sided while books are two-sided. Two-sided documents differentiate between the left (even) and right (odd) pages and can have different margins for both, to take care of amount of page space used up by the binding. There can also be rules like all chapters must start on an odd page.

Typesetting Mathematical Content in LaTeX

Ability of automatically laying out complex mathematical content is a major strength of LaTeX. The most common way of laying out mathematical content in LaTeX is using the packages amsmath, amssymb and amsfonts created by the American Mathematical Society.

The amsmath package defines several environments for mathematical content. There are two ways of typesetting formulas and equations. We may have them printed as part of the running text (inline) or they may be printed independently on their own lines (called display in LaTeX parlance). The former form can be obtained using the math environment, while the latter can be obtained using the displaymath environment.

The equation environment is a display environment that automatically numbers equations. A convenient way of embedding math environment in running text is to enclose the mathematical content between $...$. In the mathematical environments, each letter is treated as a mathematical variable. Hence they are different from the text environments. To better understand these environments, create a tex file by using code in listing 12.2.

- Create a new file in SciTE using the **File → New** menu option.
- Type the content shown in Listing 12.2 in a SciTE editor.

```
\documentclass[12pt]{article}
\usepackage{amsmath}
\title{Introduction to \LaTeX}
\date{May 2013}
\begin{document}
\section*{math environment}
The quadratic equation, in its general form, is
\begin{math}
ax^2 + bx + c = 0
\end{math}.
You learnt about them in class X.

The quadratic equation, in its general form, is $ax^2 + bx + c = 0$. You learnt about
them in class X.

\section*{displaymath environment}
The quadratic equation, in its general form, is
\begin{displaymath}
ax^2 + bx + c = 0
\end{displaymath}. You learnt about them in class X.

\end{document}
```

Listing 12.2 : Demonstration of the math environments

- Save the file using the **File → Save** menu option.
- Now select the **Tools → Build** menu option (shortcut key: F7) to compile your LaTeX file.
- If the compilation was successful, you may select the **Tools → Go** menu option (shortcut key: F5) to view the file in the default document viewer. Figure 12.3 shows the output of the file when viewed in document viewer.

math environment

The quadratic equation, in its general form, is $ax^2 + bx + c = 0$. You learnt about them in class X.

The quadratic equation, in its general form, is $ax^2 + bx + c = 0$. You learnt about them in class X.

displaymath environment

The quadratic equation, in its general form, is

$$ax^2 + bx + c = 0$$

. You learnt about them in class X.

Figure 12.3 : Part output of the TeX code in Listing 12.2

Using Mathematical Symbols

Mathematics uses a large number of symbols. We shall discuss how some of those symbols can be used in LaTeX. Letters of the Greek alphabet have their corresponding commands, like \alpha, \beta, \gamma, \pi that produce the lower case letters. The same commands, when used with the first letter in the uppercase, for example \Alpha produce capital Greek letters.

There are commands for other mathematical symbols as well. Figure 12.4 and Figure 12.5 show some LaTeX commands and the mathematical symbols produced by them using the AMS packages. The reason for providing commands for the trigonometric functions sin, cos and others are that they need to be recognized and typeset as functions rather than as individual variables.

$<$	$<$	$>$	$>$
$=$	$=$	\leq	$\leq$
\geq	$\geq$	\neq	$\neq$
\times	$\times$	\div	$\div$
\pm	$\pm$	\mp	$\mp$
\in	$\in$	\notin	$\notin$
\supset	$\supset$	\subset	$\subset$
\supseteq	$\supseteq$	\subseteq	$\subseteq$
\cup	$\cup$	\cap	$\cap$

Figure 12.4 : Some Mathematical Symbols in LaTeX

\cong	$\cong$	\propto	$\propto$
\rightarrow	$\rightarrow$	\parallel	$\parallel$
\leftrightarrow	$\leftrightarrow$	\leftarrow	$\leftarrow$
\angle	$\angle$	\bigodot	$\bigodot$
\triangle	$\triangle$	\overleftrightarrow{AB}	$\overleftrightarrow{AB}$
\stackrel\frown{AB}	$\stackrel{\frown}{AB}$	\overrightarrow{AB}	$\overrightarrow{AB}$
\overline{AB}	$\overline{AB}$	\perp	$\perp$
45^{\circ}	45°	\implies	$\implies$
\iff	$\iff$	\therefore	$\therefore$
\because	$\because$	\sin	$\sin$
\cos	$\cos$	\tan	$\tan$
\sec	$\sec$	\csc	$\csc$
\cot	$\cot$	\theta	θ

Figure 12.5 : Some More Mathematical Symbols in LaTeX

Using Mathematical Operators

LaTeX supports a large number of mathematical operators. The power operator (x2) and the index operator (x1) are not provided separately, but are implemented using the generic superscript operator ^ (the caret character) and the generic subscript operator _ (the underscore character) respectively. The superscript operator raises the text that follows it, while the subscript operator lowers it. Both reduce the size of the text as well. Absolute values can be denoted by enclosing the expression between two | (vertical bar) symbols.

Fractions are created using the command \frac{numerator}{denominator}, while square root of a number x is denoted using the command \sqrt{x}. These operators can be nested one inside another. That means we may have a fraction in the denominator of another fraction, whose square root is in the numerator of yet another fraction, and so on. There is no practical limit on this, and LaTeX takes care of sizing and placing the elements appropriately. Round brackets work normally. An example usage of mathematical operators is given in listing 12.3.

- Create a new file in SciTE using the **File → New** menu option.
- Type the content shown in Listing 12.3 in a SciTE editor.

```
\documentclass[12pt]{article}
\usepackage{amsmath}
\setlength{\parindent}{0pt}
\title{Introduction to \LaTeX}
\date{May 2013}
\begin{document}
\begin{math}
x^2 \\[6pt]
x_1 \\[6pt]
x_i \\[6pt]
x_i^2 \\[6pt]
x^y \\[6pt]
a^bc \\[6pt]
{a^b}^c \\[6pt]
a^{b^c} \\[6pt]
\frac{x}{y} \\[6pt]
\sqrt{b} \\[6pt]
\frac{\frac{1}{3}+\frac{3}{2}}{\frac{2}{3}+\frac{1}{2}} \\[6pt]
\frac{-b\pm\sqrt{b^2-4ac}}{2a} \\[6pt]
\frac{-b\pm\sqrt{\Delta}}{2a} \\[6pt]
\sqrt{1+\frac{1}{\sqrt{1+\frac{1}{\sqrt{1+\frac{1}{2}}}}}} \\[6pt]
\sqrt{(x_1-x_2)^2+(y_1-y_2)^2} \\[6pt]
|x-y| \\[6pt]
(\frac{x_1+x_2}{2},\frac{y_1+y_2}{2})
\end{math}
\end{document}
```

Listing 12.3 : Example of Mathematical Operators

- Save the file using the **File → Save** menu option.
- Now select the **Tools → Build** menu option (shortcut key: F7) to compile your LaTeX file.
- If the compilation was successful, you may select the **Tools → Go** menu option (shortcut key : F5) to view the file in the default document viewer. Figure 12.6 shows the output of the file when viewed in document viewer.

$$x^2 \qquad \sqrt{b}$$

$$x_1 \qquad \frac{\frac{1}{3}+\frac{3}{2}}{\frac{2}{3}+\frac{1}{2}}$$

$$x_i \qquad \frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

$$x_i^2 \qquad \frac{-b\pm\sqrt{\Delta}}{2a}$$

$$x^y \qquad \sqrt{1+\frac{1}{\sqrt{1+\frac{1}{\sqrt{1+\frac{1}{2}}}}}}$$

$$a^b c \qquad \sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$

$$a^{b^c} \qquad |x-y|$$

$$a^{b^c} \qquad \left(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2}\right)$$

$$\frac{x}{y}$$

Figure 12.6 : Output of the TeX code in Listing 12.3 broken in two parts

Using Equations

LaTeX provides a special equation environment for typesetting equations. Each equation is to be enclosed in the equation environment, which cannot be embedded in a math environment. Equations are numbered automatically and center-aligned. Listing 12.4 shows an example of using the equation environment, while Figure 12.7 shows a part of the output.

```
\documentclass[12pt]{article}
\setlength{\parindent}{0pt}
\usepackage{amsmath}
\title{Introduction to \LaTeX}
\date{May 2013}
\begin{document}
\begin{equation}
\sin^2\theta + \cos^2\theta=1
\end{equation}
\begin{equation}
```

```
\sec^2\theta - \tan^2\theta=1
\end{equation}
\begin{equation}
\csc^2\theta - \cot^2\theta=1
\end{equation}
\end{document}
```

Listing 12.4 : Using the Equation environment

$$\sin^2\theta + \cos^2\theta = 1 \qquad (1)$$

$$\sec^2\theta - \tan^2\theta = 1 \qquad (2)$$

$$\csc^2\theta - \cot^2\theta = 1 \qquad (3)$$

Figure 12.7 : Output of the TeX code in Listing 12.4

Advantages of LaTeX

LaTeX has several benefits. Just as TeX was extended to LaTeX, LaTeX itself also can be extended. Anyone can create additional packages to enhance the features provided in LaTeX, to add new features or to provide alternate implementations. Thousands of such packages have been created by LaTeX users around the world to cater to different needs. Most of them are free. They are hosted on the CTAN (The Comprehensive TeX Archive Network) website at www.ctan.org.

LaTeX is extremely good at laying out complex mathematical formulae in nice looking and appropriate way. Because of this, it is quite popular among authors and publishers in mathematics, engineering, computer science and other technical areas. Because it is open source, highly systematic and is developed in the academic spirit of knowledge sharing and collaboration, it is popular among academicians and scholars. People in these fields use LaTeX, share their views and experiences, help one another, develop and share new packages and drive the further development of LaTeX.

LaTeX has built-in facilities to automatically update numbering and references and to automatically create table of contents, indexes, and other such needs, taking a major burden off the mind of the author.

Summary

In this chapter, we learnt how to use LaTeX for typesetting technical and mathematical documents. We studied the essentials of using LaTeX. We discussed its syntax and the structure of a LaTeX document. Then we discussed basic document creation and formatting in LaTeX. Finally, we saw how LaTeX can be used to typeset mathematical content, one of its major strength. We also discussed how it differs from word processors. We discussed the advantages of using LaTeX.

Configuring SciTE and LaTeX (Only for teachers)

- Close SciTE if it is already open.
- Open the terminal.
- Run the command

  sudo gedit /usr/share/scite/tex.properties&

  This will open the file in gedit as administrator. Edit it as follows :
- Delete the lines

  file.patterns.tex=*.tex;*.sty

  file.patterns.context=*.tex;*.tui;*.tuo;*.sty

  Note make sure that you do not delete the line

  file.patterns.latex=*.tex;*.sty;*.aux;*.toc;*.idx
- Change the line

  command.go.$(file.patterns.latex)=gv $(FileName).pdf

  to

  command.go.$(file.patterns.latex)=evince $(FileName).pdf
- Save the file and close gedit.
- Open SciTE; open a proper LaTeX file and press F7 followed by F5. The LaTeX file should compile and the PDF file should open. Close the PDF file before returning to SciTE.

EXERCISE

1. Compare LaTeX with word processors. List the strengths and weaknesses of both.
2. List key reasons for the popularity of LaTeX.
3. List the reserved characters and characters that cannot be used directly in LaTeX.
4. Explain the structure of a LaTeX document.
5. Explain the \frac command and its nesting inside another \frac command with an example.
6. Choose the most appropriate option from those given below :

   (1) Modern word processing software operates in which of the following mode ?

   (a) WIGIWIS (b) WISYWIG (c) WYSIWYG (d) WISYWYG

(2) Which of the following is not a reserved character in LaTeX ?

(a) @ (b) % (c) $ (d) ^

(3) Which of the following begins with \begin{name} and ends with \end{name}.

(a) group (b) section (c) environment (d) preamble

(4) Which of the following character is used to mark a comment in LaTeX ?

(a) $ (b) % (c) # (d) &

(5) Which of the following part of the LaTeX document contains the metadata ?

(a) preface (b) TOC (c) preamble (d) environment

(6) Which of the following web site hosts the LaTeX packages ?

(a) CTAN (b) CLAN (c) CTEN (d) CLEN

(7) Which of the following will not be automatically numbered ?

(a) \section (b) \subsection (c) \chapter* (d) \part

(8) Which of these environments displays mathematical content inline with the text ?

(a) displaymath (b) math (c) equation (d) text

(9) Which of the following commands generate the set union symbol ?

(a) \cup (b) \setunion (c) \cap (d) \union

(10) Which of the following operator is used to denote a subscript or an index ?

(a) _ (underscore) (b) ^ (caret) (c) - (minus) (d) < (less than)

(11) Which of the following refer to the very first line in the preamble ?

(a) \usepackage (b) \title (c) \maketitle (d) \documentclass

LABORATORY EXERCISE

1. Create a skeletal structure of this book in LaTeX with preface, table of contents, chapters, sections, subsections, etc. Type only a few words in each.
2. Create a LaTeX document to illustrate text formatting.
3. Create a LaTeX document to illustrate paragraph formatting.
4. Create a LaTeX document and experiment with page layout.
5. Create LaTeX documents with the given contents.

(a) If A = {x | x ∈ N, x ≤ 4}, B = {-1, 0, 1, 2, 3} and C = {0, 1, 2} then (A ∪ B) ∩ (A ∪ C) = {0, 1, 2, 3, 4}.

(b) If A = {x | x ∈ N, x ≤ 7} and B = {2, 4, 6} then B ⊂ C.

(c) $\frac{(81)^{\frac{1}{4}}}{(625)^{\frac{1}{4}}}+\frac{(216)^{\frac{1}{3}}}{(8)^{\frac{1}{3}}}-(729)^{\frac{1}{6}}$

(d) X ⊂ Y and Y ⊂ X ⇒ X = Y

(e) $\overrightarrow{PQ}\cap\overrightarrow{PR}=P$

(f) $BE\cap EG=\{E\}$

(g) $\sqrt{s(s-a)(s-b)(s-c)}$

(h) $Areaof\odot(O,r)=\pi r^2$

(i) $\sqrt{a+2\sqrt{b}}=\sqrt{x}+\sqrt{y}$

(j) $x=\frac{a+\sqrt{a^2-4b}}{2}, y=\frac{a-\sqrt{a^2-4b}}{2}$

(k) $\frac{h(\tan\alpha-\tan\beta)}{\tan\alpha\tan\beta}$

Other useful free tools and services 13

In this chapter we shall discuss some useful free tools and services. We shall have a quick introduction to data compression techniques. These techniques allow us to reduce the amount of memory and disk space required to store the information. We shall discuss the Archive Manager tool that lets us apply these techniques to files and entire directories. We shall briefly discuss VLC media player, a versatile multimedia tool. We shall also discuss the Google Maps service, which is an online service providing, maps of different areas. We will see two small but useful applications - Character Map, which allows us to insert various symbols and characters from different languages in our text. We shall then have an overview of the R environment for statistical computing. We shall learn some basic statistical operations like finding mean and median in R and also learn to draw bar graphs and histograms in R. Lastly we will look at two tools RationaPlan and Skype. While some of these applications come pre-installed with Ubuntu 10.04 LTS, others need to be installed from the Ubuntu repositories.

Data Compression

A modern computer system has lakhs of files on them. Often one needs to transfer a large number of these files or an entire directory structure to another computer or storage device. When we want to transfer computer files (containing data or programs or both) to another computer or storage device, the amount of data to be transferred or stored becomes a concern. Both computer networks and external storage devices are usually not as fast as internal components of a computer. Thus, transferring larger amount of data may take more time. If the Internet is used for such transfers, more time may be taken in the transfer due to slow Internet speed. Also, such transfers put load on the usually clogged Internet connection. Unless the user or organization has unlimited Internet plan, higher amount of data may result in higher cost as well.

Similarly, if the files and directories are transferred to a storage device, the amount of the data to be transferred again becomes an issue because of the finite capacity of storage devices and the multiple uses that they are put through. Considering these issues, there is a need to reduce the amount of storage space occupied by computer files (and entire directory structures), wherever possible. From a convenience point of view, in many cases it is also desirable to have a single file to handle rather than a large bunch of files or a complex directory structure.

Computer scientists have developed techniques to place a whole directory structure into a single file for convenience. Such a file is called an "archive". They have also developed a number of techniques for reducing the storage requirements of computer files and directory structures. These techniques are called data compression.

Data compression generally works by identifying repetition in the data and encoding the data in a way that reduces or eliminates such repetition. Some techniques also identify and eliminate less important information to conserve space. As an example, consider the children's rhyme shown in

figure 13.1. It certainly has a lot of repetition of words. We may take advantage of this and represent each word by a single digit or letter. At the beginning of our encoded file we shall have a table that tells us which digit/letter represents which word (this information is needed to convert the file back in its original uncompressed form). Afterwards, we may use a single digit/letter to represent a word. Thus, lengthy words appear only one time and subsequently they are represented by a single character. We leave punctuation as it is. We may mark the beginning of the table with a ^ (caret) symbol and end of the table with a $ symbol. The encoded file is also shown in the second half of figure 13.1.

```
One little, two little, three little Indians
Four little, five little, six little Indians
Seven little, eight little, nine little Indians
Ten little Indian boys.

Ten little, nine little, eight little Indians
Seven little, six little, five little Indians
Four little, three little, two little Indians
One little Indian boy.
```

Actual Contents

```
^0:One 1:little 2:two 3:three 4:Indians 5:Four
6:five 7:six 8:Seven 9:eight A:nine B:Ten
C:Indian D:boys E:boy$0 1, 2 1, 3 1 4
5 1, 6 1, 7 1 4
8 1, 9 1, A 1 4
B 1 C D.

B 1, A 1, 9 1 4
8 1, 7 1, 6 1 4
5 1, 3 1, 2 1 4
0 1 C E.
```

Encoded Contents

Figure 13.1 : Example of data compression

While the size of the original file was 324 bytes, the size of the encoded file is just 226 bytes. This is because each word is used in its full form only once; subsequently it is represented by a single character. The encoded file can be transformed into the original file at any time using the reverse process.

The scheme described here is fairly simple. Actual data compression programs use far more sophisticated algorithms based on information theory to achieve even greater reduction in size. Fortunately, Linux provides ready-made free and open source software for managing archive files. It is called Archive Manager.

Archive Manager

Archive Manager as shown in figure 13.2 permits us to combine many files or an entire directory tree into a single file known as an archive. On Linux systems, tar (tape archiver) is the most common archive format. It also provides the facility of compressing the archive to reduce its file size. It supports multiple compression algorithms and compressed file formats. The most common compressed file formats are the zip file format and the tar.gz file format. While the zip file format supports storing multiple files in a single zip file, on Unix/Linux systems the common practice is to combine the files into a single uncompressed file in the tar format and then compress the file in the zip format using gzip (GNU zip, an open source archiving program). Such a file commonly has the extension tar.gz and is called a "tar ball". The tar ball is a very common format for distributing software or bundles of files on the Linux platform. The zip format is also used by different applications under different names. For example, the JAR files in Java and the OpenOffice.org office suite file formats use zip compression. Archive Manager uses the extension part of the filename to identify the file format when opening an archive and to select the file format when creating a new archive.

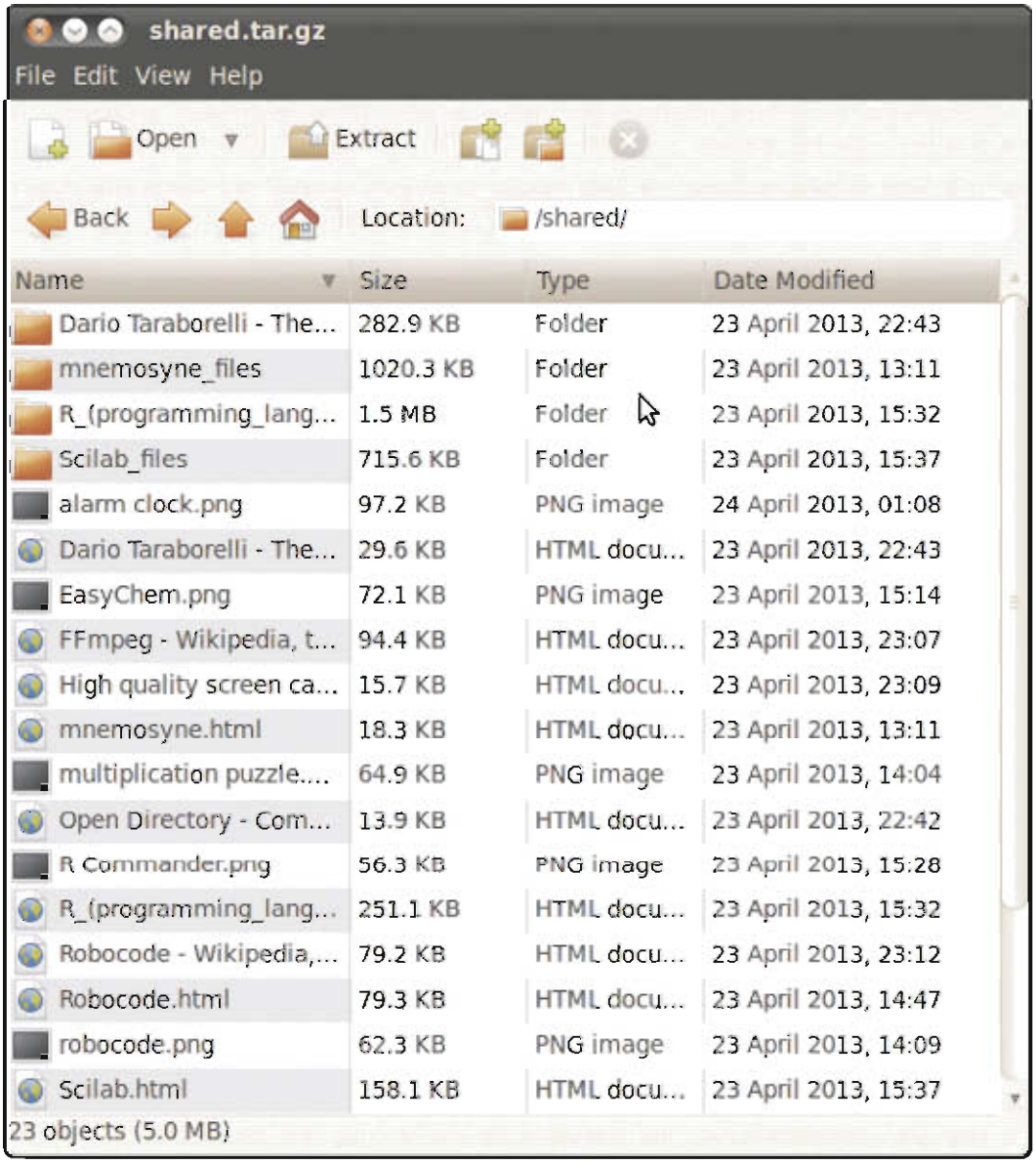


Figure 13.2 : Archive Manager

Archive Manager can also be used to explore the contents of archives, to extract files from the archives, to add new files to the archive, to delete files from the archive, and perform other such operations. There is no menu item to start the archive manager by default. One may double-click on an archive file in the Nautilus file browser to start the archive manager. A new archive can be created by right-clicking a file or directory and selecting the Compress... option from the context menu. This will create a new archive with that file or directory as its contents. This option is not available for files that are themselves archives. If we wish, we may add Archive Manager to the Accessories sub menu of the Application menu. For this, right-click the Ubuntu icon in the upper-left corner, select Edit Menu from the list, click on Accessories in the dialog box that opens and select the check box for Archive Manager and close the dialog box.

When an archive is open in the Archive Manager, the view is similar to a directory open in the Nautilus file browser as can be seen in figure 13.2. The operations and keyboard shortcuts are also similar. Double-clicking a directory opens that directory and shows its contents. The Up and Back buttons in the toolbar can be used to go to the parent directory (if it exists in the archive) and the previous directory (if any) respectively. They are enabled only when applicable.

Thus the Archive Manager helps us in creating and using archive files for backup, for transfer via an external storage device, for transfer via network, for saving disk space, etc.

The VLC Media Player

We have discussed earlier that Ubuntu comes with built-in tools to play multimedia content like audio, video, etc. However, another great open source tool called VLC (originally it was short form of VideoLAN Client) media player is shown in figure 13.3. It is also extremely popular because if its versatility and features.

Figure 13.3 : The VLC Media Player

Started as an academic project by the music-loving students of a university in Paris, it is now a community project available under multiple operating systems and has been downloaded over a billion times. One of the issues faced with multimedia content is that there are a variety of ways of converting multimedia information coming from hardware devices (audio/video streams) into computer data and then converting the computer data back into audio/video streams for playing on the hardware devices. This conversion (coding) and reverse conversion (decoding) is performed by a software component called codec (coder decoder). Each multimedia data format requires its own codec. The advantage of VLC is that it supports all the popular codecs and hence all the popular formats. It also supports all the major types of devices including web cameras, HD monitors, speakers of all types, microphones, headphones. VLC provides several options for playback of audio and video. It can convert multimedia files from one format to another. It can also stream (send) audio/video to and receive audio/video from another computer(s) over the network.

After starting VLC from the **Applications → Sound & Video** menu, we may open one or more files using the **Media → Open File...** menu item. We may open an entire directory using the **Media → Open Directory...** option. If we open a directory, VLC will actually open all media files in that directory playable by it. When one or more files are opened, they are added to the playlist. The playlist is a list of media files (tracks) to be played. We may play the media in the playlist in sequence, or in random order. We may go to the previous or next track. We may also save the playlist for playing the same tracks again. The playlist can be opened using the **View → Playlist** option and saved using the **Media → Save Playlist to File...** option. VLC supports multiple file formats for playlist, but M3U is more common.

At the bottom of the window, VLC displays a progress bar showing how much of the current track has been played and what is its total duration. By sliding the little slider on the bar, we may move back and forth in the current track. Below that, we have the play button (marked with a little right-pointing triangle). When the track is playing, this button turns into a pause button (with two parallel bars on it). If we pause the playback, the button reverts back to play mode. The middle button in the next three buttons is used to stop the playback completely, while the buttons with double arrows on the left and right allow us to jump to the previous and next track respectively. The next button acts as a toggle between watching a video in a window against watching it in full screen mode. That means, every time we click the button, it switches from one of these states to the other. In full screen mode, even the controls like play/pause, stop is hidden. They can be displayed in a temporary floating window by moving the mouse cursor over the video. The next button is for displaying the playlist. These options are available in the menu also.

While VLC has a lot of other functionalities as well, here we shall discuss how VLC can be used to convert multimedia files from one format to another. For this, select the **Media → Convert / Save** option from the menu. Using the Add button, select and add the file to be converted. Then from the list besides the Convert / Save button, select the Convert option. This will open another dialog box. Here, we need to provide the destination file (in which the converted track will be saved). We need to select appropriate file format (audio or video), from the drop down provided. Clicking on the Start button will now start the process and the progress will be shown. Sometimes, we may get error in the conversion process also.

Google Maps

Google Maps shown in figure 13.4 is a free Internet-based service provided by Google, Inc. Google has, over a period of years, collected extensive map data for the whole earth through various means like satellite imagery, cars with cameras mounted on them, data purchased from other organizations and data provided by millions of individual users around the world. Google Maps allows anybody to edit the maps and identify landmarks, buildings, etc. It also allows users to upload photographs of the place and post reviews of the place. The service is also heavily used on mobile devices like smartphones and tablets.

Figure 13.4 : The Google Maps Service

The service can be accessed from the web browser by opening the web site http://www.google.co.in and selecting the Maps option from the top menu, or directly entering the URL http://maps.google.co.in. On mobile phones, the service may be accessed from the mobile phone's web browser as well as from the Google Maps application. When accessed, the service first tries to know our current location. On the PC, a rough idea of the user's location can be obtained from the user's Internet connection. On a mobile phone, the user's location can be known with very high accuracy if the GPS (Global Positioning System) facility is used. This system finds a user's location with an estimated error of only few meters using satellite signals. Google maps also allows the user to correct the location, if it is inaccurate. Then it shows a map of the current location. We may pan (scroll) the map by dragging it in different directions or by clicking on the arrows in a circle on the left hand side. A vertical slider bar on the left allows us to zoom in and out on the map.

Google maps can show the information of a place in different forms. The default view is map view, shown in figure 13.4. By clicking on the Satellite button, we may switch to the satellite image view, shown in figure 13.5. In this view, the images taken by satellites in the sky are shown. The images are clear enough to identify most familiar buildings and roads. The service also provides the facility of getting directions on how to reach from place A to place B. The two places may be two different cities or places within the same city. The service also provides a choice of routes where available. Figure 13.6 shows an example. The service even provides turn-by-turn guidance while traveling if we use a mobile device equipped with GPS receiver. On a highway or in a crowded city area, walking or driving, the service decides the route based on the current location and destination and guides us with on-screen as well as voice (spoken) instructions.

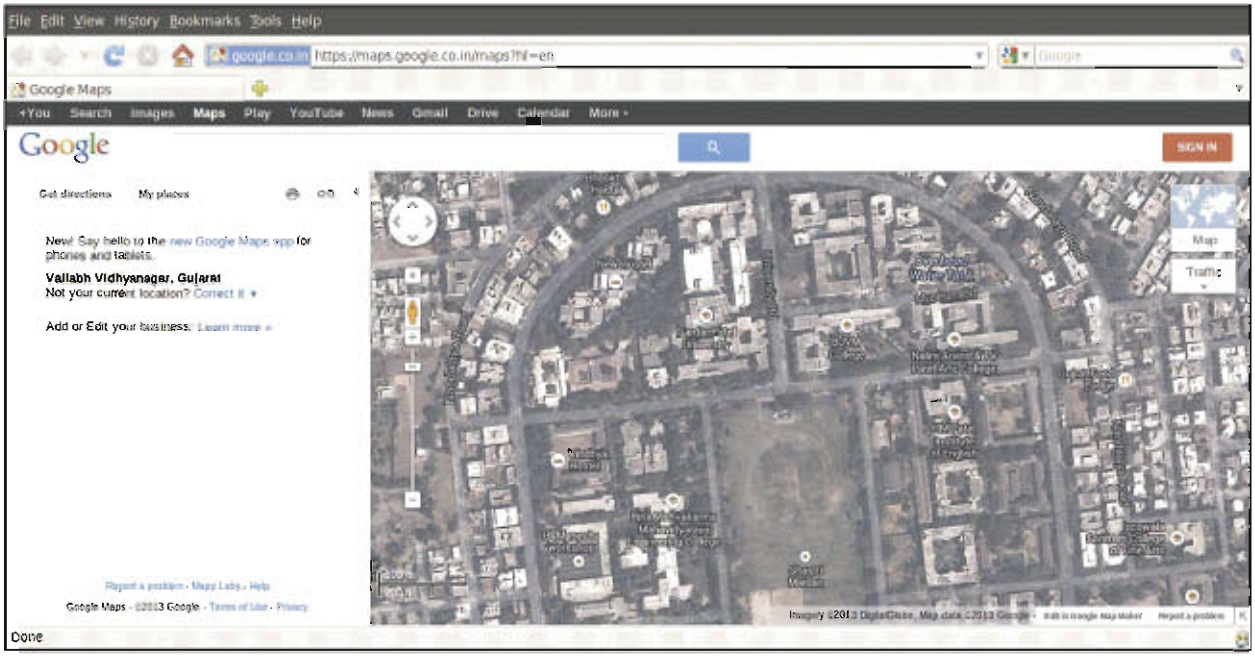


Figure 13.5 : The Google Maps Satellite Image View

There are several uses of the service. It can be used to search for a place and find its exact location. It can also be used to find directions to places unknown to us or in cities we are not familiar with. Government organizations and businesses use the service to provide people information on how to reach their offices. The service also has a facility using which anyone can display a map of any place on one's web site. Many organizations display such maps on their web sites. Tourist information web sites and Tourism departments of governments also make use of the service. It can also be used to display bus routes, current location of running trains, etc. The service also provides the facility of finding the nearest ATM centre, bank, restaurant, bus stop or any other such essential place.

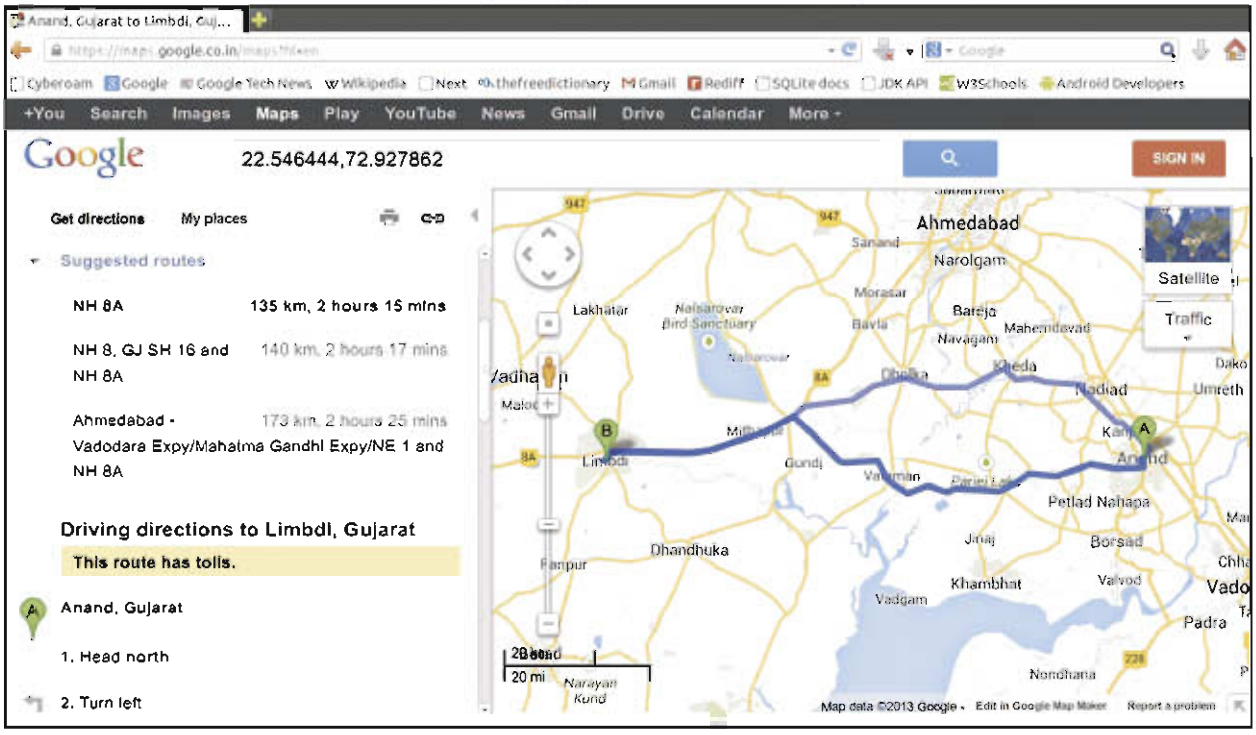


Figure 13.6 : Finding Directions using Google Maps

Character Map

The Character Map program shown in figure 13.7 can be used to enter Unicode characters into any application. We first select the script from the left pane and then double click characters to insert them into the Text to copy area below. Spaces and punctuation may be entered from the keyboard directly into the area after clicking in it. Combining characters, like the long "II" (??????) in Gujarati can also be entered and combine with the corresponding consonant appropriately. Brief details of the character currently selected with single-click or entered with double-click are shown in the status line at the bottom, while more details on the character is available in the Character Details tab. After we get a substantial amount of content, we may copy the content and paste into an application. If you only have to type a few characters in another script occasionally, then Character Map is a good solution for that.

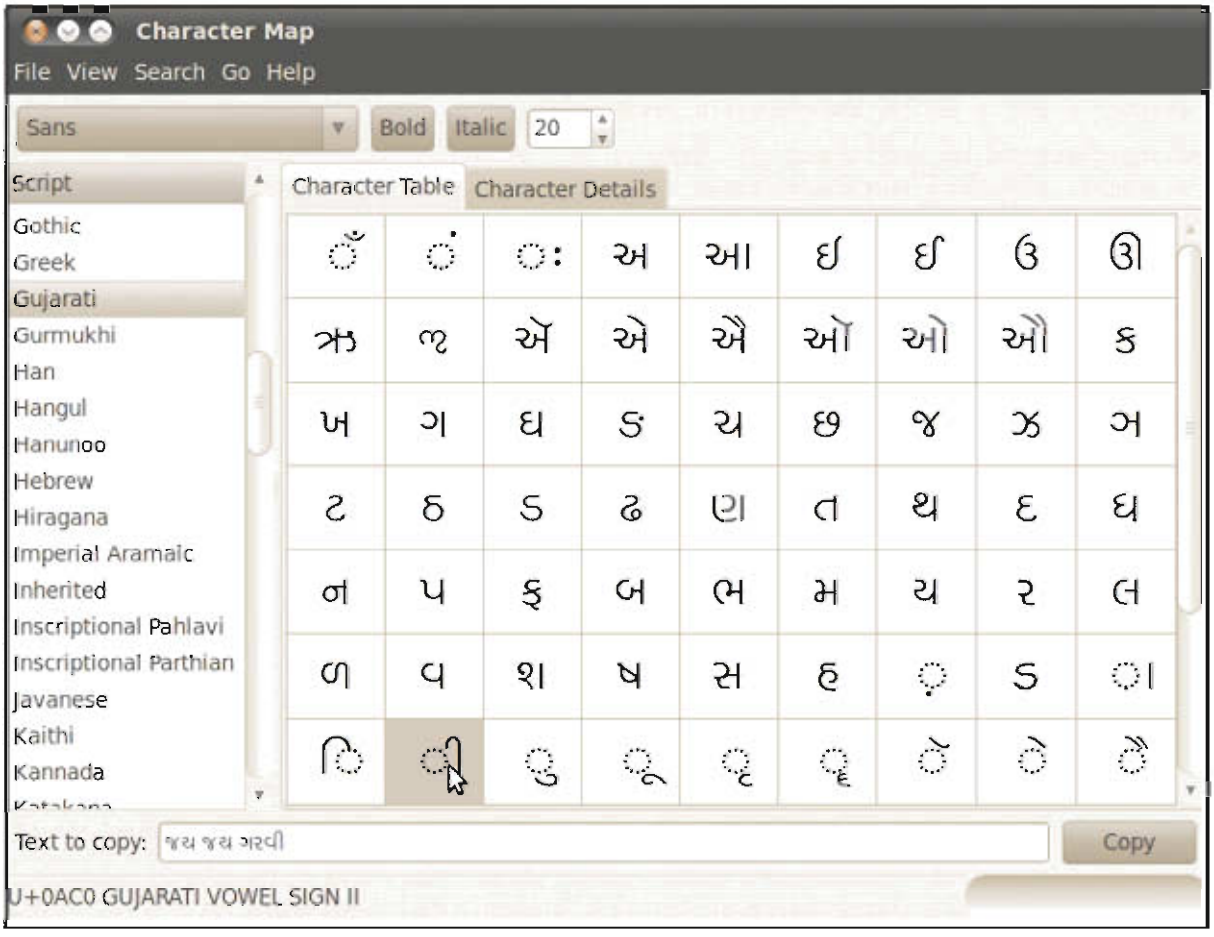


Figure 13.7 : The Character Map program

The R Software

R is a free software environment for statistical computing. It is a GNU project. It is widely used for statistical analysis. It has its own scripting language. It is a case-sensitive language. R has two work environments namely Command Line and Graphical. To use it in a GUI environment we need to install graphic editors like R Commander or RStudio from Ubuntu Software Center. The command to install R is given at the end of this chapter. To invoke the R scripts from terminal window, open terminal and type R on the command prompt. You will get a welcome message along with R prompt as shown in figure 13.8

```
harshal@HarshalAtUbuntu: ~
harshal@HarshalAtUbuntu:~$ R

R version 2.14.1 (2011-12-22)
Copyright (C) 2011 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: i686-pc-linux-gnu (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

Figure 13.8 : R Command prompt

R uses # as the comment marker, just like the Linux shell. Any text following #, up to the end of the line, is treated as a comment. Number and string are the basic data types. Strings may be enclosed in single quotes or double quotes. Common operators and functions are available, just like other programming languages. Ordered lists of items (arrays or lists) are commonly used and are also referred to as vectors. Lists are created using the c function to combine several numbers into a single list. Most operators like +, -, *, /, and others work equally well on single number as well as on lists. However, when both operands of binary operators are lists, their lengths (number of elements in them) should ordinarily be same. Entering the name of a variable and pressing ENTER displays its value. An example of this concept is shown in figure 13.9.

```
harshal@HarshalAtUbuntu: ~
harshal@HarshalAtUbuntu:~$ R

> a <- 10
> a
[1] 10
> b <- 20
> a*b
[1] 200
>
```

Figure 13.9 : Working with variables in R

Here the symbol > shows the prompt of R. The statement > a ← 10 defines a variable 'a' and assigns it a value 10. The statement > a followed by Enter key press displays the value of a. Similarly the statement > b ← 20 defines a variable 'b' and assigns it a value 20. The statement > a*b followed by Enter key press multiplies the value of variable 'a' and 'b' and display the result on the prompt.

Some common R commands include q() for quitting R, help() for accessing the on line help, demo() for viewing some demonstrations and help.start() to open the on line help in a browser. To get help on a particular function we need to use the syntax help(function name). When we quit R, we are asked whether we want to save the data (values of variables) from the current session. If we respond "yes" or "y", the data will be saved in a file in the current directory and will be available in the next session of R we start from the same directory. The function ls() displays a list of all the variables we have defined as shown in figure 13.10.

```
harshal@HarshalAtUbuntu: ~
harshal@HarshalAtUbuntu:~$ R

> a <- 10
> a
[1] 10
> b <- 20
> a*b
[1] 200
> ls()
[1] "a" "b"
>
```

Figure 13.10 : Use of ls() in R

A series of consecutive numbers (range) may be generated using the syntax start:end. For example, 1:5 is same as c(1, 2, 3, 4, 5).

R also provides basic statistical operations quite straightforward forms. For example we may used the functions min(list), max(list), mean(list) and median(list) to find the minimum value, maximum value, mean and median for the list as shown in figure 13.11.

```
harshal@HarshalAtUbuntu: ~
harshal@HarshalAtUbuntu:~$ R

> ll <- c(3,6,8,3,4,6,9,4,7,3,9)
> min(ll)
[1] 3
> max(ll)
[1] 9
> mean(ll)
[1] 5.636364
> median(ll)
[1] 6
>
```

Figure 13.11 : Statistical functions in R

Plotting in R

Assume that we want to plot a bar chart that represents faculty wise students in a university. We may define the number of students as a list and the names of faculties as another list. Then we may use the function barplot with various arguments for drawing the plot as shown in figure 13.12.

```
harshal@HarshalAtUbuntu: ~
harshal@HarshalAtUbuntu:~$ R
> no_students <- c(140,210,700,950,810,320)
> faculty_names <- c("Medical","Engineering","Science","Commerce","Arts","Law")
> barplot(no_students,main="Faculty wise students",xlab="faculty",
+ ylab="Number of students",names.arg=faculty_names,
+ ylim=c(0,1000),border="blue")
>
```

Figure 13.12 : Plotting graph in R

The first argument specifies the data list, while all other arguments have the form name=value. We use the arguments main, xlab, ylab, names.arg, ylim and border to specify the main title, X-axis label, Y-axis label, the values to be displayed for the bars, the range of values to be plotted on the Y-axis and border color respectively. Observe that for long commands, when we press ENTER before the command is complete, R issues the + prompt until we complete the command. By default the graph does not have grid lines as can be seen in figure 13.13.

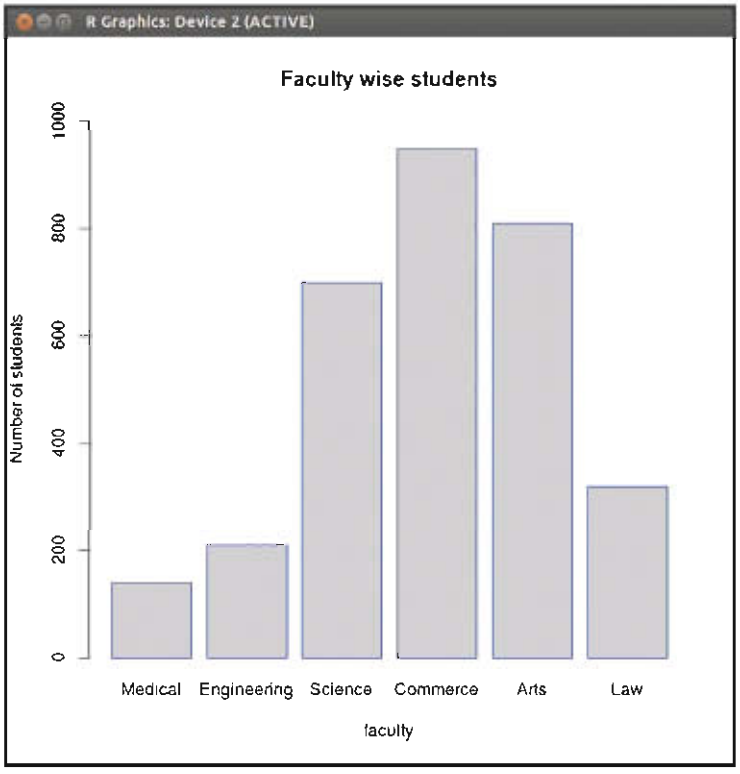


Figure 13.13 : A Bar Chart Produced in R

Producing histograms is also not difficult. Suppose we have marks out of 100 in a particular subject for several students. The histogram can be drawn using the code shown in figure 13.14 and will look like the plot shown in the same figure.

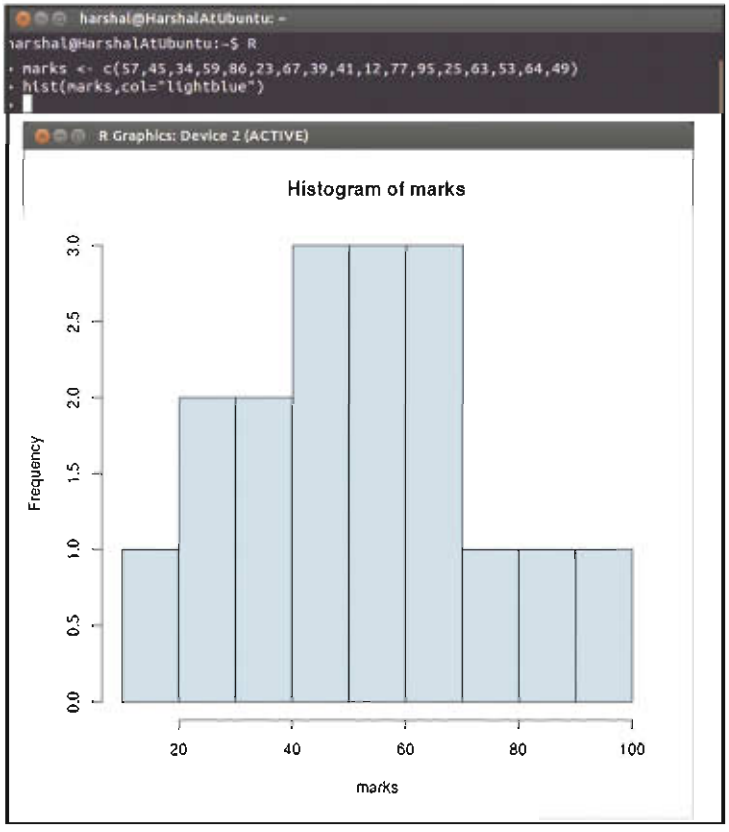


Figure 13.14 : A Histogram Produced in R

RationalPlan

People working in different areas like construction, engineering, services and consulting, business or software development usually work on a project that is having constraints of time, resource and budget. Any delay in completion of the project increases the cost of the project. At some point of time to manage this kind of project we need to prepare, maintain and follow a well defined plan. Such a plan helps us to complete the project as scheduled, on time and within budget.

RationalPlan is one such open source software designed to assist project managers in preparing, maintaining and following a well defined project plan. It assists the project managers throughout the life cycle of their projects. It comprises three different desktop products namely RationalPlan Single, Multi and Viewer.

RationalPlan Single allows us to manage independent projects that do not have common resources and has no interrelations between different projects. It allows the manager to associate general project information like name, notes, links, assumptions, constraints or risks. It allows creating, editing and deleting calendars, building schedules. Once the resources are created we can assign tasks to them. It provides project tracking tools such as critical path, mark completion value for tasks, work and cost time phased information. We can generate a printable report as well as import data from other project management tool or export our data into other formats.

RationalPlan Multi allows us to manage projects that share company resources across projects. It also manages interdependencies between projects. It includes all the features of RationalPlan Single Project. Further it calculates the resources data (work, cost, over allocation) considering their assignments in all projects. It allows links between tasks pertaining to different projects, analyzes all our projects data and creates a project Portfolio view.

RationalPlan Viewer is an additional tool developed to share the project in its original file format (.xrp), but it can be used as well to open Microsoft Project files. It is useful to anybody who needs to check and overview the project's evolution without making changes to the schedule, and for resources to see their assignments.

Though as of today the support of RationPlan on Unbuntu 10.04 has been removed it is available for Ubuntu 12.04 and later versions.

Skype

You must have used various instant messaging services like Yahoo Messenger, Google Talk or Rediff Bol. We use them for real time chatting with text, audio and video facilities. Skype is one such software that allows us to make calls over the Internet using our computer.

The Skype service allows users to communicate with peers using voice, video as well as text. We can use Skype to place phone call to recipients on the traditional telephone network also. The calls to landline telephones or mobile phones are charged via a debit based user account system while calls to other users within the Skype service are free of charge. It also provides additional features like file transfer and videoconferencing.

The Skype software can be downloaded free of cost, though its source code is proprietary and not available for modification. To use Skype we need a working sound input and output configuration. Most

modern computers today support both these features. In laptops they are inbuilt while on desktop we may use extensions like a headset, speaker and a web cam. We can install Skype from Software Center.

To start Skype, choose **Applications → Internet → Skype.** If you are opening it for the first time it will open an End User License Agreement window. Read the contents and click on "I agree" button. We will now see a window as shown in figure 13.15.

Figure 13.15 : Skype initial window

To use the Skype service you will need a Skype user account. In case you don't have one, then click on the link with contents "Don't have a Skype Name yet?", follow procedure to create a Skype name. Once you have a Skype name and password you can enter the details in the textbox of the window shown in figure 13.15 and click on the "Sign in" button. If everything goes well you can now start interacting with other Skype users or make phone calls using Skype.

Summary

In this chapter we discussed some useful free tools and services. We learnt about data compression techniques that allow us to reduce the amount of memory and disk space required to store the information. We discussed the Archive Manager tool that lets us apply these techniques to files and entire directories. We also introduced VLC media player, a feature-rich multimedia tool. We discussed the Google Maps service, which is an online service providing, maps of different areas. We learnt how to use Character Map to insert various symbols and characters from different languages in our text. We had an overview of the R environment for statistical computing. We learnt some basic statistical operations like calculating mean and median in R and learnt to draw bar graphs and histograms in R. Finally we saw the use of RationalPlan and Skype.

Instruction for Teachers

The tools covered in this chapter are only for demonstration purpose. The teachers are requested to show demo of these tools to the students. The examples given in laboratory exercise section can be used by teachers for demonstration. Teacher needs to install R software on the machines. To install R use the command given below :

sudo apt-get install r-base r-base-dev

EXERCISE

1. Write a short note on data compression.
2. Why do we need data compression tools ?
3. List main features of Archive Manager.
4. Write the main features of the VLC media player.
5. List the applications of Google Maps.
6. Explain the use of the Character Map program.
7. Choose the most appropriate option from those given below :

(1) Which of the following refers to a file that has an entire directory structure inside it ?

(a) apache (b) archie (c) archi (d) archive

(2) What is the full form of tar ?

(a) tape archiver (b) tech archiver (c) test archiver (d) tight archiver

(3) For which types of archives is the password protection option available ?

(a) zip (b) tar (c) tar.gz (d) both zip and tar.gz

(4) Which of the following is a feature-rich media player ?

(a) VAC (b) VEC (c) VLC (d) VNC

(5) What is the full form of VLC ?

(a) VideoLAN Client (b) Video Line Coder

(c) Video Length Coder (d) Video List Creater

(6) Which technology gives our location with accuracy ?

(a) GRS (b) GPRS (c) GRPS (d) GPS

(7) Which program is used to enter Unicode characters into any application ?

(a) Character Display (b) Character Insert

(c) Character Map (d) Character Select

(8) Which command is used to quit from R ?

(a) quit() (b) q() (c) exit() (d) close()

(9) Which function is used to create a bar graph in R ?

(a) bar() (b) plot() (c) bargraph() (d) barplot()

(10) Which of the following are different variants of RationalPlan ?

(a) Single, Multi, Viewer (b) Singular, Multiple

(c) View, Preview (d) Server, Client

LABORATORY EXERCISE

1. Create a zip file and store an entire directory in it.
2. Create a tar.gz file and store an entire directory in it.
3. Extract all files from the zip archive above into a subdirectory files1.
4. Extract all files from the tar.gz archive above into a subdirectory files2.
5. Locate your school on Google Maps. Can you identify the building in the satellite image?
6. Locate your home in Google Maps. Obtain directions for reaching your school from your home. Is it the same path you take?
7. Type सत्यमेवजयते in gedit using Character Map.
8. Type $\sin(\alpha - \beta) = \sin \alpha \cos \beta - \cos \alpha \sin \beta$ in gedit with help from Character Map.
9. Plot a bar graph of the number of days in each month of the (non-leap) year.
10. Plot a bar graph of the number of pages in your textbooks.
11. Plot a histogram of the frequency of letters in the names of days of the week (sunday, monday, …saturday). Only include the letters that occur at least once.
12. Create your own Skype user account and try to explore the facilities provided by Skype service.