Programming Guidelines

Learning Objectives

After studying this lesson the students will be able to:

- appreciate the importance of understanding and analyzing a problem appropriately before beginning with the application development.
- understand about some of the GUI application guidelines.
- demonstrate efficient program development practices.
- be familiar with and understand the stages of application development.
- identify different types of errors.

GUI Programming uses a simplified approach to programming. In GUI Programming, most of the components are predefined in a generic way to be adapted and incorporated according to the needs or requirements of the application. It provides a very comfortable feel to the programmer to develop applications without getting into complicated and cumbersome logic and commands. Most of the GUI tools are developed to provide simplified approach to programming and provide enormous inbuilt functionality to the programmer. This chapter will help the readers to learn how to utilize GUI tools to develop programs for various applications in efficient and effective manner.

GUI Application Development Guidelines

Some good application development guidelines are:

1. Understand the need of the application before starting the development.
2. Find out all possible inputs, which are required to produce the desired result or results.

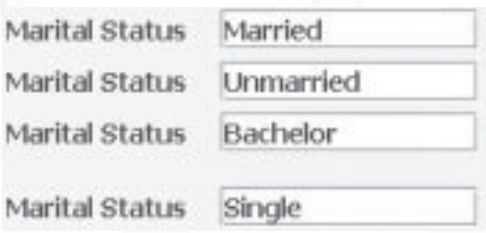


Figure 7.1 Ambiguities caused in User Input due to a Textfield

3. Make sure that the user provides appropriate information with minimum efforts to avoid ambiguity in data. The same can be done by appropriately deciding on the various input components - maximize use of radio button, checkbox, combo box, and list. Wherever possible avoid use of text field and text area for accepting inputs from the user to reduce ambiguity.

Figure 7.2 Avoiding Ambiguity by using Combo Box

4. Radio Button should be used wherever one of the option out of limited number of known set of options are required to be taken from the user. For example, for accepting gender (Male or Female), marital status (Single or Married), for accepting membership type (Monthly, Annual or Lifetime) etc.

Figure 7.3 Radio button

5. Checkbox should be used wherever multiple options are required to be selected from a limited number of known set of options. For example, for accepting multiple hobbies (Swimming, Singing, Dancing, Debating), for accepting food order in a restaurant (Pizza, Burger, Channa Kulcha, Pao Bhaji, Chowmein) etc.

Figure 7.4 Checkbox

6. Combo box should be used wherever only one of the option from a large number of known set of options is required to be taken from the user. For example, selecting state, selecting marital status, selecting schools etc.

Figure 7.5 Combobox

7. List should be used wherever multiple options are required to be selected from a large number of known set of options. For example, selecting multiple food items from a menu containing five or more number of items.

Figure 7.6 List

8. It is advisable to use List and Combo box when there are too many options as they help save space and are less cumbersome to design as compared to radio button and checkbox.

9. Options in a list or a combo box may be displayed in alphabetical order so that it is easier for the user to locate the appropriate option or may be displayed according to the probability of choice. For example, to take input of name of a state the names should be displayed according to alphabetical order, to take input of employee designation the highest level should be put at last and the lowest level should be put at top. The explanation for this is since there are more employees at lower levels as compared to higher levels, the probability of choosing the lower level is more. In short the most probable answer should be at the top.

10. It is advisable to use appropriate labels for each input and output options to help the user to correctly interpret them.

11. While writing the code do not use variable names as A, B, C etc. Instead use meaningful names and follow naming conventions. All the variables and constants must be named according to the naming conventions. They make the code easier to read, understand and maintain. For example a variable storing total marks should be named as Total. Similarly a variable holding cost price may be named as CP.

12. Ensure Clarity of expressions. All the expressions should be easy to understand for the user. There should not be a compromise to reduce the statements by loosing their clarity.

13. The conditional construct if..else should be preferred when there are very few alternative execution paths to choose from and also when decisions are to be made based on ranges of values or conditions. For example, to show gender based title etc.

14. The switch construct should be used when there are too many alternative execution paths and decisions is based only on a single integer or enumerated value (countable value). For example, to show weekday based message etc.

15. For repeating code a known number of times, the for loop is the best choice and when the number of iterations is not preknown, use the while or the do..while loop. When you want to test a condition at the end of the loop to see whether some block should be repeated, the do..while statement should be preferred. For example to sum 10 numbers for loop is the best whereas to accept password, the do..while is the best.

16. Use appropriate comments. Comments are very essential for providing internal documentation of a program. Comments can be used to explain the various complicated steps of the program thereby making the program more understandable for the user. In Java single line comments begin with '//' and multiple lines of comments are enclosed between '/*' and '*/' (quotes are not to be included).

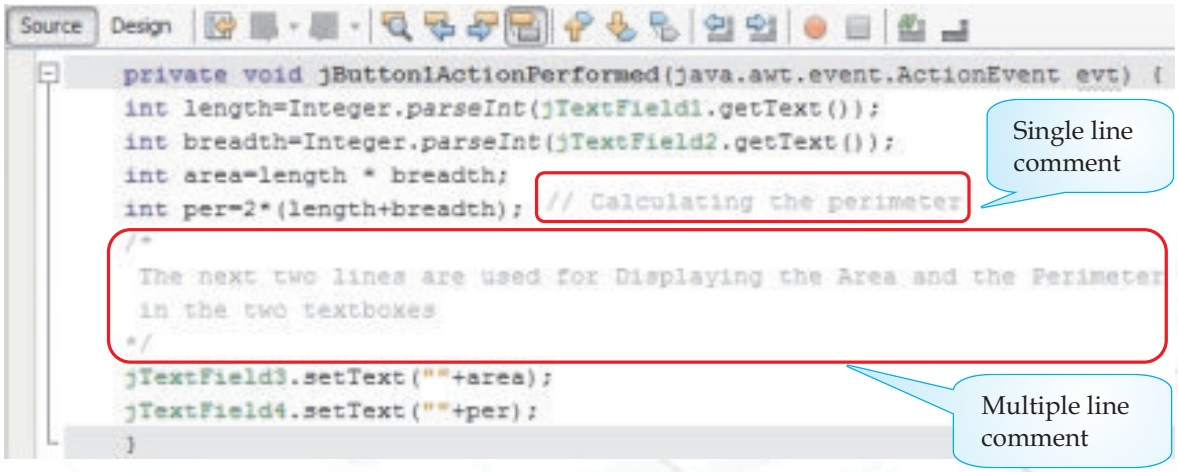


Figure 7.7 Adding Comments

17. Insert blank lines and blank spaces where necessary to separate logical group of statements.

18. Proper indentation must be used to highlight nesting of constructs like if, select or loops.

19. Avoid using Free formatting styles. In Java we can type any number of statements in the same line separated by a; (semi colon). This is called free

formatting style but it makes program less readable and difficult to debug. So we should avoid it and instead Prettyprinting should be encouraged.

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
 String txt1=jTextField1.getText();String txt2=jTextField2.getText();
 Integer num1=Integer.parseInt(txt1);Integer num2=Integer.parseInt(txt2);
 Integer sum=num1+num2; jTextField3.setText(""+sum);
}
```

Figure 7.8 Coding using Free formatting styles is difficult to read and debug

Prettyprinting is the formatting of a program to make it more readable. These formatting conventions usually consist of changes in positioning, spacing, color, contrast, size and similar modifications intended to make the content easier to view, read and understand. Prettyprinters for programming language source code are sometimes called code beautifiers or syntax highlighters. Netbeans supports prettyprinting and the shortcut key to format any source code in Netbeans is Alt+Shift+F.

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    String txt1 = jTextField1.getText();
    String txt2 = jTextField2.getText();
    Integer num1 = Integer.parseInt(txt1);
    Integer num2 = Integer.parseInt(txt2);
    Integer sum = num1 + num2;
    jTextField3.setText("" + sum);
}
```

Figure7.9 Coding using PrettyPrinting formatting style is easy to read and debug

20. The application must be reliable. It must be capable of handling situations like wrong input or no input. It should display proper error messages in case of such errors.

21. The application should be portable. It should be able to run on different platforms.

Stages of a Simple GUI Application Development

Various steps involved in the development of a new application are as follows:

1. **Analysis:** This phase involves the following steps:

   - indepth understanding of the problem

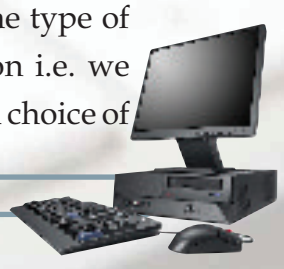
- deciding the requirements from the new system
- jotting down possible inputs and outputs that are required for obtaining the desired solution.

For example, to make the Member Counter Application in Chapter 6 we first performed a detailed analysis about what is expected from the application. Next we noted down the possible inputs and outputs i.e. we understood that the user has to input a name which is to be appended to the contents of a text area. After one run, the user is again to be asked if he wishes to continue and the application should terminate only when the user wants to.

2. **Design:** This phase involves planning of step-by-step procedures required to solve a given problem. At this stage a detailed design of the following components is to be completed:

- **Inputs:** involves defining the kind (data type) of data to enter into the application. In this stage we should also decide on the type of input components to minimize ambiguity and inconsistency.
- **Outputs:** decide on the possible data to be displayed from the application and also how, where and when it is to be displayed.
- **User Interface (Forms):** involves designing of the screen the user will see and use to enter data or display data. The placement of various input-output components on the form in an aesthetic and visually appealing manner is a major step in this phase.
- **Modular Components:** involves breaking of complex steps into simple ones to attain the target. Depending on the user interface this step will involve deciding on the functionality required from each component placed on the form to obtain the desired output.
- **Algorithms:** involves creating a simple solution in the form of steps called an algorithm and it helps in making the coding process easier.

For example, after completing the analysis stage in the Member Counter Application we proceed to the design stage where we first decide on the type of input and output required based on the requirement of the application i.e. we decide that the input will be accepted using a dialog box (to give the user a choice of



adding or cancelling) and the output will be displayed in a text area (so that lots of names can be displayed). Each input and output type is decided based on the analysis done in stage 1. Next step is to design the user interface or the Form and place the relevant components as we did in the case of our Member Counter Application.

3. **Coding:** This phase involves actual writing of programs using appropriate programming languages. A good programmer will make an optimum code, which is readable, easy to understand and maintain with appropriate error handling, comments and indentation.

   For example, after designing the form, we wrote the actual code using java programming language.

4. **Testing and Debugging:** Virtually all applications have defects in them called 'bugs' and these need to be eliminated. Bugs can arise from errors in the logic of the program specification or errors in the programming code created by a programmer. Testing means the process of executing the application with possible set of input data in order to find probable errors. Debugging means correction of those errors in the application. In the testing and debugging stage, we should try out all possible inputs in order to make our application error free.

   For example, in our Member Calculator Application, what will happen if we input numbers instead of a Name in the dialog box? Trying out and fixing up all such errors is the aim of this stage of application development.

5. **Documentation:** Documentation means the instructions and information about the usage of the application. Providing the documentation makes it easier for the end user to understand the functionality of the application.

   For example, giving appropriate comments in all our applications is part of documentation as it clearly tells the user and the programmer about what a particular part of the code is doing.

6. **Application Delivery and Maintenance:** The completed software is packaged with full documentation and delivered to the end users. When the end users use the software, bugs that were not found during testing may appear. The maintenance involves the rectification of previously undetected errors and changes that are to be

made for the enhancement of the functionality of the application. An updated version of the software with the reported bugs corrected and enhancements is then sent as a replacement to the end user.

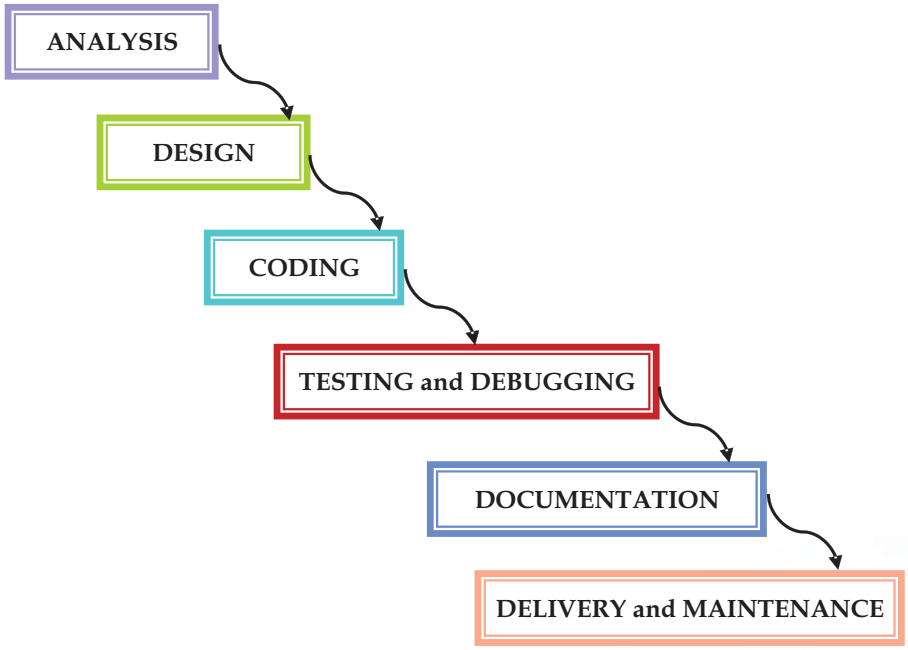


Figure 7.10 Stages of Simple GUI Application Development

Please note that the single-sided arrow on the right side of the stage indicates that we should proceed to the next stage only when the preceding phase is completed and perfected.

Know more

This model is known as the waterfall model of Application development. There are several other modifications of the model explained above.

Types of Errors

The different types of errors encountered while developing any application are explained below:

1. **Syntax Errors:** Formal set of rules defined for writing any statement in a language is known as syntax. Syntax errors occur when syntax rules of any programming language are violated. These errors occur during compilation of the application but

in Netbeans these errors are highlighted in design stage itself using the error indicator as shown in Figure 7.11. Some of the common examples of syntax errors are missing semicolon, missing parenthesis and using incompatible data types.

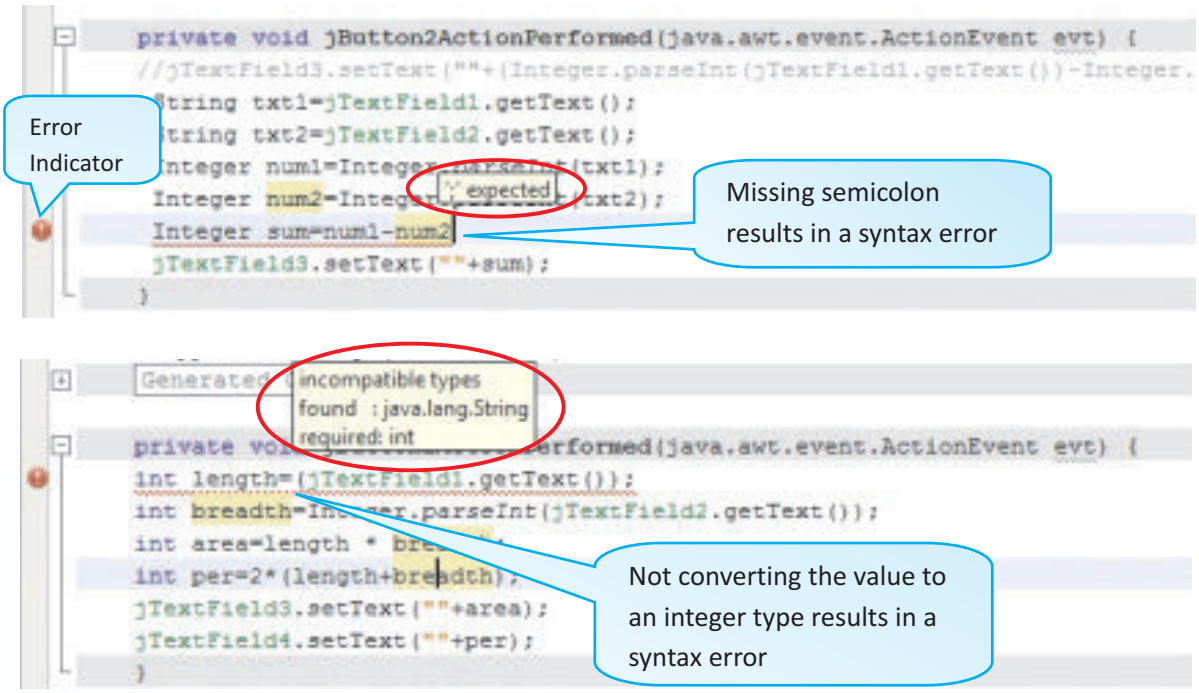


Figure 7.11 Common Syntax Errors

2. **Run time errors:** If an application is syntactically correct then it is compiled and translated into machine language and is ready to be executed. Run time errors occur during the execution of an application. These errors will result in abnormal termination of the application. Some common examples of runtime errors are Division by zero, trying to convert to number (int or double) from an empty jTextField etc.

3. **Logical errors:** In case the application is not giving any compilation or runtime error but still giving a incorrect output, it is due to logical errors. These Errors occur due to mistakes on part of the programmer. They are the most difficult errors to find and debug. One such common example of logical error is using a wrong formula as Eng+Maths+GK/3 instead of (Eng+Maths+GK)/3 for calculating average of 3 subject marks. A very effective technique to locate logical errors is placing output statements (for example using jMessageDialog) to print intermediate values at strategic places in your code to track down the cause of the

error during testing phase. Once tested with sample data, these output statements must be removed.

Exception Handling

Run time errors are also called exceptions, and handling such errors in the application is called exception handling. In java exception handling is done using try{ } and catch{ } blocks. Statements that can raise an error are placed inside the try{ } block and its handling code is written inside the catch{ } block.

Summary

- While designing the form for a GUI, make sure that the user provides appropriate information with minimum efforts to avoid ambiguity in data.
- Decide on the type of input components and constructs carefully after understanding the need of the application
- Use blank lines, appropriate comments and proper indentation to make a program more readable
- Prefer prettyprinting formatting style over free formatting styles
- Stages of a GUI application development includes Analysis, Design, Coding, Testing and Debugging, Documentation and Application Delivery and Maintenance
- Common types of errors encountered in an application are syntax errors, run time errors and logical errors
- Run time errors are also called exceptions
- Writing code to handle run time errors in a java application is called exception handling

Multiple Choice Questions

1. _____ is the process of translating a task into a series of commands that a computer will use to perform that task.

A. Project design

B. Installation

C. Systems Analysis

D. Coding

2. Translating the problem statement into a series of sequential steps describing what the application must do is known as:

A. Coding.

B. Debugging.

C. Creating the algorithm.

D. Writing the documentation

3. Which of the following component is the best suited to accept the country of the user?

A. List

B. Combo box

C. Radio button

D. Check box

4. Which type of loop is best suited to check whether the password input by the user is correct and display an error message?

A. for

B. do..while

C. while

D. All of the above

5. Which construct will you use to find the sum of the first 10 natural numbers?

A. switch statement

B. for loop

C. if..else statement

D. None of the above

6. Which of the following is not a good programming guideline?

A. Adding lots of comments

B. Using prettyprinting

C. Using text fields to accept input of marital status

D. Designing visually appealing forms

Exercises

1. Excessive comments add time to the execution of your program. (True/False). Justify your answer.
2. Differentiate between compile time and run time errors.
3. Which error is harder to locate and why?
4. Explain the following terms:
   a) Exception handling
   b) Syntax
   c) Portability
   d) Prettyprinting
   e) Syntax error
5. The code given below will give an error on execution if the value entered in t2 is 0. Identify the type of the error and modify the code to handle such an error.

```
int a,b,c;
a= Intger.parseInt(t1.getText());
b= Intger.parseInt(t2.getText());
c= a / b;
```