THE BEGINNER'S GUIDE TO PROGRAMMING

Terms and Conditions

LEGAL NOTICE

The Publisher has strived to be as accurate and complete as possible in the creation of this report, notwithstanding the fact that he does not warrant or represent at any time that the contents within are accurate due to the rapidly changing nature of the Internet.

While all attempts have been made to verify information provided in this publication, the Publisher assumes no responsibility for errors, omissions, or contrary interpretation of the subject matter herein. Any perceived slights of specific persons, peoples, or organizations are unintentional.

In practical advice books, like anything else in life, there are no guarantees of income made. Readers are cautioned to reply on their own judgment about their individual circumstances to act accordingly.

This book is not intended for use as a source of legal, business, accounting or financial advice. All readers are advised to seek services of competent professionals in legal, business, accounting and finance fields.

You are encouraged to print this book for easy reading.

Table Of Contents

Foreword

There are several different concepts that an individual needs to understand before being able to tackle the issue of programming concepts and how they unfold.

The Beginner's Guide to Programming

Chapter 1:

Understanding Programming Concepts And How They Work

Synopsis

Programming concepts may include information on how to write Web Logic startup scripts, creating Web services within the Web Logic site, steps on how to understand how the particular programs being written works and many more platforms that require thorough understanding.

Where To Start

Some would say that computers are actually very dumb tools that cannot function without a preset guidance module incorporated into its system. This would literally mean that a computer program will not be able to be of any use without actually being “told” what to do. A computer is unable to act or think for itself, unless the appropriate accompanying software used clearly depicts step by step what needs to be done for any particular activity. Therefore, to understand how the programming works, the user will first have to understand the concepts required to make the system work in the first place.

Some of these should ideally include understanding the terms and programs. Among which would be the famous and basic word processor program, where the input of characters typed using the keyboard will execute the formats of the text, correct any mistakes in spelling and grammar and create the output of displays and points that are neatly organized in text form to be readable for the user.

There are also the game programs where the keystrokes or joystick movements are part of the input which constitutes the calculation of speed and distance the animations take on the screen. These calculations have the output of actions and movements of the particular animation on the screen being viewed.

The stock market predictor is another program that has great input capacity and the ability to give information on the current and past

prices of stocks. This is where the programs tries to recognize the trend in the price fluctuation and has the capability of predicting the future probable price of stocks.

Chapter 2:

Techniques Of Writing A Program

Synopsis

There are several important aspects to consider before actually commencing on writing a particular program. Without these considerations in place, the individual could end up writing a program that will be rendered useless or inadequate in its ability to perform its required actions.

Helpful Techniques

Some of the steps taken first should be to identify the problem so that the program can be designed to tackle the basic needs of the user who chooses to use the programs. Then, there would also be a need to identify the user in terms of their reasons for actually using the program in question. There will also be a need to identify the computer that is being used with the program. Computers such as Windows, Macintosh, mainframe computers, and computers running Linux, Handheld Palm or Pocket PC or any other super computers should also be taken into account. This is due to the fact that certain programs are not compatible with some of the tools the users are

currently using. The programming skill of the individual should also be considered, as the competency level of the eventual outcome of the programs should be able to address all the various issues that might arise during the use of the particular program.

Understating that a program is only useful if it is able to address the issues it is being used for is the ultimate goal for its existence. If the program is not able to address this in its completed form, then the programs is usually deemed unfit for use. Therefore, it is important to ensure the writing of the programs covers every detail and reason for the program's usage before it is released for use.

Chapter 3:

The List Of Programming Languages

Synopsis

As with the multitude of human languages used for communication purposes, the computer systems also requires and uses various languages to communicate its information to the user. These languages are usually based on certain syntactic and semantic rules, which contribute to the defining of the meaning of each programming language and its individual implementation.

How Its Made

The programming languages are put in place to provide the computer programmers with the means to create and express computer algorithms.

The following are some of the more popularly used languages:

APL – this array of programming languages works simultaneously on a multiple array of data platforms and is interpretive, interactive and functional as a programming language.

AutoIt – this freeware automation language is for Microsoft Windows and its main intent is to create automation scripts that can be used for the execution of some repetitive tasks that are within the Windows application.

Basic – also otherwise referred to as the beginner's all-purpose symbolic instruction code, this was designed to provide non scientifically inclined users access to computers. Microsoft Visual Basics is predominantly using this program.

Eiffel – this is an object orientated programming language that is ISO standardized and developed to be able to provide extensible and reusable software. It is popularly used in industries such as finance, aerospace and video gaming. It has since evolved to include many other functional programming features such as Ubercode, which is a high level platform sharing language.

Forth – this structured imperative programming language is based on the implementation idea of stacks, while supporting an interactive execution of commands, as well as a compilation of sequences of commands at each use.

Frink – this program is based on the Java Virtual Machine and focuses mainly on science and engineering elements. Its main feature of tracking unit measurable through calculations enables the quantities to contain their respective unit's within the measurements keyed.

Chapter 4:

How To Choose The Right Compiler

Synopsis

Being able to understand and select the best compiler is pivotal to the final outcome of the program designed. There is a need to ensure the systems used within the compiling exercise are compatible.

Choose The Right Stuff

When initiating the compiling exercise, the designer should ideally be aware of two very important points, which are what platforms will eventually be used to run the programs created and what assumptions are to be made in the code used. This will ultimately determine the compiler option to be used.

The target and choice of platforms will directly determine points such as 32 bit or 64 bit instruction sets, instruction set extensions which the compiler will eventually need to use and instruction scheduling, which will depend on the instruction execution of the time and cache configuration.

The target platform will specify the processor that the application is eventually expected to run on, along with the minimum processor that it is required, and whether the application is going to be 32 bit or 64 bit. It will be better to clearly specify the target architecture to avoid unforeseeable changes in the compiler flag. There are usually a number of compiler flags that work together to specify the target architecture, where some are more appropriate as default value for the given target processor, while some are for support and others for instructions. The order of these compiler flags are also significant, as the flags accumulate form left to right in the event that there is conflicting settings, the flag on the right will override the values of the flags which have been specified earlier on the command line.

Chapter 5:

What Is An Interpreter

Synopsis

This is basically a program extension language, AEPL, which is designed to allow complex user input, it describes complex word outflows and easily extends available allocation logic. This makes the interpreted language a phase of programming language in which programs are indirectly executed by the interpreter used at the given time.

Interpreter Basics

Although, theoretically any language can be done in a compilation format or interpreted as it is designed, in actuality it would be applied because of a common platform that facilitates the implementation practice and not some essential property language element. In most cases, the programming language has little performance differences between an interpretive or compiled based approach to the implementation style.

There are a variety of languages that can be implemented, using both compilers and interpreters, and this includes Basic, Pascal, C, Lisp and Python. The Java language is usually translated to a format that is interceded to be ideally interpreted, in the “just in time” compilation which is used to generate the machine code.

The interpreting platform created by the language used gives the implementations some additional flexibility over compiler implementations. Some of the features that are often easier to implement through the interpreters instead of the compilers would include platform independence, which uses Java’s byte code as an example, reflection and reflective use of the evaluator, which is a first order eval function, dynamic typing, smaller executable program sizes, which are primarily used since implementations have some level of flexibility.

Chapter 6:

Writing Your Program With An Editor

Synopsis

Using an editor for program writing is useful as the additional services executed will differ from the general use of the word processor, which would include elements such as Microsoft word or WordPerfect as these usually do not contain tools to change formatting, fonts, margins and general layout styles.

Having Some Help

By default, Word puts the formatting and layout information directly into the file used, which actually confuses the compiler and complicates certain usage elements.

A doc file in a text editor format will usually be found with its own formatting code, while with the text editor there would not be a need for added formatting, which in turn makes the compile code easier to read. Ideally, it has a set of different features when compared to that of the traditional word processing programs, which may include the inability to use or include pictures, tables or double spaced writing. The general features of the text editors will vary from implementation to implementation, but are mostly there in the kinds of frequently used features, and one of which is the syntax highlighting feature, which is very useful indeed. This means the editor will be able to highlight certain words or types or syntax specific language. Also, the editor will be able to have all quoted text show up in colored font. The editor can also indicate mismatched parentheses or brackets by turning them red.

Versatility is also another element of the editor, as the editor knows which language is being used for the programming. This is made possible by either “telling” the programs or by the detection of the suffix within the file. If the file being worked on is code.cc, it will see the .cc and automatically use the C++ rules, however if it is one called code.html, it will automatically apply the HTML rule instead.

Chapter 7:

The Functions Of A Debugger

Synopsis

A debugger is supposed to function as a program that “runs” other programs, while allowing the user to exercise some form of control over the programs downloaded. The bugger is also supposed to function as a detecting agent that examines elements within a program when it runs into trouble or goes amiss.

What It Does

The following are some of the basic features or functions of a debugger:

- When the user initiates use of a particular program that has errors, these errors will usually manifest themselves during the actual running of the program. Here, the user should be able to identify which statement or expression the program is using for its execution when the errors occurred through the debugger.

- The debugger will also assist the user if and when a fatal error occurs while the program is running and identify the line of program that contains the call to the particular function shown.

- The values of the programs variable, including the parameters of a particular point within the usage of the execution of the program is also part of the debugger function.

- Being able to certify the results gained from the evaluation of a particular expression at some point within the program execution is also possible with the debugger. The sequence of statements actually executed within the program is also depicted by the debugger's actions.

All these function are actually in practice to ensure the debugger is able to examine the program data, where a track back action can be initiated based on the information noted.

Chapter 8:

Ease Your Burden With Components

Synopsis

Basically, component orientate programming is a new element of extensibility, in which the development and integration of extensions are a distributed activity and not a centralized action as in previously used software applications, developments and paradigms.

Make Things Simple

Component focused programming for computer languages must be designed to have some level of basic properties that will eventually lend support, rather than cause impediments to the user. This is important to the distribution platform applicable to the extensibility of the software system in place. Although most would agree that component orientated programming is a good extension tool to have available for use, there is less enthusiasm when it comes to agreeing on what the fundamentals of component orientated programming is really based on. Over time, there has been a number of approaches and technologies that have been proposed for component oriented programming, which some have deemed useful while other feel there is a lot that still needs attention.

Based on three very fundamental elements, the component oriented programming language starts with the need for interfaces, the need for modules and finally the need for polymorphism. The interface element will consist of all possible implementations that can fill a certain and specific role within the already composed system. This is possible as they are often the only form of coordination between the frameworks and components used and the only means by which the whole composition can be validated.

These modules define the static structure of a particular system being used by providing rather rigid boundaries which are designed not to be crossed arbitrarily. This ensures the interaction between the

framework and components are kept isolated and this makes dependencies explicit.

Polymorphism, on the other hand, supports the dynamic structure of a system by allowing various different instances of different implementation types to be bound to the same interface at runtime.

Chapter 9:

Optimizing Your Program With Profiler

Synopsis

Profiling, as the term implies, is gathering and analyzing material. In this case, it is related to the software engineering process where the profiling is a form of dynamic programs analysis.

Profiling

This process is done by measuring the usage of memory, the usage of particular instructions or the frequency and duration of function calls. However, it seems that the most common use of the profiler tool is to aid program optimization. The profiling exercise is usually managed by instrumenting either the program source code or by its binary executable form, using the tool called the profiler or the code profiler. The methodology used is usually classified under the profiler as an event based, as statistical based, as instrumentation based or as simulation based. The program analysis tools are essential in providing some level of understating of the program's behavior. Computer architects and designers need the assistance of such tools to help evaluate the level of efficiency that the programs will eventually be able to achieve. Software writers also need these tools to analyze their programs to evaluate critical sections of code, while compiler writers use these tools to find out how their respective instruction scheduling or branch prediction algorithm is capable of performing once the programs are in use.

A trace for sequential programs can usually be accommodated with the summary profile but performance problems in parallel programs often depend on the time lines and the connection to the event. Therefore, there is a need to have a full trace done to better understand the actions caused and the implications that will be produced.

Chapter 10:

Installing Your Program

Synopsis

Understating the elements within the installing process will help the individual better tackle the task. This can usually be done quite adequately with the least amount of frustration, even for the less technologically savvy individual.

Relax! It's Not That Hard

It is possible to change the language Windows uses to display text in wizards, dialog boxes, menus and other items within the user's interface platform. Although most display languages are usually installed by form of default, there are leeways provided to the user to have the choice to install the additional language files as well as when to do so.

There are basically two types of language files that are popularly chosen for installing the programs. These will come in the form of the Windows 7 Language Interface Packs or otherwise referred to as LIPs and the Windows Language Packs. The Windows Language Packs or LIPs provides a translated version of the most widely used areas of the user interface and are usually available for free to download whenever needed. Within this, there is a rather long list of languages that are available and these languages can be found on sites such as Microsoft Local Language Program.

The Windows 7 Language Packs provide a translated version of the elements needed most for the user's interface execution. However, before any of the installation of the display languages can be done, the user will need to access the language files first. These files are usually located on the computer, a computer within the network or from Windows DVD. The user can also access these language packs by downloading them directly from the web; however, there are some differences on how to go about acquiring these language interface packs.

Wrapping Up

Correctly programming something can indeed be very frustrating. That is why you have been provided with the helpful hints and tricks you read about above. Take some time to relax, think about your objective and approach it with a game plan in mind. Remember to keep the tips you just learned in mind while making your game plan. Good luck!